Computación y Sistemas

Vol. 21 No. 2 April - June, 2017

Editorial

We have the pleasure to introduce this number of Computación y Sistemas that includes nine regular papers and five special issue articles. On the one hand, regular articles cover a wide range of computer science subfields, such as computer vision, robotics, software engineering and signal processing. On the other hand, special issue papers focus in text authorship analysis.

It is well known that the Web is not only the greatest repository of digital information ever invented but also the largest communication platform. Its current development has allowed the exchange of information among people from different places, ages, cultures and conditions. However, it also has facilitated the dissemination of anonymous texts as well as the inappropriate copy and use of content from others. This situation has raised great research interest in authorship analysis approaches, which focus on answering important questions such as, given a document, who wrote it? And what are its author's traits? This special issue presents to the reader five papers devoted to authorship analysis. In the following lines we outline the main contribution of each article.

Hernández-Castañeda and Calvo (Mexico) propose using semantic information to solve the author verification problem. Particularly they create a semantic space model by means of Latent Dirichlet Allocation (LDA). Their experimental results on the corpus used at PAN 2014 and PAN 2015 showed that LDA aids to verify authorship when there is limited training data, i.e., less than five short texts written by a specific author.

Castro et al. (Cuba, Spain) further develop the topic of authorship verification. They propose a neighborhood classification method that considers the similarities between a document of unknown authorship and some sample documents from the author of interest. Similar to the previous paper, they also show results in datasets used at PAN 2014 and 2015.

Alekseev and Nikolenko (Russia) address the problem of author profiling. In their paper entitle “Word Embeddings for User Profiling in Online Social Networks”, they consider the application of word embeddings to user profiling. In particular, they study the effectiveness of such representations to predict the age and gender of social media users.

Tutubalina and Sergey Nikolenko (Russia) continue the topic of author profiling but in a domain specific scenario. They propose several approaches to extract demographic information, such as gender and age, from user reviews concerning medical products or services. In the experiments, they compare some modern natural language techniques, including feature rich classifiers, topic models and deep neural networks.

Finally, Litvinova et al. (Russia, Mexico) propose a method for identifying suicidal behaviors of individuals based on an analysis of their blogs. They mainly describe the design of a mathematical model to classify texts as suicidal or non-suicidal. This model considers a set of linguistic and stylistic features that are not significantly dependent on the content. Experiments are on Russian blogs; the obtained results are comparable to state-of-the-art for English texts.

This issue of the journal also contains eleven regular papers.

Amarnath Pathak et al. (India, Mexico) describe a system for search and retrieval of scientific papers that contain mathematical expressions. In their paper “MathIRs: Scientific Documents Retrieval System” they propose a method for identification and matching mathematical expressions according to the user’s query. While

Computación y Sistemas, Vol. 21, No. 2, 2017, pp. 163–165
ISSN 1405-5546
doi: 10.13053/CyS-21-2-2592

mathematical text may seem "logical" and thus easy to follow by the computer, in fact the way mathematicians express their ideas is very difficult to formalize, which makes retrieval of mathematical texts a difficult and important task. The authors show that their method achieves high performance.

Pacheco-Morales et al. (Cuba, Spain) present a work in image processing. In their paper entitle "Un método para el reconocimiento de objetos ocluidos", they propose a method to recognize occluded objects in digital images. The proposed method considers the use of Hidden Markov Models for segmentation of overlapping objects. Experiments on a set of different databases demonstrate the high effectiveness of the proposed method.

Cruz-Matías and Ayala (Mexico, Spain) present a contribution to the field of computer graphics. In their paper entitle "Compact Union of Disjoint Boxes: An Efficient Decomposition Model for Binary Volumes", they propose a decomposition model for binary volumes called Compact Union of Disjoint Boxes (CUDB). They analyze some of the main characteristics of this model in comparison to previous works and present algorithms for conversion to and from other models. Experimental results show that CUDB is smaller in number of elements and so in storage size than existing models.

Platas-Garza and Rodríguez-Maldonado (Mexico) present a paper on halfband filters (HBF), which are very relevant to perform several signal-processing tasks. In their paper entitle "Design of Flat Halfband Filters With Sharp Transition and Differentiators Through Constrained Quadratic Optimization", they propose an alternative method for the design of type I Halfband FIR filters with flat magnitude and narrow transition bands. They also show the design of type IV FIR digital differentiators through the proposed method. Design examples are presented to demonstrate the effectiveness of the proposed methodology.

Domínguez-Guerrero et al. (Mexico) contribute with an article in robotics. In their paper, "Identificación del campo de trabajo de un robot hexápodo utilizando optimización multiobjetivo", they present a method to find the workspace of a hexapod mobile robot. This method applies multi-objective optimization to estimate the optimal set of configurations for walking in any viable direction. The proposed method is validated by simulation using the Robotics toolbox of Matlab.

Martínez-García et al. (Mexico) present a paper entitled "Bridging The Gap Between Model-Based Design And Reliable Implementation Of Feedback-Based Biocircuits: A Systems Inverse Problem Approach". This paper focuses on the tuning of mathematical models describing the design of synthetic biological circuits. The proposed tuning methodology combines exact algebraic parameter reconstruction with nonlinear observed-based parameter estimation. This methodology is illustrated via computer–based simulations involving the tuning of a state–based model describing a well–known cyclic feedback bio-circuit.

Arcos-Argudo (Ecuador) contributes with an article in small-world networks, a topic with lot of applications in different sciences such as sociology, geology and neuroscience. This paper mainly presents a comparative study of two algorithms, Kleinberg and Biased Selection, which aim to establish a direct connection between any node in a graph and a randomly "long-range neighbor".

Serna et al. (Colombia) present a paper entitled "A model for determining the maturity of automation of software testing as a research and development area". This work describes the results of a research on the maturity level of test automation. It is based on a systematic review of the related literature and its main conclusion is that the maturity level of automation of software testing is teenager.

Toriz-Palacios and Sánchez-López (Mexico) present a new proposal to build maps of complex

environments. In their work “A B-Splines Based Data Association Method for the SLAM Problem in Complex Environments”, they describe a proposal to use B-spline curves to extract characteristic points of the detected obstacles. They show the results obtained using real and simulated information, and conclude that the proposed method achieves great precisions in map construction of complex environments, a difficult task with techniques that currently exist.

Cuevas et al. (Mexico) depict a method for parameter identification in chaotic fractional systems. In the paper entitle “Parameter estimation for chaotic fractional systems by using the Locust Search Algorithm”, they describe a method that uses the evolutionary technique based on the behavior of swarms of locust for parameter identification of fractional order chaotic systems. This method avoids the concentration of individuals in the best positions, eliminating critical flaws such as the premature convergence to sub-optimal solutions and the limited exploration-exploitation balance.

Ortiz-Rangel et al. (Mexico) contribute with an article in image processing. In their paper entitle “Filtrado de Ruido Gaussiano mediante Redes Neuronales Pulso-Acopladas”, they propose an algorithm to reduce the effect of Gaussian noise in grayscale images. This algorithm is based on the Intersection Cortical Model (ICM), but it also considers a Time Matrix (TM) that provides information about the iteration when the neuron (pixel) fires for first time. Simulation results varying the degree of Gaussian noise show better effectiveness of the proposed method than well-known filters.

Special Issue editors,

Manuel Montes-y-Gómez
Luis Villaseñor-Pineda

Author Verification Using a Semantic Space Model

Ángel Hernández-Castañeda[1,2], Hiram Calvo[1]

[1] Instituto Politécnico Nacional (IPN),
Centro de Investigación en Computación (CIC),
Mexico

[2] Tecnológico de Estudios Superiores de Tianguistenco (TEST),
Tecnológico Nacional de México (TecNM), Estado de México,
Mexico

ahernandez_a12@sagitario.cic.ipn.mx, hcalvo@cic.ipn.mx

Abstract. In this work we propose to solve the author verification problem using a semantic space model through Latent Dirichlet Allocation (LDA). We experiment with the corpus used in the author identification tasks at PAN 2014 and PAN 2015. These datasets consist of subsets in the following languages: English, Spanish, Dutch and Greek. Each problem contained in these corpora is formed by one to five known documents which were written by one author and one unknown document. The task is to predict whether the unknown document was written by the author who wrote the known documents. We processed the documents in the dataset and captured the fingerprint of authors by generating a probabilistic distribution of words in the documents. In PAN 2015 classification, we achieved 81.6%, 75.4%, 74.1%, 67.1% accuracy for each English, Spanish, Dutch and Greek subset respectively. In particular for the English subset, we outreached the best result reported in both competitions.

Keywords. Author verification, semantic space model, cross-genre, cross-topic, latent Dirichlet allocation.

1 Introduction

Author verification is an important problem to solve, since many tasks require recognizing the author who wrote a specific text. For example, knowing which author wrote an anonymous book, or identifying notes of a serial killer. In this paper we deal with an author verification challenge in a more realistic setting.

Specifically, datasets used (two of them) consists of one to five documents by a known author and one document by an unknown author. Datasets are formed by subsets in different languages (English, Spanish, Dutch and Greek).

The aim is to identify whether a written unknown text was written by the same author which wrote the known texts. It is important to note that this task becomes more difficult when the dataset is composed of short documents, since common approaches are not able to capture effective models with few amounts of words [22]. However, in real cases within the forensic field, long texts rarely exist.

Several approaches have been conducted to generate more informative features based on text style; it is possible to generate features by extracting lexical, syntactic, or semantic information among others. Usually, lexical information is usually limited to word counts and occurrence of common words. On the other hand, syntactic information is able to consider, to a certain extent, the context of the words.

In this work we use semantic information to find features that help to discriminate texts. For this purpose, we create a model using Latent Dirichlet Allocation. By using this method, we take into account all vocabulary from all texts at the same time, and after a statistical process, find to what extent the relations between words are given in each document.

LDA is a statistical algorithm that considers a text collection as a topics mixture. Processing a

set of documents by LDA returns a set of distributions of topics. Each distribution can be seen as a vector of features and a fingerprint of each document within the collection. Then we use machine learning algorithms to classify the obtained patterns.

We evaluate the proposed approach on two datasets where an author verification task is tackled: the corpus PAN 2014 and PAN 2015.

2 Related Work

Several works have attempted to study the authorship identification challenge by generating different kinds of features [10, 12]. The nature of each dataset can determine the difficulty of the task, that is, how hard it will be to extract appropriate features [1, 8]. In [13], a study of techniques can be found, which show that, while the number of authors increases and the size of training dataset decreases, the classification performance decreases.

This seems logical, since the size of training data is smaller, and thus, the identification of helpful features is affected.Many works address author identification through their writing style [15, 16, 25]. For instance, in [9], style-based features are compared to the BoW (Bag of Words) method. They attempt to discriminate authors from texts in the same domain obtained from Twitter.

Style markers such as characters, long words, whitespaces, punctuation, hyperlinks, or parts of speech, among others, are included. The authors found that a style-based approach was more informative than a BoW-based method; however, their best results were obtained by considering two authors, so there was an accuracy decrease when the number of authors was increased. This suggests that, depending on how big is the training set, there will be stylistic features that help to distinguish an author from another, but not from every other author.

Stylistic features can be also applied to other tasks. In [2], Bergsma et al. combined features to address two-class problems. They attempted to obtain style, Bag of Words (BoW) and syntax features to classify native and non-native English writers, texts written for conferences or workshops (classification of venue) and texts written by male or female writers (classification of gender).

Their dataset consists of texts that are scientific articles—this kind of texts is more extensive, unlike e-mail, tweets, and other short texts. So, this could have led to identify non-native written texts with promising accuracy. Nevertheless, long texts do not ensure good results, since their classification tasks on venue and gender obtained low accuracy.

The purpose of identifying authorship can vary. For example, Bradley et al. [4] attempt to prove that it is possible to find out which author wrote an unpublished paper (for conference or journal); only by considering the cited works in it. By using latent semantic analysis (LSA) [6], the authors propose to create a term-document matrix wherein possible authors are considered as documents while cited authors are considered as terms. LSA assumes that words that are close in meaning will occur in similar pieces of text. The results of Bradley et al. showed that the blind review system should be questioned.

Another example is Castro and Lindauer [5], with the task of finding out whether Twitter user's identity can be uncovered by their writing style. The authors focused in features as word shape, word length, character frequencies, and stop words' frequencies, among others. With an RLSC (regularized least square classification) algorithm, the authors correctly classified 41% of the tweets.

In Pimas [18] the author verification task is addressed by generating three kinds of features: stylometric, grammatical and statistical. Pimas et al. study is based on the PAN 2015 authorship verification challenge.

In that work, topics distribution is considered as well, but they argue against using it because the dataset is formed of topic mixtures in a way that affected their results. A cross validation model (10 folds) shows good performance; on the other hand, the model showed overfitting using the specified training and test sets.

3 Proposed Approach

As we found in the previous section, features based on stylistics, syntactic and lexical information consider separately each written

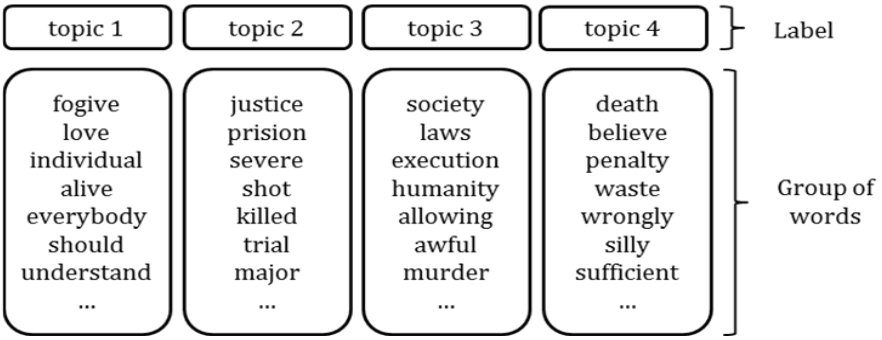


Fig. 1. Example of generated topics by using LDA

document. For instance, it is possible to obtain stylistic features by counting the number of stop words and characters by document.

These occurrences of symbols, in short and cross-domain texts, can merely be random and they do not allow capturing writing style.

In a collection of cross-domain texts, we infer that linked words and the distribution of them in the texts may provide more informative features, since, in first instance, LDA's estimated distributions of topics will depend on the content of each text in the collection.

In this section we present our method for generating features. First, in Section 3.1 we detail the source of features we use. Next, in Section 3.2 we describe the datasets used in this work for evaluation, and finally in Section 3.3 we give details on our feature vector construction.

3.1 Source of Features

Latent Dirichlet Allocation [3] is a probabilistic generative model for discrete data collections such as a collection of texts; it represents documents as a mix of different *topics*.

Each topic consists of a set of words that keep some link between them. Words, in turn, can be chosen based on probability. The model assumes that each document is formed word-by-word by randomly selecting a topic and a word for this topic. As a result, each document can combine different topics. Namely, simplifying things somewhat, the generation process assumed by the LDA consists of the following steps:

1. Determine the number *N* of words in the document according to the Poisson distribution.
2. Choose a mix of topics for the document according to Dirichlet distribution, out of a fix set of *K* topics.
3. Generate each word in the document as follows:

    a) choose a topic;
    b) choose a word in this topic.

Assuming this generative model, LDA analyzes the set of documents to *reverse-engineer* this process by finding the most likely set of topics, which may compose a document. LDA automatically generates the groups of words (topics); see Figure 1.

Accordingly, LDA can infer, given a fixed number of topics, how likely that is each topic (set of words) appears in a specific document of a collection. For example, in a collection of documents and 3 latent topics generated with the LDA algorithm, each document would have different distributions of 3 likely topics. That also means that vectors of 3 features would be created.

3.2 Datasets

The proposed approach was tested with different datasets. Included datasets are PAN 2015, which is based on cross-topic and cross-genre documents, and PAN 2014, consisting of specific-domain documents.

Table 1. Dataset details for the PAN 2015 task of author identification

Language	Training problems			Test problems			Type
	Problems	No. docs	Avg. words per doc.	Problems	No. docs	Avg. words per doc.	
English	100	200	366	500	452	536	Cross-topic
Spanish	100	500	954	100	1000	946	Cross-topic/genre
Dutch	100	276	354	165	380	360	Cross-genre
Greek	100	100	678	100	500	756	Cross-topic

3.2.1 PAN 2015 Corpus

To conduct experiments with our approach, we used the corpus proposed in the author identification task of PAN 2015 [23]. The dataset consists of four subsets, each set written in different languages: English, Spanish, Dutch and Greek. Subsets have significant differences.

The English subset consists on dialog lines from plays; the Spanish subset consists on opinion articles of online newspapers, magazines, blogs and literary essays; the Dutch subset is formed by essays and reviews; and the Greek subset is formed by opinion articles of categories as politics, health, sports among others. The corpus also has different amount of documents per subset, as detailed in Table 1. In addition, each language consists of a number of problems to solve which are specifically defined below (Section 3.3).

Due to its nature, this dataset focused on problems that require capturing more specific information about the way of writing of the author. For example, suppose we know a person who worked for newspaper writing articles about sports; nevertheless, that person decides to be independent and spends life writing horror novels.

One possible task could be to find out which articles belong to the sport ex-writer among sport articles of different authors. In this case, the vocabulary of the documents can uncover the author; for instance, by his n-grams-usage rate, this kind of task is called **cross-topic**.

On the other hand, another possible task is to discover whether a horror novel was written by the novelist, based on the sport articles which she or he wrote before. Consequently, there is a drastic change in genre and topic of the documents, i.e., the intersection between vocabularies of the documents would be substantially reduced. This is called **cross-genre**.

3.2.2 PAN 2014 Corpus

The PAN 2014 corpus [24], like PAN 2015, consists of documents written in four languages: Dutch, English, Greek, and Spanish. However, PAN 2014 documents are in the same domain, that is, they do not contain mixtures of genre. Each dataset contains one up to five known documents and one unknown document; the challenge is to find out whether the unknown document was written by the same author who wrote the known documents. We show in Table 2 details about the PAN 2014 corpus.

3.3 Method

There are different works that have used LDA as source of features. For example, Pacheco et al. [14] faced author verification challenge of PAN 2015 by proposing a scheme based on the universal background model [19]. By that scheme three feature vectors were created: a vector for author, a vector for each set of known documents and a vector for each unknown document. The three vectors are encoded to produce a new vector. Then, features (including those generated by LDA) were selected by using a random forest which hierarchically determines the importance of each feature.

In the same way, Moreau et al. [11] used a genetic algorithm for selecting the best sources of features (strategies) for author verification of PAN 2015. One of the strategies was LDA, with which two vectors of five topics were generated by

Table 2. Dataset details for the PAN 2014 task of author identification

Language	Training problems			Test problems			Genre
	Problems	No. docs	Avg. words per doc.	Problems	No. docs	Avg. words per doc.	
Dutch	96	268	412.4	96	287	398.1	Essays
Dutch	100	202	112.3	100	202	116.3	Reviews
English	200	729	848.0	200	718	833.2	Essays
English	100	200	3,137.8	200	400	6,104.0	Novels
Greek	100	385	1,404.0	100	368	1,536.6	Articles
Spanish	100	600	1,135.6	100	600	1,121.4	Articles

executing the following steps: first, the authors split documents on character n-grams; next, a set of five topics for each author's text is sought; finally, their algorithm tries to find a set of five topics for each document to be verified (unknown document). With these two vectors the authors obtain a measure that indicates how similar the vectors are, with regard to the style; that measure is used as fitness for a genetic algorithm.

Savoy [20], in a way similar to us, uses distribution of topics calculated by LDA. The challenge is to find out which author, of a set of authors, wrote a specific document. The symmetric Kullback-Leibler distance between the distribution of the unknown document and the distribution of the documents of the authors was calculated.

Then, the author is assigned with regard to the closest distance. This task is called authorship attribution due to the fact that the challenge is to identify the authors of anonymous text. The authorship attribution task was also studied by Seroussi et al. [21]; in their research topic distributions were viewed as vectors of features (patterns), which were directly classified.

The main difference between other works mentioned above is that, in this work we generate features by using LDA under the assumption that one specific author has a specific way of relating words, and thus generate a text.

That assumption led us to analyze which words are linked in a written text, and to which extent the set of linked words are present in the text. With this information we subtract document vectors to create new vectors that form a training set; this process is detailed below.

Specifically, we propose to use Latent Dirichlet Allocation for extracting semantic information from the corpus. As mentioned before, given a collection of texts, LDA is able to find relations between words based on the way they are used in the text. On the other hand, common stylistics approaches use symbol rates in the documents for distinguishing between two documents written by different authors.

As we stated before (Section 2), while texts become shorter, the amount of symbols tend to be not enough to produce effective discriminating features. This worsens when the number of authors is increased.

We infer that writers have different ways of linking words due to the fact that each writer makes use of specific phrases. In addition, they use words at different rates, and thus, written texts keep a background structure.

For example, some author usually may use the phrase "the data gathered in the study suggests that..." in contrast to another author who uses "the data appears to suggest that." We can see that the words "the, in, to, that" can be included in different topics since, unlike LSA [6], LDA can assign the same word to different topics to better handle polysemy.

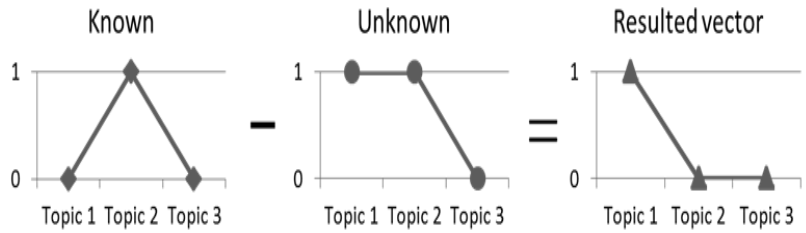


Fig. 2. Example of subtraction between known-document's vector and unknown-document's vector

As a result, to use those words at different rates shall result in different topic distributions for each document.

The task of the dataset used for this study is as follows. For each language or subset of the dataset there is a specific number of problems; for each problem in turn, there are one to five documents considered as *known* and one document considered as *unknown*. These known documents are written by the same author. To solve a specific problem we have to find out whether the unknown document was written by the same author who wrote the known documents.

To represent each problem, all documents in the dataset are processed with LDA. Then, we obtain vectors (with real values—probability of each topic) which represent known and unknown documents. Based on a specific problem, we do a subtraction between each known-document's vector and the unknown-document's vector (let us remember that there is only one unknown document by problem; however there are one to five known documents).

We found that converting the real values to {0, 1} values slightly improved final results, so we binarized them using the arithmetic mean as threshold; 0 represents topic absence and 1 represents the presence of the topic. Therefore, the subtraction between vectors resulted in two possible values: 0 when topics are equal and 1 when topics are different (see Fig.).

4 Results

In the following experiments we used Naïve Bayes for classification. In addition, different *n*-topics for LDA were specified. Therefore, patterns of *n* features were generated for each document. We found that varying the topics number also changed the performance of classification. There is no a priori method for determining how many topics we should choose for incrementing performance; thus, we have to fix an interval until we achieve the best results.

Due to the fact that LDA is a stochastic method, the obtained result for each experiment can be different, so we show the average of 100 experiments for each number of topics tested. In addition, standard deviation is shown. Note that each average of 100 experiments was calculated independently for each measure; that is, product of accuracy and ROC area, on the corresponding tables, will not be proportional to FS (final score) measure.

The FS measure is the product of two values: c@1 [17] and the area under the ROC curve (AUC) [7]. The former is an extension of the accuracy metric and the latter is a measure of classification performance that provides more robust results than accuracy.

Under the assumption that in a real case the test set is unknown, a five-cross validation on the training set was done to find the optimal number of topics, and then we used that parameter to evaluate on the test sets.

4.1 Results of PAN 2015 Classification

We show in Table 3, Table 4, Table 5 and Table 6 a detailed analysis of the classification performance of each language on author identification PAN 2015. As can be seen, we can reach the best performance setting different

Table 3. Performance obtained for English, setting different number of topics

Topics	Binary values				Original values			
	c@1	ROC area	FS	FS-SD	c@1	ROC area	FS	FS-SD
2	0.743	0.733	0.554	0.129	0.495	0.407	0.202	0.019
3	**0.816**	0.853	0.697	0.041	0.672	0.699	0.474	0.090
4	0.800	**0.847**	**0.679**	0.049	0.633	0.637	0.407	0.080
5	0.781	0.835	0.654	0.067	0.631	0.646	0.412	0.085
6	0.759	0.811	0.618	0.075	0.622	0.621	0.389	0.064
7	0.778	0.790	0.586	0.084	0.595	0.598	0.358	0.064
8	0.746	0.800	0.599	0.064	0.591	0.594	0.356	0.083
9	0.735	0.787	0.581	0.075	0.616	0.624	0.388	0.073
10	0.724	0.767	0.559	0.084	0.604	0.603	0.367	0.067

Table 4. Performance obtained for Spanish, setting different number of topics

Topics	Binary values				Original values			
	c@1	ROC area	FS	FS-SD	c@1	ROC area	FS	FS-SD
2	0.633	0.632	0.402	0.037	0.637	0.672	0.428	0.032
3	0.730	0.765	0.561	0.080	0.673	0.702	0.473	0.046
4	**0.750**	**0.783**	**0.589**	0.066	0.678	0.697	0.474	0.053
5	0.740	0.776	0.576	0.071	0.678	0.698	0.475	0.047
6	0.751	0.777	0.586	0.072	0.661	0.694	0.463	0.061
7	0.754	0.777	0.589	0.075	0.664	0.697	0.465	0.068
8	0.726	0.756	0.551	0.072	0.646	0.676	0.439	0.064
9	0.719	0.747	0.540	0.079	0.639	0.667	0.429	0.070
10	0.715	0.738	0.530	0.075	0.635	0.658	0.420	0.066

topics. For instance, the method achieves the best result in the case of English when we set four topics (see Table 3).

For each table, accuracy, ROC area, FS, and FS-SD (final score – standard deviation) are shown. Furthermore, a comparison between the performance obtained with original values and binary values of LDA is shown; this comparison suggests that, at least for the test environment in this study, binary values obtain better performance on classification. It is interesting to note that with vectors formed for few topics (one up to ten), we are able to obtain over 64% accuracy.

As a matter of fact, one may suppose that documents could have been categorized by subject; however, that assumption seems unlikely, due to the fact that, as we showed in Section 0, the used dataset is conformed by a mixture of subjects. We conducted two experiments aiming to find whether two documents written by the same author will be similar based on their distribution of topics.

Fig. 3 shows the sum of all differences by topic in the test dataset for English. As can be seen, the number of differences is high when texts are written by different authors.

In Fig. 4 the same differences are shown for the Spanish language. We classified the dataset without pre-processing and found the following values shown in table 7: Accuracy, F-measure (F), Precision (P), Recall (R). In this table we

Table 5. Performance obtained for Dutch, setting different number of topics

Topics	Binary values				Original values			
	c@1	ROC area	FS	FS-SD	c@1	ROC area	FS	FS-SD
2	0.722	0.721	0.521	0.030	0.682	0.703	0.479	0.023
3	**0.741**	**0.751**	**0.557**	0.030	0.685	0.710	0.487	0.029
4	0.717	0.736	0.530	0.067	0.671	0.707	0.475	0.035
5	0.675	0.730	0.496	0.067	0.636	0.676	0.432	0.050
6	0.668	0.724	0.485	0.049	0.617	0.661	0.409	0.044
7	0.681	0.730	0.498	0.043	0.619	0.651	0.404	0.046
8	0.690	0.730	0.505	0.041	0.626	0.653	0.410	0.045
9	0.696	0.727	0.508	0.050	0.633	0.660	0.419	0.040
10	0.690	0.715	0.494	0.045	0.629	0.657	0.414	0.038

Table 6. Performance obtained for Greek, setting different number of topics

Topics	Binary values				Original values			
	c@1	ROC area	FS	FS-SD	c@1	ROC area	FS	FS-SD
2	0.616	0.616	0.384	0.079	0.594	0.611	0.364	0.054
3	0.643	0.674	0.440	0.090	0.564	0.591	0.336	0.060
4	0.668	0.695	0.469	0.087	0.564	0.592	0.336	0.053
5	0.665	0.696	0.466	0.077	0.561	0.599	0.338	0.059
6	**0.671**	**0.709**	**0.479**	0.086	0.567	0.600	0.344	0.075
7	0.661	0.696	0.468	0.114	0.566	0.601	0.344	0.074
8	0.664	0.703	0.471	0.089	0.561	0.595	0.337	0.066
9	0.654	0.690	0.455	0.084	0.585	0.603	0.356	0.071
10	0.669	0.703	0.476	0.100	0.593	0.612	0.366	0.071

Table 7. Results of each subset classification. Accuracy (Acc), Precision (P), Recall (R) and F-measure (F)

Subset	Type	Acc	P	R	F
English	Cross-topic	85.6	0.864	0.856	0.855
Spanish	Cross-topic/genre	76.0	0.760	0.760	0.760
Dutch	Cross-genre	70.9	0.733	0.709	0.702
Greek	Cross-topic	64.0	0.646	0.640	0.640

show values corresponding to the experiment closest to the average result for each language.

While accuracy is a measure used in many works on author identification and provides a point of comparison with other results, we also opted for showing precision, recall, and F-measure; this allows for a deeper analysis of results: precision shows the percentage of selected texts that are correct, while recall shows the percentage of correct texts that are selected.

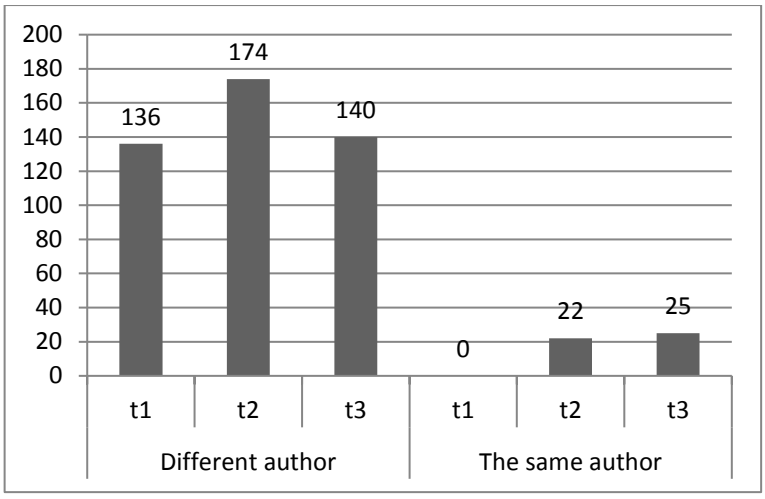


Fig. 3. Topic differences between documents written either by the same or by different author (English subset)

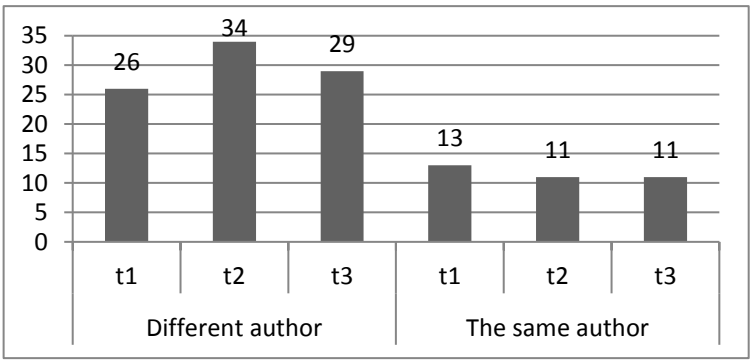


Fig.4. Topic differences between documents written either by the same or by different author (Spanish subset)

Finally, F-measure is the combined measure to assess the P/R trade-off.

According to Table 7, we obtained the best result for the English subset with 85.6% accuracy, even when it has the largest testing set (500 problems) of the corpus. The Spanish subset ranks second with 76.0% accuracy, the Dutch subset reached 70.9% accuracy and finally the Greek subset reached 64.0% accuracy.

We obtained our best and worst performance with the English and Greek subsets respectively; therefore, we cannot infer that the cross-topic setup made the difference in results since actually both subsets are of the same type (see Table 7) and one of them was not affected. Similarly, for both Spanish and Dutch subsets (second and third place respectively), results do not lead to conclude that the genre mixture has some correlation with them. We compared our results with those obtained in author identification task at PAN 2015 [23]. Therefore, we calculated the same as PAN-2015 task's authors, a final score.

We show in Table 8 the best results obtained for each language subset by participants of the PAN-2015 task. In addition, the worst FS scores of PAN-2015's participants are shown. According to those results, our method seems to perform better for both English and Dutch languages.

Additionally, our method is able to outperform FS results with regard to the English subset and has better performance than Bartoli et al. and Bagnall's result with regard to the Dutch subset.

Table 8. Results comparison with other authors. FS=c@1*AUC.

Author	Measure	Subset			
		English	**Spanish**	**Dutch**	**Greek**
Bagnall 2015	c@1	0.757	0.814	0.644	0.851
	AUC	0.811	0.886	0.700	0.882
	FS	0.614	0.721	0.451	**0.750**
Bartoli et al. 2015	c@1	0.559	0.830	0.689	0.657
	AUC	0.578	0.932	0.751	0.698
	FS	0.323	**0.773**	0.518	0.458
Moreau et al. 2015	c@1	0.638	0.755	0.770	0.781
	AUC	0.709	0.853	0.825	0.887
	FS	0.453	0.661	**0.635**	0.693
This work	c@1	0.816	0.750	0.741	0.671
	AUC	0.853	0.783	0.751	0.709
	FS	**0.697**	0.589	0.557	0.479
Worst result PAN'15	FS	0.201	0.095	0.089	0.212

On the other hand, for both Spanish and Greek subsets the proposed method did not show good performance; however, our results are not among the worst FS scores of PAN-2015's participants and ROC curve results show that predictions are acceptable.

We exclude that the bad results may caused by the length of vocabulary since we counted around 1000 and 600 types on Spanish and Greek documents respectively; and just around 300 and 200 types on English and Dutch documents.

Thus, if the performace depended on the size of vocabulary, better results would be obtained on Spanish and Greek documents. We infer that results depend on the documents source.

That is, English articles were taken from plays, containing dialogs; therefore, those documents may keep certain similar latent structure since all of them were written for a specific purpose. Unlike English documents, the other documents have different purposes; for example, Spanish documents are taken from newspapers and even from personal blogs.

4.2 Results of PAN 2014 Classification

The task of PAN 2014 is very similar to the PAN 2015 task explained above; however there is a main difference with regard to the dataset: in PAN 2014 there is no merging of domains. Thus, it contains six subsets, and each subset consists of texts of the same genre (for example, essays, articles, novels).

In this section, we show the results of PAN 2014 classification. Table 9 and Table 10 show detailed results for the subsets where the PAN 2014 baseline was reached, and where it was not surpassed, respectively.

In this case, we outperformed the baseline, set by PAN 2014 challenge, for Dutch reviews, English novels and English essays (three of six subsets, see Table 11).

Table 9. Detailed results of classification on genres where baseline was reached

	Dutch reviews		English novels		English essays	
Topics	**FS**	**FS-DS**	**FS**	**FS-DS**	**FS**	**FS-DS**
10	0.315	0.041	**0.391**	0.062	0.517	0.042
20	0.312	0.058	0.358	0.051	0.533	0.044
30	**0.325**	0.054	0.333	0.030	0.539	0.040
40	0.301	0.048	0.327	0.054	0.535	0.040
50	0.310	0.066	0.296	0.045	0.540	0.043
60	0.283	0.052	0.292	0.041	**0.551**	0.039
70	0.261	0.050	0.279	0.041	0.540	0.032
80	0.256	0.052	0.282	0.043	0.535	0.024
90	0.281	0.046	0.270	0.038	0.536	0.028
100	0.296	0.051	0.268	0.048	0.546	0.034

Table 10. Detailed results of classification on genres where baseline was not reached

	Dutch essays		Greek articles		Spanish articles	
Topics	**FS**	**FS-DS**	**FS**	**FS-DS**	**FS**	**FS-DS**
10	0.661	0.052	**0.436**	0.050	**0.322**	0.047
20	**0.667**	0.045	0.406	0.064	0.310	0.056
30	0.648	0.053	0.360	0.046	0.280	0.050
40	0.628	0.053	0.352	0.060	0.277	0.054
50	0.595	0.037	0.339	0.042	0.280	0.044
60	0.565	0.059	0.329	0.058	0.261	0.046
70	0.548	0.055	0.304	0.052	0.255	0.045
80	0.544	0.064	0.298	0.039	0.259	0.044
90	0.494	0.057	0.288	0.051	0.275	0.061
100	0.475	0.053	0.267	0.045	0.261	0.047

Table 12 shows the best scores achieved by the PAN-2014's participants. Our method only outperforms on English essays classification with regard to the best scores.

In order to find why other languages had a lower performance, we manually analyzed randomly selected Spanish texts where the proposed method yielded bad classification results; we manually looked for cues to find out a consistent characteristic that lead to a missclasification. Nonetheless, texts did not show spelling errors, Spanish slang, or other signs which could help us to differ correctly from incorrectly classified texts. We consider a deeper analysis in a future work is necessary.

5 Conclusions

A common approach to verify authorship is by attempting to model the author's writing style. The assumption is that, by using that approach, it is possible to capture specific features to discriminate one author from others.

That hypothesis is hard to prove; nevertheless it is known that certain amount of data is

Table 11. Performance comparison of this work and PAN-2014 baselines

Genre	c@1	ROC area	FS of this work	FS baseline of PAN 2014
Dutch essays	0.778	0.855	0.667	**0.685**
Dutch reviews	0.560	0.577	**0.325**	0.322
English essays	0.705	0.780	**0.551**	0.288
English novels	0.605	0.644	**0.391**	0.202
Greek articles	0.631	0.686	0.436	**0.452**
Spanish articles	0.555	0.576	0.322	**0.378**

Table12. Score comparison of this work and PAN-2014's participants

Genre	Best score PAN'14	Worst score PAN'14	This work
Dutch essays	0.823	0.307	0.667
Dutch reviews	0.525	0.170	0.325
English essays	0.513	0.270	**0.551**
English novels	0.508	0.225	0.391
Greek articles	0.720	0.281	0.436
Spanish articles	0.698	0.248	0.322

necessary to find more appropriate features leading to a high classification performance.

Finding suitable data is a problem, for instance, when we are talking about forensic field, since there are hardly long texts available and they are in different domains.

We showed in this work how LDA aids to verify authorship when there is limited data, i.e., only from one to five short texts written by a specific author to determine whether an unknown document belongs to the same author.

Basically we used document distributions to capture what we call the author's fingerprint. Then, by subtraction between topic distributions, we found that documents written by different authors tend have different fingerprints compared to those written by the same author.

Due to the fact that LDA is a stochastic method, it is necessary to preserve consistency on subtractions; thus, we have to process all documents at the same time. For instance, if we process the training set and after the test set, the topic *x* from the distribution of the document *z* may not correspond with the topic *x* of the distribution of the document *w*. Therefore, in a real case, to classify a new unknown document, it would be necessary to re-process all documents including the new ones.

This approach allowed us to achieve 74% accuracy on average for all different languages included in PAN 2015, and 63.9% accuracy in average on PAN 2014. In both editions, we were able to surpass the best results reported for the English author identification task. Finding a specific reason of performance decrease for other languages has been left as a future work.

Acknowledgements

We thank Instituto Politécnico Nacional (SIP, COFAA and BEIFI), CONACYT, TEST-TecNM, and Red TTL for their support.

References

1. **Afroz, S., Brennan, M., & Greenstadt, R. (2012).** Detecting hoaxes, frauds, and deception in writing style online. *IEEE Symposium on Security and Privacy*, pp. 461–475. DOI: 10.1109/SP.2012.34.

2. **Bergsma, S., Post, M., & Yarowsky, D. (2012).** Stylometric analysis of scientific articles. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* pp. 327–337, Association for Computational Linguistics.
3. **Blei, D.M., Ng, A.Y., & Jordan, M.I. (2003).** Latent Dirichlet allocation. *Journal of machine Learning research*, Vol. 3, pp. 993–1022.
4. **Bradley, J.K., Kelley, P.G., & Roth, A. (2008).** *Author identification from citations*. Dept. Computing Science, Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep.
5. **Castro, A. & Lindauer, B. (2012).** *Author Identification on Twitter.* Semanticscholar.org.
6. **Dumais, S.T. (2004).** Latent semantic analysis. *Annual review of information science and technology*, Vol. 38, No. 1, pp.188–230.
7. **Fawcett, T. (2006).** An introduction to ROC analysis. *Pattern recognition letters*, Vol. 27, No. 8, pp. 861–874.
8. **Green, R.M. & Sheppard, J.W. (2013).** Comparing Frequency-and Style-Based Features for Twitter Author Identification. *The Twenty-Sixth International FLAIRS Conference*.
9. **Layton, R., Watters, P., & Dazeley, R. (2013).** Local n-grams for Author Identification. *Notebook for PAN at CLEF.*
10. **Madigan, D., Genkin, A., Lewis, D.D., Argamon, S., Fradkin, D., & Ye, L. (2005).** Author identification on the large scale. *Proceedings of the Meeting of the Classification Society of North America,* pp. 13.
11. **Moreau, E., Jayapal, A., Lynch, G., & Vogel, C. (2015).** Author verification: basic stacked generalization applied to predictions from a set of heterogeneous learners. *Working Notes Papers of the CLEF*.
12. **Narayanan, A., Paskov, H., Gong, N.Z., Bethencourt, J., Stefanov, E., Shin, E.C.R., & Song, D. (2012).** On the feasibility of internet-scale author identification. *IEEE Symposium on Security and Privacy*, pp. 300–314.
13. **Nirkhi, S. & Dharaskar, R.V. (2013).** *Comparative study of authorship identification techniques for cyber forensics analysis.* ArXiv preprint 1401.6118.
14. **Pacheco, M.L., Fernández, K., & Porco, A. (2015).** Random Forest with Increased Generalization: A Universal Background Approach for Authorship Verification. *CLEF Working Notes.*
15. **Pateriya, P.K. (2012).** A Study on Author Identification through Stylometry. *International Journal of Computer Science & Communication Networks*, Vol. 2, No. 6, pp. 653–657.
16. **Pavelec, D., Justino, E., & Oliveira, L.S. (2007).** Author identification using stylometric features. *Revista Iberoamericana de Inteligencia Artificial*, Vol. 11, No. 36, pp. 59–66.
17. **Peñas, A. & Rodrigo, A. (2011).** A simple measure to assess non-response. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* Vol. 1, pp. 1415–1424, Association for Computational Linguistics.
18. **Pimas, O., Kröll, M., & Kern, R. (2015).** Know-Center at PAN author identification. *Working Notes Papers of the CLEF.*
19. **Reynolds, D.A., Quatieri, T.F., & Dunn, R.B. (2000).** Speaker verification using adapted Gaussian mixture models. *Digital signal processing*, Vol. 10, No. 3, pp. 19–41.
20. **Savoy, J. (2013).** Authorship attribution based on a probabilistic topic model. *Information Processing & Management*, Vol. 49, No. 1, pp. 341–354.
21. **Seroussi, Y., Zukerman, I., & Bohnert, F. (2014).** Authorship attribution with topic models. *Computational Linguistics*, Vol. 40, No. 2, pp. 269–310.
22. **Stamatatos, E. (2009).** A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, Vol. 60, No. 3, pp. 538–556.
23. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B. (2015).** Overview of the Author Identification Task at PAN. *CLEF Working Notes.*
24. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B. (2014).** Overview of the Author Identification Task at PAN. *CLEF Working Notes*, pp. 877–897.
25. **Verhoeven, B. & Daelemans, W. (2014).** CLiPS Stylometry Investigation (CSI) corpus: A Dutch corpus for the detection of age, gender, personality, sentiment and deception in text. *LREC*, pp. 3081–3085.

Article received on 14/11/2016; accepted on 17/03/2017.
Corresponding author is Ángel Hernández-Castañeda.

Verificación de autoría, clasificación por vecindad

Daniel Castro[1], Yaritza Adame[1], María Pelaez[1], Rafael Muñoz[2]

[1] Desarrollo de Aplicaciones, Tecnología y Sistemas (DATYS), Santiago de Cuba, Cuba

[2] Universidad de Alicante, España, Departamento de Lenguajes y Sistemas Informáticos, España

daniel.castro@cerpamid.co.cu, {yaritza.adame, maria.pelaez}@datys.cu, rafael@dlsi.ua.es

Resumen. El análisis de autoría se ha convertido en una herramienta determinante para el análisis de documentos digitales en las ciencias forenses. Proponemos un método de Verificación de Autoría mediante el análisis de las semejanzas entre documentos de un autor por vecindad, sin estimar umbrales a partir de un entrenamiento, implementamos dos estrategias de representación de los documentos de un autor, una basada en instancias y otra en el cálculo del centroide. Evaluamos colecciones según el número de muestras, los géneros textuales y el tema abordado. Realizamos un análisis del aporte de cada función de comparación y de cada rasgo empleado así como una combinación por mayoría de los votos de cada par función-rasgo empleado en la semejanza entre documentos. Las pruebas se realizaron usando las colecciones públicas de las competencias PAN 2014 y 2015. Los resultados obtenidos son prometedores y nos permiten evaluar nuestra propuesta y la identificación del trabajo futuro a desarrollar.

Palabras clave. Análisis de autoría, verificación de autoría, funciones de comparación, rasgos lingüísticos.

Authorship Verification, Neighborhood-based Classification

Abstract. The Authorship Analysis task has become a determining tool for the analysis of digital documents in forensic sciences. We propose a neighborhood classification method of Authorship Verification analyzing the similarities of a document of unknown authorship between samples documents of one author, without estimating parameters values from a training data, we implemented two strategies of representation of the documents of an author, an instance based and a profile based one. We will evaluate the methods in different data collections according to the number of samples, the textual genres and the topic addressed. We perform an analysis of the contribution of each function of comparison and each feature used to take as final decision a combination by majority of the votes of each function-feature pair used in the similarity between documents. The tests were carried out using the public data sets of the Authorship Verification PAN 2014 and 2015 competitions. The results obtained are promising and allow us to evaluate our proposal and the identification of future work to be developed.

Keywords. Authorship detection, author identification, similarity measures, linguistic features.

1. Análisis de autoría

El mundo actual está matizado por grandes avances tecnológicos que abarcan casi todas las esferas de la sociedad. Un ejemplo de esto, es el desarrollo de las tecnologías de la información, donde desempeña un papel importante internet, el cual rápidamente se ha convertido en la principal forma de intercambio de información, permitiendo la comunicación casi en tiempo real, sin tener en cuenta la distancia. La mayor parte de esta información se encuentra almacenada en forma textual no estructurada y escrita en diferentes idiomas, posibilitando que muchos documentos digitales puedan servir de fuentes de consulta. Esta disponibilidad de información conlleva a que muchas veces las personas para un bienestar propio incurran en abusos, como es el caso de la apropiación del conocimiento. Estos “abusos” de la información constituyen un robo de material intelectual [11,14].

En las ciencias forenses, cada día aumenta la necesidad del empleo de métodos computacionales que humanicen y aligeren el trabajo desarrollado por los peritos. El análisis documental es una de las disciplinas que tradicionalmente presenta, entre sus esferas de investigación, la construcción e identificación de perfiles de autores y, más en detalle, la identificación de autoría de documentos sospechosos. Desde sus inicios y aún en la actualidad, se analizan los rasgos caligráficos en los textos manuscritos.

A partir del auge de la digitalización de la sociedad, se comienzan a presentar investigaciones en las que es necesario identificar los rasgos de autores de documentos digitales, aprovechando para esto el creciente desarrollo de métodos de Inteligencia Artificial (IA), que involucran algoritmos de áreas del Procesamiento del Lenguaje Natural (PLN), la Minería de Textos (MT), el Reconocimiento de Patrones (RP), entre otros.

La comunidad científica, fundamentalmente a partir de la década de los 90, dedica esfuerzos crecientes a la investigación y desarrollo de métodos y algoritmos en la tarea de Análisis de Autoría (AA), profundizando en diferentes subtareas como por ejemplo: el Agrupamiento de muestras de autores, la Detección de Plagio, la Detección y Verificación de Autoría, entre otros [8, 9, 14, 16].

Un impulso importante en las investigaciones y en el desarrollo de algoritmos de AA se logra a partir de la plataforma de experimentación y colaboración PAN[1], principalmente en las ediciones que han tenido lugar desde el 2012 hasta la actualidad [2, 3, 12].

Las principales etapas del desarrollo de un sistema computacional [15] se basan en la siguiente metodología:

- **Formulación** del problema no matemático, es decir, el problema que se quiere resolver.
- **Formalización** del problema, es decir, creación del problema matemático.
- **Selección** de la forma de solución del problema.
- **Solución** del problema matemático.
- **Análisis e interpretación** de los resultados, respecto al problema no matemático original que se quiere resolver.

La mayoría de los trabajos consultados sobre la tarea de Análisis de Autoría, dedican los esfuerzos a las etapas de **Formalización**, **Selección** y **Solución**. Sin embargo, pocos parten, o no lo publican, de un análisis de las características en situaciones reales y la solución dada. Por supuesto, una de las complejidades radica en la obtención y luego publicación de colecciones reales de problemas a resolver. En este sentido cabe destacar nuevamente la plataforma de prueba e intercambio para las investigaciones en este tema, que se brinda en las competencias PAN. Los organizadores se esfuerzan por proporcionar colecciones variadas tanto en género textual y longitud de los textos como en temas abordados; y esto constituye un recurso y una oportunidad de incalculable valor para experimentar y desarrollar aproximaciones.

Los principales esfuerzos en las investigaciones de análisis de autoría se han centrado en las etapas de [4,14]:

1. Selección de rasgos y características de la redacción: captura el estilo y los patrones de redacción que lo identifican y diferencian del resto de los autores. Si solo se cuenta con muestras del autor en análisis (más desafiante y complejo), pues no se obtendrían características que lo diferencien.

2. Representación computacional del estilo de redacción: elemento este de suma importancia, pues impone o canaliza la riqueza de información y rasgos que se almacenan.

3. Método de aprendizaje para la clasificación e identificación de autor, empleando un clasificador binario: es la etapa en la que se toma la decisión sobre la autoría de un documento sospechoso o anónimo, se respondería la pregunta ¿Es o no redactado por el autor?

A modo de resumen, las características principales de los trabajos presentados en las ediciones de PAN del 2012 al 2015, y recogidas en los resúmenes de los organizadores, son:

[1] http://pan.webis.de/, PAN is a series of scientific events and shared tasks on digital text forensics.

1. Rasgos y características de redacción

La mayoría de las aproximaciones utiliza algunos de los rasgos expuestos por [14], donde se plantean agrupados en diferentes capas o niveles de análisis del contenido escrito. Niveles de análisis de Caracteres, Léxicos, Sintácticos, Semánticos y específicos de un Dominio de Aplicación. De estos niveles, se analizan y usan con mayor sistematicidad los rasgos léxicos y de caracteres, debido a la facilidad y disponibilidad de herramientas de Procesamiento de Lenguaje para varios idiomas; de ahí, la generalidad de las soluciones. Además, según los resultados experimentales y las consideraciones de los autores de los trabajos, se han obtenido buenos resultados con estos, sin embargo la incorporación de otros rasgos sintácticos y semánticos no aporta significativos aumentos de precisión.

2. Representaciones computacionales

La propuesta más abordada y utilizada se corresponde con la Bolsa de Palabras (del inglés Bag of Words), de manera general es un n-uplo de rasgos lingüísticos y de contenido [14]. Se han presentado aproximaciones haciendo uso de representaciones con grafos, pero estas son las más escazas [5].

Otro elemento a considerar es el espacio de representación de las muestras de cada autor, en este escenario se han presentado trabajos orientados al análisis de cada una de las instancias (instance based) o documentos o a la construcción de representantes de autores (profile based) [14].

3. Métodos de clasificación y decisión

Los enfoques han sido, de manera general, distribuidos en dos grupos, aquellos considerados perezosos (lazy) o de poco esfuerzo y los del grupo de algoritmos con esfuerzo (eager), siendo los primeros los que menos parámetros ajustan o que basan su análisis considerando únicamente los datos que se ofrecen a clasificar sin entrenamiento, y los segundos los que necesitan de muestras recogidas con anterioridad o entrenamiento para el ajuste de los modelos [2, 3, 14, 16].

Los trabajos presentados utilizan en gran medida métodos de clasificación basados en máquinas de vectores soporte (SVM), árboles de decisión, métodos probabilísticos, estrategias de vecindad y una buena parte emplean métodos de combinación de varios clasificadores homogéneos o heterogéneos.

Los clasificadores basados en instancias responden sorpresivamente bien en dominios de clasificación de documentos [14] y el AA puede considerarse una sub-tarea de la clasificación de documentos, en la que se debe hacer especial énfasis en las etapas de la representación de los documentos y la identificación de los rasgos.

Resaltamos entre otros, los trabajos [8, 9, 12], los que presentan estrategias de clasificación basadas en instancias a partir de la vecindad de los objetos de la clase. Nosotros presentamos una implementación basada en instancias y vecindad con similitudes a las expuestas en los trabajos mencionados, con la diferencia que evaluamos tres variantes de decisión de pertenencia a partir de umbrales calculados por la semejanza intra clase de los documentos. Proponemos estudiar una estrategia basada en instancias, como una basada en prototipo. Una diferencia importante que evaluamos es la combinación de varios clasificadores, utilizando más de una función de comparación, permitiendo que la combinación no sea sensible a la eficacia de una sola función de comparación cuando se reciben documentos de diferente tema, género textual o tamaño.

1.1. Motivación de nuestro trabajo

Un caso de estudio práctico en las ciencias forenses se manifiesta cuando el perito debe evaluar la autoría de un documento desconocido y solo cuenta con muestras certificadas de un autor. Ante esto deberá: responder si fue redactado o no por el consiguiente autor, abstenerse o definir en qué grado pudo ser redactado, entre otros elementos, atendiendo a la semejanza con las muestras conocidas. Este caso de estudio se corresponde con las investigaciones realizadas en la Verificación de Autoría (VA).

En nuestra propuesta evaluaremos los siguientes aspectos:

1. Utilizar un método de clasificación basado en el promedio de semejanza entre objetos de un grupo, sin necesidad de ajustar parámetros para la comparación y decisión de la clasificación de un documento de autoría desconocida. Estudiar la semejanza del documento desconocido con respecto a las muestras del autor y determinar el mecanismo de pertenencia al grupo.
2. Método de clasificación calculando el centroide entre objetos de un grupo. Estudiar la semejanza del documento desconocido con respecto al centroide y a las muestras del autor y determinar mecanismo de pertenencia al grupo.
3. Estudiar la efectividad de la clasificación para las diferentes respuestas que se esperan, siendo estas: Sí redactado, No redactado o Abstención.
4. Evaluar en colecciones escritas en idioma español variando el número de muestras de un autor.
5. Evaluar con colecciones cuando varían la homogeneidad en cuanto a los géneros textuales.
6. Evaluar el impacto del uso de cada función de comparación y rasgo empleado.

2. Métodos implementados

El problema que nos proponemos evaluar se corresponde con una tarea de Verificación de Autoría, donde implementamos un método que determina la autoría de un documento desconocido usando una estrategia Intrínseca (donde solo se cuenta con muestras de un autor), con rasgos de los presentados en la literatura a partir de un análisis de caracteres, léxico y sintaxis; emplearemos una aproximación basada en Instancias y otra basada en Representantes, que no dependa de realizar la construcción de un modelo entrenado o la calibración de umbrales con colecciones de entrenamiento.

Proponemos para esto dos algoritmos apoyados en el cálculo de la semejanza entre pares de objetos, definiendo una función de comparación y estableciendo una representación vectorial de los documentos a partir de un tipo de rasgo escogido [10].

Específicamente, restringimos el dominio de aplicación a un entorno donde solo se cuenta con documentos de muestra de un autor (una clase) y dado un documento desconocido, debemos responder si fue redactado por este autor, no redactado o abstenerse. Nos queda definir bajo qué criterios un objeto nuevo pertenece o no la clase, ya sea usando un algoritmo por promedio o uno por el centroide.

De manera formal definimos los siguientes elementos:

Autor o Grupo: conjunto de documentos redactados por una sola persona (documentos conocidos) y lo representamos con la notación $A = \{D_1, D_2, \dots, D_n\}$, donde los D_i se corresponden con cada uno de los documentos redactados por el autor.

Un documento será representado por un conjunto de Rasgos Lingüísticos extraídos a partir de un procesamiento realizado bien a nivel de caracteres, léxico o sintáctico, utilizando para cada caso herramientas de PLN. En nuestro trabajo vamos a considerar un total de 10 Clases de Rasgos (F), los que se describirán en secciones siguientes, y denotaremos con la siguiente expresión $F = \{F_1, F_2, \dots, F_{10}\}$. Para un F_i, cada documento se representa como $F_i(D) = (x_1(D), x_2(D), \dots, x_n(D))$, donde n denota el total de rasgos en el espacio de representación de los documentos para un F_i, siendo $F_i(D)$ la descripción del documento D y cada $x_i(D)$ el valor del rasgo x_i.

Semejanza entre un par de documentos $\beta(D_i, D_j)$ $i \neq j$: utilizamos tres funciones de comparación, Jaccard, Coseno y Minmax. Estas funciones han sido usadas en diversos trabajos presentados en las competencias PAN [5, 10, 13].

2.1. Arquitectura propuesta

Las colecciones de verificación de autoría ofrecidas en la competencia PAN [2, 3, 12], se estructuran por un conjunto de autores (problemas) y por cada autor se brinda una lista de documentos redactados por este y un documento de autoría desconocida. La tarea consiste en responder si el texto desconocido es redactado por el autor en análisis, no redactado por él o en abstenerse de responder.

Este escenario es similar al problema práctico que nos enfrentamos y queremos resolver, por lo que la base de nuestra propuesta radica en la implementación de un clasificador que sea capaz

de dar una respuesta de la autoría de un documento desconocido partiendo, únicamente, de las muestras conocidas de un autor (Verificación de Autoría Intrínseca VAI).

El objetivo que nos trazamos se corresponde con utilizar una combinación de respuestas de cada clasificador implementado y dar una respuesta final usando un voto por mayoría. Dividimos el total de respuestas en que se dice Sí sobre el total de respuestas. Obtenemos un valor entre 0 y 1, si la respuesta es menor a 0.5 entonces la decisión final es que no fue redactado por el autor, si es igual a 0.5 lo consideramos una abstención y el resto de los casos, o sea, cuando es mayor que 0.5 entonces se considera redactado por el autor.

Cada clasificador debe tomar una decisión a partir de las muestras que se tienen en el instante de la clasificación, sin contar para esto con fases de entrenamiento donde se puedan ajustar parámetros o realizar selección de rasgos o identificación de objetos no representativos.

2.2. Clasificador

En cada clasificador construido definimos 3 etapas necesarias, una primera etapa para la representación de los documentos; una segunda donde se comparan estas representaciones de cada documento y se analiza el grado de semejanza entre cada par de documento; y una tercera etapa en la que se determina si el documento desconocido ha sido redactado por el autor del que se dispone de muestras, utilizando una regla de decisión propia para este clasificador, ver Figura 1.

La etapa de representación es el paso inicial y una de las etapas más importantes en toda tarea de Análisis de Autoría. Para nuestro trabajo se propone emplear diferentes familias de rasgos a partir de analizar el contenido y la redacción de los documentos. Debemos aclarar que en un clasificador, se define un Tipo de rasgo de una de las familias de rasgos del contenido. Se emplean 3 familias de rasgos, basados en Caracteres, Léxico y Gramatical y en cada una diferentes Tipos de rasgos. Para la ejecución de un clasificador se debe contar con los documentos de muestra del autor y un documento de autoría desconocida. Las representaciones escogidas se explican con detalles en el epígrafe “Representaciones de los objetos”.

Luego, procedemos al cálculo de la semejanza entre cada par de documentos, con el propósito de conocer en qué medida son similares dos documentos a partir de la coincidencia de rasgos y a la frecuencia de uso de los mismos. Cobra vital importancia la identificación e implementación de las funciones de comparación entre documentos, aspecto este explicado con detalles en el epígrafe “Cálculo de la semejanza entre objetos, funciones de comparación”.

Proponemos dos estrategias de clasificación para el análisis de la semejanza de los objetos; una orientada a considerar cada documento como una instancia del problema y la segunda a partir de la construcción de un representante o prototipo de las muestras conocidas. Para cada una de estas estrategias definimos 3 reglas de decisión que nos permiten evaluar la pertenencia del documento desconocido como un documento redactado por el autor del que tenemos muestras conocidas. Los aspectos relacionados con la estrategia de clasificación basada en instancias y las reglas de decisión adoptadas en esta, se exponen en el epígrafe “Regla de decisión utilizando el promedio de semejanza entre objetos de una clase”; y en el epígrafe: “Regla de decisión utilizando la semejanza con centroide de una clase” se exponen detalles de la estrategia basada en prototipos.

El clasificador debe dar como respuesta: documento desconocido es redactado por el autor de las muestras conocidas (valor mayor a 0.5), se abstiene en determinar si fue redactado por este autor (valor 0.5) o determina que el documento de autoría desconocida no fue redactado por el autor de las muestras (valor menor de 0.5). Estos datos numéricos son los valores que permiten obtener un voto por mayoría en la combinación final de los clasificadores.

2.3. Representaciones de los objetos

Los rasgos lingüísticos son el núcleo de la tarea de análisis de autoría (independientemente de la subtarea de las mencionadas en la que se trabaje), ellos permiten codificar los documentos con algún modelo matemático, siendo tradicionalmente el más usado el modelo de bolsa de palabras (Bag of

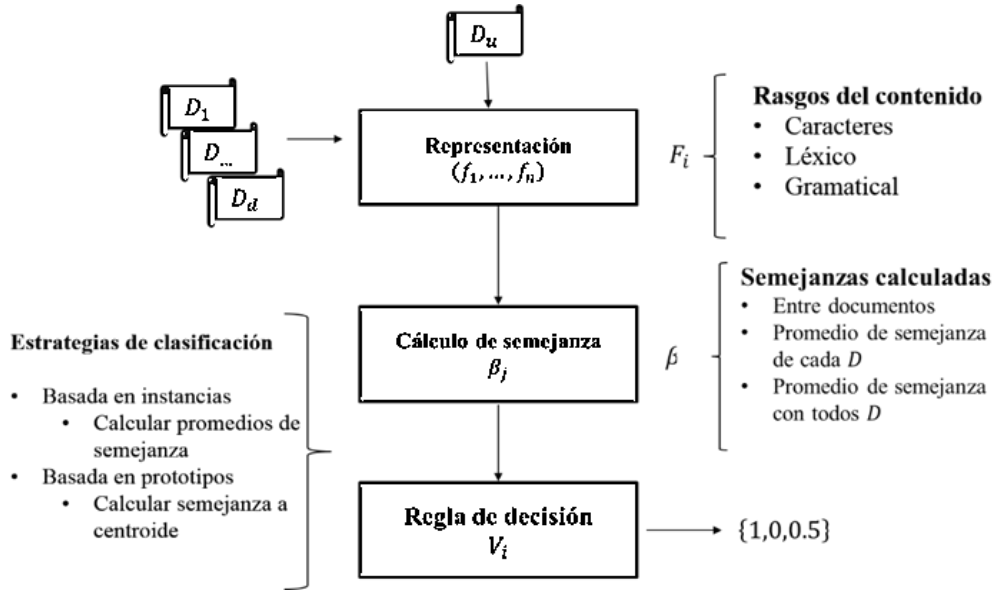


Fig. 1. Etapas del clasificador de Verificación de Autoría Intrínseco

Word, BoW), empleando como representación un n-uplo de rasgos. El propósito radica en intentar identificar un estilo propio de redacción para cada autor que lo diferencie del resto, en nuestro enfoque solo que lo caracterice a él, puesto que no dispondremos de muestras de otros autores.

Existe una gran cantidad de rasgos que han sido tomados en cuenta para la tarea de análisis de autoría por los investigadores, en la generalidad o mayoría, se usa una distribución o identificación de rasgos por capas lingüísticas (podemos llamarlos además, rasgos obtenidos a partir del contenido de la redacción).

Para nuestra propuesta escogimos 10 Tipos de Rasgos lingüísticos de los reportados en la literatura [2, 10, 14] agrupados en las siguientes Familias o Capas de rasgos de análisis lingüístico.

– Capa de caracteres:
  — **N-gramas de caracteres**: se obtienen como rasgos, todas las secuencias de n caracteres, sin eliminación de elementos en el texto. Para los experimentos se probó con varios valores de N y los mejores resultados se aprecian para N 3 y 4. Se construye una BoW con N = 3 y otra con N = 4. Para mostrar los resultados en los experimentos, relacionamos 3-grama de caracteres con (*F2*) y 4-grama de caracteres con (*F3*).
  — **N-gramas de Prefijos de tamaño k**: se construye una representación BoW tomando solo las N secuencias de caracteres de tamaño k a inicio de palabras. Para mostrar los resultados en los experimentos, relacionamos 2-grama-prefijo-tamaño-2 (*F5*).
  — **N-gramas de Sufijos de tamaño n**: esta es similar a la representación anterior, pero tomando las N secuencias de caracteres de tamaño k al final de cada palabra. Para mostrar los resultados en los experimentos, relacionamos 2-grama-sufijo-tamaño-2 (*F6*).

Los rasgos de esta capa son sencillos de calcular y nos permiten emplear herramientas no dependientes de un idioma. Para su cálculo se utilizan herramientas sencillas como los segmentadores de texto, que son usados para buscar patrones de redacción a través del uso de sufijos, prefijos, signos de puntuación, secuencias consecutivas de caracteres, entre otros.

– Capa léxica:
  — **N-gramas de palabras**: secuencias de N términos consecutivos luego de segmentado un texto. Construimos dos representaciones, una

con N = 1 y otra con N = 3. Para mostrar los resultados en los experimentos, relacionamos 1-grama de palabras (*F1*) y 3-grama de palabras (*F4*). Se toma N con 1 y 3 luego de probar con N de 1 a 5 y obtener los mejores resultados con 1 y 3.

Al igual que los rasgos de la capa de caracteres, los rasgos léxicos se pueden obtener empleando herramientas sencillas como los segmentadores de texto y son usados para buscar patrones de redacción a través del uso de palabras, secuencias consecutivas de palabras, entre otros.

- Capa gramatical:
  - **N-gramas de lemas**: secuencias de N lemas consecutivos luego de lematizado un texto. Construimos dos representaciones, una con N = 1 y otra con N = 3. Para mostrar los resultados en los experimentos, relacionamos 1-grama de lemas (*F7*) y 3-grama de lemas (*F9*).
  - **N-gramas de Etiquetas Gramaticales (PoS)**: secuencias de N etiquetas gramaticales consecutivas luego de etiquetado un texto. Construimos dos representaciones, una con N = 1 y otra con N = 3. Para mostrar los resultados en los experimentos, relacionamos 1-grama de PoS (*F8*) y 3-grama de PoS (*F10*).

Los rasgos de esta capa son un poco más complejos, dependiendo de herramientas de etiquetado y lematización de textos, son dependientes del idioma, requieren más tiempo para ser calculados y son usados para determinar patrones de redacción a través del uso de las categorías gramaticales y lematización de las palabras.

Para ilustrar el proceso de representación interna de los documentos usando cada uno de los rasgos lingüísticos supongamos que disponemos de un documento.

D1: *Me gusta pescar y navegar en las profundas aguas del mar Caribe.*

F1: *{(me, 1); (gusta, 1); (pescar, 1); (y, 1); (navegar, 1); (en, 1); (las, 1); (profundas, 1); (aguas, 1); (del, 1); (mar, 1); (caribe, 1); (.,1)}.*

F7: *{(me, 1); (gustar, 1); (pescar, 1); (y, 1); (navegar, 1); (en, 1); (el, 1); (profundo, 1); (agua, 1); (del, 1); (mar, 1); (caribe, 1); (.,1)}.*

F8: *{(PP1CS00, 1); (VMIP3S0, 1); (VMN0000, 1); (CC 1); (VMN0000, 1); (SP, 1); (DA0FP0, 1); (AQ0FP00, 1); (NCFP000, 1); (SP, 1); (NCCS000, 1); (NP00000, 1); (FP,1)}.*

En cada clasificador se determina como un parámetro de configuración, el Tipo de Rasgo con el que se representarán los documentos. Se construye un n-uplo de rasgos (términos) binario o pesado por la Frecuencia de su uso en el documento en análisis (term frequency, TF), dependiendo de la función de comparación que se empleará para el cálculo de la semejanza entre los objetos.

2.4. Cálculo de la semejanza entre objetos, funciones de comparación

Debido a que en nuestro problema práctico podemos encontrar documentos en las muestras de un autor con características muy variables, como, el tamaño, el género literario, la temática que abordan, entre otras. Además de la necesidad de un método general para ser usado en cualquier entorno de aplicación en la tarea de análisis de autoría decidimos escoger 3 funciones de comparación reportadas en la literatura, con el objetivo de tener un marco flexible capaz de adaptarse a cualquier entorno de aplicación.

Las funciones de comparación pueden dividirse en funciones de semejanzas y funciones de distancia. Las primeras evalúan la similitud entre dos objetos otorgando un valor cercano a 1 mientras más semejantes sean; en contraposición las de distancia determinan que dos objetos son semejantes a medida que el cálculo se acerca a 0. Para el desarrollo de los experimentos se implementaron funciones de semejanza para n-uplos binarios y para n-uplos pesados y una función de distancia.

El índice de Jaccard (1), mide la proporción existente entre la cantidad de elementos de la intersección de dos conjuntos sobre el total de elementos de la unión.

Siempre toma valores entre 0 y 1, correspondiente este último a la igualdad total entre ambos conjuntos. En informática se utiliza para medir la distancia entre vectores principalmente definidos sobre un espacio

vectorial booleano (las componentes del vector sólo pueden ser 0 o 1):

$$\frac{A \cap B}{|A \cup B|}. \quad (1)$$

La medida de similitud coseno (2) es usada para medir el valor del coseno ángulo comprendido entre dos vectores en un espacio, mientras menor sea el ángulo mayor es el coseno y en consecuencia mayor es la similitud entre los dos vectores. Es una medida ampliamente usada en la literatura. En comparación con el índice de Jaccard es una medida más exigente ya que no mide solamente la presencia de una determinada característica sino el nivel de importancia de esa característica en ambos vectores:

$$\frac{\sum_{i=0}^{n} x_i * y_i}{\sqrt{\sum_{i=1}^{|X|}(x_i)^2} + \sqrt{\sum_{i=1}^{|Y|}(y_i)^2}}. \quad (2)$$

En las funciones de distancia mientras más pequeño es el valor más cercano están los dos vectores y viceversa, mientras mayores sean los valores más alejados se encuentran. Las funciones de distancia pueden ser fácilmente convertidas en funciones de semejanza mediante la resta del valor 1 con el valor de la función de distancia.

A pesar de que en la literatura, la distancia euclidea es una de las más usadas, no la empleamos debido a que obtiene valores semejantes a la función coseno cuando los n-uplos están normalizados como en nuestro problema [1].

La distancia MinMax (3) determina la proporción existente entre los valores mínimos y los valores máximos pero tiene el inconveniente que solo toman en cuenta aquellas características que se encuentran en ambos documentos; ha sido utilizada en el algoritmo de [13], el cual se ubicó entre los primeros trabajos de la edición PAN 2013:

$$\frac{\sum_{i=1}^{r} \mathrm{Min}(x_i, y_i)}{\sum_{i=1}^{r} \mathrm{Max}(x_i, y_i)}. \quad (3)$$

2.5. Regla de decisión utilizando el promedio de semejanza entre objetos de una clase

La etapa final del clasificador se corresponde con las reglas de decisión que nos permiten obtener la respuesta, en nuestra tarea responder si el documento desconocido fue redactado por el autor de las muestras (responder 1), abstenerse de dar una respuesta (responder 0.5) o determinar que no fue redactado por él (responder 0). A continuación, expondremos los detalles de la clasificación basada en instancias y las tres reglas de decisión propuestas a partir de esta estrategia. A modo de resumen, se puede observar la figura 2.

Dados dos grupos de documentos A_1 y A_2, donde A_1 contiene el conjunto de muestras de un autor y A_2 el documento desconocido D_u, construimos un grupo nuevo $A = A_1 \cup A_2$ formado por la unión de todos los documentos de A_1 y A_2 y calculamos el promedio de semejanza del grupo A (PS_A) y el promedio de semejanza de cada documento D_i con el resto del grupo A ($PS_{D_i}^A$):

$$PS_{D_i}^A = \frac{\sum_{j=1, j\neq i}^{|A|} \beta(D_i, D_j)}{|A| - 1}, \quad (4)$$

$$PS_A = \frac{\sum_{i=1}^{|A|} PS_{D_i}^A}{|A|}. \quad (5)$$

Evaluamos las siguientes tres estrategias:

1. Se calcula el promedio de semejanza de todos los objetos del grupo, considerando al desconocido; si el promedio del documento desconocido es mayor que la media de los promedios del grupo, entonces, es bien semejante a la mayoría de los objetos conocidos y se considera redactado por este autor. Ver figura 2. Esta aproximación debe permitir que si el documento no fue redactado por el autor, entonces, aunque la semejanza del desconocido afecte a las semejanzas del resto de los objetos del grupo, este tendrá el menor promedio de semejanza, o su promedio no será mayor que la media del grupo. El punto débil se debe presentar en que se equivoque en aquellos objetos que se deben considerar redactados por el

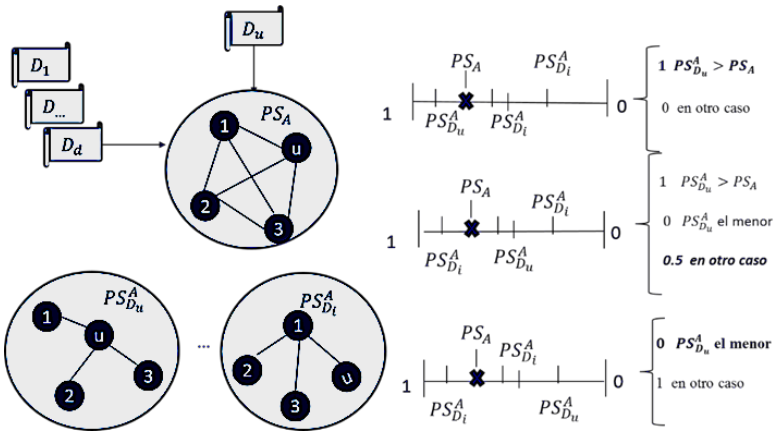


Fig. 2. Clasificador basado en instancias, calculando el promedio de semejanzas entre todos los documentos de muestra y el desconocido. Reglas de decisión a partir del promedio de semejanza del desconocido con las muestras del autor

autor y que su semejanza de promedio no es mayor que la media. Puede ser la menor semejanza promedio o no ser la menor, pero tampoco mayor que la promedio. Estos casos se darían como errores.

2. Una segunda estrategia implementada, considera como respuesta de redactado por el autor, los casos en que el promedio de semejanza del desconocido, sea mayor que el promedio del grupo, la respuesta de no redactado sería cuando el promedio de semejanza sea el menor y daría una respuesta de Abstención (Abs) si no es el menor promedio, pero no supera el promedio del grupo.

   En este caso, se busca que aquellos documentos desconocidos que son redactados por el autor y que no son semejantes a la mayoría, pero más semejantes que el menos semejante, no los dé como una respuesta de no redactado y evaluar en qué grado, los documentos no redactados tendrían el menor promedio de semejanza o se incluirían entre estos de Abs.

   Si se consideran entre los Abs es una señal de que la representación no está diferenciando los no redactados por algunas muestras. Este sería un indicio de que se pudiera trabajar en evaluar los rasgos obtenidos por cada tipo de rasgo, emplear selección de rasgos y evaluar técnicas de análisis de objetos no representativos.

3. La tercera idea reside en considerar como redactado por el autor a la muestra desconocida que no tenga el menor promedio de semejanza, por el criterio de que el documento no redactado debe tener el menor promedio de semejanza, lo que no quita que documentos si redactados tengan el menor promedio de semejanza.

2.6. Regla de decisión utilizando la semejanza con centroide de una clase

La segunda propuesta de clasificación empleada está basada en la construcción de un representante o prototipo de un grupo de documentos de muestra. La idea es que este representante permitiría agrupar todas las características presentes en las redacciones de los documentos de muestra.

La decisión de la pertenencia de un documento desconocido a las muestras conocidas parte entonces de evaluar la semejanza de este documento desconocido con el representante construido. A continuación, se explican los detalles de la clasificación basada en prototipo y las tres reglas de decisión para obtener la respuesta de un clasificador. Ver figura 3.

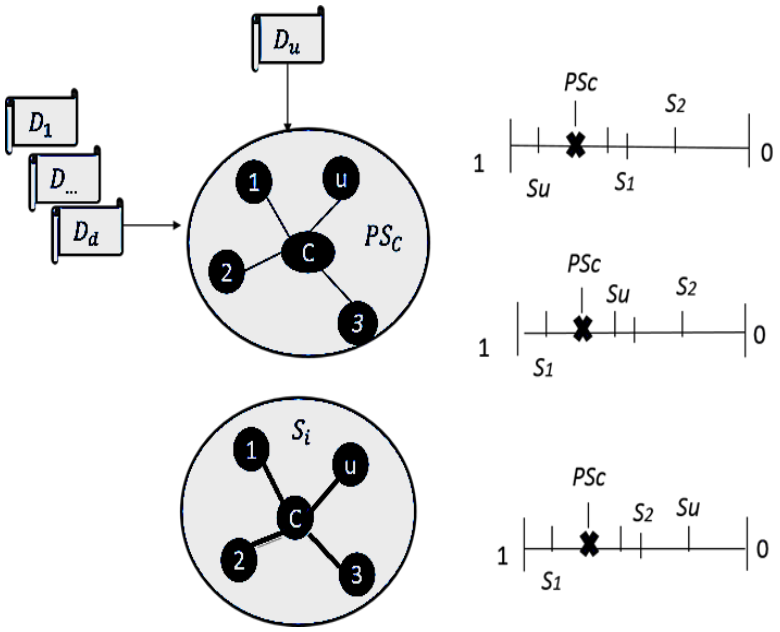


Fig. 3. Clasificador basado en prototipo, calculando la semejanza de los documentos con respecto al prototipo. Reglas de decisión a partir de la semejanza del desconocido con el representante del autor

Dados dos grupos de documentos A_1 y A_2, donde A_1 contiene el conjunto de muestras de un autor y A_2 el documento desconocido D_u, construimos un grupo nuevo $A = A_1 \cup A_2$ formado por la unión de todos los documentos de A_1 y A_2 y calculamos el centroide suma [11] del grupo A (C_A) y la semejanza de cada documento D_i del grupo A con C_A (6). Por último obtenemos el promedio de las semejanzas con el centroide PS_c(7):

$$S_{D_i}^{C} = \beta(D_i, C), \tag{6}$$

$$PS_c = \frac{\sum_{i=1}^{|A|} S_{D_i}^{C}}{|A|}. \tag{7}$$

Evaluamos las siguientes tres estrategias:

1. Se construye un centroide del grupo integrado por los documentos redactados y el desconocido, luego se calcula la semejanza de cada documento con el centroide. Para la comparación calculamos el promedio de las semejanzas con el centroide. Si la semejanza calculada del desconocido con el centroide es mayor que este promedio de semejanzas calculado, entonces se considera redactado por el autor. Se responde No redactado en caso contrario.
2. Si la semejanza del desconocido con el centroide es mayor que el promedio de las semejanzas al centroide, entonces es redactado. Si no es superior al promedio pero no es la menor semejanza entre el resto, entonces se considera una abstención. Es No redactado cuando presenta la menor semejanza con el promedio.
3. Si la semejanza del desconocido al centroide no es la menor semejanza, entonces consideramos al documento desconocido como redactado por el autor. No redactado en caso contrario.

3. Resultados experimentales

Para los experimentos se consideraron las colecciones de la competencia PAN de los años 2014 y 2015 para el español. En la edición del 2014 [3], la colección de autores para español presentaba como característica que las muestras

Tabla 1. Estructura y distribución de documentos y problemas de verificación de autoría en PAN 2014

	Idioma	Género	#Autores	#Docs	Promedio de docs conocidos por autor	Promedio de palabras por docs
Entrenamiento	Holandés	Ensayos	96	268	1.8	412.4
	Holandés	Comentarios	100	202	1.0	112.3
	Inglés	Ensayos	200	729	2.6	848.0
	Inglés	Novelas	100	200	1.0	3137.8
	Griego	Artículos	100	385	2.9	1404.0
	Español	Artículos	100	600	5.0	1135.6
	Total		696	2,384	2.4	1091.0
Prueba	Holandés	Ensayos	96	287	2.0	398.1
	Holandés	Comentarios	100	202	1.0	116.3
	Inglés	Ensayos	200	718	2.6	833.2
	Inglés	Novelas	200	400	1.0	6104.0
	Griego	Artículos	100	368	2.7	1536.6
	Español	Artículos	100	600	5.0	1121.4
	Total		796	2,575	2.2	1714.9

Tabla 2. Estructura y distribución de documentos y problemas de verificación de autoría en PAN 2015

	Idioma	Tipo	# Autores	#Docs	Promedio de docs Conocidos por problemas	Promedio de palabras por docs
Entrenamiento	Holandés	multi-género	100	276	1.76	354
	Inglés	multi-tópico	100	200	1.0	366
	Griego	multi-tópico	100	393	2.93	678
	Español	mixto	100	500	4.0	954
	Total		400	1369	2.42	588
Prueba	Holandés	multi-género	165	452	1.74	360
	Inglés	multi-tópico	500	1000	1.00	536
	Griego	multi-tópico	100	380	2.80	756
	Español	mixto	100	500	4.00	946
	Total		865	2332	2.3	

de cada autor eran homogéneas en cuanto a género textual y tema abordado, a diferencia de la colección presentada en 2015 [2], donde las muestras de cada autor podían ser de diferente género textual y tema.

En la tabla 1 y 2 se presentan las características de las colecciones que usaremos para los experimentos descritos en las próximas secciones. Se aprecia que la colección con mayor número de muestras por autores es la de español, además de contar con un número alto de palabras por documentos. Estos dos elementos son esenciales para los resultados que esperamos obtener en los experimentos que se describirán en las siguientes secciones.

La medida de evaluación que empleamos se corresponde con la medida de *accuracy* (8) propuesta y usada en las ediciones de las competencias PAN [3]

$$c@1 = \frac{n_c + \left(\frac{n_u * n_c}{n}\right)}{n}, \qquad (8)$$

donde n_c es la cantidad de respuestas correctas, n_u la cantidad de abstenciones y n el total de problemas a responder.

3.1. Centroide vs Promedio

Evaluamos inicialmente la efectividad de cada una de las estrategias de clasificación que proponemos, basada en instancias y en el centroide y en las siguientes secciones de experimentos, solo utilizaremos el enfoque que brinde mejores resultados.

Mostramos los resultados de las ejecuciones de las estrategias de centroide y promedio de semejanza sobre 4 colecciones de PAN que usamos, así como del empleo de las combinaciones de todos los pares funciones de comparación y rasgos. En la tabla 3 los valores obtenidos para la variante 1 de comparación. En la tabla se resaltan valores en los que los resultados son superiores con diferencia entre una estrategia de promedio de las instancias y la de centroide.

A modo de resumen se aprecia que la estrategia usando el promedio entre las instancias obtiene los mejores valores de *accuracy* que la estrategia de calcular el centroide. Con respecto a los idiomas se aprecian los mejores resultados en las colecciones de español y griego en PAN 2014 y español para PAN 2015, correspondiendo estas a colecciones con mayor cantidad de muestras conocidas por autores y textos con una longitud considerable.

3.2. Evaluando respuestas Sí, No y abstenciones

A continuación, vamos a ilustrar algunos resultados de las tres variantes propuestas de umbrales, especificando en el número de respuestas de Sí, No y las Abstenciones. Los resultados mostrados se corresponden a ejecuciones realizadas con la colección de textos en español de las datas de PAN 2014 test2.

En los resultados de la sección anterior no podemos apreciar donde se equivoca más la estrategia de clasificación, si en determinar los documentos que sí son redactados por el autor o en responder qué documento no fue redactado por el autor. Debemos señalar que en las colecciones de la competencia se brindan dos clases de problemas: clases de autores para los que el documento desconocido que se debe evaluar no fue redactado por él, y en este la respuesta positiva es decir No, y problemas en los que el documento desconocido si fue redactado por el autor y la respuesta positiva es decir Sí.

Los resultados mostrados en la tabla 4, son obtenidos solo considerando que el promedio de semejanza del desconocido supere la media de la semejanza entre todos. **Correcto** representa el total de respuestas positivas ya sea que se responda Sí cuando es Sí y No cuando no fue redactado. **Incorrecto** representa el total de respuestas negativas, o sea, dijo Sí cuando no fue redactado y viceversa. **Abstenciones** cuando la respuesta es una abstención, con esta estrategia, solo se da abstención en la combinación final, cuando entre los 10 tipos de rasgos, se alcancen 5 respuestas Sí y 5 No, pero para cada par rasgo-función siempre será 0. **Correcto Sí** cantidad de respuestas positivas en las que se debía responder Sí. **Correcto No** cantidad de respuestas positivas cuando se debía responder que No. **Accuracy** representa el valor de la medida *accuracy*, tal como se propone en las evaluaciones de las competencias PAN 2014 y 2015.

Tabla 3. Comparación de los resultados obtenidos con variante 1 entre promedio y centroide

Año	Colección	Idioma	Género	Promedio	Centroide
2014	Entrenamiento	Español	artículos	**0,84**	0,71
		Griego	artículos	0,53	0,46
		Inglés	novelas	**0,57**	0,4
		Inglés	ensayos	0,55	0,56
		Holandés	comentarios	0,49	0,5
		Holandés	ensayos	0,5	0,49
	Prueba	Español	artículos	**0,74**	0,64
		Griego	artículos	0,62	0,56
		Inglés	novelas	0,46	0,48
		Inglés	ensayos	0,6	0,54
		Holandés	comentarios	0,49	0,51
		Holandés	ensayos	0,58	0,45
2015	Entrenamiento	Español	mixto	**0,77**	0,69
		Griego	multi-tópico	0,58	0,55
		Inglés	multi-tópico	0,5	0,48
		Holandés	multi-género	0,57	0,54
	Prueba	Español	mixto	0,66	0,52
		Griego	multi-tópico	0,57	0,54
		Inglés	multi-tópico	0,5	0,5
		Holandés	multi-género	0,5	0,5

Tabla 4. Estrategia 1 empleando medida de comparación jaccard

Rasgos	Correcto	Incorrecto	Abstenciones	Correcto Sí	Correcto No	Accuracy
F1	67	33	0	22	45	0.67
F2	67	33	0	26	41	0.67
F3	69	31	0	27	42	0.69
F4	63	37	0	17	46	0.63
F5	59	41	0	18	41	0.59
F6	64	36	0	25	39	0.64
F7	68	32	0	23	45	0.68
F8	60	40	0	23	37	0.6
F9	67	33	0	23	44	0.67
F10	63	37	0	25	38	0.63
Combinación	65	31	4			0.67

Tabla 5. Estrategia 2 empleando medida de comparación jaccard

Rasgos	Correcto	Incorrecto	Abstenciones	Correcto Sí	Correcto No	Accuracy
F1	22	5	73	22	0	0.38
F2	26	9	65	26	0	0.42
F3	27	8	65	27	0	0.44
F4	17	4	79	17	0	0.30
F5	18	9	73	18	0	0.31
F6	25	11	64	25	0	0.41
F7	23	5	72	23	0	0.39
F8	23	13	64	23	0	0.37
F9	23	6	71	23	0	0.39
F10	25	12	63	25	0	0.40
Combinación	21	6	73			0.36

En la data que estamos mostrando del español, se cuenta con un total de 100 problemas de verificación y en cada problema un total de 5 muestras de documentos redactados por el autor y un documento desconocido.

Para la evaluación se conoce si el documento desconocido fue redactado o no por este autor. Se puede responder Sí redactado, No redactado o Abstenerse. Se presentan 50 problemas en los que la respuesta debe ser Sí y 50 en los que la respuesta debe ser No.

Como la restricción de esta estrategia es que solo se responda Sí cuando se supere la media de semejanza del grupo, se busca que el documento desconocido sea bien semejante a la mayoría de las muestras conocidas, según esta idea, debe responder positivo a todas las muestras desconocidas que no fueron redactadas, o sea decir No, y evaluar en qué grado es capaz de responder correctamente Sí, ya que para las respuestas de Sí es una restricción fuerte que supere la media. Se aprecia, en sentido general, que para la mayoría de las respuestas No, es positiva la respuesta y que en casi la mitad de las respuestas Sí, los documentos pasaban la frontera de la media.

Las principales respuestas negativas están en los Sí que se respondió que No por no superar la media y bastante interesante es ver cómo algunos documentos desconocidos en los que se debe responder No, superaron la media de su grupo de muestras de autor que en principio sería más semejante a la mayoría de las conocidas.

La estrategia en este caso (tabla 5) es responder que Sí, si el promedio del desconocido es mayor que la media del grupo, decir abstención (Abstención) si no supera la media del grupo, pero no es el menor promedio de semejanza del grupo y se responde que No cuando el promedio de semejanza del desconocido es el menor.

En este experimento, podemos ver cómo la mayoría de las respuestas son de abstención, casi todas en las que debía responder que No y el resto de las que debía responder que Sí, esto identifica que casi todas las respuestas en que debe decir Sí, o están por encima de la media del grupo o por debajo de la media, pero sin ser el menor promedio de semejanza, y que es bastante fácil que, un objeto no redactado por el autor sea, incluso, más semejante a sus muestras que algunas de las conocidas, por lo que es bastante difícil que tengan un promedio de semejanza mayor a la media, pero no fueron las muestras con menor promedio.

Si se considera la abstención como una respuesta más favorable a equivocarse, entonces el resultado es positivo, porque la cantidad de respuestas **Correcto** es para la mayoría de los

Tabla 6. Estrategia 3 empleando función de comparación Jaccard

Rasgos	Correcto	Incorrecto	Abstenciones	Correcto Sí	Correcto No	Accuracy
F1	50	50	0	50	0	0.5
F2	50	50	0	50	0	0.5
F3	50	50	0	50	0	0.5
F4	50	50	0	50	0	0.5
F5	50	50	0	50	0	0.5
F6	50	50	0	50	0	0.5
F7	50	50	0	50	0	0.5
F8	50	50	0	50	0	0.5
F9	50	50	0	50	0	0.5
F10	50	50	0	50	0	0.5
Combinación	50	50	0			0.5

rasgos mucho más alta que los errores **Incorrecto.**

Esta estrategia (tabla 6), determina como respuesta Sí, cuando el promedio de semejanza del desconocido no es el menor. Se responde No en caso contrario. Estamos tomando como frontera de decisión el objeto con menor promedio de semejanza.

Se aprecia que para todos los documentos desconocidos en los que se debe decir que Sí, estos nunca tienen el menor promedio de semejanza, y entonces el **Correcto Si** es igual al total de **Correcto**, pero se equivocó en los que debe responder No, porque estos tampoco son los objetos de menor promedio de semejanza, contrario a lo que se debía esperar.

Esto ilustra que podemos estar en presencia de situaciones en las que tenemos documentos (outliers) en los bordes de la distribución en el espacio de característica de los rasgos, probablemente debido a la cantidad de rasgos que pueden ser redundantes con respecto a los documentos desconocidos.

A modo de resumen para situaciones prácticas forenses sería conveniente utilizar la segunda estrategia de decisión, puesto que se equivoca menos que las otras estrategias, aunque obtiene un volumen alto de abstenciones.

Consideramos que introduciendo estrategias para determinar los documentos menos representativos de las muestras, se pudiera discriminar mejor con la tercera estrategia (tabla 6) y que en nuestro trabajo no usamos métodos de selección de rasgos que pudieran permitir una diferencia mayor, entre las muestras conocidas y el documento desconocido cuando este no pertenece al autor en análisis.

3.3. Comparación con trabajos presentados en la edición de PAN 2014

Queremos a continuación comparar los resultados obtenidos con las propuestas presentadas en la edición del PAN 2014. Para esto se presentan problemas en los que solo se cuenta con una muestra de documento conocido para el autor y esto no permitiría realizar la comparación, porque se necesitan al menos dos documentos de muestra para el cálculo de los promedios o del centroide.

Ante esta situación decidimos dividir el documento a la mitad y generar dos documentos. Esta es una idea muy simple y burda y reconocemos que podemos utilizar estrategias de segmentado más elaboradas, pero nos quedará para trabajo futuro. La dificultad mayor se concentra cuando tenemos una sola muestra y esta es corta.

Tabla 7. Resultados alcanzados de los participantes en la edición PAN 2014 y nuestro enfoque

Posición	Holandés-ensayo		Holandés-comentario		Griego-artículos	
	Trabajo	**c@1**	**Trabajo**	**c@1**	**Trabajo**	**c@1**
1	Frery et al.	0,9	Satyam et al.	0,69	Khonji & Iraqi	0,81
2	Mayor et al.	0,88	Khonji & Iraqi	0,65	Mayor et al.	0,75
3	Castillo et al.	0,86	Moreau et al.	0,59	Castillo et al.	0,73
4	Khonji & Iraqi	0,84	Zamani et al.	0,59	Moreau et al.	0,7
5	Jankowska et al.	0,84	Frery et al.	0,57	Jankowska et al.	0,68
6	Moreau et al.	0,83	Jankowska et al.	0,56	Zamani et al.	0,66
7	BASELINE	0,79	Halvani & Steinebach	0,55	**Nuestro+**	0,66
8	Satyam et al.	0,75	BASELINE	0,53	Frery et al.	0,642
9	**Nuestro+**	0,73	Mayor et al.	0,525	BASELINE	0,64
10	Vartapetiance & Gillam	0,71	Layton	0,52	**Nuestro**	0,62
12	Modaresi & Gross	0,63	Modaresi & Gross	0,5	Halvani & Steinebach	0,6
13	Halvani & Steinebach	0,617	**Nuestro**	0,49	Satyam et al.	0,6
14	Harvey	0,615	Harvey	0,48	Modaresi & Gross	0,54
15	**Nuestro**	0,58	Castillo et al.	0,37	Vartapetiance & Gillam	0,53
16	Layton	0,56			Harvey	0

Esta situación se refleja fundamentalmente en las colecciones de documentos del holandés. En la tabla 7 incluimos los resultados para tres colecciones y el resto en la tabla 8.

Nuestros resultados se observan con el nombre **Nuestro** y además adicionamos un **Nuestro+** que se corresponde con evaluar problemas en los que se tiene más de una muestra conocida.

Observamos que los resultados más bajos se alcanzan en las colecciones de novela en inglés, a partir de que todos los problemas de esta colección contienen un solo documento conocido a pesar de ser documentos extensos, y para el holandés en comentarios, donde los textos son bien cortos y una muestra conocida por autor.

Podemos apreciar que en las colecciones donde eliminamos el análisis de los problemas de una sola muestra, se mejoran los valores de *accuracy* y se alcanzan los mayores valores en la colección de español donde se presenta un mayor número de documentos de muestra por autor.

3.4. Verificación de autoría para todos los idiomas de las colecciones

En las colecciones que se brindan en las competencias PAN, se incorporan muestras para la verificación de autoría en los idiomas inglés, holandés y griego. La propuesta que implementamos es dependiente de las Clases de Rasgos con las que se representan los documentos y, como se expone en la descripción de los rasgos empleados, estos se obtienen en dependencia de determinadas herramientas de PLN disponibles.

Realizamos experimentos para los 4 idiomas brindados: español, inglés, griego y holandés. Como salvedad, debemos mencionar que al no disponer de lematizador y etiquetador morfológico para el griego y el holandés, solo se utilizaron combinaciones de 6 clases de rasgo, [*F1- F6*] y para el inglés, al igual que para español, desde [*F1- F10*].

Anteriormente, comprobamos que esta aproximación del promedio es sensible cuando se

Tabla 8. Resultados alcanzados de los participantes en la edición PAN 2014 y nuestro enfoque

Posición	español,-artículos		inglés,-ensayos		inglés,-novelas	
	Trabajo	**c@1**	**Trabajo**	**c@1**	**Trabajo**	**c@1**
1	Khonji & Iraqi	0,77	Frery et al.	0,71	Modaresi & Gross	0,71
2	Castillo et al.	0,76	Satyam et al.	0,65	Zamani et al.	0,65
3	Moreau et al.	0,75	Layton	0,61	Castillo et al.	0,615
4	Frery et al.	0,75	**Nuestro**	0,6	Mayor et al.	0,614
5	**Nuestro**	0,74	Moreau et al.	0,6	Khonji & Iraqi	0,61
6	Jankowska et al.	0,73	Khonji & Iraqi	0,583	Frery et al.	0,58
7	Mayor et al.	0,71	Modaresi & Gross	0,58	Satyam et al.	0,57
8	Vartapetiance & Gillam	0,66	Castillo et al.	0,58	Moreau et al.	0,525
9	Harvey	0,65	Mayor et al.	0,557	Harvey	0,525
10	Modaresi & Gross	0,65	Zamani et al.	0,55	Halvani & Steinebach	0,515
11	Zamani et al.	0,64	Jankowska et al.	0,548	Layton	0,51
12	Halvani & Steinebach	0,64	Harvey	0,54	Vartapetiance & Gillam	0,49
13	Satyam et al.	0,56	Halvani & Steinebach	0,538	**Nuestro**	0,46
14	Layton	0,54	BASELINE	0,53	Jankowska et al.	0,45
15	BASELINE	0,53	Vartapetiance & Gillam	0,52	BASELINE	0,44

dispone de una sola muestra conocida, y en las colecciones de los idiomas griego, holandés ensayo e inglés ensayo se presentan problemas (autores) en los que se dispone de una sola muestra conocida.

Para estos casos, elaboramos una sub-colección eliminando esos problemas y en la tabla de los resultados se llaman igual que la anterior pero con un +. Incluiremos los valores obtenidos en las dos primeras variantes y usando la combinación de los 30 pares de función-rasgo. Ver tabla 9.

Es interesante en estos resultados, apreciar los valores obtenidos para las colecciones de español y holandés ensayo con más de una muestra. En estos se reduce en gran medida en la variante 2 de decisión el número de respuestas en que se equivoca, aunque se incrementan considerablemente las abstenciones. No obstante considero para una situación práctica pericial que es preferible que se abstenga a que dé respuestas equivocadas.

3.5. Influencia de las funciones de comparación

Al ser tres funciones de comparación las propuestas a usar, debemos analizar la influencia de cada una, o sea, evaluar cuál aporta en las decisiones correctas, siempre empleando todas las clases de rasgos con los que se representan los documentos.

En la tabla 10, podemos ver los valores de *accuracy* en las diferentes colecciones, cuando se emplean todas las funciones (30 pares función-rasgo), dos funciones (20 pares función-rasgo) y solo una función de comparación (10 pares función-rasgo). Los resultados presentados se corresponden con la variante 1 propuesta.

Se observa, como resumen, que los valores alcanzados, cuando utilizamos las tres funciones de comparación, en su mayoría son superiores a los alcanzados cuando se emplean dos o una, pero no son significativamente más altos. De todas las funciones de comparación se pueden resaltar los valores obtenidos cuando empleamos la

Tabla 9. Valores de accuracy para todas las colecciones y todas las combinaciones de pares rasgo-función. Variantes 1 y 2. En la variante 2 se expone accuracy(respuestas Positivas, Negativas, Abstenciones)

Año	Colección	idioma	género	todo (variante1)	todo (variante2)
2014	Entrenamiento	Español	artículos	**0,84**	**0.58(36/2/62)**
		Griego	artículos	0,53	0.26(15/11/73)
		Griego +	artículos	0,55	0.16(7/0/73)
		Inglés	novelas	0,57	0.47(33/23/44)
		Inglés	ensayos	0,55	0.43(58/45/97)
		Inglés +	ensayos	0,53	0.34(31/28/85)
		Holandés	comentarios	0,49	0.49(49/49/1)
		Holandés	ensayos	0,5	0.47(37/37/21)
		Holandés +	ensayos	0,67	0.57(12/1/21)
	Prueba	Español	artículos	**0,74**	**0.47(28/4/68)**
		Griego	artículos	0,62	0.41(26/13/61)
		Griego +	artículos	0,66	0.34(15/2/61)
		Inglés	novelas	0,46	0.33(42/40/118)
		Inglés	ensayos	0,6	0.46(62/40/98)
		Holandés	comentarios	0,49	0.49(49/50/1)
		Holandés	ensayos	0,58	0.5(37/29/29)
		Holandés +	ensayos	**0,73**	**0.5(13/1/29)**
2015	Entrenamiento	Español	mixto	**0,77**	**0.45(26/0/74)**
		Griego	multi-tópico	0,58	0.33(20/11/69)
		Griego +	multi-tópico	0,63	0.35(18/2/69)
		Inglés	multi-tópico	0,5	0.5(48/46/6)
		Holandés	multi-género	0,57	0.49(37/30/33)
		Holandés +	multi-género	0,6	0(0/0/33)
	Prueba	Español	mixto	0,66	0,63(62/31/7)
		Griego	multi-tópico	0,57	0,57(57/39/4)
		Inglés	multi-tópico	0,5	0,5(250/250/0)
		Holandés	multi-género	0,5	0,5(83/82/0)

función de distancia MinMax y la semejanza Coseno.

3.6. Influencia de cada clase de rasgo empleado

Otro aspecto importante que evaluamos es la influencia o aporte de las representaciones con cada clase de rasgo. Para esto, analizamos la variación de los resultados de *accuracy* cuando mantenemos la combinación de los resultados de emplear una función de comparación y solo eliminamos una clase de rasgo. Los resultados se aprecian en las tablas 11, 12 y 13.

En la columna se denota como **No 1** a no considerar el empleo del Tipo de Rasgo *F1*, de forma similar el resto de las columnas.

En los resultados, no se aprecia una marcada disminución de los valores de *accuracy* cuando

Tabla 10. Variante 1 de promedio para colecciones de español, variando las funciones de comparación, y manteniendo todas las clases de rasgos

Año	Colección	todo	Jacc-coseno	Jacc-Minmax	Cose-Minmax	Jaccard	Coseno	MinMax
2014	Entrenamiento	**0,84**	0,8	0,85	0,8	0.8	0.82	0.73
	Prueba	**0,74**	0,67	0,73	0,72	0.67	0.71	0.73

Tabla 11. Variante 1 de promedio para colecciones de español, usando Jaccard como función de comparación y dejando de usar un Rasgo en la combinación

Año	Colección	todo	No 1	No 2	No 3	No 4	No 5	No 6	No 7	No 8	No 9	No 10
2014	Entrenamiento	0,8	0,82	0,8	0,81	0,78	0,8	0,78	0,78	0,79	0,77	0,77
	Prueba	0,67	0,64	0,64	0,64	0,66	0,7	0,68	0,64	0,68	0,66	0,66

Tabla 12. Variante 1 de promedio para colecciones de español, usando Coseno como función de comparación y dejando de usar un Rasgo en la combinación

Año	Colección	todo	No 1	No 2	No 3	No 4	No 5	No 6	No 7	No 8	No 9	No 10
2014	Entrenamiento	0,82	0,83	0,81	0,82	0,8	0,82	0,81	0,84	0,81	0,8	0,81
	Prueba	0,71	0,71	0,71	0,69	0,7	0,72	0,7	0,68	0,7	0,69	0,7

Tabla 13. Variante 1 de promedio para colecciones de español, usando MinMax como función de comparación y dejando de usar un Rasgo en la combinación

Año	Colección	todo	No 1	No 2	No 3	No 4	No 5	No 6	No 7	No 8	No 9	No 10
2014	Entrenamiento	0,73	0,73	0,71	0,7	0,73	0,71	0,7	0,72	0,71	0,72	0,72
	Prueba	0,73	0,72	0,7	0,72	0,73	0,72	0,71	0,7	0,7	0,72	0,73

dejamos de emplear alguno de los Tipos de rasgo propuestos.

En resumen, en las secciones de los experimentos, cuando evaluamos el uso de algunas funciones de comparación y cada uno de los Tipos de Rasgos, se observa que la combinación de varios rasgos o de varias funciones de comparación, nos permite obtener valores similares sin mucha afectación. Se debe analizar en detalle cada uno de los rasgos de forma independiente.

3.7. Resultados según la cantidad de muestras conocidas por autor

En los experimentos realizados ocurre que todos los autores presentan la misma cantidad de muestras de documentos conocidos, a pesar de ser pocas.

Con esto, no podemos analizar el impacto que se produce cuando se varía la cantidad de las muestras. La idea que subyace es que mientras mayor sea la cantidad de muestras conocidas, debe equivocarse menos el método, pero también pasa que la dispersión de los objetos en el espacio es mayor.

Con el próximo experimento vamos a evaluar los valores de *accuracy* a medida que incrementamos la cantidad de muestras. Comenzamos con una sola muestra conocida, hasta el total de las muestras. Para esto, promediamos los valores de accuracy para cada autor con una estrategia de validación Leave-one out.

Tabla 14. Variante 1 de promedio para colecciones de español, variando la cantidad de documentos conocidos de muestra y variando las funciones de comparación

# de Docs	todo	Jacc-coseno	Jacc-Minmax	Cose-Minmax	Jaccard	Coseno	MinMax
uno	0,5	0,5	0,5	0,51	0,5	0,54	0,49
dos	0,71	0,7	0,71	0,7	0,68	0,69	0,66
tres	0,77	0,76	0,71	0,73	0.74	0.74	0.73

Tabla 15. Variante 1 de promedio para colecciones de español, variando la cantidad de documentos conocidos de muestra y variando las clases de rasgo empleadas

# de Docs	todo	No 1	No 2	No 3	No 4	No 5	No 6	No 7	No 8	No 9	No 10
uno	0,5	0,5	0,52	0,52	0,51	0,52	0,52	0,5	0,5	0,5	0,5
dos	0,71	0,71	0,71	0,7	0,72	0,72	0,71	0,71	0,72	0,71	0,73
tres	0,71	0,7	0,7	0,7	0,72	0,72	0,7	0,71	0,71	0,72	0,71

Vamos a realizar dos corridas, una en la que empleamos todos los rasgos y vamos eliminando funciones de comparación y otra, en la que mantenemos todas las funciones de comparación y eliminamos un rasgo a la vez. Los valores se reflejan en las tablas 14 y 15. Según las estrategias planteadas para el cálculo del promedio de semejanza y la definición de los umbrales de decisión en base a estos promedios, mínimo necesitamos contar con dos documentos. Para las evaluaciones en que dejamos un solo documento conocido, lo que hicimos fue dividir el documento a la mitad y construir dos documentos. Intuitivamente, esto conformaría dos documentos bien parecidos por lo que el promedio de semejanza debe ser bien alto.

Se observa, como era de esperar, que el cambio en los valores de *accuracy* entre tener un solo documento y más de uno es significativo, por lo que se debe trabajar en estrategias más elaboradas cuando se presenta un problema de un solo documento conocido. A partir de contar con dos documentos o más, no se evidencian diferencias de los resultados.

Tendríamos que estudiar otros fenómenos con respecto a la distribución de las muestras en el espacio de representación, en nuestra aproximación podríamos estudiar la desviación que se experimenta en los valores de los promedios de semejanza de los documentos con respecto al resto, evaluando la dispersión de los documentos de muestra. Esto permitiría definir el uso de algunas de las decisiones de comparación atendiendo a la desviación de las muestras en la clase.

4. Conclusiones y trabajo futuro

Implementamos un método de Verificación de Autoría, atendiendo solo a las muestras conocidas de un autor y sin la calibración de parámetros en fases de entrenamiento. Para este implementamos dos estrategias de representación de las muestras, una basada en instancias y la segunda en un centroide. Debemos evaluar con mayor detalle la aproximación usando centroide.

Definimos tres variantes para calcular la decisión de cuando un documento desconocido pertenece a las muestras del autor, o sea que fue redactado por este o no y consideramos que las variantes 1 y 2 son las más adecuadas, aunque de estas para una situación práctica pericial la variante 2 es menos estricta que la 1 presentándose menos equivocaciones aunque un número alto de abstenciones.

Consideramos que es necesario dedicar esfuerzos en la incorporación de técnicas de

selección de rasgos, que permitan diferenciar mejor los documentos no redactados por el autor de sus muestras conocidas y que la semejanza de sus muestras conocidas sea mayor entre ellas.

Se resalta que la propuesta es sensible al número de muestras conocidas y al tamaño de las mismas. La combinación de varias funciones de comparación y tipos de rasgos para la representación permite que el modelo no se afecte cuando con alguna de estas no se obtienen valores similares al resto.

Debemos además, evaluar en qué medida se obtienen mayorías simples o altas tanto para responder que sí fue redactado, como para responder que no y devolver con esto, un grado de certeza de la respuesta.

No es suficiente con los resultados alcanzados para determinar, con absoluta certeza, cuando un documento No es escrito por un autor, siendo este el detalle en que más debemos trabajar, puesto que con la variante 1 y 2 de decisión, se equivoca poco en responder que sí fue redactado.

Referencias

1. **Manning, C.D., Raghavan, P., & Schütze, H. (2008)**. *Introduction to information retrieval.* Cambridge University Press, Vol. I-XXI, pp. 1–482.
2. **Stamatatos, E., Daelemans, W., Verhoeven, B., Stein, B., Potthast, M., Juola, P., Sánchez-Pérez, M.A., & Barrón-Cedeño, A. (2014).** Overview of the Author Identification Task at PAN 2014. *CLEF Working Notes,* pp. 877–897.
3. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B. (2015).** Overview of the Author Identification Task at PAN 2015. *CLEF Working Notes.*
4. **Castillo, E., Vilariño-Ayala, D., Cervantes, O., & Pinto, D. (2015):** Author attribution using a graph based representation. *CONIELECOMP*, pp. 135–142. DOI: 10.1109/CONIELECOMP.2015.7086940.
5. **Castillo, E., Cervantes, O., Vilariño-Ayala, D., Pinto, D., & León, S. (2014)**. Unsupervised Method for the Authorship Identification Task. *CLEF Working Notes*, pp. 1035–1041.
6. **Ghaeini, M.R. (2013).** Intrinsic Author Identification Using Modified Weighted KNN. *Notebook for PAN at CLEF Working Notes.*
7. **Jankowska, M., Keselj, V., & Milios, E. (2014).** Ensembles of Proximity-Based One-Class Classifiers for Author Verification. *Notebook for PAN at CLEF CLEF Working Notes,* pp. 1069–1072.
8. **López-Monroy, A.P., Montes-y-Gómez, M., Pineda-Villaseñor, L., Carrasco-Ochoa, J.A., & Martínez-Trinidad, J.F. (2012)**. A New Document Author Representation for Authorship Attribution. *Pattern Recognition 4th Mexican Conference, MCPR,* Huatulco, Mexico, Springer, pp. 283–292. DOI: 10.1007/978-3-642-31149-9_29.
9. **López-Monroy, A.P., Montes-y-Gómez, M., Pineda-Villaseñor, L., Carrasco-Ochoa, J.A., & Martínez-Trinidad, J.F. (2012)**. A new document author representation for authorship attribution. *Lect Notes Comput Sci*, Vol. 7329, pp. 283–292. DOI:10.1007/978-3-642-31149-9_29.
10. **Halvani, O., Steinebach, M., & Zimmermann, R. (2013).** Authorship Verification via k-Nearest Neighbor Estimation. *Notebook PAN at CLEF*.
11. **Juola, P. (2006)**: Authorship Attribution. *Foundations and Trends in Information Retrieva*l, Vol. 1, No. 3, pp. 233–334.
12. **Juola, P. & Stamatatos, E. (2013).** Overview of the Author Identification Task at PAN. *CLEF, Working Notes*.
13. **Ruiz-Shulcloper, J. (2009)**. *Reconocimiento Lógico Combinatorio de Patrones: Teoría y Aplicaciones.* Tesis en opción al grado científico de Doctor en Ciencias, La Habana.
14. **Seidman, S. (2013).** Authorship Verification Using the Impostors Method. *Notebook for PAN at CLEF, CLEF Working Notes.*
15. **Stamatatos, E. (2009)**. A survey of modern authorship attribution methods. *J Am Soc. Inf. Sci. Technol.,* Vol. 60, No. 3, pp. 538–556. DOI:10.1002/asi.21001.
16. **Sapkota, U., Bethard, S., Montes-y-Gómez, M., & Solorio, T. (2015).** Not All Character N-grams are Created Equal: A Study in Authorship Attribution. *HLT-NAACL*, pp. 93–102.

Artículo recibido el 14/11/2016; aceptado el 17/03/2017.
Autor de correspondencia es Daniel Castro.

Word Embeddings for User Profiling in Online Social Networks

Anton Alekseev[1,2], Sergey Nikolenko[1,2]

[1] National Research University Higher School of Economics, St. Petersburg, Russia

[2] Steklov Mathematical Institute at St. Petersburg, St. Petersburg, Russia

anton.m.alexeyev@gmail.com, sergey@logic.pdmi.ras.ru

Abstract. User profiling in social networks can be significantly augmented by using available full-text items such as posts or statuses and ratings (in the form of likes) that users give them. In this work, we apply modern natural language processing techniques based on word embeddings to several problems related to user profiling in social networks. First, we present an approach to create user profiles that measure a user's interest in various topics mined from the full texts of the items. As a result, we get a user profile that can be used, e.g., for cold start recommendations for items, targeted advertisement, and other purposes; our experiments show that the interests mining method performs on a level comparable with collaborative algorithms while at the same time being a cold start approach, i.e., it does not use the likes of an item being recommended. Second, we study the problem of predicting a user's demographic attributes such as age and gender based on his or her full-text items. We evaluate the efficiency of various age prediction algorithms based on *word2vec* word embeddings and conduct an extensive experimental evaluation, comparing these algorithms with each other and with classical baseline approaches.

Keywords. User profiling, word embeddings, distributional semantics, ranking.

1 Introduction

In the modern Web, with the advance of social interactions between users and full-scale data mining of all information related to users, user profiling has become a very important problem. In this context, user profiling means converting the recorded user behavior into a set of labels or probability distributions that capture the most important aspects of the user that can be further used for making new recommendations, providing targeted advertisement, and so on.

One could expect that user profiling can be significantly augmented with natural language processing. Much of what goes on in social networks has the form of a text, and one can use texts generated by the user him/herself such as wall posts or statuses to mine his or her interests and demographic information. The recent development of deep learning techniques for natural language processing has led to state of the art models that operate in a basically unsupervised fashion and do not require much linguistic insight; one such direction of study deals with *word embeddings*, vector representations of words that capture semantic relations between the words and can serve as an intermediate step for other models.

The contributions of this work are twofold. First, we concentrate on a novel application of word embeddings to *user profiling*, with the specific example of improving user age prediction with full-text items. Second, user profiles can incorporate knowledge such as demographic information (age, gender, location etc.) or attempt to infer it from user behavior.

However, the holy grail of user profiling is to concisely represent a user's topical interests, preferably as narrowly as possible, with obvious applications to targeted advertising and new recommendations. The methods of summarizing information about users lie at the core of many personalized search and advertisement engines

Computación y Sistemas, Vol. 21, No. 2, 2017, pp. 203–226
ISSN 1405-5546
doi: 10.13053/CyS-21-2-2734

and various recommender systems. Being able to make predictions based on appropriately summarized prior user-system interaction allows, among other things, to alleviate the so-called *cold start* problem, which is one of the main problems of recommender systems: how do you recommend a new item that has not been rated before or has had very few ratings? Given user profiles and a way to match the new item to these profiles, one can make recommendations when collaborative filtering is inapplicable.

We believe that huge corpora of user-generated texts stored in forums and social networks can be used to produce interpretable, semantic user profiles and improve interests-based recommendations for full-text items. We develop new age prediction methods and algorithms for users interacting with full-text items based on distributed word representations and a novel approach to user's interests profile construction. We show improvements in demographic user profiling for these algorithms, show improved results in item recommendation over collaborative recommenders and, most interestingly, show a way to extract an interpretable user interest profile.

The paper is organized as follows. In Section 2, we survey related work on user profiling with textual information, especially inferred from social media, and word embeddings.

Section 4 is devoted to the background of our experimental study, detailing the dataset collected from the Russian social network *Odnoklassniki* and *word2vec* models we have trained for this study.

In Section 5 we show our proposed algorithms for age prediction and present comprehensive experimental results that show improved age prediction.

Section 6 defines models for semantic user profiling that are also based on word embeddings; we show sample user interest profiles and present an experimental evaluation in terms of recommending new items to the users based on their textual preferences.

We conclude with Section 7. This work is a significantly extended joint journal version of two conference papers [3, 58].

2 Related Work

2.1 User Profiling with NLP

User profiling by user behavior has had a long history in many different contexts. Previous attempts at big data user profiling without deep neural networks have leaned upon external knowledge in the form of ontologies [50] and presented a general framework for using NLP in profiling [14]. There is a large classical field of authorship analysis, attribution and author verification studies [38, 91]; we refer to surveys [16, 78, 79] for details and references.

Some works use natural language processing to perform or augment user profiling. In particular, there have been several works closer to social sciences based on available anonymized datasets that do things similar to user profiling, usually mining demographic information from texts generated by a user, and attempts to mine text to establish new information regarding a user or relations between users, a field known as *social media personal analytics*. Next, we highlight some of this research.

In [43], anonymized text messaging datasets are used to investigate the demographics of texting, while in [26], author profiling for English emails uncovers basic demographic traits (gender, age, geographic origin, level of education, and native language) and five psychometric traits based on email texts.

Several Twitter-based studies have focused on mining demographic features based on tweets [24, 32]; the work [42], for instance, does it in a weakly supervised fashion, using Facebook or Google+ profiles as distant supervision. The work [59] detects personality traits from weblog texts, while the work [5] explicitly studies lexical predictors of personality type, [10] determines demographic information by social media texts, and [67] mines user relations from online discussions; an interesting extension is [28] which attempts personality profiling of fictional characters based on the texts about them.

In [70], author profiles in social media are mined to get hidden user profile information, while in [60] metadata is used to mine author profiles;

the work [85] attempts automatic collection and summarization of personal profiles from various social networks and other sources, while [20] proposes linguistic features that help determine the natural language of a person writing in English (on a dataset of the First NLI Shared Task) and [66] determines a user's occupation by his or her tweets.

In [21, 82], the user's political preferences are determined by his or her tweets, and [40] drives it further to get the user's actual voting intentions. This kind of profiling even extends to medical issues: the work [64] attempts to screen Twitter users for depression based on their tweets. Numerous works on the topic have been published based on the results of the shared Author Profiling Tasks at digital text forensics events by PAN initiative [8, 29, 71–74, 83]; we specifically note the work [8] that uses *word2vec* clustering to get features for author profiling. Finally, there are quite a few works for determining the geographical location of a user from his or her textual activity in social networks [9, 33, 44, 68, 69, 87].

As for neural NLP models, one recent work that actually uses modern neural network-based NLP to automatically construct user profiles is [80]. There, convolutional neural networks are used to construct a joint representation of users, products, and their reviews, in particular user profiles. This results in semantic user profiles that are then used to improve sentiment classification but can probably be used for other purposes as well. A recent work [58] has used word embeddings to construct user profiles from the texts they liked in a social network; the profiles were constructed as logistic regression weights of word clusters (clustered in the semantic space of word embeddings), with a special mechanism to reduce the weights of clusters with common words and bring topical clusters to the top. In [30], a *deep semantic similarity model* (DSSM) is trained to model the "interestingness" of documents. The purpose of the model is to recommend target documents that might interest a user based on a source document which she is reading at the moment. This is mostly an information retrieval model, trained on click transitions between source and target documents; this work is similar to [81] and also uses convolutional architectures. The hierarchical neural language model from [25] with a document level and a token level can also be extended to learning user-specific vectors to represent individual preferences, which can be used to give personalized recommendations.

User profiling is a special case of user modeling. For general reviews of the field and key papers, we refer to [12, 18, 27, 36, 86]. Specific techniques that have been applied to represent user interests in content-based and hybrid recommender systems include, for example, relevance feedback and Rocchio's algorithm [47, 63], where a user profile is represented as a set of words and their weights, penalized if the retrieved textual item is uninteresting, as in [62]. Ontologies and encyclopaedic knowledge sources have been used, e.g., in Quickstep and Foxtrot systems [51] that recommend papers based on browsing history, automatically classify the topics of a paper and make use of relations between the topics in ontology to obtain their similarity; rank is computed based on the correlation of user profile topics and estimated paper topics. Nearest neighbours are often used in such systems; e.g., DailyLearner [11] stores tf-idf representations of recently liked stories in a short-term memory component, using it to recommend new stories [47, 63]. Decision rules have been used, e.g., in the RIPPER system [7, 22], where rules are a conjunction of several tests against items features. Interpretable predicted user characteristics are also often utilized in practice; cf., e.g., Yandex.Crypta.

3 Word Embeddings

Recent advances in distributed word representations have made it into a method of choice for modern natural language processing [31]. Distributed word representations are models that map each word occurring in the dictionary to a Euclidean space, attempting to capture semantic relationships between the words as geometric relationships in the Euclidean space. In a classical word embedding model, one first constructs a vocabulary with one-hot representations of individual words, where each word corresponds to its own dimension, and then trains representations

for individual words starting from there, basically as a dimensionality reduction problem. For this purpose, researchers have usually employed a model with one hidden layer that attempts to predict the next word based on a window of several preceding words. Then representations learned at the hidden layer are taken to be the word's features.

The *word2vec* embeddings come in two flavors, both introduced in [52]: *Continuous Bag-of-Words* (CBOW) and *skip-gram*. During its learning, a CBOW model is trying to reconstruct the words from their contexts, while the skip-gram model operates inversely, predicting the context from the word. The idea of word embeddings has been applied back to language modeling in [53, 54, 56], and starting from the works of Mikolov et al. [52, 55], word representations have been used for numerous NLP problems, including text classification, extraction of sentiment lexicons, part-of-speech tagging, syntactic parsing etc.

Another important model for word embeddings is *Glove* (GLObal VEctors for word representations) [65].

Efficient and/or more stable algorithms for training word embeddings have been developed in [48, 49, 52, 57].

4 Background

4.1 Datasets

For this project, we have obtained a large dataset from the *Odnoklassniki* social network. The dataset has been created as follows:

— the dataset began with 486 seed users;

— for these users, their sets of friends have been extracted;

— then the friends of these friends; as a result, the dataset contains a neighborhood of depth 2 in the social graph for the original seed users.

As a result, the dataset contains information on 868,126 users of the *Odnoklassniki* social network. In particular, it contains the following data:

— demographic information on 868,126 users of the network: gender, age, and region (region info may be imprecise since there is no such explicit field in the user's profile, the region is determined by the IP addresses from which the user has logged in most often);

— the social graph that defines the "friendship" relation and contains (and indicates) several different type of links: "friend", "love", "spouse", "parent", and so on; all users with known demographic data are also present in the social graph;

— history of logins for individual users;

— data on the "likes" ("class!" marks) a user has given to other users' statuses and posts in various groups;

— texts of user posts and group statuses that have been liked by these selected users.

The mean age of all users was 31.39 years; the age distribution is shown on Fig.1. Note that there are quite a lot of users with implausible ages (ages 2 and 3, age higher than 100 years); since the user specifies the age by himself/herself, this probably represents missing, incorrect, or purposefully distorted data. Note that this is an important point for the relevance of our research: when a user has not specified his/her age, or when a user has specified an obviously incorrect age, we still need to predict his or her age in order to give age-related recommendations and enroll the user into age cohorts. For the experiments, however, we have removed from the dataset all ages below 10 and above 80 since they are likely to correspond to faulty/missing information.

Fig.2 shows the distribution of the number of friends in the *Odnoklassniki* dataset; interestingly, while the usual Pareto distribution (straight line on a log-log plot) picks up after about 100 friends, it actually increases before that point. This is probably an artifact of the data collection: naturally, the social circle (neighborhood of depth 2) of a predefined set of seed users will contain few isolated or nearly isolated users.

We began evaluation with the entire dataset as outlined above, what is called below the "extended"

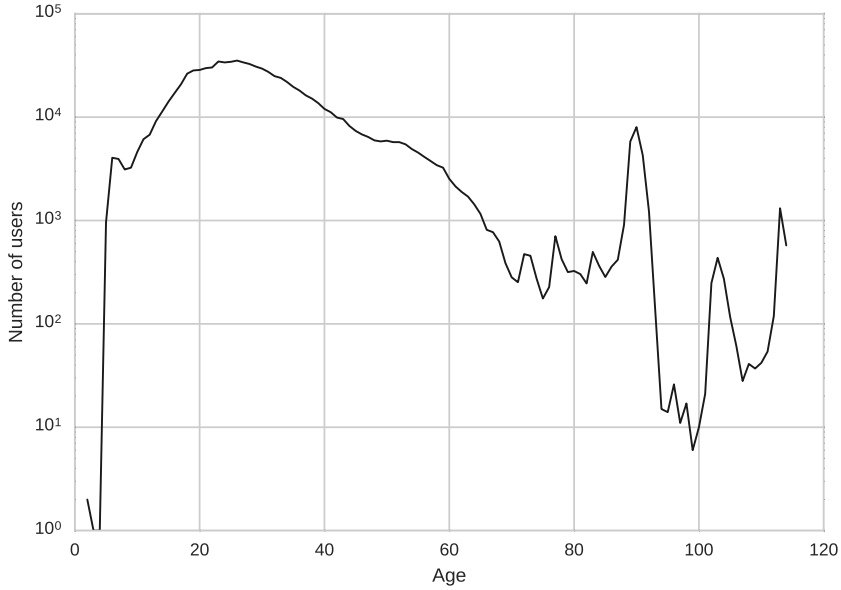


Fig. 1. Age distribution in the *Odnoklassniki* dataset

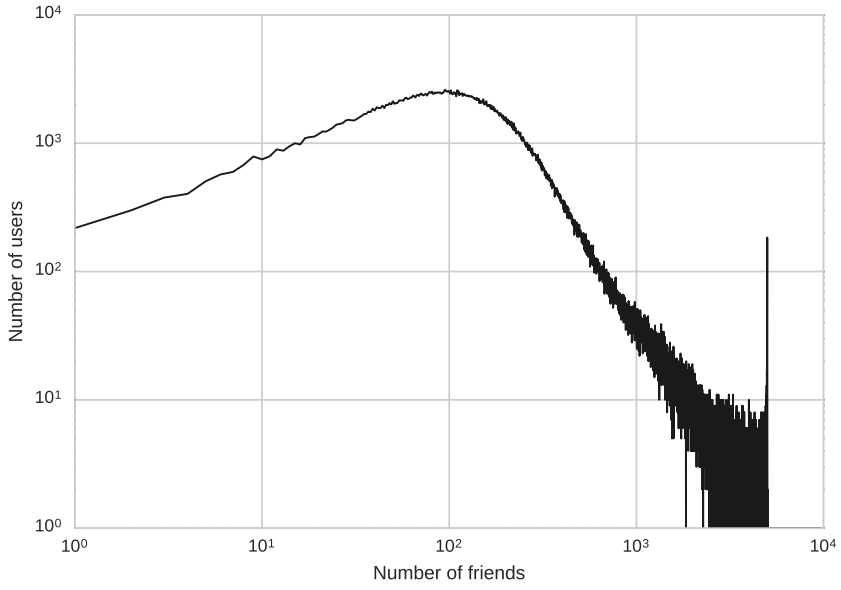


Fig. 2. Number of friends distribution in the *Odnoklassniki* dataset

dataset. However, in order to perform more experiments, be more flexible, and not get bogged down in the technicalities of fitting huge datasets into available hardware, we have also prepared a smaller "basic" dataset that we performed some experiments on.

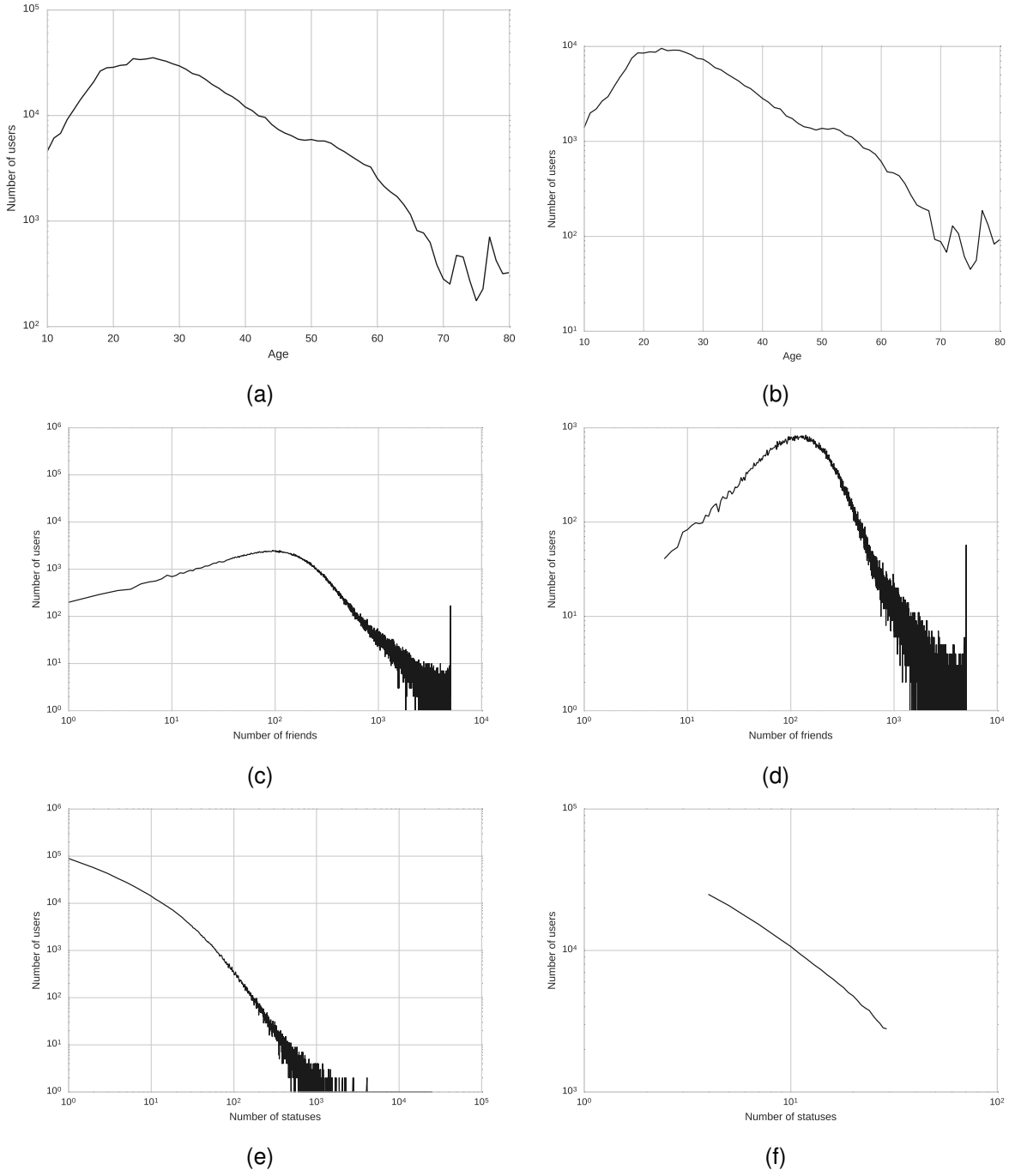


Fig. 3. Basic distributions for the datasets: (a) age distribution, extended, (b) age distribution, basic, (c) friends distribution, extended, (d) friends distribution, basic, (e) statuses distribution, extended, (f) statuses distribution, basic.

Table 1. Basic statistics for extended and basic datasets

Dataset	Training set		Test set	
	Users	Statuses	Users	Statuses
Extended	661206	10880321	165301	2704883
Basic	170856	2014983	42713	503150

Table 2. Word2vec models trained on large Russian corpora; the columns are: dimension d, window size w, number of negative samples n, vocabulary threshold v, and the resulting model size

Type	d	w	n	v	Size
CBOW	100	11	10	20	1.3G
CBOW	100	11	10	30	0.97G
skip-gram	100	11	10	30	0.97G
skip-gram	200	11	1	20	1.3G
CBOW	200	11	1	30	2.0G
skip-gram	200	11	1	30	2.0G
CBOW	300	11	1	30	2.9G
skip-gram	300	11	1	30	2.9G

The basic dataset preserves most properties of the extended dataset; the only difference is that we have filtered the users to have at least 5 and at most 300 statuses. This has let us cut off a relatively small number of highly prolific writers (or, to be more precise, prolific reposters), significantly reducing the total number of statuses, and cut off the long tail of users with very few statuses, while still preserving important properties of the data.

The basic statistics for the two datasets are shown on Table 1, and Fig. 3 indicates that all basic distributions such as age and number of friends are very similar for the two datasets, except, naturally, the distribution of the number of statuses. Both datasets were split into training and test sets randomly in the 80:20 proportion.

4.2 Word2vec Models

As a dataset for word embeddings, we have used a large Russian-language corpus (the largest we know) with about 14G tokens in 2.5M documents [4,61]. This corpus includes the Russian *Wikipedia* (1.15M documents, 238M tokens), automated Web crawl data (890K documents, 568M tokens), (main part) a huge *lib.rus.ec* library corpus (234K documents, 12.9G tokens), and, finally, user statuses and group posts from the *Odnoklassniki* social network, as described above (excluding test data used later). All of this has let us obtain what we believe to be an unprecedented quality of the resulting representations. We refer to [4, 61] for more details on the training data.

We have used continuous bag-of-words (CBOW) and skip n-gram *word2vec* models trained on a single NVidia Titan X GPU with the currently fastest *word2vec* implementation ported to CUDA (`https://github.com/ChenglongChen/word2vec_cbow`). Our previous experiments have suggested that vector sizes in the low hundreds and window size of 11 words are the best parameters on this dataset. In total, so far in the experiments we have used eight different *word2vec* models whose parameters are shown in Table 2; the models differ in the type (CBOW or skip-gram), dimension of word vectors d, window size w (later omitted since $w = 11$ in all models), number of negative samples n in the training, and vocabulary threshold v that controls the size of the vocabulary (a lower threshold means more words get vectors, but words with few occurrences will not have enough training data and might have a random-like, meaningless vector). Note also that every model can come in a "raw" form, as trained, and a normalized form where all vectors are normalized to Euclidean length 1.

5 Demographic User Profiling with Word Embeddings

5.1 User Age Prediction Algorithms

In this section, we propose several relatively simple algorithms that operate on word embeddings of the words in social network statuses of the users, aiming to predict a user's age from his or her writing.

First, folklore among social network researchers says that to predict a user's age it is usually sufficient to take the mean age of his or her friends: it will predict the age with outstanding accuracy. We have tested this hypothesis on the *Odnoklassniki* dataset. To investigate, we have trained the following models:

(1) MEANAGE: predict age with the mean of friends' ages and the global mean if no friends ages are known;

(2) LINEARREGR: linear regression with a single feature (mean friends age);

(3) ELASTICNET: elastic net regressor with a single feature (mean friends age);

(4) GRADBOOST: gradient boosting with a single feature (mean friends age).

Results of these simple models are shown in Table 3 in two variations: basic, where we substitute zeros instead of missing features (when there are no friends' ages) and "nonzero", where we train and test only on a subset of data with nonzero features (at least one friend with known age). It appears that LINEARREGR performs worse than MEANAGE in its first variation because linear regression cannot implement the condition "if the feature is zero (default value in the absence of neighbors) do something completely different", and GRADBOOST is noticeably better because it is powerful enough to handle such case-by-case conditions.

However, we should note that the errors here are quite significant: in terms of MAE, we are more than nine years off on average even if we restrict ourselves to cases with friends with known ages. Hence, we expect that subsequent work is not meaningless and can bring substantial improvements.

Note that while the idea to use the sum and/or mean of word embeddings to represent a sentence/paragraph is, indeed, the simplest idea for the representation of a larger chunk of text, due to the geometric properties of the *word2vec* and GloVe models this idea is not as naive as it sounds. This approach has been used as a baseline in [41] but was proposed as a reasonable method for short phrases in [55] and has been shown to be effective for document summarization in [37].

Thus, we propose three basic algorithms:

(1) MEANVEC: train on mean vectors of all statuses for a user;

(2) LARGESTCLUSTER: train on the centroid of the largest cluster of statuses;

(3) ALLMEANV: train on every status independently, with the mean vector of a specific status and the mean age of friends as features and the user's demography as target; at the testing stage, we compute predictions for every status and average the predictions.

The MEANVEC algorithm simply computes the mean vector of all statuses and adds it as features to the classification/regression model. Formally speaking, we introduce the following notation:

— W is the vocabulary, with words $w \in W$;

— U is the set of users, a user will usually be denoted as $u \in U$;

— S_u is the set of texts "belonging to" user u (either written by u or liked by him/her), with a single text usually denoted as $s \in S_u$; the s stands for either "string" or, more specifically, "status";

— $\boldsymbol{v}_w^m$ is the vector (word embedding) of word w in model m (we will omit the superscript when it is not important or clear from context);

— $\bar{\boldsymbol{v}}_A = \frac{1}{|A|}\sum_{w\in A}\boldsymbol{v}_w$ is the mean vector of a set of word embeddings A;

— MAF_u is the mean age of the friends of a user $u \in U$; in the algorithms, this is the only feature we use from the social graph.

In this notation, the MEANVEC algorithm operates as follows: for a machine learning (regression for age) algorithm ML,

(1) for every user $u \in U$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|}\sum_{w\in S}\boldsymbol{v}_w$;
   — compute the mean vector of all statuses $\bar{\boldsymbol{v}}_u = \frac{1}{|S_u|}\sum_{s\in S_u}\bar{\boldsymbol{v}}_s$;

(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_u)$ for every $u \in U$.

Table 3. Baseline results: predictions by mean age of the friends

Model	Train		Test		Train, nonzero		Test, nonzero	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEANAGE	9.701	6.725	9.661	6.707	8.833	5.865	8.787	5.840
LINEARREGR	11.252	8.659	11.226	8.650	8.794	5.951	8.752	5.930
ELASTICNET	11.251	8.660	11.226	8.651	8.795	5.969	8.752	5.948
GRADBOOST	9.602	6.743	9.569	6.730	8.683	5.879	8.645	5.861

The LARGESTCLUSTER algorithm operates as follows: for a machine learning algorithm ML,

(1) for every user $u \in U$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
   — cluster the set of vectors $\{\bar{\boldsymbol{v}}_s \mid s \in S\}$ into two clusters with agglomerative clustering; denote by $C \subseteq S$ the larger cluster;
   — compute the mean vector of statuses from C $\bar{\boldsymbol{c}}_u = \frac{1}{|C|} \sum_{s \in C} \bar{\boldsymbol{v}}_s$;
(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{c}}_u)$ for every $u \in U$.

Prior experiments (e.g. see 6.3) showed that clustering *word2vec* representations may yield semantically related groups of words/n-grams, and it appears natural to try a similar approach to statuses representations. Hence, the largest cluster could be expected to be the most descriptive.

The ALLMEANV algorithm operates as follows: for a machine learning algorithm ML,

(1) for every user $u \in U$ and every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$ for every $u \in U$ and $s \in S_u$;
(3) on the prediction stage, for a user $u \in U_{\mathrm{test}}$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
   — predict the age for this status, $a_s = \mathrm{ML}(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$;
   — return the average predicted age, $a = \frac{1}{|S_u|} \sum_{s \in S_u} \mathrm{ML}(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$.

5.2 Evaluating User Age Prediction

In the first experiment, we took the simplest MEANVEC algorithm and compared how various *word2vec* models perform. The results are shown in Table 4. We can draw the following conclusions:

— naturally, the MEANAGE algorithm does not care about *word2vec* at all, it is only included as a sanity check;

— *word2vec* models do help all models, both linear and GRADBOOST – compare these results with Table 3;

— it appears that CBOW models outperform skip-gram models in this task (quite significantly);

— by increasing the dimension d, we also get some improvements, but these improvements are rather small;

— a decrease in v, although it makes the *word2vec* model significantly larger and longer to train, has absolutely no effect on the end result.

Generally speaking, these conclusions mean that for the purposes of demographic analysis and similar problems we can concentrate on relatively small *word2vec* models, with dimensions 100 or 200, and perhaps further increase v, which would lead to much smaller models and faster training.

In the second experiment here, we have compared raw and normalized *word2vec* models in the same setting; some of the results are shown in Table 5; for convenience, raw and normalized versions are shown immediately next to each other. The results are rather interesting: the 'farther' the classifier is from linear models, the better normalized versions are. For LINEARREGR, raw vectors slightly outperform normalized ones,

for ElasticNet there is almost no difference, and GradBoost makes (sometimes significantly) better use of the normalized versions. This result can probably be attributed to the fact that while normalized vectors are indeed usually recommended for use, raw vectors can have larger absolute values, including rather large outliers, and simple linear models are better at picking on larger absolute values. Still, the conclusion is to use mostly normalized models in the future since we are after the best model rather than the best linear regression.

The next step was to compare baseline algorithms with each other. Table 6 shows the comparison results between MeanVec and LargestCluster algorithms (marked MV and LC) on the original (extended) dataset, shown for a selection of normalized *word2vec* models.

Interestingly, the LargestCluster algorithm invariably loses to MeanVec in all experiments. One possible reason for this might be that the largest cluster of all statuses turns out in many cases to be the least meaningful (e.g., consisting of similar reposts from an online game or of extremely brief statuses, e.g., consisting of a single smiley); we have verified this idea with a direct examination of the data but believe that in the future, variations on the idea of clustering statuses might yet prove to be useful.

This comparison has been performed on the smaller "basic" dataset that we have presented above. Results of this comparison are shown in Table 7, which marks the MeanVec, LargestCluster, and AllMeanV algorithms as MV, LC, and AV respectively.

As for the results, the LargestCluster algorithm, again, loses in almost all cases to both MeanVec and AllMeanV. What is much more interesting, however, is that AllMeanV, while performing roughly on par with MeanVec in LinearRegr and ElasticNet, begins to lose significantly to MeanVec and even LargestCluster when we use GradBoost as the classifier. This result was quite surprising since we expected that more data and more detailed status vectors (individual for each status rather than averaged over all statuses of a user) will actually bring an improvement. One possible reason for this behavior is that in passing from MeanVec to AllMeanV we have, in essence, "moved" the averaging from the semantic space of word embeddings to averaging prediction results. Hence, this result can be interpreted as showing that simple averaging works very well in the semantic space (this is not surprising given that many semantic relations become linear in the space of embeddings), even better than building an ensemble of predictions from individual statuses afterwards.

5.3 *Word2vec* Trained on Different Data

Another interesting question is whether to use generic *word2vec* models trained on large text corpora externally or train word embeddings specifically for this problem. To answer this question, we have trained *word2vec* models on the user statuses and group posts themselves with the *gensim* library. Table 8 shows a comparison for our three basic classifiers, LinearRegr, ElasticNet, and GradBoost, for the MeanVec algorithm with these "local" word embeddings and "global" word embeddings trained externally (they were used in all previous experiments). We see that while the difference for ElasticNet is nonexistent, both LinearRegr and GradBoost consistently make better use of "local" *word2vec* models. Hence, in future studies we recommend to train word embeddings locally or fine-tune global embeddings with the local dataset.

6 Mining User Interests

6.1 Problem Setting

Apart from demographic predictions, a harder and arguably even more commercially attractive task of user profiling is to concisely represent a user's topical interests, preferably as narrowly as possible. The methods of summarizing information about users lie at the core of many personalized search and advertisement engines and various recommender systems. Being able to make predictions based on appropriately summarized prior user-system interaction allows, among other things, to alleviate the so-called *cold start* problem,

Table 4. A comparison of *word2vec* models, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEANAGE											
				9.672	6.707	9.668	6.711	8.778	5.835	8.800	5.848
LINEARREGR											
cbow	100	10	20	10.401	7.776	10.397	7.785	8.514	5.792	8.541	5.810
cbow	100	10	30	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
skip	100	10	30	10.818	8.219	10.813	8.232	8.624	5.863	8.645	5.879
skip	200	1	20	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
cbow	200	1	30	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
skip	200	1	30	10.735	8.143	10.724	8.148	8.592	5.846	8.613	5.859
cbow	300	1	30	10.338	7.722	10.329	7.727	8.492	5.779	8.512	5.794
skip	300	1	30	10.689	8.088	10.675	8.096	8.583	5.837	8.601	5.854
ELASTICNET											
cbow	100	10	20	10.810	8.208	10.799	8.217	8.694	5.903	8.719	5.921
cbow	100	10	30	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	200	1	30	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
skip	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	300	1	30	11.026	8.433	11.017	8.445	8.741	5.938	8.766	5.956
skip	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST											
cbow	100	10	20	9.089	6.352	9.065	6.344	8.399	5.697	8.394	5.699
cbow	100	10	30	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
skip	100	10	30	9.294	6.527	9.277	6.529	8.495	5.766	8.491	5.770
skip	200	1	20	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
cbow	200	1	30	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
skip	200	1	30	9.365	6.583	9.344	6.580	8.520	5.784	8.512	5.785
cbow	300	1	30	9.048	6.323	9.025	6.316	8.380	5.683	8.371	5.681
skip	300	1	30	9.387	6.596	9.367	6.595	8.536	5.799	8.532	5.804

which is one of the main problems of many recommender systems: how do you recommend a new item that has not been rated before or has had very few ratings? Given user profiles and a way to match the new item to these profiles, one can make recommendations when collaborative filtering is inapplicable.

This motivation ties in well with full-text recommendations. When users interact with items that have actual text associated with them, this allows for a possibility to infer topical user profiles based on automated mining of the texts they interact with. This problem has become especially relevant in recent years due to the growth of the social Web, where users interact with various texts all the time, not only reading but actively rating them.

As for possible solutions, recent advances in natural language understanding, especially in distributional semantics, provide many promising new methods for this problem. This is precisely the path that we take in the second part of the work, using topical clusters based on distributed word representations to construct user profiles.

In this work, we propose a novel method for user profiling in full-text recommender systems, constructing a user profile as an interpretable summary of the user's interests that can also be utilized for recommending new items solely based on the prior state of the system.

6.2 Brief Outline of the Approach

First, we cluster all word representations trained on an external corpus. We have obtained high-quality clusters that are easy to interpret as possible indicators of user's interests, so they were chosen to serve as a basis for user profiling; a user

Table 5. A comparison of *word2vec* models with their normalized versions, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR											
cbow	100	10	20	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
n.cbow	100	10	20	10.426	7.807	10.418	7.807	8.535	5.807	8.560	5.824
skip	100	10	30	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
n.skip	100	10	30	10.753	8.156	10.736	8.159	8.601	5.853	8.620	5.864
skip	200	1	20	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
n.skip	200	1	20	10.363	7.748	10.351	7.746	8.512	5.795	8.532	5.808
cbow	200	1	30	10.735	8.143	10.724	8.148	8.592	5.846	8.613	5.859
n.cbow	200	1	30	10.750	8.152	10.733	8.155	8.601	5.853	8.620	5.864
skip	200	1	30	10.338	7.722	10.329	7.727	8.492	5.779	8.512	5.794
n.skip	200	1	30	10.333	7.720	10.319	7.717	8.501	5.789	8.518	5.800
cbow	300	1	30	10.689	8.088	10.675	8.096	8.583	5.837	8.601	5.854
n.cbow	300	1	30	10.687	8.083	10.668	8.084	8.586	5.840	8.603	5.853
ELASTICNET											
cbow	100	10	20	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
n.cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	20	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
n.skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	30	11.026	8.433	11.017	8.445	8.741	5.938	8.766	5.956
n.skip	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.cbow	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST											
cbow	100	10	20	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
n.cbow	100	10	20	9.043	6.320	9.020	6.313	8.375	5.680	8.367	5.677
skip	100	10	30	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
n.skip	100	10	30	9.204	6.458	9.177	6.449	8.456	5.742	8.448	5.742
skip	200	1	20	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
n.skip	200	1	20	8.998	6.290	8.973	6.281	8.350	5.660	8.337	5.658
cbow	200	1	30	9.365	6.583	9.344	6.580	8.520	5.784	8.512	5.785
n.cbow	200	1	30	9.204	6.457	9.176	6.447	8.455	5.743	8.448	5.743
skip	200	1	30	9.048	6.323	9.025	6.316	8.380	5.683	8.371	5.681
n.skip	200	1	30	8.983	6.274	8.965	6.271	8.341	5.656	8.333	5.656
cbow	300	1	30	9.387	6.596	9.367	6.595	8.536	5.799	8.532	5.804
n.cbow	300	1	30	9.172	6.438	9.150	6.430	8.442	5.732	8.430	5.730

would be characterized by his or her affinity to these clusters.

For the recommender system, we used a large dataset from the "Odnoklassniki" online social network; we used group posts (texts in online communities written by their members) and individual user posts (texts published by a user on his/her profile page) as full-text items and user likes for these posts as ratings.

There are two important obstacles along this way.

1. First, the dataset contains only positive signals from the users (likes), which is common in real life recommender systems but which makes it hard to train.

   While recommender systems based on such implicit information do exist, e.g., recommender systems based on max-margin nonnegative matrix factorization [39], it is unclear how to adapt them to full-text recommendation and user profiles in the semantic space.

2. Whatever technique one tries for the problem,

Table 6. A comparison of the MEANVEC and LARGESTCLUSTER algorithms on the extended dataset for various normalized *word2vec* models

	Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
	type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR												
MV	cbow	100	10	20	10.401	7.776	10.397	7.785	8.514	5.792	8.541	5.810
LC	cbow	100	10	20	10.548	7.926	10.542	7.929	8.573	5.834	8.601	5.852
MV	cbow	100	10	30	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
LC	cbow	100	10	30	10.553	7.933	10.551	7.939	8.574	5.832	8.603	5.853
MV	skip	200	1	20	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
LC	skip	200	1	20	10.849	8.251	10.833	8.255	8.629	5.870	8.650	5.883
MV	cbow	200	1	30	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
LC	cbow	200	1	30	10.493	7.878	10.494	7.886	8.554	5.822	8.581	5.839
ELASTICNET												
MV	cbow	100	10	20	10.810	8.208	10.799	8.217	8.694	5.903	8.719	5.921
LC	cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	cbow	100	10	30	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
LC	cbow	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
LC	skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	cbow	200	1	30	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
LC	cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST												
MV	cbow	100	10	20	9.089	6.352	9.065	6.344	8.399	5.697	8.394	5.699
LC	cbow	100	10	20	9.122	6.388	9.100	6.381	8.427	5.720	8.419	5.720
MV	cbow	100	10	30	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
LC	cbow	100	10	30	9.141	6.399	9.120	6.396	8.431	5.720	8.428	5.723
MV	skip	200	1	20	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
LC	skip	200	1	20	9.277	6.520	9.250	6.508	8.493	5.772	8.490	5.770
MV	cbow	200	1	30	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
LC	cbow	200	1	30	9.090	6.361	9.069	6.353	8.406	5.700	8.398	5.702

user profiles always tend to be dominated by clusters/topics consisting of common words that occur often in the texts of various topics, but are useless for recommendations.

The second problem was especially hard to solve; we solved it with a novel approach to user profiling based on logistic regression trained multiple times on random subsets of the dataset; this approach is described in detail below.

6.3 Clustering Word Vectors

In our experiments, we use a skip-gram *word2vec* model of dimension 500 trained on a large Russian language corpus [4, 61].

To get a finite set of possible user interests or document topics, we clustered the word vectors directly. Note that while for some other applications topic modeling [13, 84] might prove to be more useful, but in our case the basic underlying texts were too short and of too poor quality to hope for a good topic model, decisions regarding the topics would often have to be made on the basis of one or two keywords. Besides, we wanted to develop a top-down general approach that would be applicable directly even without a large and all-encompassing collection of texts available directly in the recommender system.

The embeddings of terms that occurred in our social network posts dataset resulted in 111281 vectors to be clustered in the $\mathbb{R}^{500}$ space. We have tried several methods for large-scale data clusterization, including Birch [90], DBSCAN [76], and mean shift clustering [23], but despite being generally able to process 100K+ items, these methods have proven to be not fast enough for high-dimensional data (for dimension 500 in our case), coupled with 2000 clusters.

Hence, the best option turned out to be classical k-means clustering. We applied

Table 7. A comparison of the MEANVEC, LARGESTCLUSTER, and ALLMEANV algorithms on the basic dataset, normalized *word2vec* models

	Word2vec params				Train		Test	
	type	d	n	v	RMSE	MAE	RMSE	MAE
LINEARREGR								
MV	cbow	100	10	20	8.778	6.066	8.791	6.087
LC	cbow	100	10	20	8.927	6.174	8.946	6.200
AMV	cbow	100	10	20	8.981	6.096	8.991	6.125
MV	skip	100	10	30	9.051	6.256	9.054	6.268
LC	skip	100	10	30	9.153	6.332	9.146	6.335
AMV	skip	100	10	30	9.186	6.249	9.189	6.270
MV	skip	200	1	20	8.974	6.216	8.983	6.225
LC	skip	200	1	20	9.084	6.292	9.091	6.299
AMV	skip	200	1	20	9.132	6.210	9.137	6.231
MV	cbow	200	1	30	8.734	6.035	8.736	6.052
LC	cbow	200	1	30	8.898	6.158	8.902	6.169
AMV	cbow	200	1	30	8.944	6.071	8.952	6.098
MV	skip	200	1	20	8.976	6.217	8.985	6.226
LC	skip	200	1	20	9.086	6.296	9.094	6.301
AMV	skip	200	1	20	9.133	6.211	9.139	6.232
ELASTICNET								
MV	cbow	100	10	20	9.390	6.498	9.397	6.522
LC	cbow	100	10	20	9.390	6.498	9.397	6.522
AMV	cbow	100	10	20	9.398	6.428	9.398	6.451
MV	skip	100	10	30	9.390	6.498	9.397	6.522
LC	skip	100	10	30	9.390	6.498	9.397	6.522
AMV	skip	100	10	30	9.399	6.428	9.398	6.451
MV	skip	200	1	20	9.390	6.498	9.397	6.522
LC	skip	200	1	20	9.390	6.498	9.397	6.522
AMV	skip	200	1	20	9.398	6.428	9.398	6.451
MV	cbow	200	1	30	9.390	6.498	9.397	6.522
LC	cbow	200	1	30	9.390	6.498	9.397	6.522
AMV	cbow	200	1	30	9.399	6.428	9.398	6.451
MV	skip	200	1	20	9.390	6.498	9.397	6.522
LC	skip	200	1	20	9.390	6.498	9.397	6.522
AMV	skip	200	1	20	9.399	6.428	9.398	6.451
GRADBOOST								
MV	cbow	100	10	20	8.075	5.398	8.068	5.399
LC	cbow	100	10	20	8.163	5.456	8.172	5.467
AMV	cbow	100	10	20	8.228	5.447	8.275	5.486
MV	skip	100	10	30	8.277	5.534	8.271	5.537
LC	skip	100	10	30	8.333	5.572	8.336	5.577
AMV	skip	100	10	30	8.362	5.548	8.408	5.583
MV	skip	200	1	20	8.242	5.526	8.231	5.513
LC	skip	200	1	20	8.308	5.562	8.307	5.559
AMV	skip	200	1	20	8.347	5.541	8.392	5.572
MV	cbow	200	1	30	8.032	5.368	8.026	5.366
LC	cbow	200	1	30	8.142	5.447	8.144	5.446
AMV	cbow	200	1	30	8.220	5.443	8.264	5.479
MV	skip	200	1	20	8.240	5.518	8.230	5.510
LC	skip	200	1	20	8.308	5.560	8.302	5.558
AMV	skip	200	1	20	8.347	5.542	8.392	5.573

mini-batch k-means that samples subsets of data (mini-batches) and then applies standard k-means to then: they are assigned to centroids, and centroids are "moved" to actual centers; the updates are done stochastically, after every mini-batch [77]. For initialization, we used the k-means++ approach that initializes cluster centers as far from each other as possible and then applies standard k-means to a random data subset to refine initialization [6].

Table 9 shows sample clusters together with their *idf* (inverse document frequency) values. It is clear that the most frequent clusters largely consist of common words that do not represent any specific topic that could be used for recommendations; they will be our major problem in the next section.

We begin with the following notation: D is the set of documents, C, set of clusters, T, set of all words, T_c, set of words in a cluster $c \in C$, $\mathrm{word2vec} : T \to \mathbb{R}^d$, function assigning each word its embedding, $\mathrm{df}(t)$, number of documents $t \in T$ occurs in, $\mathrm{clust} : T \to C$, function returning the cluster of a word, I_u^{like}, set of all items user u liked.

To produce user profiles, we first constructed fixed-dimensional vector representations of documents $v_{doc} \in \mathbb{R}^d$ for each document $doc \in D$, representations of clusters of documents $v_c \in \mathbb{R}^d$ for each cluster $c \in C$ based on the representations of documents, and finally representations of users $v_u \in \mathbb{R}^d$ for each user $u \in U$ based on representations of the documents they liked and the corresponding clusters; in our experiments, $d = 500$. To build vector representations, we used a straightforward approach based on averaging and idf-like weighting. Suppose that we know word2vec word embeddings for a large proportion of words in our data (not all due to typos, proper names and the like), $v_c = \frac{1}{|T_c|} \sum_{t \in T_c} \mathrm{word2vec}(t)$ for each $c \in C$. Then we define

$$\mathrm{df}_c = \sum_{t \in T_c} \mathrm{df}(t), \quad IDF_c = \log\left(\frac{\sum_{c^* \in C} \mathrm{df}_{c^*}}{\mathrm{df}_c}\right),$$

$$v_{doc} = \sum_{t \in doc} \frac{IDF_{\mathrm{clust}(t)} \cdot v_{\mathrm{clust}(t)}}{IDF^{doc}_{sum}},$$

and $IDF^{doc}_{sum} = \sum_{t\ indoc} IDF_{\mathrm{clust}(t)}$. Finally, the user representation is

$$v_u = \sum_{doc \in I^u_{liked}} \frac{\sum_{t \in doc} IDF_{\mathrm{clust}(t)} \cdot v_{\mathrm{clust}(t)}}{Z},$$

Table 8. *word2vec* trained on local and global dataset, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR											
Global, cbow	100	10	20	10.426	7.807	10.418	7.807	8.535	5.807	8.560	5.824
Local, cbow	100	10	100	10.321	7.703	10.302	7.701	8.518	5.783	8.505	5.794
Global, cbow	200	1	30	10.750	8.152	10.733	8.155	8.601	5.853	8.620	5.864
Local, cbow	200	10	10	10.261	7.646	10.230	7.632	8.529	5.807	8.473	5.771
ELASTICNET											
Global, cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
Local, cbow	100	10	100	11.254	8.656	11.225	8.649	8.771	5.951	8.758	5.952
Global, cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
Local, cbow	200	10	10	11.253	8.666	11.226	8.649	8.803	5.965	8.750	5.946
GRADBOOST											
Global, cbow	100	10	20	9.043	6.320	9.020	6.313	8.375	5.680	8.367	5.677
Local, cbow	100	10	100	8.986	6.270	8.951	6.263	8.348	5.651	8.318	5.649
Global, cbow	200	1	30	9.204	6.457	9.176	6.447	8.455	5.743	8.448	5.743
Local, cbow	200	10	10	8.983	6.277	8.906	6.235	8.351	5.656	8.289	5.628

Table 9. Sample *word2vec* clusters

IDF	Terms
3.276	decide family leave buy parent read case week ...
4.469	smile work answer appreciate state goal given inside brain remind ...
5.703	comment Quran culture union Kim German note interview East forum historical ...
5.902	salt pepper garlic sour-cream greens vegetables carrot cucumber ...
6.126	pain disease shock depression abortion cardiac dense muscular insomnia ...
7.608	stick axe thunder arrow sword boomerang shield spike steel armor paddle ...
8.239	lead fly move once drive run walk ...
9.650	bacteria molecule spermatozoid leukocyte chromosome mitochondria amoeba ...
11.004	scaffold gallows pardon torture quartering hanging beheading ...

where Z is a corresponding normalization value.

Then we constructed a new representation of a document, designed as a vector of cluster likelihoods $p(c \mid d)$; namely, for every document $doc \in D$ and every cluster $c \in C$ we computed

$$L(doc \mid c) = e^{\frac{-\|v_{doc} - v_c\|}{2\sigma^2}},$$

$$p(c_i|doc_j) = \frac{L(doc_j|c_i)}{\sum_{doc} L(doc|c_i)}.$$

Then, to construct the profile of a user u by his or her set of liked items I_u^{like}, we repeated the following procedure N times independently (in the experiments below, we used $N = 100$):

(i) on step k, draw a random sample from the documents the user u didn't like, taking the size of the sample equal to the number of documents the user actually liked; we denote it by $I_{u,k}^{\text{non-like}}$;

(ii) train logistic regression with the following data: I_u^{like} are the positive examples, $I_{u,k}^{\text{non-like}}$ are the negative examples, and the features are document affinities to clusters $p(c \mid d)$, $d \in I_u^{\text{like}} \cup I_{u,k}^{\text{non-like}}$;

(iii) as a result of this logistic regression, we get a set of weights $w_{u,c,k}$ for each cluster c.

Then, for every user u his or her profile is defined as the parameters of the normal distribution for every weight $\{(c, \mu_{u,c}, \sigma_{u,c}) \mid c \in C\}$, each $(\mu_{u,c}, \sigma_{u,c})$ trained on the set $\{w_{u,c,k} \mid c \in C\}$.

In other words, logistic regression here is used to approximate the probability of a like; it trains a hyperplane separating liked items from items that

have not been liked in the semantic feature space. This simple approach would be sure to fail if we simply trained liked documents against non-liked documents since the dataset is vastly imbalanced (a single user can be expected to view but a tiny fraction of all items); hence the random sampling of non-liked documents.

However, there is one more purpose to the random sampling apart from balancing the problem. We would like to solve the problem of common-word clusters, clusters that contain common words, are ubiquitous in the dataset, and tend to dominate all user profiles simply because by random chance, a user will like more than their fair share of some of common word clusters. In randomly sampling the negative examples, we get a certain distribution of "concentrations" of different clusters in the negative example. Note that:

— "topical" clusters that contain rare words will seldom occur in negative examples and thus the variance of the resulting weight distribution $\sigma_{u,c}$ with respect to these clusters will be low;

— "common word" clusters that contain words that are widely distributed across the entire dataset will sometimes appear more often and sometimes less often in the negative examples, and thus the variance of the resulting weight distribution $\sigma_{u,c}$ with respect to these clusters will be high.

Hence, this approach lets us distinguish between common word clusters and topical clusters by the value of $\sigma_{u,c}$. The higher the standard deviation, the more likely it is that the cluster consists of common words. As the final scoring metric for the user profile, we propose to use the mean weight penalized by its variance; we used $\mu - 2\sigma$ as the final score in the examples below. This scoring metric can also be thought of as the lower bound of a confidence interval for the cluster affinity. Figure 4 shows sample results of two user profiles. Note how common-words clusters have high average affinity but also high standard deviation that drags them down in the final scoring and lets topical clusters come out on top.

6.4 Recommender Algorithm and Evaluation

Here, we present an actual recommender algorithm based on the user profiles mined as shown above. This serves as both a sample application for our user profiling system and as a way to evaluate our results numerically, by comparing it to baseline recommender algorithms.

We propose the following item-based algorithm to make recommendations based on a user profile in the form $\{(c, \mu_{u,c}, \sigma_{u,c}) \mid c \in C\}$:

(1) penalize the mean of a cluster's weight distribution with its variance, $w^\star_{u,c} = \mu_{u,c} - \alpha\sigma_{u,c}$, where α is a coefficient to be tuned for a specific system;

(2) predict the probabilities of likes according to the logistic regression model with modified weights,

$$p(\text{Like} \mid doc, \boldsymbol{w}'_u) = \left(1 + \exp(\boldsymbol{v}^\top_{doc}\boldsymbol{w}^\star_u)\right)^{-1};$$

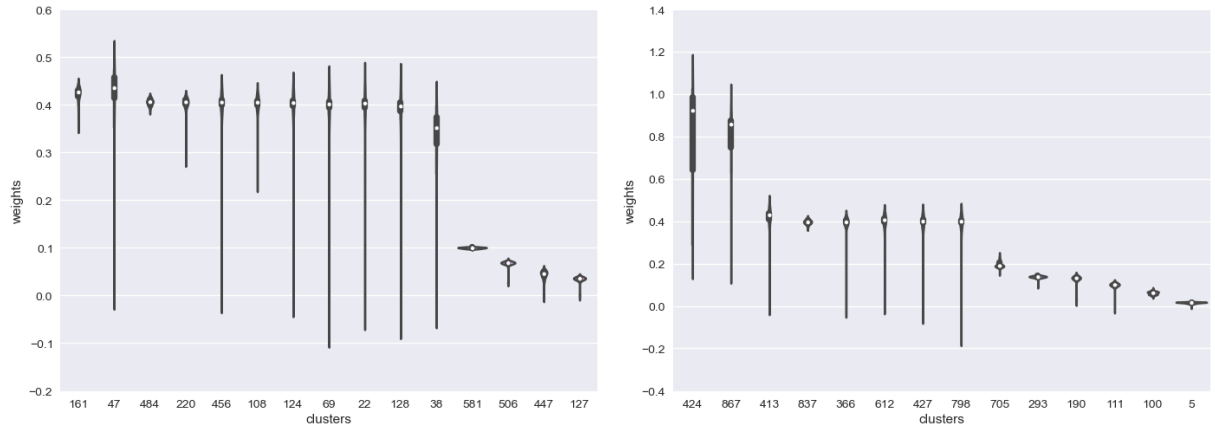
(3) rank the items according to the predicted probability.

Note that this is a cold start algorithm for the items: it does not use an item's likes at all, only the likes of a user to construct his or her profile.

We have conducted experimental evaluation with a large dataset provided by the "Odnoklassniki" social network. For the experiment, we have chosen to use posts in groups (online communities) and likes provided by the users for these posts since a post in a group, as opposed to a post in a user's profile, is likely to be evaluated by many users with different backgrounds, and the users are more likely to like it based on its topic and content rather than the person who wrote it.

Thus, the dataset consists of texts of posts in the communities (documents) and lists of users who liked the posts. The dataset contains 286K words in the vocabulary (after stemming and stop words removal), 14.3M documents (group posts), and 284.6M total tokens in these documents.

As the user set U, we chose top 2000 users with most likes from a randomly sampled subset of users (so that we get users with a lot of likes but not outliers with huge number of likes that are



#	μ	σ	Words
161	0.422	0.017	uni din tel tine adam riga van eden loc publ etc judo art professor polis...
47	0.421	0.074	Nido Josep Jordi Victor Fbio Paulinho Avito Oswaldo Oliveira Julio...
484	0.407	0.007	Virgo Aries Taurus Dey hill branch statue strand Hon patron figure Chakhon...
220	0.405	0.014	Christ soul spirit God pray purify verse forgiven salvation devote Saturday...
108	0.396	0.035	Daniel Vladimir chronicle Tver Svyatoslav Ryazan Novgorod Pskov veche...
124	0.391	0.062	ask talk call listen report try calm assuring prompt pleading convince...
69	0.386	0.081	minute managed approach this autumn song male children ages anxiety...
#	μ	σ	Words
867	0.772	0.165	hours two-hour break minute half-hour five-minute two-hour ten-hour...
424	0.833	0.202	kissing call cry silent scream laughing nod dare restrain angry slam...
837	0.399	0.010	youtube blog net mail facebook player online yandex user tor ado...
366	0.396	0.042	associate attitude seems quite horoscope ideal religious face era...
413	0.406	0.080	feel glad remember worrying offended jealous inhale pity envy suffer autumn...
427	0.385	0.073	hijack bombing raid to steal loot bomb
798	0.385	0.080	uro missile air defense mine RL submarine Vaenga Red Banner Pacific Fleet...

Fig. 4. Sample user profile produced by resampling logistic regression. Top: graphical representations of the two user profiles. Middle: sample clusters for the profile depicted on the left. Bottom: sample clusters for the profile depicted on the right

most probably bots or very uncharacteristic users). We divided their likes into disjoint training and test sets; there were 16000 likes by these users in the training set and 4797 likes in the test set.

We carried out our evaluation procedures on three algorithms: two baseline collaborative algorithms and the new algorithm described above.

User-based collaborative algorithm finds k nearest neighbors for a user and recommends documents according to the users' likes. Specifically, for each user u we build a list of k nearest neighbours $N(u)$ by cosine distance (via LSHForest) in the space of their vector representations in $\mathbb{R}^d$. Then we set the affinity between users as cosine distance between their vector representations:

$$w(u_1, u_2) = v_{u_1}^\top v_{u_2}, \quad \forall u_1, u_2 \in U.$$

Documents are ordered by the following ranking function:

$$\mathrm{rank}(u, doc) = \frac{\sum_{u' \in N(u) \cap \mathrm{Liked}(doc)} w(u, u')}{\sum_{u' \in N(u)} w(u, u')},$$

where $\mathrm{Liked}(doc)$ is the set of users who liked document doc. Thus, we rank documents

according to the weighted sum of representations of users who liked it.

Item-based collaborative algorithm finds k nearest neighbors for a document and recommends documents similar to the ones a user liked. Specifically, for each $doc \in D$ we build the set of k nearest neighbors $N(doc)$ by cosine distance in the space of vector representations for the documents, compute similarities between the documents, $w(doc_1, doc_2) = v_{doc_1}^\top v_{doc_2}$, and rank documents as

$$\mathrm{rank}(u, doc) = \frac{\sum_{doc' \in N(doc) \cap \mathrm{Like}(u)} w(doc, doc')}{\sum_{doc' \in N(doc)} w(doc, doc')},$$

where $\mathrm{Like}(u)$ is the set of documents user u liked.

In all algorithms that use k nearest neighbors approach we used an empirically chosen $k = 5$.

Finally, in our *regression-based algorithm* we recommend according to the negative-biased posterior: given a user profile $\{(\mu_{u,c}, \sigma_{u,c} \mid c \in C\}$, we rank documents according to

$$\mathrm{rank}(u, doc) = \frac{1}{1 + e^{-\sum_{c \in C} p(c|doc) w^*_{u,c}}},$$

where $w^*_{u,c} = \mu_{u,c} - \alpha\sigma_{u,c}$.

All users and documents vector representations are normalized before applying each of the algorithms above. Each of the evaluated recommender algorithms provides the ranking of documents for a given user. We build a set of all likes from test set and the same number of unliked documents for each user. It is expected that the liked documents will be ranked higher than others on average, which is a common ranking task. Hence, we used standard ranking evaluation metrics to evaluate the algorithms:

— NDCG (Normalized Discounted Cumulative Gain) is a unified metric of ranking quality [35]; the discounted cumulative gain is defined as

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{\mathrm{liked}_i}{\log_2(i+1)},$$

where $\mathrm{liked}_i = 1$ iff item i in the ranked list is recommended correctly, and NDCG normalizes this value to the maximal possible: $\mathrm{NDCG}_p = \mathrm{DCG}_p / \mathrm{IDCG}_p$, where IDCG_p is the perfect DCG of a ranked list with all correct items on top;

— Top1, Top5, and Top10 metrics show the share of liked documents at the first place, among the top five, and among the top ten recommendations respectively; these metrics are important for real-life recommender systems since an average user commonly views only a very small number of recommendations.

Results of our experimental evaluation are shown in Table 10. We see that the simple cold start recommender algorithm based on our user profiles performs virtually on par with collaborative algorithms that actually take into account the likes already assigned to this item. These are very good results for a cold start algorithm; note, however, that actual recommendations of full-text items in the same system are not the only or even the main purpose of our approach: the ultimate goal would be to employ user profiles to make outside recommendations for other items with textual content or tags that could be related to the interest profile, such as targeted advertising.

Another way to demonstrate that the regression method learns new things about the users and items being recommended is to show its contribution into the performance of ensembles of rankers. We used the following blending method: first, we normalized the scores obtained by the methods in the blend ($Score_m$ for each ranking method m):

$$Score^m_{norm}(doc) = \\ = \frac{Score^m(doc) - \min_{doc} Score^m(doc)}{\max_{doc} Score^m(doc) - \min_{doc} Score^m(doc)}$$

for every document doc and ranking method m, and then constructed the final scoring function as

$$Score_{blended}(doc) = e^{\sum_m e^{Score^m_{norm}(doc)} \alpha_m},$$

where α_m are blending weights to be found. We use hill climbing to tune parameters $\alpha_m \in [-1, 1]$, maximizing average NDCG for a separate validation set and finally testing performance on a production set that constituted 20% of the values. Rows 4 and 5 of Table 10 show that the blends noticeably improved upon the performance of both our regression-based approach and collaborative algorithms.

Table 10. Experimental results for the recommender algorithms

Algorithm	NDCG	Top1	Top5	Top10
User-based CF	0.7817	0.4557	1.9440	2.6557
Item-based CF	0.7904	0.4934	1.9636	2.6589
Regression-based cold start	0.7777	0.4852	1.8741	2.5960
User-based CF + regr.	0.8089	0.5508	2.0130	2.6920
Item-based CF + regr.	0.8043	0.5364	1.9834	2.6589

7 Conclusion and Future Work

In this work, we have prepared and preprocessed a huge Russian language free text dataset with a number of different sources ranging from literature to user statuses in social networks, trained a number of *word2vec* models, obtained and preprocessed a large user profiling dataset from the social network *Odnoklassniki*, suggested a number of user profiling algorithms based on *word2vec* embeddings, and performed a large-scale comparison of these algorithms and different *word2vec* models, drawing conclusions important for subsequent work on user-generated texts. We have also presented a new approach to user profiling based on logistic regression on randomly resampled subsets of items, which leads to readily interpretable user profiles; our experiments have shown that a simple cold start recommender algorithm based on user profiles produces results comparable to collaborative approaches and can be blended with them for further improvement.

While the proposed age prediction algorithms did bring certain improvements as compared to the "zero baseline" of training with the mean age of a user's friends, these improvements were not huge in absolute terms: we have been able to shave off about 0.2 years in terms of mean absolute error. Therefore, we remain optimistic that these results can be much improved in the future. In further work, we plan to

(1) develop new features for user profiling algorithms based on text embeddings (embedding larger portions of text than a word); here we hope to train a deep text understanding model for the Russian language and apply it to user profiling,

(2) develop and train a character-level word embedding model for the Russian language; we expect this model to be very important for studies of user-generated texts replete with typos, intentional misspellings, and so on.

Also, apart from developing new user profiling algorithms, we plan to investigate other variations of word embeddings. For example, one such is given by the Polyglot system [2], and a completely different direction with a graph-based model is proposed in [1].

We also note recent efforts in *word sense disambiguation* for word embeddings: the same word can have several very different meanings, and it would be natural to try to model it with several vectors in the semantic space [15, 17, 19, 34, 45, 46, 75, 88, 89]. In further work, we plan to perform an even more extensive comparison between various word embedding variations; a comparison across these models might provide valuable insight into the use of *word2vec* models for subsequent applications such as user profiling, sentiment analysis, or full-text recommendations.

Acknowledgements

We thank Dmitry Bugaichenko, Philipp Fedchin, and the *Odnoklassniki* social network (from the *Mail.Ru* holding) for the social network dataset and Alexander Panchenko and Nikolay Arefyev for general-purpose Russian-language training data.

The work of Anton Alekseev was supported by the Russian Federation grant 14.Z50.31.0030. The work of Sergey Nikolenko was supported by the Basic Research Program at the National Research University Higher School of Economics (HSE) in 2017.

References

1. **Aggarwal, C. & Zhao, P.** (**2010**). Graphical models for text: A new paradigm for text representation and processing. *Proceedings of the 33rd International ACM SIGIR, Conference on Research and Development in Information Retrieval*, SIGIR '10. ACM, New York, NY, USA, pp. 899–900. DOI: 10.1145/1835449.1835672.
2. **Al-Rfou, R., Perozzi, B., & Skiena, S.** (**2013**). Polyglot: Distributed word representations for multilingual nlp. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, Sofia, Bulgaria, pp. 183–192.
3. **Alekseev, A. & Nikolenko, S. I.** (**2016**). Predicting the age of social network users from user-generated texts with word embeddings. *Proceeding oh 5th conference on Artificial Intelligence and Natural Language*.
4. **Arefyev, N., Panchenko, A., Lukanin, A., Lesota, O., & Romanov, P.** (**2015**). Evaluating three corpus-based semantic similarity systems for Russian. *Proceedings of International Conference on Computational Linguistics Dialogue*.
5. **Argamon, S., Dhawle, S., Koppel, M., & Pennebaker, J. W.** (**2005**). Lexical predictors of personality type, *Proceedings Joint Annual Meeting of the Interface and the Classification Society of North America*.
6. **Arthur, D. & Vassilvitskii, S.** (**2007**). K-means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA. pp. 1027–1035.
7. **Basu, C., Hirsh, H., & Cohen, W.** (**1998**). Recommendation as classification: Using social and content-based information in recommendation. *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 714–720.
8. **Bayot, R., Gonçalves, T.** (**1998**). Author Profiling using SVMs and Word Embedding Averages—Notebook for PAN at CLEF 2016. *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers*, pp. 5-8, September, Évora, Portugal
9. **Berggren, M., Karlgren, J., Östling, R., & Parkvall, M.** (**2015**). Inferring the location of authors from words in their texts. *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, Linköping University Electronic Press, Sweden, Vilnius, Lithuania, pp. 211–218.
10. **Bergsma, S. & Van Durme, B.** (**2013**). Using conceptual class attributes to characterize social media users. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Sofia, Bulgaria, pp. 710–720.
11. **Billsus, D. & Pazzani, M. J.** (**2000**). User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, Vol. 10, No. 2-3, pp. 147–180. DOI: 10.1023/A:1026501525781.
12. **Bjorkoy, O.** (**2010**). *User Modeling on the Web: An Exploratory Review of Recommendation Systems*, Ph.D. thesis, NTNU Trondheim.
13. **Blei, D. M., Ng, A. Y., & Jordan, M. I.** (**2003**). Latent Dirichlet allocation. *Journal of Machine Learning Research*, Vol. 3, No. 4–5, pp. 993–1022.
14. **Bloedorn, E. & Mani, I.** (**1998**). Using {NLP} for machine learning of user profiles. *Intelligent Data Analysis*, Vol. 2, No. 1–4, pp. 3–18. DOI: 10.1016/S1088-467X(98)00003-1.
15. **Boleda, G., Padó, S., & Utt, J.** (**2012**). Regular polysemy: A distributional model. *Proceedings of the First Joint Conference on Lexical and Computational Semantics, Vol. 1: Proceedings of the Main Conference and the Shared Task, and Vol. 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 151–160.
16. **Bouanani, S. E. M. E. & Kassou, I.** (**2014**). Article: Authorship analysis studies: A survey. *International Journal of Computer Applications*, Vol. 86, No. 12, pp. 22–29.
17. **Bruni, E., Tran, N. K., & Baroni, M.** (**2014**). Multimodal distributional semantics. *J. Artif. Int. Res.*, Vol. 49, No. 1, pp. 1–47.
18. **Brusilovsky, P., Kobsa, A., & Nejdl, W.**, editors (**2007**). *The Adaptive Web: Methods and Strategies of Web Personalization*. Springer-Verlag, Berlin, Heidelberg.
19. **Chen, Z., Lin, W., Chen, Q., Chen, X., Wei, S., Jiang, H., & Zhu, X.** (**2015**). Revisiting word

embedding for contrasting meaning. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 106–115.

20. **Cimino, A., Dell'Orletta, F., Venturi, G., & Montemagni, S.** (**2013**). Linguistic profiling based on general–purpose features and native language identification. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, Atlanta, Georgia, pp. 207–215.
21. **Cohen, R. & Ruths, D.** (**2013**). Classifying political orientation on twitter: It's not easy! *International AAAI Conference on Weblogs and Social Media*.
22. **Cohen, W. W.** (**1995**). Fast effective rule induction. *Twelft International Conference on Machine Learning (ML95)*, Morgan Kaufmann Publishers, pp. 115–123.
23. **Comaniciu, D. & Meer, P.** (**2002**). Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 24, No. 5, pp. 603–619. DOI: 10.1109/34.1000236.
24. **Deitrick, W., Miller, Z., Valyou, B., Dickinson, B., Munson, T., & Hu, W.** (**2012**). Gender identification on twitter using the modified balanced winnow. *Communications and Network*, Vol. 3, No. 4, pp. 189–195. DOI: 10.4236/cn.2012.43023.
25. **Djuric, N., Wu, H., Radosavljevic, V., Grbovic, M., & Bhamidipati, N.** (**2015**). Hierarchical neural language models for joint representation of streaming documents and their content. *Proceedings of the 24th International Conference on World Wide Web*, WWW '15. ACM, New York, NY, USA. DOI: 10.1145/2736277.2741643.
26. **Estival, D., Gaustad, T., Pham, S. B., Radford, W., & Hutchinson, B.** (**2007**). Author profiling for english emails. *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING'07)*, pp. 263–272.
27. **Fischer, G.** (**2001**). User modeling in human–computer interaction. *User Modeling and User-Adapted Interaction*, Vol. 11, No.1-2, pp. 65–86. DOI: 10.1023/A:1011145532042.
28. **Flekova, L. & Gurevych, I.** (**2015**). Personality profiling of fictional characters using sense-level links between lexical resources. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Lisbon, Portugal, pp. 1805–1816.
29. **Forner, P., Navigli, R., & Tufis, D.** Clef 2013 evaluation labs and workshop–working notes papers. pp. 23-26, Valencia, Spain.
30. **Gao, J., Pantel, P., Gamon, M., He, X., Deng, L., & Shen, Y.** (**2014**). Modeling interestingness with deep neural networks. *EMNLP*.
31. **Goldberg, Y.** (**2015**). A primer on neural network models for natural language processing. *CoRR*. DOI: abs/1510.00726.
32. **Green, R. M. & Sheppard, J. W.** (**2013**). Comparing frequency-and style-based features for twitter author identification. *FLAIRS Conference*.
33. **Han, B., Cook, P., & Baldwin, T.** (**2013**). A stacking-based approach to twitter user geolocation prediction. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Association for Computational Linguistics, Sofia, Bulgaria, pp. 7–12.
34. **Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y.** (**2012**). Improving word representations via global context and multiple word prototypes. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, pp. 873–882.
35. **Jarvelin, K. & Kekalainen, J.** (**2002**). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, Vol. 20, No. 4, pp. 422–446.
36. **Johnson, A. & Taatgen, N.** (**2005**). User modeling. *Handbook of human factors in Web design*, Lawrence Erlbaum Associates, pp. 424–439.
37. **Kågebäck, M., Mogren, O., Tahmasebi, N., & Dubhashi, D.** (**2014**). Extractive summarization using continuous vector space models. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pp. 31–39.
38. **Koppel, M., Schler, J., Argamon, S., & Messeri, E.** (**2006**). Authorship attribution with thousands of candidate. Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 659–660.

39. **Kumar, B. V., Kotsia, I., & Patras, I.** (**2012**). Max-margin non-negative matrix factorization. *Image and Vision Computing*, Vol. 30, No. 4–5, pp. 279 – 291. DOI: 10.1016/j.imavis.2012.02.010.
40. **Lampos, V., Preoţiuc-Pietro, D., & Cohn, T.** (**2013**). A user-centric model of voting intention from social media. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Sofia, Bulgaria, pp. 993–1003.
41. **Le, Q. V. & Mikolov, T.** (**2014**). Distributed representations of sentences and documents. *CoRR*, pp. 1405.4053.
42. **Li, J., Ritter, A., & Hovy, E.** (**2014**). Weakly supervised user profile extraction from twitter. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Baltimore, Maryland, pp. 165–174.
43. **Ling, R., Bertel, T. F., & Sundsøy, P. R.** (**2012**). The socio-demographics of texting: An analysis of traffic data. *New Media & Society*, Vol. 14, No. 2, pp. 281–298.
44. **Liu, J. & Inkpen, D.** (**2015**). Estimating user location in social media with stacked denoising auto-encoders. *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, Association for Computational Linguistics, Denver, Colorado, pp. 201–210.
45. **Liu, P., Qiu, X., & Huang, X.** (**2015**). Learning context-sensitive word embeddings with neural tensor skip-gram model. *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, AAAI Press, pp. 1284–1290.
46. **Liu, Y., Liu, Z., Chua, T.-S., & Sun, M.** (**2015**). Topical word embeddings. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 2418–2424.
47. **Lops, P., Gemmis, M. D., Semeraro, G., Lops, P., Gemmis, M. D., & Semeraro, G.** (**2011**). Content-based recommender systems: State of the art and trends.
48. **Luo, Q. & Xu, W.** (**2015**). Learning word vectors efficiently using shared representations and document representations. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 4180–4181.
49. **Luo, Q., Xu, W., & Guo, J.** (**2014**). A study on the cbow model's overfitting and stability. *Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, Web-KR '14, ACM, New York, NY, USA, pp. 9–12. DOI: 10.1145/2663792.2663793.
50. **Middleton, S. E., Shadbolt, N. R., & De Roure, D. C.** (**2004**). Ontological user profiling in recommender systems. *ACM Transactions on Information Systems (TOIS)*, Vol. 22, No. 1, pp. 54–88.
51. **Middleton, S. E., Shadbolt, N. R., & De Roure, D. C.** (**2004**). Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, Vol. 22, No. 1, pp 54–88. DOI: 10.1145/963770.963773.
52. **Mikolov, T., Chen, K., Corrado, G., & Dean, J.** (**2013**). Efficient estimation of word representations in vector space. *CoRR*, pp. 1301–3781.
53. **Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., & Khudanpur, S.** (**2010**). Recurrent neural network based language model. *INTERSPEECH*, Vol. 2, pp. 3.
54. **Mikolov, T., Kombrink, S., Burget, L., vCernockỳ, J. H., & Khudanpur, S.** (**2011**). Extensions of recurrent neural network language model. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5528–5531.
55. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J.** (**2013**). Distributed representations of words and phrases and their compositionality. *CoRR*, pp. 1310.4546.
56. **Mnih, A. & Hinton, G. E.** (**2009**). A scalable hierarchical distributed language model. *Advances in neural information processing systems*, pp. 1081–1088.
57. **Mnih, A. & Kavukcuoglu, K.** (**2013**). Learning word embeddings efficiently with noise-contrastive estimation. **Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., & Weinberger, K. Q.**, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pp. 2265–2273.
58. **Nikolenko, S. I. & Alekseyev, A.** (**2016**). User profiling in text-based recommender systems based on distributed word representations. *Proceedings 5th International Conference on Analysis of Images, Social Networks, and Texts*.
59. **Oberlander, J. & Nowson, S.** (**2006**). Whose thumb is it anyway?: classifying author personality from weblog text. *Proceedings of the COLING/ACL on Main conference poster sessions*, Association for Computational Linguistics, pp. 627–634.

60. **Paik, W., Yilmazel, S., Brown, E., Poulin, M., Dubon, S., & Amice, C.** (**2001**). Applying natural language processing (nlp) based metadata extraction to automatically acquire user preferences. *Proceedings of the 1st International Conference on Knowledge Capture*, K-CAP '01, ACM, New York, NY, USA. pp. 116–122. DOI: 10.1145/500737.500757.

61. **Panchenko, A., Loukachevitch, N., Ustalov, D., Paperno, D., Meyer, C. M., & Konstantinova, N.** (**2015**). Russe: The first workshop on russian semantic similarity. *Proceedings of the International Conference on Computational Linguistics and Intellectual Technologies (Dialogue)*, pp. 89–105.

62. **Pazzani, M. & Billsus, D.** (**1997**). Learning and revising user profiles: The identification of interesting web sites. *Mach. Learn.*, Vol. 27, No. 3, pp. 313–331. DOI: 10.1023/A:1007369909943.

63. **Pazzani, M. J. & Billsus, D.** (**2007**). The adaptive web. Chapter in Content-based Recommendation Systems, Springer-Verlag, Berlin, Heidelberg, pp. 325–341.

64. **Pedersen, T.** (**2015**). Screening twitter users for depression and ptsd with lexical decision lists. *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, Association for Computational Linguistics, Denver, Colorado, pp. 46–53.

65. **Pennington, J., Socher, R., & Manning, C.** (**2014**). Glove: Global vectors for word representation. *Proceedings Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 532–1543.

66. **Preoţiuc-Pietro, D., Lampos, V., & Aletras, N.** (**2015**). An analysis of the user occupational class through twitter content. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 1754–1764.

67. **Qiu, M., Yang, L., & Jiang, J.** (**2013**). Mining user relations from online discussions using sentiment analysis and probabilistic matrix factorization. *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Atlanta, Georgia, pp. 401–410.

68. **Rahimi, A., Cohn, T., & Baldwin, T.** (**2015**). Twitter user geolocation using a unified text and network prediction model. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 2: Short Papers, Association for Computational Linguistics, Beijing, China, pp. 630–636.

69. **Rahimi, A., Vu, D., Cohn, T., & Baldwin, T.** (**2015**). Exploiting text and network context for geolocation of social media users. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Denver, Colorado, pp. 1362–1367.

70. **Rangel, F. & Rosso, P.** (**2016**). On the impact of emotions on author profiling. *Information Processing & Management*, Vol. 52, No. 1, pp. 73 – 92. DOI: 10.1016/j.ipm.2015.06.003.

71. **Rangel, F., Rosso, P., Moshe Koppel, M., Stamatatos, E., & Inches, G.** (**2013**). Overview of the author profiling task at PAN. *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, CELCT, pp. 352–365.

72. **Rangel, F., Rosso, P., Potthast, M., Stein, B., & Daelemans, W.** (**2015**). Overview of the 3rd author profiling task at PAN. *CLEF*.

73. **Rangel, F., Rosso, P., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daeleman, W., et al.** (**2014**). Overview of the 2nd author profiling task at PAN 2014. *CEUR Workshop Proceedings*, Vol. 1180, CEUR Workshop Proceedings, pp. 898–927.

74. **Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B.** (**2016**). Overview of the 4th author profiling task at PAN cross-genre evaluations. *Working Notes Papers of the CLEF*.

75. **Reisinger, J. & Mooney, R. J.** (**2010**). Multi-prototype vector-space models of word meaning. *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 109–117.

76. **Sander, J., Ester, M., Kriegel, H.-P., & Xu, X.** (**1998**). Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Min. Knowl. Discov.*, Vol. 2, No. 2, pp. 169–194. DOI: 10.1023/A:1009745219419.

77. **Sculley, D.** (**2010**). Web-scale k-means clustering. *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, ACM, New York, NY, USA, pp. 1177–1178. DOI: 10.1145/1772690.1772862.
78. **Stamatatos, E.** (**2009**). A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, Vol. 60, No. 3, pp. 538–556.
79. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B.** (**2015**). Overview of the author identification task at PAN.
80. **Tang, D., Qin, B., & Liu, T.** (**2015**). Learning semantic representations of users and products for document level sentiment classification. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 1014–1023.
81. **tau Yih, W., He, X., & Meek, C.** (**2014**). Semantic parsing for single-relation question answering. *Proceedings of ACL*, Association for Computational Linguistics.
82. **Volkova, S., Coppersmith, G., & Van Durme, B.** (**2014**). Inferring user political preferences from streaming communications. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Baltimore, Maryland, pp. 186–196.
83. **Busger op Vollenbroek, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H. & Nissim, M.** (**2016**). GronUP: Groningen User Profiling—Notebook for PAN at CLEF. *CLEF Evaluation Labs and Workshop – Working Notes Papers*, pp. 5-8, Évora, Portugal.
84. **Vorontsov, K., Frei, O., Apishev, M., Romov, P., Suvorova, M., & Yanina, A.** (**2015**). Non-bayesian additive regularization for multimodal topic modeling of large collections. *Proceedings Workshop on Topic Models: Post-Processing and Applications*, TM '15. ACM, New York, NY, USA, pp. 29–37. DOI: 10.1145/2809936.2809943.
85. **Wang, Z., Ll, S., Kong, F., & Zhou, G.** (**2013**). Collective personal profile summarization with social networks. *Proceedings Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 715–725.
86. **Webb, G. I., Pazzani, M. J., & Billsus, D.** (**2001**). Machine learning for user modeling. *User Modeling and User-Adapted Interaction*, Vol. 11, No. 1-2, pp. 19–29. DOI: 10.1023/A:1011117102175.
87. **Wing, B. & Baldridge, J.** (**2014**). Hierarchical discriminative classification for text-based geolocation. *Proceedings Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 336–348.
88. **Wu, Z. & Giles, C. L.** (**2015**). Sense-aware semantic analysis: A multi-prototype word representation model using wikipedia. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 2188–2194.
89. **Yih, W.-t., Zweig, G., & Platt, J. C.** (**2012**). Polarity inducing latent semantic analysis. *Proceedings Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1212–1222.
90. **Zhang, T., Ramakrishnan, R., & Livny, M.** (**1996**). Birch: An efficient data clustering method for very large databases. *SIGMOD Rec.*, Vol. 25, No. 2, pp. 103–114. DOI: 10.1145/235968.233324.
91. **Zheng, R., Qin, Y., Huang, Z., & Chen, H.** (**2003**). Authorship analysis in cybercrime investigation. *Intelligence and Security Informatics*, Springer, pp. 59–73.

Article received on 14/11/2016; accepted on 17/03/2017.
Corresponding author is Anton Alekseev.

Demographic Prediction Based on User Reviews about Medications

Elena Tutubalina[1], Sergey Nikolenko[2,3]

[1] Kazan (Volga Region) Federal University, Kazan, Russia

[2] National Research University Higher School of Economics, Laboratory for Internet Studies, St. Petersburg, Russia

[3] Steklov Institute of Mathematics at St. Petersburg, Russia

tlenusik@gmail.com, sergey@logic.pdmi.ras.ru

Abstract. Drug reactions can be extracted from user reviews provided on the Web, and processing this information in an automated way represents a novel and exciting approach to personalized medicine and wide-scale drug tests. In medical applications, demographic information regarding the authors of these reviews such as age and gender is of primary importance; however, existing studies usually assume that this information is available or overlook the issue entirely. In this work, we propose and compare several approaches to automated mining of demographic information from user-generated texts. We compare modern natural language processing techniques, including feature rich classifiers, extensions of topic models, and deep neural networks (both convolutional and recurrent architectures) for this problem.

Keywords. Demographic prediction, user reviews, medications.

1 Introduction

Modern medical studies increasingly use nonstandard sources of information to obtain new data related to medical conditions, efficiency of drugs, their adverse effects, interactions between different drugs, and so on. One such source of information can be provided by the drug users themselves, in the form of free-text web reviews, social media posts, and other user-generated texts. These sources have been successfully used, for instance, to monitor adverse drug reactions (ADRs), making it possible to detect rare and underestimated ADRs through the users complaining about their health on social networks or specialized forums [36].

However, it may be important for the medical field to learn more than just the existence of an adverse reaction from a text review. Drugs may exhibit different behavior on people with different age, gender, or other parameters that will often be unknown for a text scraped from an Internet forum. Hence, the problem arises to mine demographic information from free-text medical reviews.

In this work, we make the first steps in the direction of extracting demographic information from user-generated texts related to medical subjects. We have collected databases of medical reviews from health-related Web sites with user-generated content, namely *WebMD* and *AskaPatient*, and have trained models to predict the age and gender of users who wrote these reviews. We propose a classification approach based on a classical classifier (we compare SVM and Maximum Entropy classifier, i.e., logistic regression) which is augmented by sets of features based on recently developed novel approaches to text mining: topic models, including the Partially Labeled Topic Model, and features based on word embeddings. We show that the resulting classifier performs significantly better than the baseline.

This work is a significantly extended journal version of the paper [40]; compared to the conference version, we have changed the approach to

baseline classifiers, making them into feature-rich classifiers with topics and word embeddings as features. We have also significantly extended the set of said features, adding new domain-specific information to aid the classifiers. Therefore, the experimental part of this work is new compared to [40], and the results have been substantially improved.

The paper is organized as follows. In Section 2, we survey related work on mining drug-related information from social media and other user-generated texts. Section 3.1 defines models for information extraction from text that we compare in this work: we present the features and briefly introduce topic models with user attributes and distributed word representations. We present experimental results in Section 4 and conclude with Section 5.

2 Related Work

The use of social media for medical and pharmacological data mining has been on the uprising since early 2010s; the term "pharmacovigilance" has been coined for automated monitoring of social media for potentially adverse drug effects and interactions; see also media articles about these effects [14, 37]. One of the first works on this subject [13] analyzed user posts regarding six drugs from a health-related social network. A comprehensive review of text mining techniques as applied to drug reaction detection can be found in [9]. We also note a Social Media Mining Shared Task Workshop (organized as part of the Pacific Symp. on Biocomputing 2016) devoted to mining pharmacological and medical information from social media, with a competition based on a published dataset [35].

In [6], authors identify ADRs from texts on health-related online forums. They used dictionary-based drug detection, extracting symptoms with a combination of dictionary-based and pattern-based methods. A lift measure (also known as pointwise mutual information) was computed to evaluate the likelihood of drug-ADR relation and chi-square test was used to evaluate the statistical significance of the lift measure. Several case studies of drugs showed that some ADRs were reported prior to FDA approval. One limitation of this work is the number of annotated examples in test data: less than 500 ADRs for evaluation.

In [32], existing machine learning dictionary-based approaches were used to identify disease names from user reviews about top 180 most frequently searched medications on the forum WebMD, using a rule-based system to extract beneficial effects of the drug. In order to identify candidates for drug repurposing, authors removed known drug indications and manually reviewed the comments without FDA reports. The main limitation of this work is the lack of an annotated corpus to evaluate the proposed method. The work [42] shows an experiment for ten drugs and five ADRs to examine associations between them on texts from online healthcare communities using association mining techniques. The FDA alerts served as a gold standard to evaluate the associations discovered between drugs and ADRs. We also note a series of works specifically on Spanish language social media [15, 36].

Usually, pharmacovigilance studies employ simple classifiers to extract information on drug effects or interactions. For example, to mine drug-related information from a stream of *Twitter* data, a recent work [24] uses a cascade of simple input filters followed by an SVM classifier, reporting good discovery results, while [44] proposes a weighted average ensemble of four classifiers: one based on a handmade lexicon, two on n-grams, and one on word embeddings.

On the other hand, drug testing and discovery of drug effects and interactions requires one to know demographic information about a user since drug effects can differ significantly depending on the user. This leads to the need to mine demographic information about the authors together with the user-generated texts themselves. When such information is provided, e.g., when the texts are collected from *facebook* users with explicitly known age and gender, there is no problem. However, in many situations user reviews for drugs and medical services are found anonymously on review web sites such as *WebMD* or *AskaPatient*; often demographic information can be known for a minority of users but not all. Hence, the problem

arises to predict user demography based on the texts of user reviews.

In natural language processing, predicting demographic features based on free text falls into a large classical field of authorship analysis, attribution and author verification studies [12, 45]; we refer to surveys [3, 38, 39] for details and references. Numerous works on the topic have been published based on the results of the shared Author Profiling Tasks at digital text forensics events by PAN initiative [2, 5, 7, 27–30]. However, authorship analysis seldom extends to medical issues: for example, the work [23] attempts to screen Twitter users for depression based on their tweets, but to the best of our knowledge, previous work has not attempted to automatically mine demographic information unless it was provided explicitly. In this work, we begin to fill this gap, providing first results on automated predictions of demographic based specifically on medical reviews.

3 Classification Methods

3.1 Models

In this section, we describe two different approaches for demographic prediction applied to a collection of user comments about medications. First, we describe our feature-rich machine learning classifiers. Second, we describe neural networks that rely on word representations learned from unannotated reviews.

3.2 Basic Classifiers and their Features

We formulate the prediction of user attributes as a classification problem. In order to perform the classification, we apply two supervised approaches with a set of hand-crafted features:

(1) support vector machine (SVM);

(2) logistic regression, also called the Maximum Entropy classifier (MaxEnt).

These approaches have been known to achieve the best results in various classification tasks, including sentiment and subjectivity classification [11, 41], ADR classification [34], and demographic prediction [22, 31]. Our classifiers leverages a variety of surface-form, semantic, cluster-based, distributed and lexicon features described below.

The entire set of features used in our classifiers consists of the following subsets:

— **Word ngrams** (NGR): occurrence of contiguous sequences of 1, 2, and 3 tokens; the maximum number of features are 25,000;

— **Drug classification groups** (ATC): drug names are classified in groups at five different levels using the DrugBank database and the ATC classification system;

— **Automatically generated lexicons** (PMI): for each token occurring in a text and present in our automatic lexicon, we use its score to compute the number of tokens with $\mathrm{score}(w) > 0$ and sum of these scores, the number of tokens with $\mathrm{score}(w) < 0$ and sum of these scores, the total score, and maximal and minimum scores; all scores and sums are averaged for each review;

— **Sentiment lexicons** (SENT): for each of the sentiment lexicons (Bing Lius Lexicon and MPQA Subjectivity Lexicon), we compute the following two features: average sum of positive scores for the tokens and average sum of negative scores for the tokens;

— **ADR lexicon** (ADR): presence/absence of ADR mentions using the lexicon;

— **Clusters** (CL): presence/absence of tokens from each of the 150 clusters;

— **Topics** (TPC): presence/absence of tokens from each of 150 topics;

— **Word embeddings** (EMB): the real-valued vector of each word as described in Section 3.4.

In the remainder of this subsection, we define each of these items in detail.

ATC classification. In the Anatomical Therapeutic Chemical (ATC) classification system, biomedical and chemical entities are divided into different groups according to the organ on which they act and their therapeutic, pharmacological, and chemical properties. Using the DrugBank database, we find the presence of a drug in each class up to 5 levels. For example, Prozac (Fluoxetine) is associated with the ATC code N06AB03 and classified into this code and the following codes from higher levels: 'elective serotonin reuptake inhibitors' (N06AB), 'antidepressants' (N06A), 'psychoanaleptics' (N06), 'nervous system' (N). We use these features to incorporate domain-specific medical knowledge into the classification process.

Automatically generated lexicon. The key idea of this automatically generated lexicon is to take advantage of a large corpus of weakly labeled texts, where authors assign several predefined labels to each text. Following state-of-art approaches for sentiment analysis [11], we automatically generated a lexicon based on the score for each token (w) (with frequency greater or equal than 10) in the Health dataset:

$$\text{score}(w) = \text{PMI}(w, cat) - \text{PMI}(w, oth),$$

where

$$\text{PMI}(w, cat) = \log \frac{p(w, cat)}{p(w) * p(cat)}$$

is the pointwise mutual information, cat denotes all texts associated with the particular category, oth denotes all texts in other categories, and $p(w, pt)$ are probabilities of w occurring in the texts labeled with a particular category. As categories we separately use age and gender attributes.

Sentiment lexicons. We used Bing Lius Lexicon[1] and the MPQA Subjectivity Lexicon[2]. We assign the score of +1 for positive entries and the score of -1 for negative entries from the Bing Lius Lexicon. For the MPQA Subjectivity Lexicon, we assign scores +0.5/-0.5 and +1/-1 for weak and strong associations respectively.

[1] http://www.cs.uic.edu/ liub/FBS/opinion-lexicon-English.rar
[2] http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

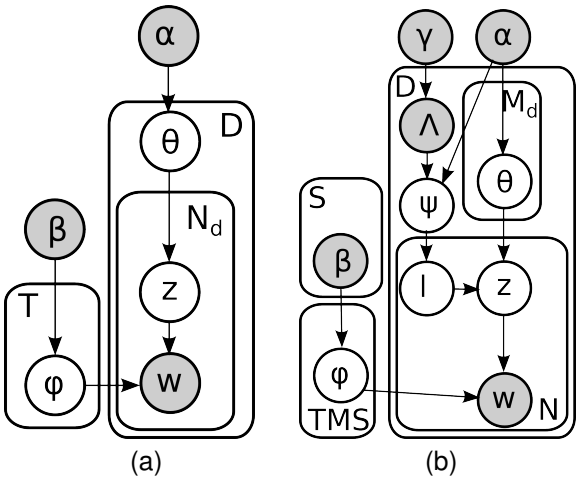


Fig. 1. Probabilistic graphical topic models: (a) the basic LDA model; (b) PLDA

ADR lexicon. We assume that patients experience different adverse drug reactions that may depend on age and gender. In order to use medical information specific to demographic groups, we develop the exact lookup based on ADR lexicon from the paper [34]. The lexicon contains 16,183 ADRs from several resources: the COSTART vocabulary created by the FDA for post-market surveillance of ADRs, the SIDER side effect resource, and the Canada Drug Adverse Reaction Database, SIDER II and the Consumer Health Vocabulary.

Cluster-based features. Clusters reduce the sparsity of the token space as an alternative representation of text. We use the Brown algorithm, i.e., a hierarchical clustering algorithm [4]. The algorithm partitioned the words into a set of 150 clusters, and we add features corresponding to the presence or absence of specific clusters in the review.

Next, we discuss the last two classes of features that come from topic models and word embeddings respectively.

Table 1. Sample topics discovered by PLDA for the tag "female" and "male"

#	topic words
	male
1	muscle left pain legs hands joint neck feet pains burning arms aches body ingling walk
2	effect sexual longer however difficult positive side sex negative control reduced libido
3	stomach diarrhea food eat acid cramps nexium gas upset nausea reflux pains
4	infection throat days rash itching reaction sinus body nose cough face fever
5	meds wife make gave finally times god home big people care end rest things house stay
	female
1	stomach nausea diarrhea eat food cramps upset sick vomiting acid bloating gas constipation
2	body hands rash feet reaction legs itching face swelling arms burning tingling allergic swollen
3	days infection throat prescribed sinus sore cough headache nose antibiotic fever ear
4	feel things happy person family dont anymore husband care longer crying depressed job

3.3 Topic Models

For topic-based features, we employ the latent Dirichlet allocation (LDA) model, a classical topic model. We assume that a corpus of D documents contains T topics expressed by W different words. Each document $d \in D$ is modeled as a discrete distribution $\theta^{(d)}$ on the set of topics: $p(z_w = t) = \theta_{td}$, where z is a discrete variable that defines the topic of each word $w \in d$. Each topic, in turn, corresponds to a multinomial distribution on words: $p(w \mid z_j = t) = \phi_{wt}$ (here w denotes words in the vocabulary and j denotes individual instances of these words). The probabilistic graphical model of basic LDA is shown on Fig. 1a. The model introduces Dirichlet priors with parameters α for topic vectors θ, $\theta \sim \mathrm{Dir}(\alpha)$, and β for word distributions ϕ, $\phi \sim \mathrm{Dir}(\beta)$ (we assume the Dirichlet priors are symmetric, as they usually are). A document is generated word by word: for each word, first sample its topic index t from θ_d, $t \sim \mathrm{Mult}(\theta_d)$, then sample the word w from ϕ_t, $w \sim \mathrm{Mult}(\phi_t)$. We denote by $n_{w,t,d}$ the number of words w generated with topic t in document d; partial sums over such variables are denoted by asterisks, e.g., $n_{*,t,d} = \sum_w n_{w,t,d}$ is the number of all words generated with topic t in document d, $n_{w,*,*} = \sum_{t,d} n_{w,t,d}$ is the total number of times word w occurs in the corpus and so on; we denote by $\neg j$ a partial sum over "all instances except j", e.g., $n^{\neg j}_{w,t,d}$ is the number of times word w was generated by topic t in document d except position j (which may or may not contain w). In the basic LDA model, inference proceeds with *collapsed Gibbs sampling*, where θ and ϕ variables are integrated out, and z_j are iteratively resampled as follows:

$$p(z_j = t \mid \boldsymbol{z}_{-j}, \boldsymbol{w}, \alpha, \beta) \propto \frac{n^{\neg j}_{*,t,d} + \alpha}{n^{\neg j}_{*,*,d} + T\alpha} \cdot \frac{n^{\neg j}_{w,t,*} + \beta}{n^{\neg j}_{*,t,*} + W\beta},$$

where $\boldsymbol{z}_{-j}$ denotes the set of all z values except z_j. Samples are then used to estimate model variables:

$$\theta_{td} = \frac{n_{w,t,d} + \alpha}{n_{w,*,d} + T\alpha}, \quad \phi_{wt} = \frac{n_{w,t,*} + \beta}{n_{*,t,*} + W\beta}.$$

We also experimented with *Partially Labeled Topic Model* (PLDA) [26]. PLDA incorporates user meta-data tags (e.g., location, gender, or age) together with topics. In this model, each document is assigned with an observed tag or a combinations of tags, topics are generated conditioned on the document's tags, and words are conditioned on the latent topics and tags. The probabilistic graphical model of PLDA is shown on Fig. 1b. The Gibbs sampling step proceeds as

$$p(z_j = t, a_j = m \mid \nu) \propto$$
$$\frac{n^{\neg j}_{*,t,m,d} + \alpha}{n^{\neg j}_{*,*,*,d} + TM_d\alpha} \cdot \frac{n^{\neg j}_{w,t,m,*} + \beta}{n^{\neg j}_{*,t,m,*} + W\beta} \cdot \frac{n^{\neg j}_{w,t,m,*} + \beta_{wk}}{n^{\neg j}_{*,t,m,*} + \sum_w \beta_{wk}} \cdot$$

An important characteristic feature of topic models is that they can be mined for qualitative results that are easy to interpret and can validate their performance. For example, Table 1 shows topics discovered by the PLDA model based on a unigram representation of reviews related to each gender; note that the distinction between "male" and "female" topics does indeed reflect common medical knowledge.

3.4 Distributed Word Representations

The other class of models in our study is very different in nature from topic models. We compare results produced by topic models with classification models based on *word2vec* embeddings processed by recurrent and convolutional neural networks (RNNs and CNNs).

Recent advances in distributed word representations have made it into a method of choice for modern natural language processing [8]. Distributed word representations are models that map each word occurring in the dictionary to a Euclidean space, attempting to capture semantic relationships between the words as geometric relationships in the Euclidean space. In a classical word embedding model, one first constructs a vocabulary with one-hot representations of individual words, where each word corresponds to its own dimension, and then trains representations for individual words starting from there, basically as a dimensionality reduction problem. For this purpose, researchers have usually employed a model with one hidden layer that attempts to predict the next word based on a window of several preceding words. Then representations learned at the hidden layer are taken to be the word's features.

The *word2vec* embeddings come in two flavors, both introduced in [16]: *Continuous Bag-of-Words* (CBOW) and *skip-gram*. During its learning, a CBOW model is trying to reconstruct the words from their contexts with a network whose architecture is shown on Fig. 2a; the training process for this model proceeds as follows:

(1) each of the inputs of this network is a one-hot encoded vector of size $|V|$, where V is the vocabulary;

(2) when computing the output of the hidden layer, we take an average of all input vectors; the hidden layer is basically a matrix of vector embeddings of words, so the nth row represents an embedding of the nth word in the vocabulary;

(3) the output layer represents a score u_j for each word in the vocabulary; to obtain the posterior, which is a multinomial distribution, we then use the softmax

$$\hat{P}(w_t|w_1^{t-1}) = \frac{\exp(u_j)}{\sum_{j'=1}^{t}\exp(u_{j'})},$$

so the loss function is

$$E = -\log p(w_t|w_1^{t-1}) = -u_j + \log\sum_{j'=1}^{|V|}\exp(u_{j'}).$$

The skip-gram model operates inversely, predicting the context from the word, which can be seen from its network architecture shown on Figure 2b. Here the target is an input word, and the output layer, in turn, now represents C multinomial disctibutions

$$\hat{P}(w_1^{t-1}|w_t) = \frac{\exp(u_{cj})}{\sum_{j'=1}^{i}\exp(u_{j'})}$$

with the loss computed as

$$E = -\log p(w_1^{t-1}|w_t) = \\ = -\sum_{c=1}^{C} u_{jc} + C\log\sum_{j'=1}^{|V|}\exp(u_{j'}).$$

The idea of word embeddings has been applied back to language modeling in [17, 18, 21], and starting from the works of Mikolov et al. [16, 19], word representations have been used for numerous NLP problems, including text classification, extraction of sentiment lexicons, part-of-speech tagging, syntactic parsing, and others.

Word embedding models represent each word using a single real-valued vector. Such representation groups together words that are semantically and syntactically similar [20]. We used word2vec from Gensim library to train embeddings on the

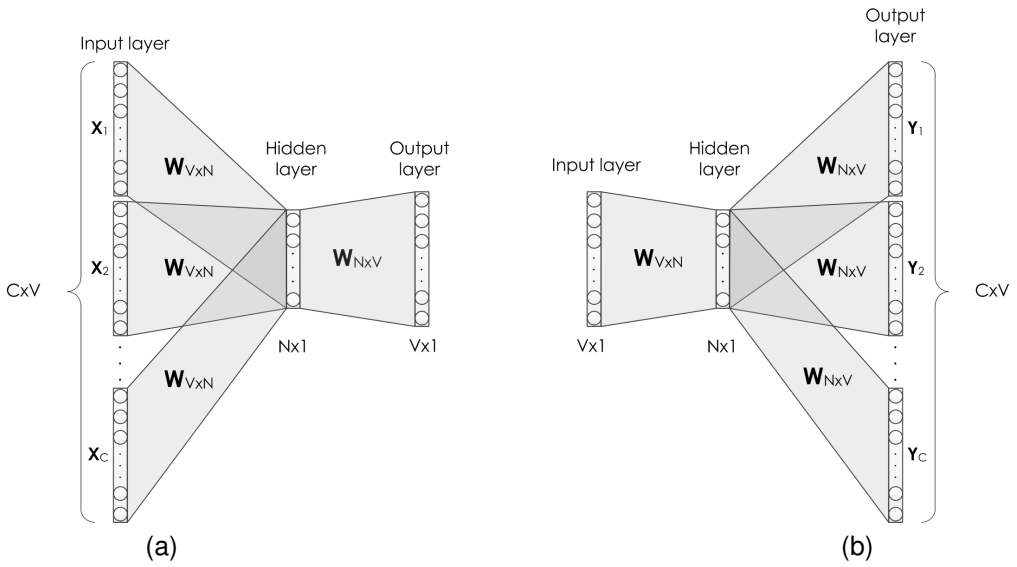


Fig. 2. Illustration of the *word2vec* models: (a) CBOW, (b) skip-gram [16, 33]

Health Dataset. We applied Continuous Bag of Words model with the following parameters: vector size of 200, the length of local context of 10, negative sampling of 5, vocabulary cutoff of 10. Below, we refer to our pre-trained vectors as **HealthVec**. We also experimented with another published word vector PubMedVec (2,351,706 terms) trained on biomedical literature indexed in PubMed [25]. PubMed comprises more than 26 million citations for biomedical literature from bibliographic database MEDLINE, journal articles, life science journals, and online books.

3.5 Neural Network Classifiers

In this work, we compare two modern approaches to natural language processing with neural networks: traditional recurrent architectures, specifically LSTM-based recurrent networks, and convolutional neural networks (CNN). In the recurrent part, we use an architecture with multiple LSTM layers, where higher layers use the sequence of outputs from the previous layer of LSTMs, and on the top level the LSTM outputs are combined into the final layer which does the actual prediction.

While CNNs have been most successfully used for image processing, recent applications of CNNs to natural language processing also produce state of the art results. In an NLP task, convolutional layers are still interleaved with subsampling max-pooling layers, but this time the convolutions are one-dimensional rather than two- or three-dimensional as in images and video. Here, we use a convolutional model similar to the one recently presented in [10] for semantic sentence classification; this model has the following characteristic features:

— it is not as deep as computer vision models and involves only one convolutional layer with max-over-time pooling and a softmax output;

— regularization is achieved through dropout; the authors report a consistent and significant improvement in accuracy with dropout across all experiments;

— the model is trained on prepared *word2vec* word embeddings and does not attempt to tune word representations for better results;

— still, the authors report better results on such tasks as sentiment analysis and sentence classification than baseline techniques that

include recursive autoencoders and recursive neural networks with parse trees.

4 Evaluation

4.1 Datasets

For experimental evaluation, we have crawled health-related reviews from two health hotel review sites: (i) *WebMD*[3] and (ii) *AskaPatient*[4].

WebMD is a health information services website that aims to provide objective, trustworthy, and valuable health information. We have crawled 217,485 reviews from authors tagged as "Patient". Each review contains the following fields: 1) date when the review was written, 2) condition for taking treatment, 3) free-text review given for the effects caused due to the use of the drug, 4) user attributes such as gender and age. Gender tags are "Male" or "Female", and predefined age tags in the dataset are "19–24", "25–34", "35-44", "45-54", "55-64", "65-74", or "75 or over". In this study, we combine some of the age tags and divide user attributes into three major age groups: "19-34", "35-64", and "65 and over".

AskaPatient[5] website aims to empower patients by allowing them to share and compare their medical experiences. We have crawled $113{,}093$ reviews. Since users often confuse two free-text fields about a drug, we have concatenated the "side effects" and "comments" fields, treating the result as a full review. Similar to *WebMD*, reviews from *AskaPatient* contain textual information, reason for taking treatment and user attributes (without predefined list of age groups).

In contrast with our previous work, we split our corpora into training and testing parts further referred as **WebMD** and **AskaPatient** (used by the ML and DL algorithms) and a free-text corpus of in-domain texts called the **Health dataset** (used to compute PMI, topics, and word representations). In order to create robust methods and exclude drugs with highly imbalanced genders (e.g., birth control pills), we use reviews associated with 5 most commented conditions for training/testing. For WebMD, review authors select a condition from a predefined list for every drug. For AskaPatient, the "reason" is a free-text field.

Table 2 summarizes the statistics of both datasets used in our study. The WebMD dataset contains 20,693 reviews with the age group "35-64", 7,410 reviews with the age group "19-34", and 7,519 reviews with the age group "65 and over". The total numbers of tokens in the WebMD and AskaPatient datasets are 2,818,429 and 1,051,969, respectively. The total numbers of unique tokens in the WebMD and AskaPatient datasets are 33,411 and 18,825, respectively.

4.2 Model Parameters

In order to get local features from a review with CNNs we have used multiple filters of different lengths [10]. We separated out 10% of the training dataset to form the validation set which was used to evaluate different model parameters. We used a sliding max-pooling window of size 2 to get features through filters. Pooled features are then fed to a fully connected feed-forward neural network (with dimension 100) which uses rectified linear units as output activations. Then we apply a softmax classifier with the number of outputs equal to the number of classes. We applied dropout rate of 0.5 to the fully connected layer and trained the network for 20 epochs; on the other hand, we did not apply dropout after the embedding layer since in our experiments this led to lower results achieved by CNNs on the validation set. The width of the convolution filters is set to 3, each with 100 filters. Additionally, we employ early stopping after two epochs with no improvement on the validation set. Embedding layers are not trainable for all networks. We set mini-batch size to 256 and 128 for the WebMD and AskaPatient datasets, respectively.

In our experiments with recurrent neural networks, we used a standard GRU or LSTM architecture on top of the embedding layer that implemented pre-trained word embeddings. Similar to [1], the resulting sequence of vectors serves as the input to the network. We experimented with shallow GRU/LSTM, two-layer GRU, and used 100 units on each layer with the Adam optimizer and rectified linear units as output

[3] http://www.webmd.com
[4] http://www.askapatient.com
[5] http://www.askapatient.com

Table 2. Summary statistics for the experimental datasets; number of reviews with a given label in shown in parentheses

Dataset	Top-5 Conditions	Gender	Age groups
WebMD	high blood pressure (10201) pain (9306) depression (7340) chronic trouble sleeping (3454) attention deficit disorder with hyperactivity (3021)	Female (23343) Male (9979)	45-54 (8430) 55-64 (7056) 35-44 (6207) 19-34 (7410) 65 or over (4219)
AskaPatient	depression (3170) anxiety (1603) uti (1545) insomnia (1329) high blood pressure (1270)	Female (6356) Male (2561)	

activation. Similar to CNN, GRU layer is then fed to a fully connected feed-forward neural network (with dimension 100). Other parameters are adopted from CNN settings.

We tested and compared the following vectors:

— **NewsVec**: commonly used word embeddings *GoogleNews-vectors-negative300*[6] trained on part of Google News dataset (about 100 billion words).

— **PubmedVec**: word vectors trained on biomedical scientific literature *PubMed* [25];

— **HealthVec**: word vectors trained on product reviews from the Health dataset.

The general statistics are presented in Table 3. We also observed better classification results after normalizing each vector by dividing it by its 2-norm.

For SVM and MaxEnt classifiers, we used LinearSVC and LogisticRegression with default parameters from the NLTK library[7]. We used Liang's implementation of the Brown hierarchical word clustering algorithm[8].

We used the Mallet[9] library to generate topics. The number of sampling iterations was set to 1000. We used default hyperparameters, took top 20 words for each topic, and evaluated 50, 100, and 150 topics on the validation set. The best results were achieved with 150 topics. We also implemented PLDA for comparison, adopting its parameters from LDA. For further evaluation, we selected topics from LDA rather than PLDA since they produced better results on validation data.

Table 3. Statistics of *word2vec* embeddings

Embeddings	Dimension	# of tokens
GoogleNews	300	3,000,000
PubMed	200	2,351,706
HealthVec	200	31,482

4.3 Results

In this section, we describe our experiments with feature-rich classifiers and deep learning models. We performed pre-processing by lower-casing all words. We performed 5-fold cross-validation and computed precision (P), recall (R), and F1-measure (F1), showing the macro-averaged results in Table 4 (gender prediction) and Table 5 (age prediction). The tables also show the best results for each model type in every column highlighted in bold.

The main result, which might look surprising at first, is that standard classifiers, when enriched with a large number of various features, outperform even the best neural network approaches that we have been able to train. Specifically, CNNs and RNNs are able to achieve better precision than SVM and MaxEnt but lose significantly in recall and therefore in the aggregate F1 measure. Moreover, Tables 4 and 5 show the variances

[6] https://code.google.com/archive/p/word2vec/
[7] http://www.nltk.org
[8] https://github.com/percyliang/brown-cluster
[9] http://mallet.cs.umass.edu

Table 4. Gender prediction (macro-averaged, 2 classes)

Model and features	WebMD			AskaPatient		
	P	R	F1	P	R	F1
SVM classifier (first column shows feature set)						
NGR (1-, 2-, and 3-grams)	0.645	0.649	0.647±0.006	0.651	0.646	0.648±0.012
NGR+ATC	0.647	0.651	0.649±0.006	0.658	0.652	0.655±0.016
NGR+EMB+TCS+CL	0.651	0.653	0.655±0.007	0.650	0.650	0.650±0.009
NGR+EMB+TCS+CL+SENT+PMI+ADR	**0.674**	**0.676**	**0.675±0.003**	**0.665**	**0.657**	**0.660 ±0.009**
MaxEnt classifier (first column shows feature set)						
NGR	0.671	0.662	0.666±0.005	0.675	0.653	0.660±0.008
NGR+ATC	0.674	0.664	0.668±0.006	0.679	0.657	0.665±0.013
NGR+EMB+TCS+CL	0.676	0.670	0.673±0.006	0.670	0.661	0.664±0.010
NGR+EMB+TCS+CL+SENT+PMI+ADR	**0.702**	**0.691**	**0.695±0.006**	**0.683**	**0.665**	**0.672±0.009**
Neural networks for end-to-end classification						
CNN, HealthVec, $[1, 2, 3]$ filters	0.706	0.651	0.663±0.011	0.684	0.635	**0.643±0.027**
CNN, HealthVec, $[1, 2, 3]$, trainable emb.	0.678	0.649	0.657±0.007	0.655	**0.636**	0.642±0.011
CNN, HealthVec, $[1, 2, 3, 4]$ filters	0.702	0.653	0.658±0.007	0.684	0.614	0.620±0.012
CNN, HealthVec, $[1, 2, 3, 4]$, trainable emb.	0.674	**0.664**	**0.668±0.007**	0.673	0.634	0.642±0.021
CNN, NewsVec, $[1, 2, 3]$ filters	**0.707**	0.645	0.653±0.029	**0.701**	0.610	0.612±0.048
CNN, PubmedVec, $[1, 2, 3]$ filters	0.705	0.637	0.646±0.021	0.635	0.607	0.614±0.055
2-layer GRU, HealthVec	0.674	0.648	0.654±0.019	0.622	0.588	0.590±0.024
1-layer GRU, HealthVec	0.680	0.632	0.640±0.015	0.599	0.545	0.523±0.077

Table 5. Age prediction (macro-averaged, 3 classes)

Model and features	WebMD		
	P	R	F1
SVM classifier (first column shows feature set)			
NGR (1-, 2-, and 3-grams)	0.514	0.513	0.513±0.011
NGR+ATC	0.526	0.524	0.525±0.002
NGR+EMB+TCS+CL	0.516	0.518	0.517±0.005
NGR+EMB+TCS+CL+SENT+PMI+ADR	**0.540**	**0.539**	**0.539±0.007**
MaxEnt classifier (first column shows feature set)			
NGR	0.562	0.521	0.536±0.004
NGR+ATC	0.566	0.527	0.542±0.003
NGR+EMB+TCS+CL	0.560	0.529	0.541±0.004
NGR+EMB+TCS+CL+SENT+PMI+ADR	**0.574**	**0.544**	**0.557±0.008**
Neural networks for end-to-end classification			
CNN, HealthVec, $[1, 2, 3]$ filters	0.615	0.490	0.510±0.014
CNN, HealthVec, $[1, 2, 3]$, trainable emb.	0.585	0.512	0.532±0.013
CNN, HealthVec, $[1, 2, 3, 4]$ filters	0.637	0.482	0.504±0.012
CNN, HealthVec, $[1, 2, 3, 4]$, trainable emb.	0.588	**0.514**	**0.536±0.006**
CNN, PubmedVec, $[1, 2, 3]$ filters	**0.648**	0.467	0.488±0.009
2-layer GRU, HealthVec	0.618	0.485	0.483±0.017
1-layer GRU, HealthVec	0.530	0.409	0.396±0.046

Table 6. Representative MaxEnt features for male and female patients with different conditions (WebMD dataset)

Pain				**High blood pressure**			
Female		**Male**		**Female**		**Male**	
for fibromyalgia	1.069	my wife	2.027	my hair	1.904	old male	1.660
fibromyalgia pain	0.937	for shoulder	0.976	hair loss	1.351	my wife	1.478
my migraines	0.918	back fusion	0.814	have gained	1.217	lower my blood	0.991
for arthritis	0.902	my knee pain	0.693	so tired	1.066	sex drive	0.910
muscle relaxer	0.848	sleep at night	0.658	terrible cough	0.925	sexual desire	0.841
severe migraine	0.747	pain level	0.651	swollen ankles	0.875	erectile dysfunction	0.811
old female	0.715	scar tissue	0.643	leg cramps	0.863	frequent urination	0.662
my headaches	0.700	kidney stones	0.578	hot flashes	0.859	heart attack	0.585
throwing up	0.688	chronic knee	0.578	severe headaches	0.706	my kidney	0.557
allergic to	0.676	bulging disk	0.564	muscle pain	0.676	ankle swelling	0.547
Depression				**Attention deficit disorder with hyperactivity**			
Female		**Male**		**Female**		**Male**	
my husband	1.281	my wife	2.723	my daughter	0.706	my wife	1.580
loss of appetite	0.741	some sexual	0.836	with adhd	0.675	old male	1.042
gained weight	0.739	my girlfriend	0.777	my son	0.548	very effective	0.848
very happy	0.676	anxiety disorder	0.758	weight loss	0.522	to urinate	0.661
my kids	0.656	alcohol and	0.705	my child	0.521	my brain	0.653
crying spells	0.642	an erection	0.662	to help	0.517	abdominal pain	0.478
lost weight	0.599	my marriage	0.600	my mood	0.445	over the years	0.414
hot flashes	0.550	diet and exercise	0.508	my heart	0.412	personal relationships	0.360
my moods	0.510	sexual dysfunction	0.504	my husband	0.391	an alcoholic	0.324
my daughter	0.507	lack of appetite	0.499	old female	0.339	my girlfriend	0.324

Table 7. Representative features obtained by MaxEnt for different age groups (WebMD dataset)

High blood pressure					
19-34		**35-64**		**65 and over**	
birth control	0.806	light headed	0.876	was normal	1.250
during pregnancy	0.707	lowered bp	0.820	too expensive	1.216
chest pains	0.667	frequent urination	0.811	feet and ankles	0.817
my pregnancy	0.657	for high bp	0.801	dry eyes	0.814
my headaches	0.625	my sleep	0.768	the price	0.813
low dosage	0.589	feeling tired	0.703	breathing problem	0.809
for my blood	0.588	heartburn and	0.685	my cardiologist	0.779
get pregnant	0.557	sex drive	0.671	life threatening	0.691
muscle cramps	0.524	rapid heart rate	0.629	vision problems	0.626
extreme fatigue	0.491	my insurance	0.546	my sodium	0.618

of the F1-measure in our cross-validation results, indicating that the advantage of SVM and MaxEnt in F1-measure is statistically significant.

This seemingly unexpected result is, in our opinion, due to two main reasons. First, we are free to augment standard classifiers with any features we want, thus using a wide variety of external information that is unavailable to the neural networks, which have to rely on text only. It is unclear how to introduce all of the features that we used for SVM and MaxEnt into the neural networks, and it would require a separate complex study, both theoretical and practical, to incorporate these features.

Second, the dataset size in this case is probably not large enough for the neural networks to shine. Since we used suitable regularization we did not experience strong overfitting in the neural

networks, but general rules of thumb suggest that our supervised datasets are too small for the expressive power of complex neural networks to have significant effect.

Thus, our results suggest that while neural network approaches often define the state of the art in modern natural language processing, in problems where rich additional information can be made available, especially in domain-specific problems with well defined domains (such as medicine in this case), classical machine learning approaches can still be very useful and can still be successfully used in practical settings.

Secondary results include two conclusions from Tables 4 and 5. First, while adding more features is usually obviously beneficial, this did not hold for ATC features in our experiments: they helped much less than others and even deteriorated the results. This is probably due to the fact that a relatively small dataset size combined with high dimension of ATC features led to overfitting. Second, note that the best results with neural networks are usually obtained in variations where the word embeddings are also trainable. Regardless of the dataset size (which would be much too small to train embeddings properly), in our experience making embeddings trainable (i.e., slightly fitting them in the end-to-end supervised network starting from unsupervised vectors) appears to be beneficial almost always and should be adopted in most settings.

We have also performed qualitative analysis of our results. In particular, we have extracted and analysed the most representative n-grams for various conditions. Tables 6 and 7 present the most representative features (excluding numeric features) for one gender over another and for a certain age group over other age classes, respectively. For this experiments, we used the MaxEnt classifier trained on the set of 2- and 3-grams extracted from the review texts. The tables indicate that key terms change with age or gender, reflecting quite natural progressions that match well with medical and commonsense intuition. Hence, our classifiers can also be used to mine qualitative information from a dataset of medical reviews, perhaps uncovering new common conditions or important factors in a certain user group.

5 Conclusion

In this work, we have presented the first results on the practically important problem of automatically learning demographic user features from his or her reviews concerning medical products or services.

We have compared several different models for gender classification and age prediction: baseline classifiers that operate on words and bigrams, feature-rich classifiers with additional information from topic models and word embeddings as well as domain-specific medical information, and convolutional and recurrent neural networks based on *word2vec* embeddings.

Results of our experiments suggest that in settings with relatively small datasets and available external information classical machine learning techniques can outperform neural network approaches. This is due to both dataset size and the fact that while it is hard to tailor neural networks to a specific form of external information, standard classifiers incorporate such new features trivially. We believe that this sample application shows that there is still a place for domain-specific machine learning solutions, especially for relatively small supervised datasets.

Acknowledgements

This work was supported by the Russian Science Foundation grant no. 15-11-10019. The authors are also sincerely grateful to the anonymous referees whose comments have led to significant improvements in the paper.

References

1. **Arkhipenko, K., Kozlov, I., Trofimovich, J., Skorniakov, K., Gomzin, A., & Turdakov, D.** (**2016**). Comparison of neural network architectures for sentiment analysis of Russian tweets. *Proceedings of International Conference, Computational Linguistics and Intellectual Technologies*.

2. **Bayot, R. & Gonçalves, T.** (**2016**). Author profiling using SVMs and word embedding averages – notebook for PAN at CLEF. *CLEF Evaluation Labs and Workshop – Working Notes Papers*, Évora, Portugal.

3. **Bouanani, S.E.M.E. & Kassou, I.** (**2014**). Authorship analysis studies: A survey. *International Journal of Computer Applications*, Vol. 86, No. 12, pp. 22–29.

4. **Brown, P.F., Desouza, P.V., Mercer, R.L., Pietra, V.J.D., & Lai, J.C.** (**1992**). Class-based n-gram models of natural language. *Computational linguistics*, Vol. 18, No. 4, pp. 467–479.

5. **Busger op Vollenbroek, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H., & Nissim, M.** (**2016**). Gronup: Groningen user profiling – Notebook for PAN. *CLEF 2016, Evaluation Labs and Workshop – Working Notes Papers*, pp. 5–8, Évora, Portugal.

6. **Feldman, R., Netzer, O., Peretz, A., & Rosenfeld, B.** (**2015**). Utilizing text mining on online medical forums to predict label change due to adverse drug reactions. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, ACM, New York, USA, pp. 1779–1788.

7. **Forner, P., Navigli, R., & Tufis, D. (2013).** *CLEF evaluation labs and workshop – Working notes papers*. pp. 23–26, Valencia, Spain.

8. **Goldberg, Y.** (**2015**). A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726.

9. **Karimi, S., Wang, C., Metke-Jimenez, A., Gaire, R., & Paris, C.** (**2015**). Text and data mining techniques in adverse drug reaction detection. *ACM Comput. Surv.*, Vol. 47, No. 4, 56:1–56:39.

10. **Kim, Y.** (**2014**). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

11. **Kiritchenko, S., Zhu, X., Cherry, C., & Mohammad, S.** (**2014**). NRC-Canada-2014: Detecting aspects and sentiment in customer reviews. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 437–442.

12. **Koppel, M., Schler, J., Argamon, S., & Messeri, E.** (**2006**). *Authorship attribution with thousands of candidate*.

13. **Leaman, R., Wojtulewicz, L., Sullivan, R., Skariah, A., Yang, J., & Gonzalez, G.** (**2010**). Towards internet-age pharmacovigilance: Extracting adverse drug reactions from user posts to health-related social networks. *Proceedings of the Workshop on Biomedical Natural Language Processing*, BioNLP '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 117–125.

14. **Marcus, A. D.** (**2014**). Researchers fret as social media lift veil on drug trials. *Wall Street Journal*.

15. **Martınez, P., Martınez, J. L., Segura-Bedmar, I., Moreno-Schneider, J., Luna, A., & Revert, R.** (**2016**). Turning user generated health-related content into actionable knowledge through text analytics services. *Computers in Industry*, Vol. 78, pp. 43–56.

16. **Mikolov, T., Chen, K., Corrado, G., & Dean, J.** (**2013**). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

17. **Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., & Khudanpur, S.** (**2010**). Recurrent neural network based language model, *INTERSPEECH*, Vol. 2, No. 3.

18. **Mikolov, T., Kombrink, S., Burget, L., Cernockỳ, J. H., & Khudanpur, S.** (**2011**). Extensions of recurrent neural network language model. *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, pp. 5528–5531.

19. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J.** (**2013**). Distributed

representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

20. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J.** (**2013**). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.

21. **Mnih, A. & Hinton, G. E.** (**2009**). A scalable hierarchical distributed language model. *Advances in neural information processing systems*, pp. 1081–1088.

22. **Nguyen, D., Smith, N. A., & Rosé, C. P.** (**2011**). Author age prediction from text using linear regression. *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, Association for Computational Linguistics, pp. 115–123.

23. **Pedersen, T.** (**2015**). Screening twitter users for depression and ptsd with lexical decision lists. *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, Association for Computational Linguistics, Denver, Colorado, pp. 46–53.

24. **Plachouras, V., Leidner, J. L., & Garrow, A. G.** (**2016**). Quantifying self-reported adverse drug events on twitter: Signal and topic analysis. *Proceedings of the 7th 2016 International Conference on Social Media & Society*, SMSociety '16. ACM, New York, NY, USA, pp. 6:1–6:10.

25. **Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., & Ananiadou, S.** (**2013**). Distributional semantics resources for biomedical text processing. *Proceedings of Languages in Biology and Medicine*.

26. **Ramage, D., Manning, C. D., & Dumais, S.** (**2011**). Partially labeled topic models for interpretable text mining. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 457–465.

27. **Rangel, F., Rosso, P., Moshe Koppel, M., Stamatatos, E., & Inches, G.** (**2013**). Overview of the author profiling task at pan 2013. *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, CELCT, pp. 352–365.

28. **Rangel, F., Rosso, P., Potthast, M., Stein, B., & Daelemans, W.** (**2015**). Overview of the 3rd author profiling task at PAN 2015. *CLEF*.

29. **Rangel, F., Rosso, P., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daeleman, W., et al.** (**2014**). Overview of the 2nd author profiling task at PAN 2014. *CEUR Workshop Proceedings*, Vol. 1180, CEUR Workshop Proceedings, pp. 898–927.

30. **Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B.** (**2016**). Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. *Working Notes Papers of the CLEF*.

31. **Rao, D., Yarowsky, D., Shreevats, A., & Gupta, M.** (**2010**). Classifying latent user attributes in twitter. *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, ACM, pp. 37–44.

32. **Rastegar-Mojarad, M., Liu, H., & Nambisan, P.** (**2016**). Using social media data to identify potential candidates for drug repurposing: A feasibility study. *JMIR Res Protoc*, Vol. 5, No. 2. doi:10.2196/resprot.5621.

33. **Rong, X.** (**2014**). Word2vec parameter learning explained. *CoRR*, abs/1411.2738.

34. **Sarker, A. & Gonzalez, G.** (**2015**). Portable automatic text classification for adverse drug reaction detection via multi-corpus training. *Journal of biomedical informatics*, Vol. 53, pp. 196–207.

35. **Sarker, A., Nikfarjam, A., & Gonzalez, G.** (**2016**). Social media mining shared task workshop. *Proc. Pacific Symposium on Biocomputing*, pp. 581–592.

36. **Segura-Bedmar, I., Martınez, P., Revert, R., & Moreno-Schneider, J.** (**2015**). Exploring

Spanish health social media for detecting drug effects. *BMC Medical Informatics and Decision Making*, Vol. 15, No. 2, pp. 1–9. doi:10.1186/1472-6947-15-S2-S6.

37. **Shaywitz, D. & Mammen, M.** (**2011**). The next killer app. *The Boston Globe*.
38. **Stamatatos, E.** (**2009**). A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, Vol. 60, No. 3, pp. 538–556.
39. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B.** (**2015**). *Overview of the author identification task at PAN*.
40. **Tutubalina, E. & Nikolenko, S. I.** (**2016**). Automated prediction of demographic information from medical user reviews. *Proc. 4th International Conference on Mining Intelligence and Knowledge Exploration*, Lecture Notes in Artificial Intelligence, Springer.
41. **Wilson, T., Wiebe, J., & Hoffmann, P.** (**2009**). Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics*, Vol. 35, No. 3, pp. 399–433.
42. **Yang, C. C., Yang, H., Jiang, L., & Zhang, M.** (**2012**). Social media mining for drug safety signal detection. *Proceedings of the International Workshop on Smart Health and Wellbeing*, SHB '12, ACM, New York, USA, pp. 33–40. doi:10.1145/2389707.2389714.
43. **Zhang, X., Zhao, J., & LeCun, Y.** (**2015**). Character-level convolutional networks for text classification. *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, MIT Press, Cambridge, USA, pp. 649–657.
44. **Zhang, Z., Nie, J.-Y., & Zhang, X.** (**2016**). An ensemble method for binary classification of adverse drug reactions from social media. *Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing*.
45. **Zheng, R., Qin, Y., Huang, Z., & Chen, H.** (**2003**). Authorship analysis in cybercrime investigation. *Intelligence and Security Informatics*, Springer, pp. 59–73.

Article received on 14/11/2016; accepted on 17/03/2017.
Corresponding author is Elena Tutubalina.

Identification of Suicidal Tendencies of Individuals Based on the Quantitative Analysis of their Internet Texts

Tatiana A. Litvinova[1,2,3], Pavel V. Seredin[1,3,4], Olga A. Litvinova[2,3], Olga V. Romanchenko[5]

[1] Voronezh State University, Voronezh, Russia

[2] Voronezh State Pedagogical University, Voronezh, Russia

[3] Scientific Research Centre "Kurchatov Institute", Moscow, Russia

[4] Benemerita Universidad Autonoma de Puebla, Puebla, Mexico

[5] Plekhanov Russian University of Economics, Moscow, Russia

{centr_rus_yaz, olga_litvinova_teacher, ghjd1}@mail.ru, paul@phys.vsu.ru

Abstract. Even though suicide is one of the top three causes of young people's deaths, no reliable methods of identifying suicidal behavior have been developed. One of the promising directions of research is quantitative analysis of speech. It is nowadays common to process texts by suicidal individuals (mostly suicidal notes or literary texts by famous people, e.g., poets, writes, etc.) and texts by individuals from a control group using software (mostly LIWC) and to design models for classifying texts as those by suicidal individuals or not. This kind of analysis has been mainly performed for English texts that generally have a number of restrictions due to their linguistic nature. The authors are the first to attempt to design a mathematical model to classify texts as those by suicidal or nonsuicidal individuals using numerical values of linguistic parameters as features. Texts (blogs by young people who committed suicides, similar in both genre and topic, to those by individuals of an age-corresponding control group) were processed using the Russian version of LIWC with users' dictionaries. Unlike current studies, in designing the model we mostly made use of features that are not significantly dependent on the content. This is because not all individuals who committed suicides are known to deal with the topic in their texts. The resulting model was shown to be 71.5% accurate, which is comparable with the state-of-the-art for English texts.

Keywords. Suicide language, internet texts, suicide predictors, text corpus, computational linguistics, Russian texts, RusPersonality.

1 Introduction

Over 800,000 people die of suicide every year. It is estimated that by the year 2020, this figure will have increased to 1.5 million [3]. It is considered to be one of the major causes of mortality during adolescence [34]. The data suggest that only 30% of suicidal individuals report their inclinations [34].

Thus there is a growing need for methods of identifying suicidal individuals. Speech analysis including quantitative analysis is known to be a valuable diagnostic tool.

In addressing this particular issue, suicide notes are most commonly used. Texts are automatically classified as suicide notes or texts of other genres [13; 16; 24; 27; 28]. There are attempts to design models to distinguish between genuine and fake texts of this genre using qualitative text parameters as features [12] as well as to identify the differences between suicidal notes by individuals who eventually committed

suicide and those who made a suicide attempt [10].

Despite the ultimate value of studying suicide notes, being rather short in length, they present little opportunity for a profound insight into the language of suicidal individuals. As important as these studies are for tackling both theoretical and practical issues, there is an obvious necessity to investigate various linguistic features of texts by suicidal individuals written throughout various periods of their lives in order to identify the predictors of suicidal behavior.

This could then be compared to the texts of a control group of individuals with a maximum similar education level, social status and other characteristics, but most importantly who did not commit suicide. Based on the data, mathematical models can be designed to allow one to predict the likelihood of an individual's suicidal behavior using the quantitative parameters of their texts as features. Studies identifying linguistic features of texts by individuals who committed suicide are commonly conducted using their literary texts [1, 15, 16, 23, 25, 32], less commonly – segments of speech of famous suicidal individuals [7]. However, there are certain restrictions associated with the nature of texts and their authors' personalities, which prevents the results from being extrapolated into the entire population.

Recently, given the growing use of the Internet and social media in particular, scientists have been able to access very valuable linguistic data containing texts by individuals who either committed suicide, attempted suicide, or articulated their suicidal contemplations. Text mining methods have been successfully employed of late to identify such texts as well as over the last decade in sociology in particular [22] (note that there have been attempts using text analysis to identify not only Internet users with suicidal tendencies but also those displaying other forms of destructive behavior, see [33] for details).

Machine learning has been used to identify Twitter texts dealing with suicide [4] and stressed individuals based on their tweets [11], etc. (see Review in [8]).

Most research has been performed for English texts except very few [5, 9, 17].

Note that there have been very few studies of internet texts by individuals who died of suicide and they are mostly case studies (see [14] for Review). Even though most studies have sought to identify internet texts by individuals who had contemplated suicide, not all of them are known to be reported [28], and hence in designing models to identify individuals with high risk of suicidal behavior it is important to focus particularly on implicit suicide markers, i.e. those parameters of texts (features) that are not significantly dependent on the topic and are thus known to be a valuable source of personal data of their authors [31].

The objective of this paper is to design a mathematical model which would enable the classification of texts by suicidal individuals (those with high risks of suicidal behavior) and by non-suicidal ones based on numerical values of selected linguistic parameters (features). Blogs by young people who died as a result of suicide and samples of texts by written speech of young individuals were employed.

Thus, our study aims to contribute to the literature on suicide prevention in social media by attempting to classify texts as those by suicidal individuals based on text parameters (features) that are not significantly dependent on the content. The novelty of the paper is that it is performed on Russian texts and no research of this kind is known of as part of suicide prevention in social media [18].

2 State of the Art

As was previously stated, linguistic analysis of the speech of individuals who committed suicide originally utilized authors' suicide notes [24]. Quantitative parameters of texts used to be manually calculated and later on, automatic text processing tools started being employed for speech analysis. One of the most common tools used for investigating speech of individuals who died of suicide is Linguistic Inquiry and Word Count (LIWC) developed by the American psychologist J. Pennebaker and colleagues [25]. LIWC calculates the proportions of certain grammatical, lexical, and semantic markers, as well as markers belonging to other categories (up to 80 text features depending on the version).

In earlier studies suicidal notes were mainly investigated. Thus using LIWC software, it was found that notes from completed suicides had

fewer metaphysical references, more future tense verbs, more social references (to others) and more positive emotions than did the notes from attempted suicides [10], which seems to be caused by individuals feeling relieved after making a decision to take their own lives.

Collecting texts from other genres written by individuals who committed suicide is a rather daunting task as a lot of research has been conducted using literary texts by suicidal writers. E.g., Stirman & Pennebaker [32], using the LIWC software, found that in poetic texts by suicidal individuals, written at different periods of time, the pronoun "I" is more frequently used compared to the texts by the test group. As time went by, suicidal individuals were seen to use fewer "we" pronouns as well as interaction verbs (e.g., talk, share, listen), but contrary to the common belief there were no statistically significant differences between the suicidal individuals and the test group in the number of words describing negative emotions. It is argued that the results are consistent with the suicide genesis theory, which postulates a connection between suicidal behavior and a growing alienation from other people.

In [1] it was shown that as an individual is just about to commit suicide there is an increase in first-person singular pronoun use, a decrease in first-person plural pronoun use, and an increase in negative emotion word use. In contrast, in [16] there was an increase in words expressing positive emotions, while words associated with causation and insight became less frequent.

Using LIWC, it was also found that suicidal individuals used more abstract words, fewer words overall, more verbs, fewer words relating to "Death", increased number of negations *(not, no)* and so on (see [18] for review).

Apart from literary texts, diaries by these individuals are employed in analysis [14]. Beside the descriptive method, the approach has been employed recently which classifies texts as those by suicidal individuals/not by suicidal individuals (e.g., see [23]). Texts are labelled using NLP tools, and classified using machine learning methods. Hence in [23] the models with the accuracy of 70.6% were obtained using song lyrics by suicidal individuals who committed suicide and those who did not (type/token ratio, proportions of first and second person pronouns, proportions of different vocabulary fields, etc. were employed as features).

The features that displayed differences between texts by suicidal and non-suicidal individuals were numbers (most useful across numerous algorithms), n-grams, the first-person singular + mental verb, concrete nouns, neutral terms, sensual words, total polar value semantic class features, and the first-person singular and passive construction syntactic features. The authors concluded that apart from expanding the corpus, "it would perhaps be fruitful to extend the analysis to other types of features and new lexicons since it has been demonstrated that this task could be solved using NLP" [23, p. 684].

Although written texts are mainly analyzed, there are individual studies which evaluate transcripts of recorded oral speech. A 2016 study [26] also argued the importance of using NLP techniques in automatic classification of texts, where the authors, using semi-supervised machine learning methods, recorded and analyzed the conversations of 30 suicidal adolescents and 30 matched controls.

The results shown that the machines accurately distinguished between suicidal and nonsuicidal teenagers. In another 2016 study of narrative notes, it was found that "incorporating a simple natural language processing strategy improved the ability to estimate risk for suicide and accidental death" [21].

There have recently been studies dealing with the analysis of internet texts by individuals with suicidal tendencies (e.g., see [2]). Publicly accessible blogs or Facebook provide new data sources for the study of suicidal behavior [17], however computational methods have only been used in a small number of suicide communication studies [4]. As was noted above, effort has been made to identify suicide-related content [4]. Some papers analyzed blog posts of individuals who died as a result of suicide (mostly case studies).

In [17] 193 blog entries of a 13-year-old boy posted during the year prior to his suicide were analyzed using the Chinese version of LIWC (CLIWC). Some language patterns related to suicide are similar in both Chinese and English, such as the use of first-person singular pronouns. Progressive self-referencing appeared to be a primary predictive of suicide.

Thus, in studies of texts by suicidal individuals, there is a clear shift from descriptive papers, which document differences in texts by suicidal individuals and control groups, to text classification studies of texts written by suicidal/non-suicidal individuals which make use of various machine learning algorithms. Though earlier papers analyzed mainly literary texts, recently social media texts have become a major real consideration.

Features were largely those extracted using LIWC, but some other papers employ well-known text representations like standard bag of words, character/word n-grams, etc. The overall consensus is that such techniques are fairly suited to furthering the development of methods to identify individual's suicidal tendencies based on analyzing their texts.

As social media takeover our lives, it is essential that there are methods in place to identify suicidal tendencies using Internet based texts. Unfortunately, no studies are known of that classify texts of individuals into suicidal or non-suicidal categories using Internet based texts from individuals who committed suicide and from a control group. In our study we are seeking to change this.

Additionally, most suicide studies have focused on English materials, while there is a clear lack of research into suicidal individuals of other cultural backgrounds [17].

3 Materials and Methods

3.1 Materials

In order to address this issue, what we basically need is a corpus of texts by suicidal individuals and those from a control group [18].

The publicly available blog entries of a suicide case were used as data for this study. The corpus of Russian texts *RusSuiCorpus,* written by individuals who committed suicide, currently contains texts by 45 Russian individuals aged from 14 to 25. The total volume of the corpus is 200 000 words. All the texts are manually collected and represent blog posts by individuals who committed suicide (blogs from *LiveJournal*). The fact that suicides were actually committed was checked by analyzing friends' comments, media texts, etc. Blogging is a prevalent form of communication in expressing emotion and sharing information, therefore it was chosen.

Being as the texts contained different numbers of words from each author, all the texts of blogs were joined into a single text with the length of about 200,000 words. The resulting text was divided into segments of 200 words making a total of 1,000 texts.

Ethics approval was obtained from the Human Research Ethics Committee for Non-clinical Faculties at Voronezh State University, Russia.

Samples of natural written speech from 1,000 students of various Russian universities, all a part of RusPersonality [19] (the first corpus of Russian texts containing wide metadata with the authors' personal information), were used. The average text length was 200 words with a total word count of 198045 words (further on NSUIC). All the texts were stream-of-consciousness essays. The authors were instructed to write whatever comes to their minds first and do in the manner they would normally do to their friends on social media.

We deliberately avoided using blogs for comparison because our goal in this study was to compare samples of natural written speech from individuals who died as a result of suicide with those from individuals whose results of psychological testing, identifying auto-aggressive behavior (with suicide as its extreme), are known [20]. It is obviously challenging to obtain this information about bloggers. At this point of the analysis we deliberately chose to include texts displaying varying risks of auto-aggressive behavior into the NSUIC corpus.

In addition, even though they were a part of different genres, the compared texts are quite homogeneous: they are all samples of what is called natural written speech, which is generally unrehearsed (the control group was instructed to write whatever first comes to mind for 30 minutes). Before proceeding to designing the model, the text corpora were divided into a learning set (900 texts in NSUIC, 900 in SUIC) and test set (100 in SUIC and 100 in NSUIC) texts.

3.2 Text Processing

All the texts were labelled using LIWC 2007 with the Russian users' dictionary. Users' dictionaries were also compiled. We selected features that were not significantly dependent on the topic of a text on purpose.

Hence the following features were selected: general LIWC descriptor categories (words per sentence (WPS), percent of words longer than six letters (Sixltr), from dimension I STANDARD LINGUISTIC DIMENSIONS (total function words, total pronouns, adverbs, prepositions, etc.), II PSYCHOLOGICAL PROCESSES (Social processes with subcategories, affective processes with subcategories, cognitive processes with subcategories, perceptual processes with subcategories and so on), AllPunc (the proportion of all the punctuation marks in a text overall and each mark individually).

Users' dictionaries were also compiled according to the user manual:

- *Deictic,* a dictionary of demonstrative pronouns and adverbs (where 1 feature represents the proportion of words per the total word length of a text), a dictionary of intensifiers and downtowners
- *Intens* (2 features), a dictionary of perception vocabulary
- *PerceptLex* (1 feature), a dictionary of pronouns and adverbs describing the speaker - *Ego* (I, my, in my opinion; 1 feature), and a dictionary of emotional words
- *Emo* (negative and positive; 2 features). All in all, there are 85 features.

The users' dictionaries were compiled using available dictionaries and Russian thesauri. As a Russian dictionary that came with the software was a translation of a corresponding English dictionary and did not stand independent testing, there are doubts as to the semantic category of the second group and thus they have to be evaluated independently and objectively, so in turn we have to check it manually and make some corrections.

4 Results and Discussion

4.1 Mathematical Processing of the Results of Text Analysis

Mathematical processing of the results of text analysis was performed using the professional software SPSS 13.0.

Originally features with a frequency of less than 50% were excluded from the total list (in the SUIC as well as in the NSUIC corpora). The frequency of a text parameter (feature) is the percentage proportion of the number of non-zero values of a certain feature compared to the total number of the analyzed texts in the corpus the feature was computed for.

A preliminary data analysis of the SUIC and NSUIC corpora, using the Shapiro-Wilk test, showed that most of the features have a non-normal distribution. In order to identify statistically significant differences between the features from SUIC and identical features from NSUIC, a method of comparing dispersions of the analyzed sets was employed. For that a one-factor dispersion analysis was used (ANOVA).

Originally we made use of Kruskal-Wallis one way analysis of variance on ranks that is a non-parametric alternative to F-criterion in our one-factor dispersion analysis. In the Kruskal-Wallis criterion, average ranks of each of the groups are compared with the total rank computed using all of the data. The Kruskal-Wallis test for the significance level $p<0.05$ allowed us to identify differences in the medial values of the groups of the features from SUIC and NSUIC corpora.

After a significant difference between the overall groups has been identified, it is then advisable to compare the average values of the existing groups. This is called a posteriori criterion of pairwise comparison [30].

Table 1. Features selected for designing the model and their calculation values

Features /Values of tests	**Function words**	**Pronouns**	**Verbs**	**Preposition**	**Conjunctions**
Frequency, % SUIC\ NSUIC	99.9\100	99.6\99.71	99.8\99.81	99.31\99.81	99.8\99.33
Kruskal Wallis: H p-level	82.232 <0.001	18.886 <0.001	102.932 <0.001	72.381 <0.001	258.808 <0.001
Median: SUIC\ NSUIC	**49.500** 47.075	12.000 **13.010**	**15.000** 12.900	12.000 **12.745**	**12.000** 9.480
Tukey: q p-level	12.824 <0.001	6.145 <0.001	14.346 <0.001	12.029 <0.001	22.749 <0.001
Fit criterion of Kolmogorov-Smirnov SUIC\ NSUIC	0.033\0.057	0.057\0.018	0.055\0.03	0.056\0.031	0.062\ 0.018
Features	**Cognitive processes**	**Inclusive**	**Comparison**	**Space**	**Comma**
Frequency % SUIC\ NSUIC	99.8\99.9	99.5\98.66	99.7\100	99.5\99.71	99.7\99.33
Kruskal Wallis: H p-level	123.480 <0.001	188.843 <0.001	35.480 <0.001	18.994 <0.001	75.673 <0.001
Median: SUIC\ NSUIC	**21.500** 19.345	**8.500** 6.640	19.000 **20.085**	10.000 **10.695**	**12.000** 10.530
Tukey: q p-level	15.713 <0.001	19.431 <0.001	8.423 <0.001	6.162 <0.001	12.301 <0.001
Fit criterion of Kolmogorov-Smirnov SUIC\ NSUIC	0.041\0.024	0.065\0.02	0.04\0.034	0.06\0.03	0.066\ 0.027

For that the Tukey test with $p < 0.05$ was used, which is a modified Student criterion.

As a result, based on the data of two tests (Kruskal-Wallis and Tukey), we excluded the features that did not meet neither the Kruskal-Wallis nor Tukey criterion from those features which passed the original analysis, and thus whose values do not have statistically significant differences in the SUIC and NSUIC corpora.

In the next phase, as part of the mathematical analysis of the data, we checked whether the distributions of the features selected at the first two stages from SUIC and NSUIC corpora are normal. One of the most effective criteria for testing the normality of the distribution is the Kolmogorov-Smirnov test that is more efficient than the alternative criteria and is designed for large selections [30].

Using the standard procedures of the SPSS software as well as a visual analysis of data by designing distribution histograms of the features selected by this point of the analysis from both corpora, we determined whether the empirical distributions of the analyzed features were normal. For this we computed a fit criterion of Kolmogorov-Smirnov / Lillefors and compared it with the critical value typical of a data set of this size. The critical coefficient calculated according to [30] in the Kolmogorov-Smirnov test for a data set with a dimensionality of $n = 1000$ and significance level $p < 0.01$ is $Dc = 0.033$.

4.2 Features Selected for the Model

The calculation results indicated that almost all of the features that are normally distributed in NSUIC, are not in SUIC. A visual test of the distribution of such features using histograms showed that the experimental histograms are asymmetric. In order to account for the differences in the type of distribution of the same features in SUIC (S) and NSUIC (N) corpora, it was decided that the features with the fit criterion of Kolmogorov-Smirnov/Lillefors of no more than the critical value ($Dc = 0.033$) in one of the selections is excluded out of those features left following the first stage of the mathematical processing and those no more than the doubled value of the critical coefficient, i.e. 0.066 in the second selection.

Hence, in order to design a model to appropriately measure the likelihood of a text being written by an individual who committed suicide, only feature which met the following criteria were employed:

1. Highly frequent ones;
2. Those that passed the Kruskal-Wallis and Tukey tests;
3. Those that, according to the Kolmogorov-Smirnov/Lillefors criterion, have a distribution close to the normal one where $p<0.01$, considering the initial assumptions.

The features that were selected in accordance with all of the above criteria, including their descriptive statistics are identified in Table 1.

As the analysis indicated, out of 130 features only 10 met all of those requirements (see Table 1). We found that in texts by suicidal individuals, compared to those by individuals from a control group, there are more *function words, verbs, conjunctions, words describing cognition overall, inclusion words, more commas, fewer prepositions, more words describing comparison, words describing space* and *pronouns*.

It was previously shown [20] that texts produced by individuals with a greater likelihood of self-destructive behavior (suicide is its extreme) typically show less lexical diversity, fewer prepositions, more pronouns overall (and particularly personal ones), a higher coefficient of coherence (due to more conjunctions and deictic particles). Blogs also displayed fewer prepositions and more conjunctions in suicidal individuals, but also fewer pronouns. The difference between these values was also less significant than for the other features.

Let us assume that texts by suicidal individuals are more abstract, contextual, have fewer spatial references, which we think is indicative of their self-centeredness.

The following disparities, which were not included in the model due to these features failure of the normality test, are worth mentioning: texts by suicidal individuals have more negations, fewer words describing social and perceptive processes (particularly vision), more vocabulary from the LIWC group “Body”, fewer words for positive emotions, and more words for negative emotions.

All in all, the above data are consistent with the hypothesis that suicidal individuals are more self-centered and less focused on seeing the world around them. However, unlike a lot of studies of English texts, Russian texts were not found to contain more "I" pronouns compared to those in the control group. Thus our significant finding is not that there are more self-references, but instead that the authors are more self-centered and less focused on the world around them.

4.3. Designing the Model

Let us denote a set of elements (i=1.10) selected for designing the above model as S_{Si} and S_{Ni} (i=1..10). These elements are mean values of ten selected features of the text from SUIC and NSUIC corpora respectively.

We can safely say that S_S and S_N are a set of numerical values [S_{S1}… S_{S10}] and [S_{N1}… S_{N10}] whose elements are mean values of the features from SUIC and NSUIC corpora.

For a text under analysis, a set of values of the same 10 features S_T must be determined.

The deviation of a set S_T from a set of numerical values S_S and S_N that are typical of the texts from SUIC and NSUIC corpora respectively is determined as follows.

We calculate the deviation of a set of values $\mathbf{S_S}$ from a set of values $\mathbf{S_T}$ as follows:

$$\chi_S^2 = \frac{1}{n}\sum_i^n \frac{(S_{Si} - S_{Ti})^2}{S_{Si}}. \quad (1)$$

Similarly, let us determine the deviation of the distribution S_N from S_T:

$$\chi_N^2 = \frac{1}{n}\sum_i^n \frac{(S_{Ni} - S_{Ti})^2}{S_{Ni}}. \quad (2)$$

Let us assume that in order to determine which type (suicidal/non-suicidal) a particular individual is, it would suffice to χ_N^2 and χ_S^2. If $\chi_N^2 / \chi_S^2 > 1$ than the text under analysis would have been written by a suicidal individual.

4.4 Accuracy of the Model

Checking the model on the test selection showed that it was 71.5% accurate (according to the number of the correctly classified texts), with a baseline of 50%. Reported accuracy results are comparable with the ones obtained with English texts (70.6) [23].

5 Conclusion and Future Work

The suggested approach showed to be fairly accurate in classifying texts, even despite the fact that we selected the features maximally independent of the content (proportions of commas, function words, etc.), which indicates that natural language processing and data mining are promising for use in the identification of suicidal behavior. The proposed method certainly has some restrictions associated with the varying number of individuals in SUIC and NSUIC as well as a relatively small number of features.

There are plans to extend SUIC and the list of features using common tools for labelling texts, including those developed by the team of authors, as well as tools for syntactic labelling of texts. It is known that «the use of syntactic n-grams… gives good results when predicting personal traits» [29]. We are also looking into utilizing certain indicators of lexical diversity that, as shown in [31], are critical for identifying the psychological condition of authors in emotional auto-reflexive writing.

Additionally, there is more work that needs to be done to validate and adapt the Russian version of LIWC.

In the future we plan to compare the proposed model with well-known text representations like standard bag of words, character/word n-grams, as well as with different weighting schemes and different learning algorithms to appreciate and compare the real predictive potential of the proposed model. It would have been interesting to note how the 10-features model performs with respect to the full (130 features) model or with respect to models using standard feature selection techniques from the machine learning area [23, 28].

We also plan to perform a comparative analysis of texts by suicidal individuals with high and low

risks of autoaggressive behavior using the corpus *RusPersonality* [19, 20].

In this corpus the negative and positive texts have a slightly different composition: the positive texts come from a reduced group of authors with a sparse range of ages; the negatives texts, however, come from a more controlled group with a bigger base of authors. As such there are plans to compare the blog posts of suicidal individuals with those written by the control group.

Since "one of the main tasks of computational linguistics is to provide models for the development of applied systems with various kinds of automatic linguistic analysis" [31], what we ultimately are seeking to do is to design software for assessing risks of suicidal behavior based on the linguistic parameters of texts, which would be instrumental in online analysis of internet texts and could potentially send letters of warning to authors of texts or to their family and friends on social media. This research could also be applicable in the psychological assessment of suicide risk.

Acknowledgments

This research is supported by the grant of the Russian President awarded to young Russian scientists (Candidates of Sciences), project № MK-4633.2016.6 "Predicting Suicidal Tendencies of Individuals Based on their Speech Production".

References

1. **Baddeley, J.L., Daniel, G.R., & Pennebaker, J.W. (2011).** How Henry Hellyer's use of language foretold his suicide. *Crisis*, Vol. 32, No. 5, pp. 288–292.
2. **Barak, A. & Miron, O. (2005).** Writing characteristics of suicidal people on the Internet: A psychological investigation of emerging social environments. *Suicide and Life-Threatening Behavior*, Vol. *35,* Num. 5, pp. 507–524. DOI: 10.1521/suli.2005.35.5.507.
3. **Bertolote, J.M. & Fleischmann, A. (2009).** A global perspective on the magnitude of suicide mortality. *Oxford Textbook of Suicidology and Suicide Prevention, A Global Perspective*, Oxford University Press, pp. 91–99.
4. **Burnap, P., Colombo, W., & Scourfield, J. (2015).** Machine Classification and Analysis of Suicide-Related Communication on Twitter. *Proceedings of the 26th ACM Conference on Hypertext and Social Media*, Guzelyurt, TRNC, Cyprus, pp. 75-84. DOI: 10.1145/2700171. 2791023.
5. **Desmet, B. & Veronique, H. (2014).** Recognizing Suicidal Messages in Dutch Social Media. *Proceedings of Ninth international conference on language resources and evaluation* (*LREC 2014*), Reykjavik, Iceland, pp. 830–835.
6. **Dethlefs, N. & Schoene, A.M. (2016).** Automatic Identification of Suicide Notes from Linguistic and Sentiment Features. *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities* LaTeCH@ACL, Berlin, Germany, pp. 128–133.
7. **Fernández-Cabana, M., García-Caballero, A., Alves-Pérez, M., García-García, M., & Mateos, R. (2012).** Suicidal traits in Marilyn Monroe's fragments: An LIWC analysis. *Crisis*, Vol. 34, No. 2, pp. 124–130. DOI: 10.1027/0227-5910/a000183.
8. **Gomez, J.M. (2014).** Language technologies for suicide prevention in social media. *Proceedings of the Workshop on Natural Language Processing in the 5th Information Systems Research Working Days (JISIC 2014).* Quito, Ecuador, pp. 21–29.
9. **Guan, L., Hao, B., Cheng, Q., Yip, P.S., & Zhu, T. (2015).** Identifying Chinese Microblog Users with High Suicide Probability Using Internet-Based Profile and Linguistic Features: Classification Model. *JMIR Ment Health,* Vol. 2, Núm. 2:e17. DOI: 10.2196/mental.4227.
10. **Handelman, L. D. & Lester, D. (2007).** The Content of Suicide Notes from Attempters and Completers. *Crisis*, Vol. 28, No. 2, pp. 102–104. DOI: 10.1027/0227-5910.28.2.102.
11. **Homan, C., Johar, R., Liu, T., Lytle, M., Silenzio, V., & Ovesdotter-Alm, C. (2014).** Toward macro-insights for suicide prevention: Analyzing fine-grained distress at scale. *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology,* Baltimore, Maryland USA, pp. 107–117.
12. **Jones, N. & Bennell, C. (2007).** The Development and Validation of Statistical Prediction Rules for Discriminating Between Genuine and Simulated Suicide Notes. *Archives of Suicide Research: official journal of the International Academy for Suicide Research,* Vol. 11, No. 2, pp. 219–225. DOI: 10.1080/13811110701250176.
13. **Leenaars, A.A. (1988).** *Suicide notes: Predictive clues and patterns*. Human Sciences Press, New York, USA.

14. **Lester, D. (2014).** *The "I" of the storm. Understanding suicidal mind*. De Gruyter, Warsaw, Poland.
15. **Lester, D. & McSwain, S. (2010).** Poems by a suicide: Sara Teasdale. *Psychological Reports,* Vol. 106, No. 3, pp. 811–812. DOI: 10.2466/pr0.106.3.811-81.
16. **Lester, D. & McSwain, S. (2011).** A text analysis of the poems of Sylvia Plath. *Psychological Reports,* Vol. 109, No. 1, pp. 73–76. DOI: 10.2466/09.12.28.PR0.109.4.73-7.
17. **Li, T.M., Chau, M., Yip, P.S., & Wong, P.W. (2014).** Temporal and Computerized Psycholinguistic Analysis of the Blog of a Chinese Adolescent Suicide. *Crisis*, Vol. 35, No. 3, pp. 168–75. DOI: 10.1027/0227-5910/a000248.
18. **Litvinova, T. (2016).** Corpus studies of speech of individuals who committed suicides. *Russian Linguistic Bulletin*, Vol. 7, No. 3, pp. 133–136. DOI: 10.18454/RULB.7.16.
19. **Litvinova, T., Litvinlova, O., Zagorovskaya, O., Seredin, P., Sboev, A., & Romanchenko, O. (2016).** "RusPersonality": a Russian corpus for authorship profiling and deception detection. *Proceedings of International FRUCT Conference on Intelligence, Social Media and Web (ISMW FRUCT 2016)*, Saint Petersburg, Russia, pp. 1–7. DOI: 10.1109/FRUCT. 2016.7584767.
20. **Litvinova, T., Zagorovskaya, O., Litvinova, O., & Seredin, P. (2016).** Profiling a set of personality traits of a text's author: a corpus-based approach. *Lecture Notes in Computer Science,* Vol. 9811, pp. 555–562. DOI: 10.1007/978-3-319-43958-7_67.
21. **McCoy, Th.H., Castro, V.M., Roberson, A.M., Snapper, L.A., & Perlis, R.H. (2016).** Improving Prediction of Suicide and Accidental Death after Discharge from General Hospitals With Natural Language Processing. *JAMA Psychiatry,* Vol. 73, No. 10, pp. 1064–1071. DOI: 10.1001/j amapsychiatry.2016.2172.
22. **Montes y Gómez, M., López-López, A., & Gelbukh A. (1999).** Text mining as a social thermometer. *Proc. Text Mining workshop at 16th International Joint Conference on Artificial Intelligence (IJCAI'99),* Stockholm, Sweden, pp. 103–107.
23. **Mulholland, M. & Quinn, J. (2013).** Suicidal Tendencies: The Automatic Classification of Suicidal and Non-Suicidal Lyricists Using NLP. *Processing of International Joint Conference on Natural Language Processing*, Nagoya, Japan, pp. 680–684.
24. **Osgood, C.E. & Walker, E.G. (1959).** Motivation and language behavior: A content analysis of suicide notes. *The Journal of Abnormal and Social Psychology,* Vol. 59, No. 1, pp. 58–67. DOI: http://dx.doi.org/10.1037/ h0047078.
25. **Pennebaker, J.W., Chung, C.K., Ireland, M., Gonzales, A., & Booth, R.J. (2007).** *The development and psychometric properties of LIWC2007*. Texas, USA.
26. **Pestian, J.P., Grupp-Phelan, J., Bretonnel-Cohen, K., Meyers, G., Richey, L.A., Matykiewicz, P., & Sorter, M.T. (2016).** A Controlled Trial Using Natural Language Processing to Examine the Language of Suicidal Adolescents in the Emergency Department. *Suicide Life Threat Behav,* Vol. 46, pp. 154–159. DOI:10.1111/sltb.12180.
27. **Pestian, J.P., Matykiewicz, P., Linn-Gust, M., South, B., Uzuner, O., Wiebe, J., Cohen, K.B., Hurdle, J., & Brew, Ch. (2012).** Sentiment analysis of suicide notes: A shared task. *Biomedical Informatics Insights,* Vol. 5, No. 1, pp. 3–16. DOI: 10.4137/BII.S9042.
28. **Pestian, J., Nasrallah, H., Matykiewicz, P., Bennett, A., & Leenaars, A. (2010).** Suicide note classification using natural language processing: a content analysis. *Biomed Inform Insights,* Vol. 3, pp. 19–28.
29. **Posadas-Durán, J.P., Markov, I., Gómez-Adorno, H., Sidorov, G., Batyrshin, I., Gelbukh, A., & Pichardo-Lagunas, O. (2015).** Syntactic N-grams as Features for the Author Profiling Task. *Notebook for PAN at CLEF 2015,* Toulouse, France.
30. **Salkind, N. (2007).** *Encyclopedia of Measurement and Statistics*. Sage Publications Inc., California, USA. DOI: 10.4135/9781412952644.
31. **Sidorov, G. & Castro-Sánchez, N.A. (2006).** Automatic Emotional Personality Description using Linguistic Data. *Research in Computing Science,* Vol. 20, pp. 89–94.
32. **Stirman, W. & Pennebaker, J. (2001).** Word use in the poetry of suicidal and non-suicidal poets. *Psychosomatic Medicine*, Vol. 63, No. 4, pp. 517–522.
33. **Villatoro-Tello, E., Juarez-Gonzalez, A., Escalante, H.J., Montes-y-Gomez, M., & Villaseñor, L. (2012).** A Two-step Approach for Effective Detection of Misbehaving Users in Chats. *Paper presented at the meeting of the CLEF (Online Working Notes/Labs/Workshop),* Rome, Italy.
34. **World Health Organization (2014).** *Preventing suicide: A global imperative.* WHO Publications, Luxemburg.

Article received on 14/11/2016; accepted on 17/03/2017.
Corresponding author is Tatiana A. Litvinova.

MathIRs: Retrieval System for Scientific Documents

Amarnath Pathak[1], Partha Pakray[1], Sandip Sarkar[2], Dipankar Das[3], Alexander Gelbukh[4]

[1] National Institute of Technology Mizoram, Aizawl, India

[2] Hijli College, Kharagpur, India

[3] Jadavpur University, Kolkata, India

[4] CIC, Instituto Politécnico Nacional, Mexico City, Mexico

{amar4gate, parthapakray, sandipsarkar.ju, dipankar.dipnil2005}@gmail.com, www.cic.ipn.mx/∼gelbukh

Abstract. Effective retrieval of mathematical contents from vast corpus of scientific documents demands enhancement in the conventional indexing and searching mechanisms. Indexing mechanism and the choice of semantic similarity measures guide the results of Math Information Retrieval system (MathIRs) to perfection. Tokenization and formula unification are among the distinguishing features of indexing mechanism, used in MathIRs, which facilitate sub-formula and similarity search. Besides, the scientific documents and the user queries in MathIRs will contain math as well as text contents and to match these contents we require three important modules: Text-Text Similarity (TS), Math-Math Similarity (MS) and Text-Math Similarity (TMS). In this paper we have proposed MathIRs comprising these important modules and a substitution tree based mechanism for indexing mathematical expressions. We have also presented experimental results for similarity search and argued that proposal of MathIRs will ease the task of scientific document retrieval.

Keywords. Natural language processing, information retrieval, MathIRs, indexing.

1 Introduction

Tremendous increase in scientific documents repositories and enormous amount of queries intended for retrieving such documents, necessitate the requirement of developing specialized tools and techniques which could handle such documents.

Scientific documents, unlike normal text documents, contain mathematical expressions and formulas which posses different form and meaning in different contexts. Several distinct appearing math expressions may actually turn out to be semantically similar and equally probable is the other situation where a given expression may resolve to several different interpretations in different contexts. For instance, the mathematical expression "h(x)" may get interpreted as function 'h' having 'x' as argument or a variable 'h' multiplied to another variable 'x'. Adding further, "$(x + y)^{1/2}$" and "$\sqrt{x+y}$ " look syntactically different but semantically they are the same. Process of indexing takes care of such visual and semantic ambiguities, present in mathematical expressions, by using a universal canonical representation for all the expressions followed by tokenization and storing of tokens in the index database to facilitate their retrieval on users' demand. In fact, the effective search for an expression inside a document and the accuracy of retrieval is determined by proficiency of the approach

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2743

adopted for indexing and semantic similarity measures used to compare query expression with the mathematical expressions inside documents. Thus, the task of mathematical information retrieval boils down to designing an effective indexing technique and similarity matching technique which could supplement and escalate the searching process.

Five major steps in indexing mathematical documents have been identified as: Preprocessing, Canonicalization, Tokenization, Structural Unification and Representation of Math for Indexing [23, 14]. However, these steps have undergone several changes and some intermediate steps have also been introduced as the indexing process has evolved with time.

Measuring semantic similarity between sentences or short texts is equally important for MathIRs and finds its application in many other natural language processing tasks such as machine translation, opinion mining, text summarization, and plagiarism detection. Need for semantic similarity detection at a very early stage is justified by the fact that the underlying documents may contain synonym, hyponym or hypernym of the query term rather than the exact query term. Comparatively less number of relevant documents will be retrieved, if semantic similarity between query expression and the indexed terms of the document is not detected by the searcher module.

Taking into account above facts and requirements, we have proposed an enhanced modular architecture for MathIRs which contains separate modules for semantic similarity detection between (text/math) contents in the query and (text/math) contents inside documents. We have also proposed an enhanced substitution tree based indexing mechanism which is presumed to overcome shortcomings of a similar indexing technique [18].

Rest of the paper is organized as follows: Section 2 describes past works on math information retrieval in general and indexing in particular. Section 3 briefs about semantic textual similarity. Section 4 details proposed system architecture for MathIRs and Section 5 describes features for similarity modules used in MathIRs. Section 6 contains experimental results for similarity modules and a comparison between winner score and our system's best score. Section 7 contains detailed discussion on working principle of indexing mechanism used in MathIRs and dataset and query set description for MathIRs. Section 8 concludes the paper and points directions for future research.

2 Related Work

Indexing of mathematical expressions, contained inside scientific documents, was first attempted using a substitution tree based method – node of the tree referring to a substitution and the leaves of the substitution tree referring to actual terms that have been indexed [3]. Moving along a path in the tree and applying substitution yields the indexed term. Such an indexing method reduces memory requirement, owing to the fact that we only need to store the substitutions and not the actual terms. It also promises paced indexing of terms and with such an indexing method in hand, the task of searching boils down to performing tree traversal in depth first fashion.

A later work identified several areas in which normalization is applicable for MathML[1] documents [8]. Documents converted to MathML usually contained MathML and XML errors, important of which include malformed tags, improper number of attribute values and child counts. An error correcting parser has been designed to address these issues which produces well formed XML documents by removing unrecognized elements, illegal attribute values and inserts missing entities, wherever required. Ambiguity in MathML documents has been further diminished by choosing single canonical representation for mathematical expressions and decimal numbers having multiple popular representation conventions.

An enhancement to the full text search engines to facilitate search for mathematical content, has been proposed in [9]. A normal text search engine uses a recognizer which recognizes text and parses it to sentences and words. Likewise, use of formula recognizer has been proposed which could recognize mathematical formulas

[1] https://www.w3.org/Math/

and convert it to MathML form, Presentation or Content[2]. Mathematical formulae are stored in postfix notation so as to avoid use of parenthesis and to facilitate similarity detection. Mathematical expression "(a+b)-(c+d)", for example, is first converted to a postfix notation "ab+cd+-" followed by storing of tokens "ab+", "cd+" and "–" in the index database. Use of postfix notation for indexing offers flexibility of searching similar expressions.

Design principle of Math Indexer and Searcher (MIaS) system [5] get motivation from the fact that conventional search engines are incapable and inefficient at handling math information retrieval from Digital Mathematics Library. To build an MIaS, different mathematical representations such as TEX/LATEX, MathML, OpenMath[3] and OmDoc[4] have been explored and their advantages and disadvantages have been discussed. Moreover, existing Math search engines have been compared on the grounds of indexing core, internal representation of document, used converters, query languages and query. Unfortunately, none of the existing systems could offer search for exact mathematical expression, equivalent expressions with different notation, sub-expressions and mixed mathematical contents.

Math Indexer and Searcher (MIaS) system was later implemented to facilitate indexing and searching of mathematical documents [23]. MIaS facilitates searching of not only the complete expression but also the sub-expressions and similar expressions by exploiting techniques of tokenization and structural unification respectively. Tokenization, a process of plucking out sub-expressions from a given input expression, is accomplished by traversal of the expression tree encoded using presentation MathML. Tokenization is followed by unification to generate generalized expressions for tokens and the same is achieved using ordering, variable unification and constant unification, whereby all the variables and constants present in the expression are substituted by single unified variable and constant.

Indexing process has been further delved into and the processes of ordering, variable unification and constant unification have been exemplified [24]. Ordering introduces an alphabetical order between arguments of an operator, whereas unification avoids risk treating two or more similar expressions, which differ only in terms of variable names or constant values, as different. Expressions, "$a+b^a$" and "$x+y^x$", are semantically same but differ in terms of variable names. Proposed system substitutes variable names with "ids", thus normalizing both these expressions to expression, "$id1 + id2^{id1}$" and causing the two expressions to match.

Canonicalization is act of transforming semantically similar MathML expressions into one common form [1]. Different ways of canonicalizing the MathML expressions have been discussed and detailed, salient ones include:

(i) removing unnecessary elements and attributes which only contribute to the appearance and not to the semantics,

(ii) minimizing the number of <mrow> elements,

(iii) subscript and superscript handling, and

(iv) avoiding use of entity "&#x2061" for function name.

Pre-processing, canonicalization and representation of math for indexing are salient issues related to indexing process of Math Indexer and Searcher (MIaS) [13]. Indexing process involves comparison of expressions to be indexed with the pre-indexed tokens and assigning weights to the token based on percentage similarity. The weights associated with the tokens later help in ranking the documents.

Design of MIaS has been further extended by incorporating modules for structural and operator unification [14] – two important features which were absent in the previous architectures of MIaS. An open source tool named MathML Unificator, is used to generate series of structurally unified variants of the input expression. Original expression and the structurally unified variants are added to the index database afterwards. However, appropriate care is taken not to overfill the index database with all possible variants of the input expression.

[2] https://www.w3.org/TR/WD-math-980106/chapter2.html
[3] http://www.openmath.org/overview/technical.html
[4] http://www.omdoc.org/

3 Semantic Textual Similarity

Retrieval of specific information requires an intelligent retrieval mechanism which is suitable in some specific domain. Many researchers have used semantic based approaches in information retrieval system to enhance performance of their system. Word based similarity is the primary similarity measure for measuring similarity between sentences. Similarity between two texts can be measured using four distinct similarity measurement approaches: lexical similarity, syntactic similarity, WordNet similarity and distributed semantic [6, 10]. Lexical similarity measure is concerned with words and is normally suitable for measuring semantic similarity of all languages [16, 17], whereas syntactic similarity measure is predominantly concerned with similarity of sentence structure. WordNet is an external resource which has been used to find similarity between sentences. On the other hand, distributed semantic similarity is primarily concerned with minimizing the difference between syntactic and semantic similarity. Machine learning tool (METEOR) and Levenshtein ratio find their application in measuring semantic similarity between sentences [15].

4 System Architecture

Our system architecture, shown in Figure 1, is enriched by three modules: Indexing, Math Processing and Similarity.

4.1 Indexing Module

Apache Lucene[5] is to be used for the purpose of indexing which is a free and open-source information retrieval software library, supported by the Apache Software Foundation and released under the Apache Software License. It performs operations such as indexing, reverse indexing and analysis on the documents fed to it, thus making the documents searchable and easy to retrieve.
Three core components of Lucene comprise Document Analyzer, Tokenizer and IndexWriter. Document analyzer analyzes the documents to recognize their content. Tokenizer separates the content into several small components called tokens which are written to the index database using IndexWriter. Writing to index is based upon positional information of the tokens in the document, a technique known as full inverted indexing. Lucene based search engines have been enabled to recommend similar documents based on current search interest of the user. In the recent release of Apache Lucene, few changes have been made with respect to fuzzy querying mechanism and query parser. Apache Lucene 6.4.0 is the latest version which is characterized by enhanced features of spellchecking, hit highlighting, advanced analysis and tokenization capabilities.

4.2 Math Processing Module

Mathematical documents originating from different sources attain heterogeneous forms such as .pdf, .tex etc. and hence it becomes necessary to pre-process the documents, converting them to one common MathML form. Automated tools such as Infty Reader[6], MaxTract[7], Tralics[8] and LatexML[9] are used to accomplish this task.

MathML, a form of XML concerned with encoding syntax and semantics of mathematical expressions, has got two major forms: presentation and content[10]. Presentation MathML markup comprises 30 elements accepting around 50 attributes. Most of these elements are concerned with syntax or layout of representation and can be categorized into three broad categories:

(i) Script elements: <msub>, <munder> and <mmultiscripts>,

(ii) Layout elements: <mrow>, <mstyle> and <mfrac>,

(iii) Table elements.

[5] https://lucene.apache.org/

[6] http://www.inftyreader.org/

[7] https://www.cs.bham.ac.uk/research/groupings/reasoning/sdag/maxtract.php

[8] http://www-sop.inria.fr/marelle/tralics/

[9] http://dlmf.nist.gov/LaTeXML/

[10] https://www.w3.org/TR/WD-math-980106/chapter2.html

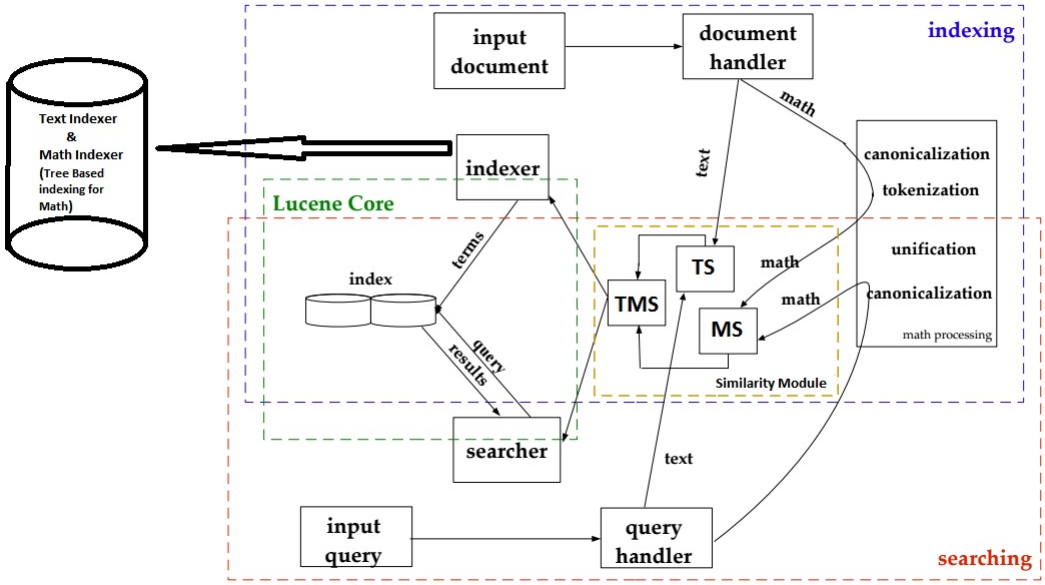


Fig. 1. System Architecture

```
<mrow>
  <msup>
    <mfenced>
      <mrow>
        <mi> a </mi>
        <mo> + </mo>
        <mi> b </mi>
      </mrow>
    </mfenced>
    <mn> 2 </mn>
  </msup>
</mrow>
```

Fig. 2. $(a+b)^2$ represented using Presentation MathML

```
<apply>
  <eq/>
  <ci>y</ci>
  <apply>
    <power/>
    <apply>
      <plus/>
      <ci>a</ci>
      <ci>b</ci>
    </apply>
    <cn>2</cn>
  </apply>
</apply>
```

Fig. 3. $(a+b)^2$ represented using Content MathML

Mathematical expression, "$(a+b)^2$" expressed using presentation MathML is shown in Figure 2.

Content markup comprises about 120 elements which accept around one dozen attributes and majority of them are used to represent operators, relations and named functions. <apply> is among most important content MathML elements, primarily used to apply functions to expressions and perform arithmetic and logical operations on collection of arguments. Mathematical equation, $y = (a+b)^2$, expressed using content MathML is shown in Figure 3.

Pre-processing is followed by canonicalization in which MathML expressions are normalized to single canonical form. Different ways of canonicalization include [1] :

(i) **Removing unnecessary elements and attributes:** Many elements such as <mspace>, <mpadded>, <mphantom> and <malignmark> contribute only to the appearance of documents and not to the semantics, thus being the preferred candidates for removal.

(ii) **Minimizing the number of <mrow> elements:** <mrow> element is used for grouping

other elements. Number of sub-expressions present in an expression can be found by counting the number of <mrow> elements. However, if a parent expression contains only one child expression then use of <mrow> element is redundant and the same should be omitted and avoided.

(iii) **subscript and superscript handling:** Use of <msubsup> element should be discouraged and <msub> and <msup> elements should be used instead. If both the functionalities are required then <msub> should be placed inside <msup>.

(iv) **Applying Functions:** Function name should be placed in the <mo> element and for the arguments we can use <mfenced> elements or parenthesis or both. Other ways of representing functions such as use of Entity '&#x2061' and use of function name inside <mi> element should be discouraged for the sake of avoiding ambiguity.

Most often, a user might be interested in search for a sub-expression rather than the complete expression. Tokenizer facilitates such functionality by plucking out sub-expressions from the input math expression and indexing them separately at a later stage. Consider a user querying for the expression "$\sqrt{(x+y)}$ " and a document 'd' containing Equation 1:

$$y = \frac{\sqrt{(x+y)}}{\sqrt{y}}. \tag{1}$$

MathIRs will not retrieve document 'd' for the given query expression, if Equation 1 has not been tokenized into tokens "$\sqrt{(x+y)}$ " and "$\sqrt{y}$ ".
Further, a user might probably be interested in similarity search rather than complete expression search or sub-expression search. This may correspond to a situation where expressions look syntactically different but semantically they are the same. Unifactor copes up with such an issue by generating all meaningful structural variants of expressions and sub-expressions and indexing them separately by assigning appropriately lowered weights to the structural variants [14]. Consider, for example, an input expression of the form "$x^2 + \frac{\sqrt{x}}{z}$". Different structural variants of this expression will be:

(i) $\odot^2 + \frac{\sqrt{x}}{z}$,

(ii) $\odot^2 + \frac{\odot}{z}$.

These variants, when indexed, are helpful in searching an expression which is similar to original expression but not exactly same. With this added functionality, if a user searches for the expression "$x^2 + \frac{\sqrt{x}}{z}$" then the document containing expression "$x^2 + \frac{\sqrt{x}}{z}$" is likely to get retrieved.

Final indexing of tokenized and structurally unified math formulas and sub formulas is proposed to be done using a substitution tree based indexing mechanism. An index tree is going to be used to systematically store math expressions. When a new expression is encountered, the existing tree is searched for the expression and if absent, extra nodes are appended in the tree to accommodate the expression. However, the tree need not be modified if parent expression of the given expression has already been indexed. Such an indexing mechanism is presumed to minimize our memory requirement by eliminating the need to store all the structurally unified variants of an expression.

4.3 Importance of Similarity Modules

Similarity modules constitute essential components of proposed architecture because user query may contain text and math contents and to retrieve documents it becomes necessary to compare (text/math) contents in the query with (text/math) contents of the underlying documents. These mandatory comparisons, if not performed, will result in retrieval of relatively less number of relevant documents.

4.4 Similarity Modules

The idea of text-text, math-math and text-math similarity matching is driven by the system architecture of MIaS [11], containing separate modules for Text-Text Entailment (TE), Math-Math Entailment (ME) and Text-Math Entailment (TME). A text or math query serves as 'hypothesis' and the content present in scientific documents serves as 'text'. An ME module, for example, will retrieve a document if some part of its mathematical content entails the mathematical query hypothesis. On the same grounds, we have proposed an architecture comprising TS, MS and TMS modules which perform text-text, math-math and text-math similarity matching for scientific document retrieval.

Our first module is TS module which defines the degree of similarity between text query and the text in document. The similarity between text and text can be computed using four approaches: lexical, syntactic, wordnet and distributed semantic. In knowledge-based approach the query is represented using synonym, hypernyms and hyponyms. Detailed description about the feature set and the experiment is provided in section 5 and section 6 respectively.

Our second module is MS module which finds the similarity between math query and the the document containing mathematical expressions. MS module helps in comparing canonicalized query expressions with canonicalized and structurally unified math expressions present in the document. A query expression might not match the exact math expression inside a document but may match one of its structurally unified versions which eventually results in retrieval of the underlying document. For example, the query expression "$a + \frac{x}{y}$" will match one of the structurally unified versions of "$a + \frac{\sqrt{x}}{y}$" and our MS module will retrieve the corresponding document.

Our third and final module is TMS module which finds similarity between text query and the mathematical expressions inside the documents. Text-Math similarity matching can be made feasible by assigning valid names to math expressions at the time of indexing. Text query will be matched to indexed names of mathematical expressions and if any suitable match is found, the corresponding document gets retrieved.

5 Features for Similarity Module

5.1 Cosine Similarity

Cosine similarity is a well known similarity measurement feature for finding similarity between two sentences. For our purpose we can represent each sentence using vectors. Similarity score of two sentences can be computed using dot product of corresponding vectors divided by the product of length of vectors.

Cosine similarity for two sentences, represented using vectors S_1 and S_2, can be computed using Equation 2:

$$S = \frac{S_1.S_2}{||S_1||.||S_2||}. \tag{2}$$

5.2 Levenshtein Ratio

Levenshtein distance, also known as edit distance, can be computed using three basic operations: insertion, deletion and substitution. To improve system performance, we can use normalization process to convert Levenshtein distance into Levenshtein ratio. Suppose 'a' and 'b' are two strings, then the Levenshtein ratio can be computed using the Equation 3:

$$\mathrm{EditRatio}(a,b) = 1 - \frac{\mathrm{EditDistance}(a,b)}{|a| + |b|}. \tag{3}$$

5.3 METEOR

METEOR is normally used for machine translation and it tries to include grammatical and semantic knowledge. METEOR score depends on outcome of types of matching: exact matching, stem matching, synonym matching and paraphrase matching [4]. Exact matching counts the similar words present in the hypothesis and referent part.

Stem matching matches words after they have been stemmed to their root words. Words such as organize, organizes, and organizing stem to the same root word 'organize' and hence they are said to match. Third module defines matching in terms of synonymy, meaning that two or more given

words are said to match if they are synonyms. The last kind of matching is based on the paraphrase matching between the hypothesis and reference translation.

5.4 Word2vec

In distributed semantic similarity approach, each word is represented using vector [12]. The main advantage of distributed semantic similarity is its ability to capture semantic representation of words after analyzing large corpora of text.

Distributed semantic similarity is based on the hypothesis: the meaning of a word is represented using the surrounding of that word [7]. Word2vec is a well-known model to produce word embedding and can be implemented using Gensim framework[11].

5.5 Jaccard Similarity

Suppose S and T are two sets containing words from two sentences. Jacardian similarity of the two sets is expressed using Equation 4:

$$JaccardSimilarity = \frac{|S \bigcap T|}{|S \bigcup T|} = \frac{|S \bigcap T|}{|S| + |T| - |S \bigcap T|}. \quad (4)$$

5.6 WordNet

WordNet is a lexical resource composed of synsets and semantic relations. Synset is a set of synonyms representing distinct concepts. Synsets are linked with basic semantic relations like hypernymy, hyponymy, meronymy, holonymy, troponymy, etc. and lexical relations like antonymy, gradation etc.

[11] https://radimrehurek.com/gensim/.

5.7 TakeLab

We have implemented TakeLab feature which was made available[12]. TakeLab system made use of various semantic features based on lexical, syntactic and external resources. The TakeLab 'simple' system obtained 3rd place in overall Pearson correlation and 1st for normalized Pearson in STS-12.

The source code was used to generate all its features namely *n-gram overlap, WordNet-augmented word overlap, vector space sentence similarity, normalized difference, shallow NE similarity, numbers overlap*, and *stock index features*.

TakeLab system predicts semantic similarity of two sentences using SVM regression approach which required full LSA vector space models provided by the TakeLab team. The word counts required for computing Information Content were obtained from Google Books Ngrams.

6 Text Similarity Module: Comparison between Winner Score and Baseline Score

Text based similarity module constitutes an important component of any Information Retrieval (IR) system. Sometimes relevant documents can't be retrieved, the reason being the absence of knowledge based similarity or lexical similarity between the text in the query and the text in document. The choice of an efficient and effective similarity measure for measuring text based similarity can boost the performance of our proposed system.

Our text similarity modules have been evaluated using SemEval 2016 dataset[13]. To find the semantic similarity, we have used two different approaches: lexical and distributed semantic similarity. Final semantic score has been computed after training using feed-forward neural

[12] http://takelab.fer.hr/sts.

[13] http://alt.qcri.org/semeval2016/task1/index.php?id=data-and -tools.

network and layer-recurrent network which are available in Matlab toolkit [14].

Five types of SemEval-2016 test datasets were used in monolingual sub-task: News, Headlines, Plagiarism, Post-editing, Answer-Answer and Question-Question.

The comparison between winner score[15] and our system's best score is given in Table 1. Besides, Table 1 shows that our system gives better result than TakeLab system which was the winner system in 2012. For plagiarism and question-question datasets, our system gave better results using distributional approach whereas best scores in answer-answer and headline datasets were obtained using lexical approach which is based on unigram matching ratio, Levenshtein ratio and METEOR.

7 Proposed Method for MathIRs

7.1 Motivation

For MathIRs, we have proposed substitution tree based indexing method which is an enhancement of the previous works in this regard [2, 18]. Substitution tree indexing was introduced by Graf where each node of the index tree represents predicates. Non-leaf nodes represent generalized substitution variables whereas leaf nodes represent specific ones. Depth first traversal of tree and applying substitutions in parallel yield specific predicates. Underlying idea has been further extended and the substitution trees have been used to represent mathematical expressions [18]. Working principle is exactly the same except for the fact that nodes in substitution trees now represent an expression in place of predicate. Each node of substitution tree is a Symbol Layout Tree (SLT) corresponding to some mathematical expression and edges of the SLT labeled as: NEXT, ABOVE and BELOW, indicating the terms next, super-scripted and sub-scripted to the given term, respectively.

[14] http://nl.mathworks.com/help/nnet/ref/feedforwardnet.html, http://nl.mathworks.com/help/nnet/ref/trainrp.html, http://nl.mathworks.com/help/nnet/ug/design-layerrecurrent-neural-networks.html.

[15] http://alt.qcri.org/semeval2016/task1/index.php?id=results.

Further simplification involves representing the nodes using substitutions to be applied while traversing the tree from root to leaf. Existing substitution tree indexing method offers minimum memory requirement benefits as it only stores the substitution at each node and not the complete terms. Memory is an important and deciding parameter while trying to build index databases and substitution tree indexing being best at minimizing memory requirements, motivates us to further investigate its structure. Besides, the proposed system in [18] suffers from obvious disadvantages of indexing only Latex formulas and representing the expressions purely based upon their syntax. Aim of our proposed system is to overcome such shortcomings and to further delve into structure of substitution trees for exploring other possible advantages.

7.2 Working Principle

The proposed system is characterized by underlying principles of SLT, substitution [18] and an improved indexing mechanism. SLT is 5 tuple (t, A, B, {x1,x2,...xn},N) where 't' refers to term, 'A' refers to above expression (or super-scripted expression to t), 'B' refers to below expression or (sub-scripted expression), "x_1,x_2,....x_n" refer to arguments and 'N' refers to next term. Any of these five tuples, if absent, are represented by 'ϕ'. Shorthand notation, (t) is used for SLTs containing only term and (t,N) for SLTs containing term plus next term, to avoid unnecessary ϕ entries. For example, "$x^2 + y_1$" represented using SLT tuples is shown in Figure 4.

$$(x, (2), \phi, \phi, (+, (y, \phi, (1), \phi, \phi)))$$

Fig. 4. SLT tuples for, $x^2 + y_1$

However, for the purpose of indexing mathematical expressions, SLTs are replaced by their substitution counterparts. In subsubsection 7.2.3, we have explained the difference between proposed indexing method and the indexing method used in [18], through an illustrative example.

Table 1. Performance comparison with other Systems on Monolingual Dataset

Corpus	Winner Score (Samsung Poland NLP Team)	TakeLab Score	Our System		Baseline Score
			Best Score	Approach	
answer-answer	0.69235	0.43301	**0.57454**	Lexical	0.48023
headlines	0.82749	0.55870	**0.72567**	Lexical	0.70749
plagiarism	0.84138	0.75086	**0.79877**	Distributional	0.76752
postediting	0.83516	0.64140	**0.81060**	Cosine	0.77196
question-question	0.68705	0.48409	**0.63612**	Distributional	0.43751

7.2.1 Dataset Description for MathIRs

We evaluated our system on the dataset from NTCIR-12 MathIR task[16]. Two corpora were used for the evaluation of our system: arXiv corpus and Wikipedia Corpus. arXiv corpus has been written by technical users assuming some level of mathematical understanding from users whereas Wikipedia corpus contain mathematical formulas written for normal users.

(i) **arXiv corpus:** arXiv corpus contains 105120 scientific articles converted from LaTeX to HTML+MathML format. Technical document from several arXiv categories[17] such as math, cs, physics:math-ph,stat, physics:hep-th are contained in arXiv corpus. Each document is divided into paragraphs resulting in 8,301,578 search units equating to 60 million math formulae.

(ii) **Wikipedia corpus:** Wikipedia corpus contains 319,689 articles from English Wikipedia converted into simpler XHTML format with images removed. These articles are not split into smaller documents with 10 % of the sampled article containing <math> tag and 90 % of the article containing complete English without <math> tag. There are around 590,000 formulas in this corpus encoded using presentation and content MathML. This corpus having uncompressed documents is 5.15 GB in size.

[16]http://ntcir-math.nii.ac.jp/.

[17]https://arxiv.org/list/math-ph/1101.

7.2.2 Query Set Description for MathIRs

arXiv corpus query set comprises 29 queries, 5 of them having only formula query whereas rest 24 having formula query plus keyword(s) and Wikipedia corpus query set comprises 30 queries – 3 of them having only formula whereas rest 27 having formula plus keyword(s)[18]. Queries are characterized and identified by unique IDs assigned to each of them. MathIRs is evaluated by comparing results of retrieval with Gold Dataset and checking values of the evaluation parameters obtained as result of comparison. Mean Average Precision (map), P_5 (Precision@5), P_10 (Precision@10) and bpref are some of the commonly used evaluation parameters.

7.2.3 Proposed Indexing Method

As per the previous indexing method [18], the expression to be indexed is first normalized by introducing an absolute ordering of variables present in the expression – for example, replacing 'x' by '$*_1$' , 'y' by '$*_2$' and so on. The expression to be inserted, is compared with the root and if match is successful then the indexing process is recursively called upon the children. If at any node, mismatch between the existing substitution at a node and the syntax of expression is found, a new node having a new substitution strategy is created to accommodate the expression. 'ϕ' symbol is used as root if no generalization is possible for all the expressions. Consider , for example, that expressions shown in Figure 7 are to be indexed and they appear in the documents in the same order as they have been written.

[18]http://ntcir-math.nii.ac.jp/

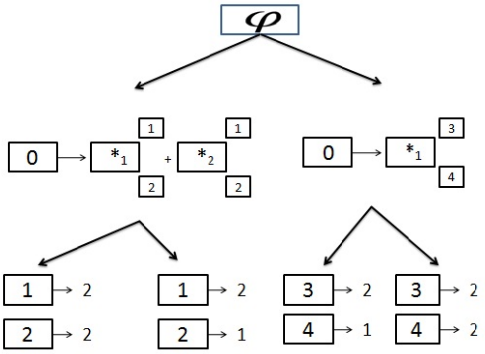


Fig. 5. Substitution Tree for expressions created using previous indexing method

In normalized forms of these expressions, 'x' is replaced by '$*_1$' and 'y' is replaced by '$*_2$'. Complete substitution tree for these expressions is shown in Figure 5.

The obvious disadvantage worth noticing about this indexing method is its specificity. The method treats two expressions, "$x_1^2 + y_1^2$" and "$x_2^2 + y_2^2$" as different although semantically they are the same – both representing sum of squares of two variables. Such an extreme specificity might not be desired in MathIRs. The user query "$x_1^2 + y_1^2$" might probably benefit from all the documents containing expressions similar to "$x_1^2 + y_1^2$". Such a shortcoming is probably an outcome of the syntax based approach of indexing which overlooks semantic similarity of two expressions, while indexing. Moreover, it only indexes math formulas written using LatexML, a serious bug which needs to be fixed keeping in mind enormously growing corpus of MathML documents.

As explained in subsection 4.2, our proposed substitution tree based indexing method comes into role after the math expressions have been tokenized and structurally unified. In our proposed indexing method, we index one of the structurally unified versions of an expression rather than indexing its normalized version. However, utmost care should be taken in selecting the structurally unified version of an expression otherwise we might end up generating an index tree which will yield absurd results or huge number of unnecessary results when searched for a query expression. Structurally unified version of expression should be such that it preserves syntax of the original expression. For example, if we have to index the expression: "$a + \frac{\sqrt{(b)}}{c}$", it will be a wise decision to select "$\odot + \frac{\sqrt{(\odot)}}{\odot}$" over "$\odot + \frac{\odot}{\odot}$", because the former one retains the syntax of original expression while the latter one completely deforms it.

For indexing the expressions: "x_1^2", "x_2^2", "$x_1^2 + y_1^2$" and "$x_2^2 + y_2^2$", we will structurally unify them to obtain "$\odot^2$", "$\odot^2$", "$\odot^2 + \odot^2$" and "$\odot^2 + \odot^2$", respectively. Substitution tree for these unified expressions is shown in Figure 6.

Substitution tree shown in Figure 6, has comparatively less number of branches, causing minimization of memory requirement and offering more generality. It also ensures generation of relatively more number of relevant results when searched for a query expression. Moreover, the proposed indexing method is able to index MathML formulas after they have been tokenized and structurally unified – an important feature which was absent in the previous indexing method [18].

8 Conclusion and Future Works

In this paper we have proposed system architecture for MathIRs which comprises separate

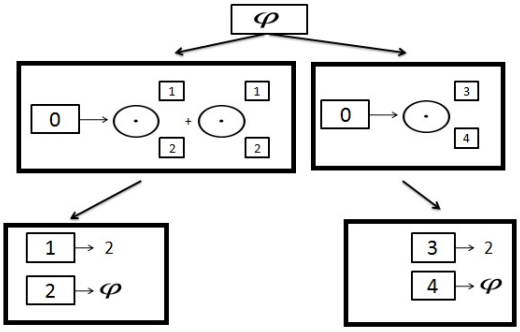


Fig. 6. Substitution Tree for expressions created using proposed indexing method

$$x_1^2,\ x_2^2,\ x_1^2 + y_1^2 \text{ and } x_2^2 + y_2^2$$

Fig. 7. Mathematical Expressions

modules for text-text, math-math and text-math similarity matching. An existing substitution tree based indexing [18] for mathematical expressions has been modified to make the indexing process more effective and generalized.

Unlike the previous approach [18], our proposed indexing method treats two semantically equivalent expressions as similar and the method is presumed to minimize memory requirement and increase number of relevant results. However, further modification in structure of substitution tree and careful selection of structural variant of an expression for indexing, can further improve effectiveness of MathIRs.

In addition, we plan to consider applying similarity measures based on the soft cosine measure [21] and syntactic n-grams [22, 19, 20].

Acknowledgement

The work presented here falls under the Research Project Grant No. YSS/2015/000988 and supported by the Department of Science & Technology (DST) and Science and Engineering Research Board (SERB), Govt. of India. The authors would like to acknowledge the Department of Computer Science & Engineering, National Institute of Technology Mizoram, India for providing infrastructural facilities and support. The fifth author acknowledges the suuport of Mexican Government though Instituto Politécnico Nacional SIP grant 20172008.

References

1. **Formánek, D., Líška, M., Růžička, M., & Sojka, P. (2012).** Normalization of digital mathematics library content. *Proceedings of the Conference on Intelligent Computer Mathematics (CICM)*, Bremen, Germany, pp. 91–103.
2. **Graf, P. (1995).** Substitution tree indexing. *Proceedings of the International Conference on Rewriting Techniques and Applications*, Springer, Kaiserslautern, Germany, pp. 117–131.
3. **Kohlhase, M. & Sucan, I. (2006).** A search engine for mathematical formulae. *Proceedings of the International Conference on Artificial Intelligence and Symbolic Computation*, Springer, Beijing, China, pp. 241–253.
4. **Lavie, A. & Agarwal, A. (2007).** Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 228–231.
5. **Líška, M. (2010).** Mathematical indexing and querying. Online: https://mir.fi.muni.cz / presentations / liska-04-05-2011.pdf.

6. **Lynum, A., Pakray, P., Gambäck, B., & Jimenez, S. (2014).** NTNU: measuring semantic similarity with sublexical feature representations and soft cardinality. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval)*, volume 14, Dublin, Ireland, pp. 448–453.
7. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.
8. **Miner, R. & Munavalli, R. (2007).** An approach to mathematical search through query formulation and data normalization. In *Towards Mechanized Mathematical Assistants*. Springer, pp. 342–355.
9. **Mišutka, J. & Galamboš, L. (2008).** Extending full text search engine for mathematical content. *Towards Digital Mathematics Library*, pp. 55–67.
10. **Pakray, P., Bandyopadhyay, S., & Gelbukh, A. (2011).** Textual entailment using lexical and syntactic similarity. *International Journal of Artificial Intelligence and Applications*, Vol. 2, No. 1, pp. 43–58.
11. **Pakray, P. & Sojka, P. (2014).** An architecture for scientific document retrieval using textual and math entailment modules. *Recent Advances in Slavonic Natural Language Processing*, pp. 107–117.
12. **Pennington, J., Socher, R., & Manning, C. D. (2014).** Glove: Global vectors for word representation. *EMNLP*, volume 14, pp. 1532–1543.
13. **Růžička, M., Sojka, P., & Líška, M. (2014).** Math indexer and searcher under the hood: History and development of a winning strategy. *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 127–134.
14. **Růžička, M., Sojka, P., & Lıška, M. (2016).** Math indexer and searcher under the hood: Fine-tuning query expansion and unification strategies. *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo Japan, pp. 7–10.
15. **Sarkar, S., Das, D., Pakray, P., & Gelbukh, A. (2016).** JUNITMZ at SemEval-2016 task 1: Identifying semantic similarity using Levenshtein ratio. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval)*, Association for Computational Linguistics, San Diego, California, pp. 702–705.
16. **Sarkar, S., Pakray, P., Das, D., & Gelbukh, A. (2016).** Regression based approaches for detecting and measuring textual similarity. *Mining Intelligence and Knowledge Exploration: 4th International Conference (MIKE)*, Springer International Publishing, Mexico City, pp. 144–152.
17. **Sarkar, S., Saha, S., Bentham, J., Pakray, P., Das, D., & Gelbukh, A. (2016).** NLP-NITMZ@DPIL-FIRE2016: language independent paraphrases detection. *Shared task on detecting paraphrases in Indian languages (DPIL), Forum for Information Retrieval Evaluation (FIRE)*, Kolkata, India, pp. 256–259.
18. **Schellenberg, M. (2011).** *Layout-based substitution tree indexing and retrieval for mathematical expressions*.
19. **Sidorov, G. (2013).** *Non-linear construction of n-grams in computational linguistics: syntactic, filtered, and generalized n-grams [in Spanish]*.
20. **Sidorov, G. (2014).** Should syntactic n-grams contain names of syntactic relations? *International Journal of Computational Linguistics and Applications*, Vol. 5, No. 1, pp. 139–158.
21. **Sidorov, G., Gelbukh, A. F., Gómez-Adorno, H., & Pinto, D. (2014).** Soft similarity and soft cosine measure: Similarity of features in vector space model. *Computación y Sistemas*, Vol. 18, No. 3, pp. 491–504.
22. **Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., & Chanona-Hernández, L. (2014).** Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications*, Vol. 41, No. 3, pp. 853–860.
23. **Sojka, P. & Líška, M. (2011).** The art of mathematics retrieval. *Proceedings of the 11th ACM symposium on Document engineering*, Association for Computing Machinery, Mountain View, California, pp. 57–60.
24. **Sojka, P. & Líška, M. (2011).** Indexing and searching mathematics in digital libraries. *Proceedings of the International Conference on Intelligent Computer Mathematics*, Springer, Bertinoro, Italy, pp. 228–243.

Article received on 08/03/2016; accepted on 23/05/2017.
Corresponding author is Partha Pakray.

Un método para el reconocimiento de objetos ocluidos

Alejandro Pacheco Morales[1], Fidel Guerrero Peña[1], Alejandro Garcés Calvelo[2], Luis A. Rodriguez Reiners[1]

[1] Universidad de Oriente, Santiago de Cuba, Cuba

[2] Instituto Nueva tecnología de la imagen, Universidad Jaume I, España

apacheco@uo.edu.cu

Resumen. En el presente trabajo se propone un método para el reconocimiento de objetos ocluidos en imágenes digitales. El mismo aporta un nuevo enfoque del uso de los Modelos Ocultos de Márkov para el reconocimiento de los objetos solapados. Para validar el método propuesto se realizaron experimentos con diferentes bases de datos, obteniéndose en todos los casos altos índices de efectividad.

Palabras clave. Imágenes digitales, oclusión, modelos ocultos de Markov, segmentación.

Method For Recognition of Ocluded Objects

Abstract. This paper proposes a method to recognize occluded objects in digital images. It provides a new approach to the use of Hidden Markov Models for segmentation of overlapping objects. To validate the method proposed several experiments were executed with different databases, in all cases high levels of effectiveness were obtained.

Keywords. Digital images, occlusion, Hidden Markov Models, segmentation.

1. Introducción

Innumerables problemas introducen grandes complejidades en el reconocimiento de objetos en imágenes digitales. Algunos de ellos, como las condiciones de iluminación, la posición y orientación de los objetos, el ruido y los datos espurios han sido abordados con mayor frecuencia en la literatura científica relacionada. Sin embargo, otros como la oclusión, no han sido tratados con la misma intensidad.

La oclusión en imágenes digitales puede ser clasificada teniendo en cuenta las causas que la provocan [5] en solapamiento, cuando un objeto oculta una porción de área de otro objeto que se quiere conocer; oclusión por opacidad, cuando la geometría del objeto oculta parte de sí mismo; y oclusión por sombra, provocada por el tipo de iluminación existente, ocultando el objeto (o parte de él) debido a su propia sombra o a la de otros objetos con los que este interactúa.

La oclusión constituye uno de los mayores obstáculos para alcanzar buenos resultados en las aplicaciones de reconocimiento de objetos. Aunque muchos autores han propuesto importantes soluciones a esta problemática [12, 8, 16, 5, 11, 13], la mayoría de ellas son muy dependientes del campo de acción donde se van aplicar, por lo que obedecen a condiciones muy específicas del entorno donde son utilizadas. Otras aproximaciones [6, 10, 1, 15] se ajustan a modelos matemáticos preestablecidos, los cuales no responden con eficacia a la heterogeneidad de los objetos que se pueden encontrar en muchas imágenes del mundo real.

Por todo lo anterior, en este trabajo se propone un método general para el reconocimiento de objetos ocluidos en imágenes digitales. El mismo aporta un nuevo enfoque del uso de los Modelos

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2735

Ocultos de Márkov en la segmentación de los objetos solapados. Para validar el método propuesto se realizaron experimentos con diferentes bases de datos, obteniéndose en todos los casos altos índices de efectividad. Los Modelos Ocultos de Márkov por su característica estocástica permiten que el método presentado en este trabajo posea una mayor flexibilidad y aplicabilidad que las investigaciones revisadas.

El trabajo se ha estructurado como sigue. Sección 1 es la introducción, en la sección 2 se presentan los trabajos revisados que contribuyeron al método propuesto. En la tercer sección se detallan los principales aspectos de la propuesta, se brindan los resultados y las comparaciones con las propuestas de la literatura en las que se sostiene la investigación. Y como último momento se dan las conclusiones y las principales ideas para trabajos futuros.

2. Trabajos relacionados

La tarea de reconocer un objeto cuando solo una parte de este es visible constituye, sin lugar a dudas, uno de los problemas más complejos en las aplicaciones de visión por computador. En la literatura especializada se pueden encontrar trabajos importantes que abordan la oclusión desde diferentes perspectivas.

Han y Jan [7] proponen un método de reconocimiento basado en el uso de puntos dominantes. Este enfoque sin lugar a dudas constituye un elemento importante para cualquier sistema de visión por computadora. Sin embargo, el análisis del reconocimiento de un objeto parcialmente ocluido utilizando solamente puntos dominantes puede resultar insuficiente para formar una representación íntegra de un objeto.

Liu y Srinath [10] proponen la representación de los objetos mediante polígonos. Esta aproximación presenta un comportamiento aceptable en objetos cuya forma responde curvas cercanas a los polígonos pero en contornos no regulares no es aplicable.

Bolles, Cain [1] y Tsang [15] combinan rasgos geométricos (líneas, arcos, esquinas), este método pese a que responde de manera correcta a figuras con contornos simples, dígase por simple que se acercan a algún modelo geométrico bien prestablecido, resulta insuficiente en objetos con formas irregulares que no responden a estos modelos.

Horáčěk, Kamenický y Flusser [8] presentan un método para el reconocimiento de objetos parcialmente ocluidos de curvatura suave en imágenes binarias. Esta investigación propone dividir el objeto en partes mediante la detección de los puntos del contorno de cero curvatura. Cada parte del objeto se representa con un vector radial y se realiza el emparejamiento utilizando String Matching [2].

Mai, Chang y Hung [12] en su investigación dividen el contorno del objeto en segmentos delimitados por puntos que se obtienen como los máximos locales de construir el Espacio de Escala de Curvatura de la imagen (CSS Curvature Scale Space) y luego realizan el emparejamiento de los segmentos mediante el algoritmo de alineamiento de Smith-Waterman empleando como medida de similitud la correlación cruzada entre los vectores de rasgos de los segmentos.

Por su parte, Gonzáles [6] propone un método basado en la información de la forma para el procesamiento de solapamientos de eritrocitos. La división del solapamiento en varios objetos es realizada mediante la detección de los puntos cóncavos en el contorno y hallando la forma, circular o elongada, que más se ajusta al arco determinado por cada par de puntos cóncavos adyacentes, aplicando una serie de restricciones para determinar las circunferencias o elipses que se corresponden con cada célula dentro del agrupamiento.

3. Método propuesto

El método propuesto en este trabajo se centrará, según la clasificación de oclusión dada por [5], en los solapamientos, siguiendo una estrategia de reconocimiento de objetos mediante el análisis de la forma. El mismo se puede dividir dos etapas fundamentales: la extracción de características y el reconocimiento de objetos (ver Figura 1).

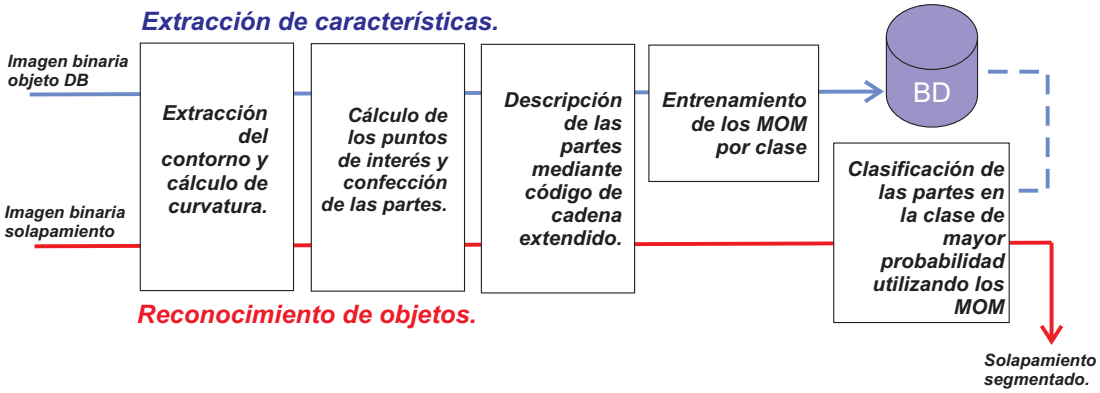


Fig. 1. Esquema del método

3.1. Extracción de características

La extracción de características parte de una imagen binaria, resultado de haber aplicado un método de segmentación a una imagen digital. En esta etapa se realizan los pasos descritos a continuación.

3.1.1. Extracción del contorno de la figura y cálculo de la curvatura del mismo

Una curva parametrizada puede ser entendida como la evolución de un punto que se mueve a lo largo del espacio 2D. Matemáticamente, la posición de un punto en el plano puede ser expresada como un vector $p(t) = (X(t), Y(t))$ donde t es un valor real llamado parámetro de la curva [3].

Sea una curva parametrizada $p(t)$ se llamará curvatura tomando la propuesta realizada en [3] a la magnitud de la proporción entre el área interior al objeto y el área total en una vecindad circular para cada punto de la curva. En la Ecuación 1 se muestra como se realiza el cálculo.

$$K(t) = \frac{A_{interior}(t,r)}{AREA\_CIRCULAR}, \qquad (1)$$

donde $A_{interior}(t,r)$ es el área de intersección entre el objeto y el círculo de radio r fijado en el punto t del contorno y $AREA\_CIRCULAR$ es el área del circulo, la cual para un parámetro r es constante.

La curvatura es un concepto extremadamente importante porque expresa la naturaleza local geométrica de la curva. Por ejemplo un valor de cero para la curvatura corresponde a una línea recta, por otro lado si el valor es constante indica un círculo o un arco de círculo. De manera general la curvatura es proporcional a la variación local de la curva. Otro hecho importante de curvatura es que resulta invariante a rotaciones, traslaciones y reflexiones de la curva original. En este paso se extrae el contorno y se calcula la curvatura siguiendo la ecuación 1 del contorno mediante una proporción entre área interior al objeto y el área total en una vecindad circular para cada punto del contorno. En la figura 2 se muestra el gráfico de la curvatura para un objeto manzana.

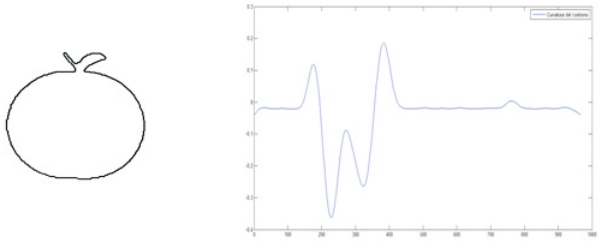


Fig. 2. Gráfico de curvatura

3.1.2. Cálculo de los puntos de interés

La división del objeto en partes se realiza tomando como puntos de interés los elementos más cercanos a una cero curvatura entre dos extremos locales. Cada segmento del contorno delimitado por los puntos de interés constituirá una parte. Así un objeto estará formado por una secuencia de partes. En la figura 3 se muestran los puntos de interés detectados para un objeto manzana.

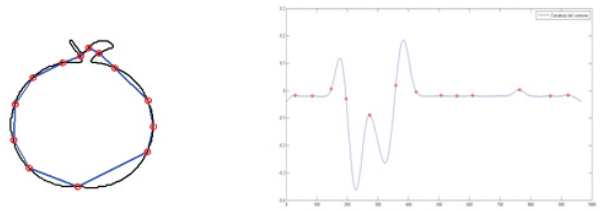


Fig. 3. Puntos de interés

Los segmentos de contorno delimitados por estos puntos serán las partes que conformarán el objeto. Cada parte es parametrizada utilizando en código de cadena extendido [14].

3.1.3. Entrenamiento de los Modelos Ocultos de Márkov

Luego de obtener la representación de los objetos de entrenamiento, mediante la detección de puntos de interés y la parametrización de los segmentos de contorno con el código de cadena, se entrena un Modelo Oculto de Márkov (MOM) por clase, utilizando las posibles direcciones del código de cadena como observaciones y como cantidad de estados la media de la cantidad de partes de las clase.

Un MOM es un modelo estadístico en el que se asume que el sistema a modelar es un proceso de Márkov de parámetros desconocidos. El objetivo es determinar los parámetros desconocidos de dicha cadena a partir de los parámetros observables.

De manera general se define un MOM [4], como un modelo probabilístico, utilizado para representar la probabilidad conjunta de un conjunto de variables aleatorias. Estas variables aleatorias son de dos tipos; el primer grupo corresponde a los posibles símbolos observables O_t, que pueden presentarse al realizar una observación del sistema. El segundo corresponde al estado en el cual se encuentra el sistema oculto durante una observación O_t. Matemáticamente un MOM se puede definir como una tupla $\lambda = \{S, O, \alpha, \beta, \pi\}$ donde:

1. S es un conjuntos de estados $S = \{s_1, s_2, ..., s_n\}$
2. O en el conjunto de observaciones $O = \{o_1, o_2, ..., o_m\}$
3. α es la matriz de probabilidades de transición de los estados s_i, $\alpha(i,j) = P[X_{t+1} = s_j | X_t = S_i]$.
4. β es la matriz de probabilidades de probabilidades de emisión de los símbolos o_i, $\beta(i,k) = P[O_t = e_k | X_t = S_i]$.
5. π es el vector de probabilidades iniciales para los estados, $\pi(i) = P[X_1 = S_i]$.

Existen tres problemas fundamentales al emplear un MOM, a saber:

Evaluación del modelo: Dada una secuencia de observaciones $O = \{o_1, o_2, ..., o_n\}$ y un modelo λ , determinar $P[O|\lambda]$. Para dar solución a este problema comúnmente son utilizado los algoritmos *Forward* o *Backward* [4].

Decodificación: Dada una secuencia de observaciones $O = \{o_1, o_2, ..., o_n\}$ y un modelo λ , encontrar la secuencia de estados ocultos $Q^* = q_1^*, q_2^*, ..., q_n^*$ tal que:

$$Q^* = argmax_{Q_i} P(Q_i | \lambda, O). \quad (2)$$

Su solución se obtiene mediante el algoritmo de *Viterbi* [4].

Aprendizaje: Dada una secuencia de observaciones $O = \{o_1, o_2, ..., o_n\}$ determinar los parámetros del modelo. El algoritmo empleado tradicionalmente para su solución es el algoritmo *Baum-Welch* [4].

3.2. Reconocimiento de objetos

En esta etapa se realiza el reconocimiento de las partes de los objetos. Para ello se parte una imagen binaria de un solapamiento, en la figura 4 se muestra el contorno de un solapamiento. Se realiza, el proceso de extracción de características y se clasifica cada parte de un solapamiento en la clase de mayor probabilidad utilizando los MOM entrenados por clase.

Fig. 4. Contorno de solapamiento de grado dos

La clasificación de las partes se realiza utilizando el algoritmo *Forward-Backward*. Una vez clasificada cada parte el objeto se segmenta mediante la unión de partes consecutivas, esto como una primera aproximación a solapamientos de grado dos (ver figura 5).

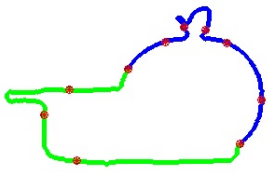


Fig. 5. Reconocimiento de las partes de un solapamiento de grado dos

3.3. Resultados

Para la evaluación se utilizaron dos bases de datos de imágenes. La primera colección de objetos se creó de manera artificial y consistió en objetos geométricos básicos, dígase cuadrados, círculos y triángulos. Como segunda colección se empleó la base de datos de imágenes MPEG-7 Core Experiment CE-Shape-1 [9]. Esta cuenta con 70 clases, con 20 objetos por clase rotados y escalados. En la figura 6 se muestran algunos ejemplos de las clases.

Fig. 6. Algunas de las clases de la base de datos *MPEG-7 Core Experiment CE-Shape-1*

La eficacia del método se comprobó mediante dos experimentos, evaluando el por ciento de partes bien reconocidas en los solapamientos formados artificialmente con objetos de las colecciones antes mencionadas.

3.3.1. Experimento 1

El primer experimento utilizó objetos de la primera base de datos por ser los más sencillos y por tener formas geométricas. Se tomaron 15 imágenes por cada clase para el entrenamiento y 5 para la validación. Los resultados fueron satisfactorios obteniendo el 94.63 % de las partes bien reconocidas. En la tabla 1 se muestra un resumen de estos, obtenidos luego de aplicarle el método a 100 solapamientos formados aleatoriamente con los objetos de la base de datos destinados para la prueba.

Tabla 1. Efectividad de la clasificación de las partes con figuras geométricas

Cantidad de partes	Clasificación		% Efectividad
	Bien	Mal	
802	759	43	94.63

3.3.2. Experimento 2

En el segundo experimento se utilizaron objetos de la segunda base de datos. La configuración fue la misma que en el primer experimento con respecto a la división de la base de datos; 15 para el entrenamiento y 5 para la evaluación. Los resultados obtenidos en este caso disminuyeron

a un 82.3 %. Este resultado se debe a que los objetos de esta base de datos poseen un nivel mayor en cuanto a complejidad de la forma.

Al igual que en el anterior experimento para esta prueba se generaron aleatoriamente 100 solapamientos con los objetos de la base de datos destinados a la evaluación. En la Tabla 2 se muestra un resumen de la efectividad de clasificación para algunos tipos de objetos.

Tabla 2. Efectividad de la clasificación de las partes (Método propuesto)

Tipo de objeto	% Efectividad de Clasificación
Manzana	82.40
Celular	**84.41**
Mariposa	81.70
Camello	80.68

Los trabajos presentados en [8] y [12] constituyeron antecedentes importantes para este trabajo. Los resultados presentados por ambos fueron los mejores encontrados en la revisión bibliográfica realizada a los métodos de reconocimiento de objetos parcialmente ocluídos. Cabe destacar que se encontraron otros métodos con mejores resultados pero en el reconocimiento de clases muy específicas de objetos dígase células, personas, etc. En la tabla 3 se muestra un comparación de estos dos métodos con la propuesta presentada en este trabajo.

Tabla 3. Efectividad de la clasificación de las partes (Método propuesto en [8])

Método	% Efectividad de Clasificación
Mai [12]	79.2
Horáčěk [8]	76.4
HMM	**82.3**

Como se puede apreciar en la tabla 3 los métodos [8] y [12] obtienen resultados por debajo del método propuesto. Estos clasifican una sección visible de un objeto y no experimentan con contornos dónde aparezcan partes de más de un elemento de la base de datos, mientras que en esta propuesta si se tiene en cuenta que pueden aparecer partes de objetos diferentes en la sección visible del contorno (figura 7). Debido a esto al enfrentarse a contornos de figuras solapadas obtienen resultados inferiores a los de este trabajo.

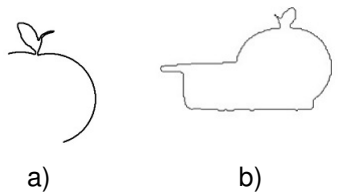


Fig. 7. a) Sección visible de un objeto b) Objetos en un solapamiento

Los trabajos [8] y [12] se comparan contra toda la base de datos de objetos empleando como medida de similitud una diferencia gaussiana y la correlación cruzada respectivamente. A diferencia de estos, la nueva propuesta presentada, responde a un esquema de clasificación supervisada por lo que para el reconocimiento no es necesario recorrer la base de datos.

4. Conclusiones y trabajo futuro

A partir del análisis de los trabajos estudiados y de la implementación de las propuestas realizadas en [8] y [12], en esta investigación se propone un nuevo método de reconocimiento de objetos solapados empleando los MOM. En la investigación se utilizan los MOM como una herramienta para el reconocimiento de objetos mediante el análisis de la forma en imágenes digitales.Se superaron los resultados de los métodos tomados como antecedentes de la investigación con un 94.63 % en un base de datos de objetos con formas geométricas simples y un 82.3 % en una base de datos de objetos con formas más complejas.

Como trabajo futuro se propone adecuar la propuesta para el reconocimiento de eritrocitos solapados en imágenes de muestras de sangre. Asimismo se desea experimentar con características de los objetos que tengan en cuenta el color y la textura.

Referencias

1. **Bolles, R. & Cain, R. A. (1982).** Partial shape clasification using contourn matching in distance transformation. *Int. J. Robotics Res.*, Vol. 1, pp. 57–81.
2. **Cormen, T. H. (2009).** *Introduction to algorithms*. MIT press.
3. **da Fontoura Costa, L. & Cesar Jr, R. M. (2010).** *Shape analysis and classification: theory and practice*. CRC press.
4. **Fink, G. A. (2007).** *Markov Models for Pattern Recognition from theory to applications*. Springer.
5. **Gil Vázquez, P. (2008).** *Estrategias para identificar oclusiones y planificación monocular para una mejora de la percepción visual de la escena*. Universidad de Alicante.
6. **González-Hidalgo, M., Guerrero-Peña, F. A., Herold-García, S., Jaume-i-Capó, A., & Marrero-Fernández, P. D. (2014).** Red blood cell cluster separation from digital images for use in sickle cell disease. *IEEE Journal of Biomedical and Health Informatics*.
7. **Han, M. & Jang, D. (1990).** The use of maximun curvature points for recognition of partially occluded objects. *IEEE Transaction on Patter Analysis and Matching Inteligence*, Vol. 23, pp. 21–33.
8. **Horáček, O., Kamenický, J., & Flusser, J. (2008).** Recognition of partially occluded and deformed binary objects. *Institute of Information Theory and Automation Academy of Sciences of the Czech Republic*.
9. **Latecki, L. J. (2004).** *Retrieval Result for Shape Similarity on the MPEG-7 Data Set*.
10. **Liu, H. & Srinath, M. (1990).** Partial shape clasification using contourn matching in distance transformation. *IEEE Transaction on Pattern Analysis and Matching Intelligence*, Vol. PAMI-12(11), pp. 21–33.
11. **Loughlin, A. (2004).** Using context to repair partial occlusions in topographic data. *Proceedings (GISRUK) Geographical Information Science Research Conference*, pp. 15–18.
12. **Mai, F., Chang, C., & Hung, Y. (2010).** Affine-invariant shape matching and recognition under partial occlusion. *IEEE International Conference on Image Processing Proceedings*.
13. **Shu, G., Dehghan, A., Oreifej, O., Hand, E., & Shah, M. (2012).** Part-based multiple-person tracking with partial occlusion handling. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 1815–1821.
14. **Szeliski, R. (2010).** *Computer vision: algorithms and applications*. Springer Science & Business Media.
15. **Tsang, P. W., Yueng, P., & Lam, F. K. (1992).** Recognition of ocluded objects. *Pattern Recognition*, Vol. 25, pp. 1107–1117.
16. **Yang, M., Zhang, L., Shiu, S. C., & Zhang, D. (2013).** Gabor feature based robust representation and classification for face recognition with gabor occlusion dictionary. *Pattern Recognition*, Vol. 46, No. 7, pp. 1865–1878.

Article received on 09/01/2016; accepted on 07/02/2017.
Corresponding author is Alejandro Pacheco Morales.

Compact Union of Disjoint Boxes: An Efficient Decomposition Model for Binary Volumes

Irving Cruz-Matías[1], Dolors Ayala[2]

[1] Universidad de Monterrey, San Pedro Garza García, Mexico

[2] Universitat Politècnica de Catalunya, Barcelona, Spain

irving.cruz@udem.edu, dolorsa@lsi.upc.edu

Abstract. This paper presents in detail the Compact Union of Disjoint Boxes (CUDB), a decomposition model for binary volumes that has been recently but briefly introduced. This model is an improved version of a previous model called Ordered Union of Disjoint Boxes (OUDB). We show here, several desirable features that this model has versus OUDB, such as less unitary basic elements (boxes) and thus, a better efficiency in some neighborhood operations. We present algorithms for conversion to and from other models, and for basic computations as area (2D) or volume (3D). We also present an efficient algorithm for connected-component labeling (CCL) that does not follow the classical two-pass strategy. Finally we present an algorithm for collision (or adjacency) detection in static environments. We test the efficiency of CUDB versus existing models with several datasets.

Keywords. Binary volumes, orthogonal polyhedra, connected-component labeling, collision detection.

1 Introduction

The model used to represent digital binary volumes (or 3D binary images) is one of the most important research topic in computer graphics, looking for a better compression for storage, analysis or visualization purposes. Moreover, there are several demanding neighborhood operations on binary volumes such as connected-component-labeling (CCL), connectivity, collision detection, between others, where the performance variability of the existing algorithms is mainly caused by the number of basic geometric elements to analyze (voxels, triangles, planes, vertices, etc.).

In most of the reported literature, the operations to study binary volumes are performed directly on the classical voxel model [31, 41]. However, in the field of volume analysis and visualization, several alternative models have been devised for specific purposes. For instance, Hierarchical structures such as octrees and kd-trees have been used for Boolean operations [46], CCL [17], and thinning [40, 58]. Octrees are used as a means of compacting regions and getting rid of the large amount of empty space in the extraction of isosurfaces [57]. Kd-trees have been used to extract two-manifold isosurfaces [24].

There are other models that store surface voxels, thereby gaining storage and computational efficiency. The semi-boundary representation affords direct access to surface voxels and performs fast visualization and manipulation operations [25]. Certain methods of erosion, dilation and CCL use this representation [20, 51].

On the other hand, a binary voxel model represents an object as the union of its foreground voxels and its continuous analog is an orthogonal pseudo-polyhedra (OPP) [33]. OPP have been used in 2D to represent the extracted polygons from numerical control data [39]. Some 3D applications of OPP are: general computer graphics applications such as geometric transformations and Boolean operations [10, 19], skeleton computation (instead of iterative

Computación y Sistemas, Vol. 21, No. 2, 2017, pp. 275–292
ISSN 1405-5546
doi: 10.13053/CyS-21-2-2737

peeling techniques) [18, 37], and orthogonal hull computation [8, 9]. OPP have been also used in theory of hybrid systems to model the solutions of reachable states [10, 16].

The Extreme Vertices Model (EVM) and the Ordered Union of Disjoint Boxes (OUDB) [1, 3] represent OPP in a compact way. EVM stores only a sorted subset of vertices of the OPP boundary, whereas OUDB keeps a sorted list of boxes that compose the whole object. EVM has been used to prove the suitability of OPP as geometric bounds in CSG [2, 1], and for many other 3D applications such as: erosion and dilation operations [42], skeleton computation [4], virtual porosimetry without skeleton computation [44], biomaterials structural parameters computation [54] and model simplification [15]. OUDB has been used to perform CCL [5, 42] and also for biomaterials structural parameters computation such as connectivity [7] and center of gravity [6].

In this paper we present in detail the Compact Union of Disjoint Boxes (CUDB), a representation model for OPP, and thus for binary volumes, that has been recently introduced, but without going into details of its implementation [14, 44]. CUDB is a special kind of cell decomposition representation which performs a spatial partition of the volume in a non-hierarchical sweep-based way consisting of a set of disjoint boxes. This model is an improved version the OUDB. We test and report the performance of CUDB and also present new algorithms for CCL and collision detection.

The paper is arranged as follows. The next section reviews related work. Section 3 introduces CUDB using the object-oriented paradigm presenting algorithms for conversion to and from other models, and for basic computations as area and volume. Section 4 presents a new CUDB-based algorithm for CCL based on the detection of connected components in graph theory. Section 5 presents a collision detection algorithm for scenarios with multiple CUDB-represented objects. Sections 6 shows the performance of all the proposed algorithms by discussing experimental results. Finally, Section 7 concludes the paper and outlines future work.

2 Background and Related Work

The classical voxel model is based on a regular decomposition of the 3D space into a set of identical cubic cells called voxels. In a voxel model, voxels are all the same size and their edges are parallel to the main axes. Formally, a voxel model V of size $n_x \times n_y \times n_z$ is defined as:

$$V = \{v_{i,j,k} \mid 0 \leq i \leq n_x, 0 \leq j \leq n_y, 0 \leq k \leq n_z\},$$

where $v_{i,j,k}$ is a voxel in location (i, j, k).

From V, all geometric and topological information can be obtained. On each voxel $v_{i,j,k}$, there is a set of associated values. For a binary voxel model, the associated value of its voxels is limited to $v_{i,j,k} \in \{0, 1\}$, where 0 corresponds to the background and 1 to the foreground.

Algorithms for the voxel model are straightforward to implement. However, just because of the size of the source models, it has the drawbacks of the loss of geometric information and high memory and computational power requirements [32]. To reduce the memory footprint and the computation time of the algorithms, many alternative models have been proposed such as hierarchical data structures and boundary representations.

Hierarchical structures made recursive subdivisions of the volume. In bintrees [47] an axis-aligned hyperplane intersects the interior of the volume producing two equal parts. The Binary Space Partition (BSP) trees [50] are similar to bintrees except that the position and direction of the subdivision hyperplanes are usually selected following an optimization heuristic. Octrees [11], like bintrees, begin with an initial volume, but it create eight sub-volumes of equal size. And kd-trees [24] are a special case of bintree, where each level is asymmetrically divided in alternate directions according to a discriminant (e.g., X, Y or Z-coordinate). In this structures, volume subdivision is repeated until a certain satisfactory level of refinement or until achieving the maximum level of recursion.

When working with binary volume datasets, it is not necessary to keep the exact scanned density values. Therefore, boundary models are also adequate to represent these kind of datasets. The basic model for representing the polygonal surface

of an object is the Boundary Representation (B-Rep) model [38] that keeps explicitly all of the relationships between geometric elements such as vertices, edges and faces. However, there are many proposed models for representing the object boundary in a more compact way.

Semi-boundary (SB) [52] is a data structure which stores only the boundary voxels of the volume keeping the information of the interior voxels in an implicit way. Shell representation [53] has the same indexing scheme used in the SB representation, but the set of voxels, which belongs to the list that contains all the boundary voxels of the volume, was redefined in order to represent objects with fuzzy boundaries. Cell-boundary [34] is also a very similar representation to SB, which consists of a set of boundary cells with their voxel configurations, so, the points of the sample in SB become vertices of the cells in the cell-boundary representation.

2.1 EVM and OUDB models

The Extreme Vertices Model (EVM) [2, 3] is a very concise representation scheme in which any OPP can be described using a subset of its vertices: the extreme vertices. EVM is actually a complete solid model with very fast Boolean operations, it is an implicit B-Rep model, i.e., all the geometry and topological relations concerning faces, edges and vertices of the represented OPP can be obtained from the EVM [43] and therefore represents OPP unambiguously [55].

Let Q be a finite set of points in $\mathbb{R}^3$, the ABC-sorted set of Q is the set resulting from sorting Q according to A-coordinate, then to B-coordinate, and then to C-coordinate. Let P be an OPP, a *brink* is the maximal uninterrupted segment built out of a sequence of collinear and contiguous two-manifold edges of P and its ending vertices are called *extreme vertices* (EV). An OPP can be represented in a concise way with the ABC-sorted set of its EV and such representation scheme is called EV.

The Ordered Union of Disjoint Boxes (OUDB) [1, 42] is a special kind of spatial partitioning representation derived from EVM, where an OPP is decomposed in a list of disjoint boxes. EVM can be obtained from the voxel model, in turn, OUDB is obtained from EVM. The conversions algorithms have been published [42]. To introduce later the CUDB model, we first briefly introduce OUDB.

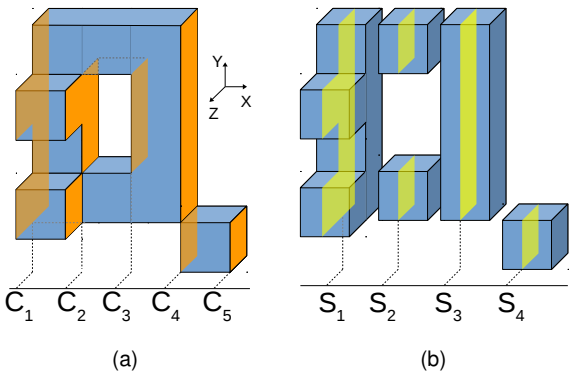


Fig. 1. (a) An orthogonal polyhedron with 5 cuts. (b) Its sequence of 4 prisms with the representative sections (X direction)

Let P be an OPP and Π_c a plane whose normal is parallel, without loss of generality, to the X axis, intersecting it at $x = c$, where c ranges from $-\infty$ to $+\infty$. Then, this plane sweeps the whole space as c varies within its range, intersecting P at certain intervals. Let us assume that this intersection changes at $c = C_1, ..., C_n$. More formally, $P \cap \Pi_{C_i-\delta} \neq P \cap \Pi_{C_i+\delta}, \forall i = 1, ..., n$, where δ is an arbitrarily small quantity. Then, $C_i(P) = P \cap \Pi_{c_i}$ is called a *cut* of P and $S_i(P) = P \cap \Pi_{C_s}$, for any C_s such that $C_i < C_s < C_{i+1}$, is called a *section* of P.

Fig. 1 shows an OP with its *cuts* and *sections* perpendicular to the X axis. Since we work with bounded regions, $S_0(P) = \emptyset$ and $S_n(P) = \emptyset$, where n is the total number of *cuts* along a given coordinate axis.

Cuts and sections are orthogonal polygons embedded in the 3D space. For each main direction, sections can be computed from cuts and cuts from sections by applying simple XOR operations. These operations are actually performed on the projections of cuts and sections onto the main plane parallel to them. From now on, we thus call the projection of the ith cut and ith section onto the main plane parallel to them C_i and S_i respectively.

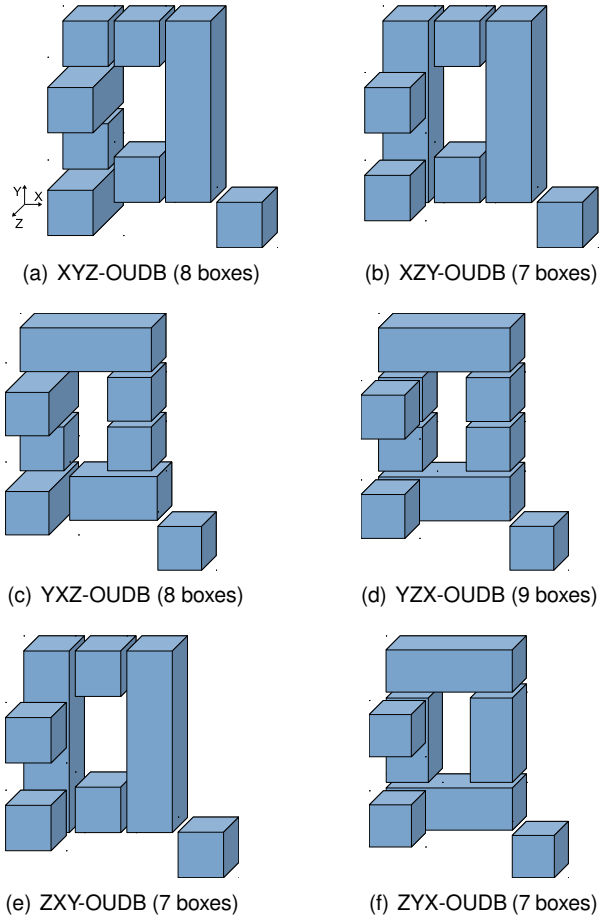
The two following expressions relate cuts and sections:

$$S_i(P) = S_{i-1}(P) \otimes C_i(P), \forall i = 1 \ldots n-1, \quad (1)$$

$$C_i(P) = S_{i-1}(P) \otimes S_i(P), \forall i = 1 \ldots n, \quad (2)$$

where $\otimes$ denotes the regularized XOR operation. An OP can be represented with a sequence of orthogonal prisms represented by their section (see Fig. 1(b)). Moreover, if we apply the same reasoning to the representative section of each prism, an OP can be represented as a sequence of boxes.

The OUDB model represents an OPP with such a sequence of boxes. OUDB is axis-aligned like octrees and bintrees, but the partition is done along the object geometry as in BSP. Depending on the order of the axes along which we choose to split the data, an OPP P can be decomposed into six different ABC-OUDB, i.e., P is subdivided by planes perpendicular to the A-axis first, and then by planes perpendicular to the B-axis. Their corresponding sets of disjoint boxes are generally different. Fig. 2 shows the possible OUDB decompositions for the OPP in Fig. 1(a).



(a) XYZ-OUDB (8 boxes) (b) XZY-OUDB (7 boxes)

(c) YXZ-OUDB (8 boxes) (d) YZX-OUDB (9 boxes)

(e) ZXY-OUDB (7 boxes) (f) ZYX-OUDB (7 boxes)

Fig. 2. The six possible OUDB decompositions for the OPP in Figure 1(a)

2.2 Connected Component Labeling

Connected Component Labeling (CCL) is a very important operation for managing volume datasets where multiple disconnected components that compose the volume need to be identified. Traditional voxel-based methods have been widely used [45].

The typical implementation of many approaches, even of the recent ones [27, 28, 35] is based in the classical two-pass strategy [45]: the *labeling pass* and the *renumbering pass*. In short, in the *labeling pass* all elements are scanned and labeled according to their already labeled neighbors. Some labeling ambiguities can be produced in this step which are properly registered in a set of equivalence classes. Then, the renumbering pass solves these ambiguities and the elements are relabeled.

There are other labeling algorithms for special volume representations as hierarchical structures or semi-boundary representations looking for a better efficiency of CCL. In this sense, OUDB has been proved to be efficient for CCL [42, 43]. With regard to semi-boundary representations, it has been concluded that CCL is better in OUDB than in semi-boundary representations when the number of boxes in the OUDB is less than the number of boundary voxels, which generally occurs.

In the OUDB-CCL process, the traversal of the boxes is performed orderly, so, checking the neighborhood of the current box involves those boxes in the immediate previous B-slice and those boxes in the immediate previous A-slice. An improvement of the OUDB-based CCL has been already proposed [5], where the

so-called OUDB-extended is computed, which allows jumping directly to the required box that needs to be tested, instead of querying and skipping several intermediate boxes.

Because of its similarity to OUDB, the two-pass strategy for CCL has been also applied in CUDB [44]. As generally the CUDB-represented object contains less boxes than its respective OUDB-representation, this approach is faster. However, the main drawback of all the aforementioned approaches is the large size of the equivalence table, because they need one entry per each new detected label.

2.3 Collision Detection

Collision detection is an important characteristic in computer graphics and simulation. Sometimes we want to determine if two or more objects collide or are adjacent. In collision detection [30] when exact accuracy is not required, typical bounding volumes like axis-aligned bounding boxes (AABB) [59], spheres [22], oriented polytopes [13] or hybrid bounding volumes are used [56].

However, when accuracy is important, a thorough analysis of the contact between the involved objects needs to be done. In complex scenes there might be several objects interacting. In such cases, an early detection phase can be applied to discard collisions between objects which are not close enough. As bounding volumes present simpler features, they are used as a preliminary test for collisions. The absence of bounding volume collisions guarantees the absence of collision between some objects and thus it significantly reduces the number of objects to be checked. Sweep and prune algorithms [12, 23] sort the objects according to the lower and upper bounds of their bounding volumes, and when a pair of objects are very close, it can be tested to exact collision.

3 Compact Union of Disjoint Boxes

Like OUDB, CUDB is also a union of disjoint boxes but a more compact one as several contiguous boxes are merged into one in several parts of the model. Let P be an OPP, to obtain the

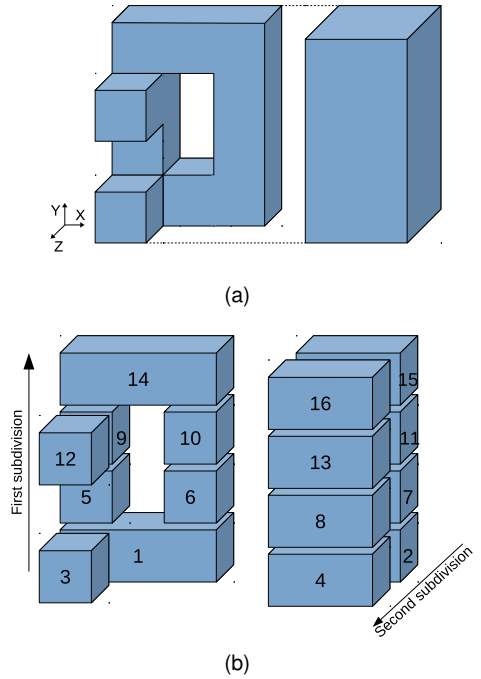


Fig. 3. (a) An OPP. (b) YZX-OUDB with 16 boxes labeled according to their YZX-coordinates

ABC-OUDB model, P is subdivided by planes perpendicular to the A-axis first, and then by planes perpendicular to the B-axis, at each cut C_i of P. Thus, every C_i splits all the geometry of P along the corresponding plane, and therefore some local regions of P, with which C_i actually has no relationship, are further unnecessarily divided. Fig. 3 shows this situation for the YZX-OUDB of an OPP P, where some cuts force unnecessary divisions. For OUDB this constraint is mandatory to keep sorted the resulting boxes. However, in order to subdivide just the pieces of P related with the cut which induces the splitting, this constraint can be relaxed.

Formally, let P be an OPP. The $CUDB(P)$ can be obtained by merging boxes in several parts of the corresponding $OUDB(P)$. Then, this model is the set of boxes obtained according to the next properties:

1. Let β_1 and β_2 be two adjacent boxes of $OUDB(P)$ in B-direction, and let $\overline{\beta_1}^B$ and $\overline{\beta_2}^B$ be their projections respectively onto the plane

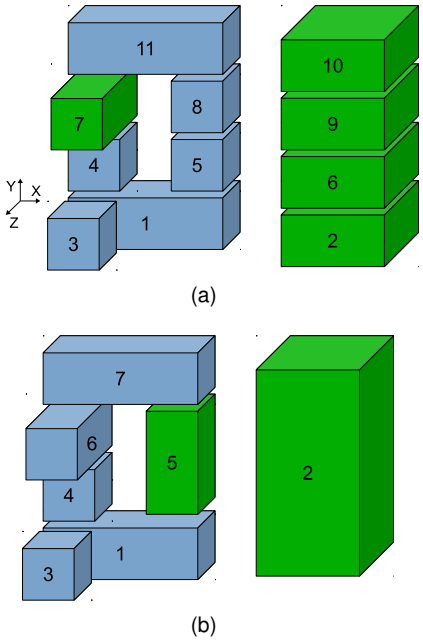


Fig. 4. (a) Result after first merging in Z-direction. (b) Resulting YZX-CUDB with 7 boxes after merging in Y-direction

perpendicular to the B-axis, then β_1 and β_2 can be merged as a single box if $\overline{\beta_1}^B = \overline{\beta_2}^B$.

2. Let β_1 and β_2 be two adjacent boxes of $OUDB(P)$ in A-direction, and let $\overline{\beta_1}^A$ and $\overline{\beta_2}^A$ be their projections respectively onto the plane perpendicular to the A-axis, then β_1 and β_2 can be merged as a single box if $\overline{\beta_1}^A = \overline{\beta_2}^A$. Note that A-direction in this property is different of B-direction of the first property.

The first property merges all unnecessary subdivisions along B-axis. Following with the previous example, Fig. 4(a) shows that the pairs of boxes (2, 4), (7, 8), (9, 12), (11, 13) and (15, 16) depicted in Fig. 3(b) can be merged applying this property for the Z-axis.

The second property merges the remaining unnecessary subdivisions along A-axis. Following with the same example, the resulting model depicted in Fig. 4(b) shows that applying this second property for the Y-axis, not only the pairs of boxes (5, 8), but also the set of boxes (2, 6, 9, 10) depicted in Fig. 4(a) can be merged.

Then, the CUDB-representation of an OPP P, is the set of disjoint boxes of the corresponding OUDB, conveniently reduced by applying the two previous merging properties. Let β_i be a box in $CUDB(P)$:

$$P = \bigcup_{i=1}^{n_b} \beta_i(P) \tag{3}$$

where n_b is the number of boxes, which is less or equal than the number of boxes in $OUDB(P)$.

Like OUDB, there are six different ABC-CUDB models for a given OPP and the number of obtained boxes depends on the order of the axes along which we choose to split the data, but we cannot know it a priori. Boxes in CUDB are sorted according to its coordinate A, then to coordinate B, and finally to coordinate C of its lower bound.

Although the implicit order among boxes in OUDB that defines their adjacency is lost, preserving the adjacency information in the CUDB model is easy with a tiny storage effort. Each box has neighboring boxes in only two orthogonal directions: A and B-direction, and for each one there are two opposite senses, so, four arrays of pointers to the neighboring boxes (two for each direction) are enough to preserve the adjacency information that is required for future operations. We define these arrays as A-backward neighbors (ABN), A-forward neighbors (AFN), B-backward neighbors (BBN) and B-forward neighbors (BFN).

CUDB has been implemented as an object with a set of properties and methods. Next, we describe the CUDB object and the algorithms for conversion to and from EVM.

3.1 CUDB Structure

Let β and Q be a box and a CUDB object respectively:

Box properties:

— point3D V_0, V_1: point3D objects representing the two diagonally opposed vertices of β, the ones with lowest and highest coordinate values.

— array ABN, AFN, BBN, BFN: Arrays of pointers to the neighbors of β.

— integer $label$: The label of β.

Box methods:

— Box(point3D V_0, point3D V_1): Constructor for a new box β with vertices V_0 and V_1. $label$ is set to 0 (undefined) and arrays ABN, AFN, BBN and BFN to $\{\emptyset\}$.

— $getLabel$(): Returns the label of β.

— $setLabel$(integer $label$): Sets the label of β as $label$.

CUDB properties:

— boolean $nonManifolds$: Flag to indicate if the boxes adjacency allows non-manifold configurations.

— array $boxes$: Vector containing the sorted boxes of Q.

— dimType dim: Dimension of Q. dimType={0D, 1D, 2D, 3D}

— integer $nBoxes$: Number of boxes in Q.

— sortingType $sort$: Sorting of Q. sortingType={XYZ, XZY, YXZ, YZX, ZXY, ZYX}

CUDB methods:

— $CUDB$(dimType dim, sortingType $sort$, boolean non_m): Constructor for a CUDB object Q of dimension dim and sorting $sort$, and flag $nonManifolds$ set as non_m.

— $getBox$(integer id): Returns the box at position id in the array $boxes$.

— $getDimension$(): Returns the dimension of Q.

— $getNBoxes$(): Returns the number of boxes of Q.

— $getNextBox$(Box β): Returns the next box to β in the array $boxes$.

— $getSorting$(): Returns the sorting of Q.

— $insertBox$(Box β): Inserts the box β into Q, at the end of the array $boxes$.

3.2 CUDB Computation

In order to obtain the CUDB-representation it is not necessary to have or compute the OUDB model. CUDB can be computed directly from the EVM, and the merging of boxes is performed on the fly. The strategy is similar to the process to compute OUDB [42]. Cuts of the EVM-represented OPP are obtained sequentially, and sections are computed from them. When this process is performed in 2D, the corresponding sections result in the boxes of the OUDB model.

For the EVM to CUDB conversion method, the same set of boxes is computed on the fly and a box is stored in the CUDB model if it is not possible to merge it with previously computed boxes applying the aforementioned Properties 1 and 2.

For a given box, the set of previous boxes that have to be considered for merging with it are those boxes belonging to the previous A-slice, which can be adjacent in A-direction, and those boxes belonging to the previous B-slice, which can be adjacent in B-direction. To facilitate this process, temporary arrays of box pointers of the current and previous slices are maintained. The corresponding algorithm is detailed next.

As most of the algorithms dealing with EVM, the corresponding conversion algorithm is also recursive over the dimension. The main function $EVMtoCUDB()$ (Algorithm 1) receives an EVM-represented OPP P and a flag to indicate if the adjacency relationship among boxes allows non-manifold configurations, and returns the CUDB-represented OPP Q containing the neighborhood information. This function initializes the temporary arrays of box pointers $prevBBoxes$, $currentBBoxes$, $prevABoxes$ and $currentABoxes$, which are defined as global variables throughout the whole conversion process, and starts the recursion by calling the function $doConversion()$ (Algorithm 2) with the original EVM-represented object P.

In function $doConversion()$, when dimension is 3D, the object is split at each cut in A-direction obtaining a set of 3D A-slices, $\Upsilon_A = \{\Upsilon_A^1, \Upsilon_A^2, \ldots, \Upsilon_A^n\}$, where n is the number of A-slices. Then the algorithm applies recursively to the 2D section representing each slice, which is

split at every internal cut in B-direction obtaining a set of 2D B-slices, $\Upsilon_B = \{\Upsilon_B^1, \Upsilon_B^2, \ldots, \Upsilon_B^m\}$, where m is the number of B-slices represented by their 1D sections, which are composed by a set of collinear brinks in C-direction. Each of these brinks defines a box. Then, each 2D slice Υ_B^i defines one o more boxes, and each 3D slice Υ_A^j contains all the boxes defined in its 2D slices.

Algorithm 1: EVMtoCUDB(**Input** P:EVM, **Input** nonManifolds:boolean, **Output** Q:CUDB)

$dim \leftarrow P.getDimension()$;
$Q =$
$CUDB(dim, P.getSorting(), nonManifolds)$;
$prevBBoxes \leftarrow \{\emptyset\}$; $currentBBoxes \leftarrow \{\emptyset\}$;
$prevABoxes \leftarrow \{\emptyset\}$; $currentABoxes \leftarrow \{\emptyset\}$;
$doConversion(P, Q, dim, \emptyset, \emptyset)$; `// First call`

In the base case, when dimension is 1D, each brink in the current slice Υ_B^i results in a box, which is inserted into $currentBboxes$. Boxes in a 2D slice Υ_B^i can be merged with boxes in the previous slice Υ_B^{i-1}, then, in the backtracking step of the recursion when dimension is 2D, function $mergeB()$ (Algorithm 3) is called, which compares all boxes β_1 in $currentBboxes$ with all boxes β_2 in Υ_B^{i-1} (stored in $prevBBoxes$) for merging. In this process, when merging property 1 is accomplished, $\beta_2 = \beta_2 \cup \beta_1$, and it is inserted into a array called $activeBoxes$. Otherwise, β_1 is inserted into $currentABoxes$ and $activeBoxes$. When the process finishes, the array $activeBoxes$ becomes $prevBBoxes$ in order to be compared with boxes in Υ_B^{i+1} in the next call.

Similar to the merging case in B-direction, boxes in a 3D slice Υ_A^j can be merged with boxes in the previous slice Υ_A^{j-1}. Once all the boxes of the current slice Υ_A^j have been computed and conveniently merged in B-direction, they are in $currentAboxes$. Then, in the backtracking step of the recursion when dimension is 3D, function $mergeA()$ (Algorithm 4) is called, which compares all boxes β_1 in $currentAboxes$ with all boxes β_2 in Υ_A^{j-1} (stored in $prevAboxes$) for merging. The steps in this function are quite similar to those in function $mergeB()$, but in this case, the merged boxes are finally inserted into the CUDB model Q.

Algorithm 2: doConversion(**Input** P:EVM, **Input/Output** Q:CUDB, **Input** dim:dimType, **Input** V_0, V_1:point3D)

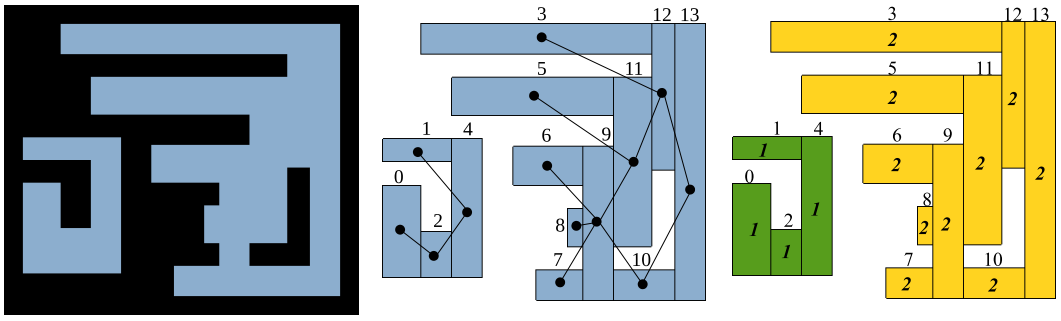
if $dim =$*1D* **then**
  foreach brink $br \in P$ **do**
    $V_0.C, V_1.C \leftarrow br.readBrink()$;
    $\beta = Box(V_0, V_1)$;
    Add β to $currentBBoxes$;
  end
else `//` $dim =$`2D or 3D`
  $Sec \leftarrow \emptyset$;
  $Cut, coordIni \leftarrow P.getNextCut()$;
  $Sec \leftarrow Sec \otimes Cut$;
  $Cut, coordFin \leftarrow P.getNextCut()$;
  while $Cut \neq \emptyset$ **do**
    if $dim =$*3D* **then**
      $V_0.A \leftarrow V_1.A$; $V_1.A \leftarrow coordFin$;
    else `//` $dim =$`2D`
      $V_0.B \leftarrow V_1.B$; $V_1.B \leftarrow coordFin$
    end
    $doConversion(Sec, Q, dim - 1, V_0, V_1)$;
    if $dim =$*3D* **then**
      $mergeA()$;
    else `//` $dim =$`2D`
      $mergeB()$;
    end
    $Sec \leftarrow Sec \otimes Cut$;
    $Cut, coordFin \leftarrow P.getNextCut()$;
  end
end

Note that, the adjacency information of boxes (ABN, AFN, BBN, BFN) is computed on the fly when tests for merging are performed.

3.3 CUDB to EVM Conversion

As in OUDB, all of the boxes in CUDB are disjoint, and according to the following EVM property:

— Let P and Q be two OPP such that $P \cap^* Q = \emptyset$, having $EVM(P)$ and $EVM(Q)$ as their respective models, then $EVM(P \cup^* Q) = EVM(P) \otimes^* EVM(Q)$.



Evolution of the boxes queue:

Label =1: {0} → {2} → {4} → {1} → { }

Label =2: {3} → {12} → {11, 13} → {13, 5, 9} → {5, 9,10} → { 9,10} → {10, 6, 7, 8} → { 6, 7, 8} → {7, 8} → {8} → { }

Fig. 5. 2D example of CUDB-CCL

Algorithm 3: mergeB()

$activeBoxes \leftarrow \{\emptyset\}$;
foreach $\beta_1 \in currentBBoxes$ **do**
  foreach $\beta_2 \in prevBBoxes$ **do**
    if $\overline{\beta_1}^B = \overline{\beta_2}^B$ **then** // Merging
     property 1
      $\beta_2 \leftarrow \beta_1 \cup \beta_2$;
      Add β_2 to $activeBoxes$;
      break;
    else if $\overline{\beta_1}^B \cap \overline{\beta_2}^B \neq \emptyset$ **then**
      Add β_2 to $\beta_1.BBN$;
    end
  end
  if β_1 was not merged **then**
    Add β_1 to $currentABoxes$;
    Add β_1 to $activeBoxes$;
    Add β_1 in BFN of each box in β_1.BBN;
  end
end
$prevBBoxes \leftarrow activeBoxes$;
$currentBBoxes \leftarrow \{\emptyset\}$;

Algorithm 4: mergeA()

$activeBoxes \leftarrow \{\emptyset\}$;
foreach $\beta_1 \in currentABoxes$ **do**
  foreach $\beta_2 \in prevABoxes$ **do**
    if $\overline{\beta_1}^A = \overline{\beta_2}^A$ **then** // Merging
     property 2
      $\beta_2.BBN = \beta_2.BBN \cup \beta_1.BBN$;
       $\beta_2.BFN = \beta_2.BFN \cup \beta_1.BFN$;
      $\beta_2.ABN = \beta_2.ABN \cup \beta_1.ABN$;
       $\beta_2.AFN = \beta_2.AFN \cup \beta_1.AFN$;
      $\beta_2 \leftarrow \beta_1 \cup \beta_2$;
      Add β_2 to $activeBoxes$;
      break;
    else if $\overline{\beta_1}^A \cap \overline{\beta_2}^A \neq \emptyset$ **then**
      Add β_2 to $\beta_1.ABN$;
    end
  end
  if β_1 was not merged **then**
    $Q.insertBox(\beta_1)$;
    Add β_1 to $activeBoxes$;
    Add β_1 in AFN of each box in β_1.ABN;
  end
end
$prevABoxes \leftarrow activeBoxes$;
$currentABoxes \leftarrow \{\emptyset\}$;

Therefore, a simple XOR operation of all the EVM-represented boxes is necessary to get the EVM-represented object. That is, let β_i be a box

in the CUDB-represented OPP P

$$EVM(P) = \bigotimes_{i=1}^{n_b} EVM(\beta_i), \tag{4}$$

where $EVM(\beta_i)$ is the EVM-representation of β_i and n_b is the number of boxes in CUDB.

3.4 Area and Volume Computation

Computing the volume (3D) or area (2D) is straightforward by doing a traversal of the boxes, and making a summation of each volume or area depending on the dimension:

$$Volume(P) = \sum_{i=1}^{n_b} Volume(\beta_i). \tag{5}$$

4 CUDB-based Connected Component Labeling

Since a CUDB-represented object contains the boxes neighborhood information, it can bee seen as an undirected graph. Thus, we proporse a new CUDB-CCL process based on the detection of connected components in graph theory.

Let $G = (V, E)$ be an undirected graph without self loops, with V being a set of vertices (the CUDB boxes) and E a set of edges defined by the neighborhood information (ABN, AFN, BBN and BFN). A connected component in G is a maximal subgraph $g = (V^g, E^g)$ in which for any two vertices $v, u \in V^g$ there exists an undirected path in G with v as start and u as end vertex [48]. A maximal subgraph means that for any additional vertex $w \in (V \backslash V^g)$ there is no path from any $v \in V^g$ to w.

Thus, the CUDB-CCL process has linear complexity, in terms of the sum of the numbers of vertices and edges of the graph, using either depth-first search or breadth-first search [29]. In either case, a search that begins at some box β, will find the entire connected component containing β. To detect all the connected components, a traversal of the boxes is performed, starting a new breadth-first search or depth-first search whenever a box that has not been already labeled is detected.

Algorithm 5: CCL(**Input/Output** Q:CUDB, **Output** cc:Integer)

```
cc ← 0 ; // Number of connected
 components
Φ = {∅} ; // Queue of box pointers
currentLabel ← 1;
foreach β ∈ Q do
    if β.getLabel() = ∅ then
        β.setLabel(currentLabel) ;
        Add β to Φ;
        while Φ ≠ ∅ do
            β1 = Φ.pop();
            foreach β2 ∈ (β1.ABN ∪ β1.AFN ∪
             β1.BBN ∪ β1.BFN) do
                if β2.getLabel() = ∅ then
                    β2.setLabel(currentLabel);
                    Φ.push(β2);
                end
            end
        end
        currentLabel + +;
    end
end
cc ← currentLabel − 1;
```

Algorithm 5 details the steps for the CUDB-CCL process using a breadth-first search strategy. Fig. 5 depicts the CUDB-CCL process for a 2D example, where the evolution of the boxes queue used by the algorithm is shown.

5 CUDB-based Exact Collision Detection

In this section, we present a CUDB-based collision detection algorithm. Given an environment composed of n objects, we assume that each object is its CUDB-representation and have the same ABC-ordering; otherwise, a preprocesing must be performed first in order to set the same ordering.

Due to the nature of the CUDB model, the proposed algorithm works only in static environments. The goal of this method is to check which objects overlap or are adjacent.

Since the CUDB-representation contains fewer elements than other representations, a straightforward solution could be to iteratively compare each of the boxes in an object with all of the boxes in the other objects (brute force). Nevertheless, taking advantage of the implicit order of the boxes in the CUDB model, unnecessary analysis can be avoided.

In the presented method, a discarding of those objects whose AABB do not collide is performed first. Then, as boxes in CUDB are ABC-sorted, all the remaining potentially colliding objects can be tested jointly, instead of testing them in pairs.

Let $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$ be a finite sequence of n CUDB-represented potentially colliding objects, and let $\Phi = \{\beta_1, \beta_1, \ldots, \beta_n\}$ be a set of box pointers, where each β_i points to a box in the object θ_i. Initially each β_i points to the first box of the corresponding object.

A collision detection between all of the boxes in Φ is performed first, followed by an iterative process. This process obtains the box β_{min} in Φ (β_i with the minimum ABC-position of its vertex V_0) and updates it with the next box in the corresponding object θ_{min}. If there are no more boxes in θ_{min}, this object is marked as not active.

Otherwise, β_{min} is compared for collision with all boxes $\beta_i \in \Phi, \forall i \neq min$ and with the subsequent neighboring boxes of each β_i, say β_t, until $\beta_t.V_0$ has an A-coordinate greater than $\beta_{min}.V_1$. Note that we do not need to compare B and C-coordinate. The main iterative process finishes when β_{min} cannot be defined, which means that all θ_i have been marked as not active. At the end, a set Δ, with object pairs (θ_i, θ_j) that collide or are adjacent has been defined.

Algorithms 6 and 7 detail the steps of this process. Function *getIndexMinBox()* returns the index min of the box in Φ with the minimum ABC-position of its vertex V_0, such that θ_{min} is marked as active. If all θ_i are marked as not active, this function returns $\emptyset$. The worst case time-complexity of the CUDB-based exact collision detection is $O(n \cdot m \cdot T)$, where n is the number of objects, m the number of boxes of the object having the maximum number of boxes, and T the total number of boxes in the n objects. In any case it holds that $n \leq m \leq T$.

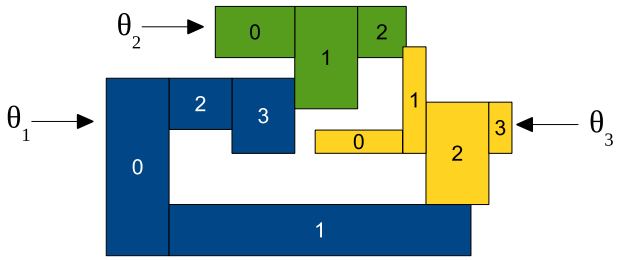
Algorithm 6: detectCollision(**Input** Θ:array or CUDB, **Output** Δ:array of CUDB pairs)

```
Δ = {∅} ; // Set of colliding object pairs
A ← {∅} ; // Vector of boolean for active θ_i
Φ ← {∅} ; // Vector of Box pointers
foreach θ ∈ Θ do
    β ← θ.getBox(0) ;
    Add β to Φ ;
    Add true to A ; // Mark all as active
end
foreach β_i ∈ Φ do // First collision test
    testCollision(Φ, β_i, Δ)
end
min ← getIndexMinBox(Φ, Act);
while min ≠ ∅ do
    β_min ← Θ[min].getNextBox();
    if β_min ≠ ∅ then
        Φ[min] = β_min;
        testCollision(Φ, β_min, S);
    else
        A[min] =false ; // Mark as not active
    end
    min ← getIndexMinBox(Φ, Act);
end
```

Algorithm 7: testCollision(**Input** Φ: Vector of Box pointers, **Input** β_i: Box, **Input / Output** Δ: array of CUDB pairs)

```
foreach β_j ≠ β_i ∈ Φ do
    β_t ← β_j;
    while β_t ≠ ∅ do
        if β_t.V0.A > β_i.V1.A then break; end;
        if β_i ∩ β_t ≠ ∅ then
            Add pair (θ_i, θ_j) to Δ; break;
        end
        β_t ← θ_j.getNextBox(β_t);
    end
end
```

Figure 6 depicts the process for a 2D example with three objects. It shows the evolution of the variables used by the algorithm.



Evolution of the variables →

Φ={0_1,0_2,0_3}	Φ={1_1,0_2,0_3}	Φ={2_1,0_2,0_3}	Φ={3_1,0_2,0_3}	Φ={3_1,1_2,0_3}	Φ={3_1,2_2,0_3}	Φ={3_1,2_2,1_3}
min=1	*min*=1	*min*=1	*min*=2	*min*=2	*min*=3	*min*=3
Δ={}	Δ={(1,3)}	Δ={(1,3)}	Δ={(1,3),(1,2)}	Δ={(1,3),(1,2)}	Δ={(1,3),(1,2),(2,3)}	Δ={(1,3),(1,2),(2,3)}
A ={T_1,T_2,T_3}	*A* ={T_1,T_2,T_3}	*A* ={T_1,T_2,T_3}	*A* ={T_1,T_2,T_3}	*A* ={F_1,T_2,T_3}	*A* ={F_1,T_2,T_3}	*A* ={F_1,F_2,T_3}

Fig. 6. 2D example of collision

6 CUDB Performance

CUDB has been compared with OUDB in number of elements and computation time for conversion to and from EVM and CCL (using OUDB-extended version [5]). The test datasets consists of 12 objects (see Fig. 7). All datasets come from public volume repositories, where from (g) to (l) are real volume models coming from CT or MRI scanners. The corresponding programs have been written in C++ and tested on a PC Intel®Core i7-4600M CPU@2.90GHz with 7.6 GB RAM and running Linux.

Table 1 shows the attributes of test datasets. For each dataset it shows size in voxels, number of foreground voxels ($|Fvoxels|$), number of triangles of a triangular surface mesh obtained via marching cubes [36], number of extreme vertices ($|EV|$), number of boxes in its XYZ-OUDB ($|OUDB|$) and XYZ-CUDB ($|CUDB|$) representation, ratio between $|OUDB|$ and $|CUDB|$, and number of connected components ($|CC|$) allowing non-manifold configurations.

Note in this table that, although the number of boxes depends on the ABC-sorting of the original EVM, CUDB produces less elements than the other representations in all cases, regarding OUDB in some of the dataset less than 10% of elements. For instance, the XYZ-CUDB representation of the Lines dataset has only 3.8% of boxes of the corresponding XYZ-OUDB representation (see Fig. 8).

Table 2 shows the performance of CUDB regarding OUDB. For each dataset it shows the time for: EVM to OUDB ($E \rightarrow O$) conversion, OUDB-CCL (O_{CCL}), EVM to CUDB ($E \rightarrow C$) conversion and CUDB-CCL (C_{CCL}). The last columns mean $t_O = E \rightarrow O + O_{CCL}$, $t_C = E \rightarrow C + C_{CCL}$, and the ratio between t_O and t_C.

Note that, although the conversion from EVM to CUDB is slightly slower than EVM to OUDB due to the extra effort to merge the boxes, the CUDB-based CCL process is much faster due to the less number of elements and that we avoid the use of a equivalence table, in all datasets more than an order of magnitude, even the temple dataset more than two orders of magnitude faster. In any case, when computing the number of connected components starting from the EVM model, CUDB is more efficient than OUDB..

In order to show the performance of the CUDB-based exact collision detection, three scenes are presented. The first one consists of seven datasets (see Fig. 9), where the size of each is around 256^3. The second scene consists of 250 spheres and 250 objects of the Star dataset randomly placed in a volume of 1000^3 voxels (see Fig. 10). The third scene consists of 2 objects of

Table 1. Attributes of the test datasets

Dataset	size	$\|Fvoxels\|$	$\|triangles\|$	$\|EV\|$	$\|OUDB\|$	$\|CUDB\|$	$\%\frac{\|CUDB\|}{\|OUDB\|}$	$\|CC\|$
(a) Cart	585×979×1000	4453852	4605468	502986	256370	100358	39.1	16
(b) Lines	500×500×500	11179011	6579736	4356	24851	954	3.8	1
(c) Cup	401×401×512	18027587	2628720	215050	236642	60429	25.5	1
(d) FanDisk	470×512×261	18706826	1336956	59290	49761	16733	33.6	1
(e) Robot	372×943×1000	29150579	4234084	126806	163281	34515	21.1	48
(f) Temple	925×1000×472	141371526	7467204	73902	260279	21084	8.1	99
(g) Aneurysm	213×215×240	69743	175064	50318	12825	10705	83.5	406
(h) Lobster	244×239×49	233509	311880	74724	27307	19322	70.8	53
(i) Engine	139×197×108	901818	663900	101114	47143	25524	54.1	9
(j) Beetle	411×371×247	1737343	567972	132184	47410	36052	76.0	17
(k) Colon	512×492×426	3995607	7772360	2142304	653717	473649	72.5	54829
(l) Mineral	376×375×206	7363953	6121828	833002	489585	232008	47.4	724

Table 2. OUDB and CUDB run time comparison in milliseconds

D.	OUDB		CUDB		t_O	t_C	$\%\frac{t_C}{t_O}$
	E→O	O_{CCL}	E→C	C_{CCL}			
(a)	406	**422**	467	**16**	828	483	58
(b)	26	**15**	37	**0.3**	41	37	90
(c)	324	**366**	360	**7**	690	367	53
(d)	76	**31**	78	**1**	107	79	74
(e)	204	**251**	253	**4**	455	257	56
(f)	263	**503**	299	**2**	766	301	39
(g)	45	**10**	53	**1**	55	54	98
(h)	95	**31**	100	**2**	126	102	81
(i)	66	**66**	96	**2**	132	98	74
(j)	88	**33**	114	**3**	121	117	97
(k)	1335	**1162**	1716	**65**	2497	1781	71
(l)	611	**926**	769	**33**	1537	802	52

the Cart dataset that have interlaced parts but do not collide (see Fig. 11).

Statistics of the collision detection test are shown in Table 3. For each scene: number of objects (n), number of boxes of the object having the maximum number of boxes (m), total number of boxes in the scene (T), number of detected collisions (i.e., number of object pairs $|pairs|$) and time to detect the collisions in milliseconds.

Note that, although scene 3 has less boxes than scene 2, the required time for collision detection is bigger. This is because there is any collision, which implies that there is not any early discarding and all boxes must be evaluated.

Table 3. Statistics of the collision scenes

Scene	n	m	T	$\|pairs\|$	Time
Scene 1	6	23410	84714	4	92
Scene 2	500	1280	546750	91	67
Scene 3	2	101743	202101	0	566

7 Conclusion and Future Work

We have presented in detail, a new decomposition model for OPP, CUDB, which is an improved version of the OUDB model. Algorithms for conversion to and from EVM, for CCL and exact collision detection have also been presented.

Experimental results show that CUDB is smaller in number of elements, and so in storage size.

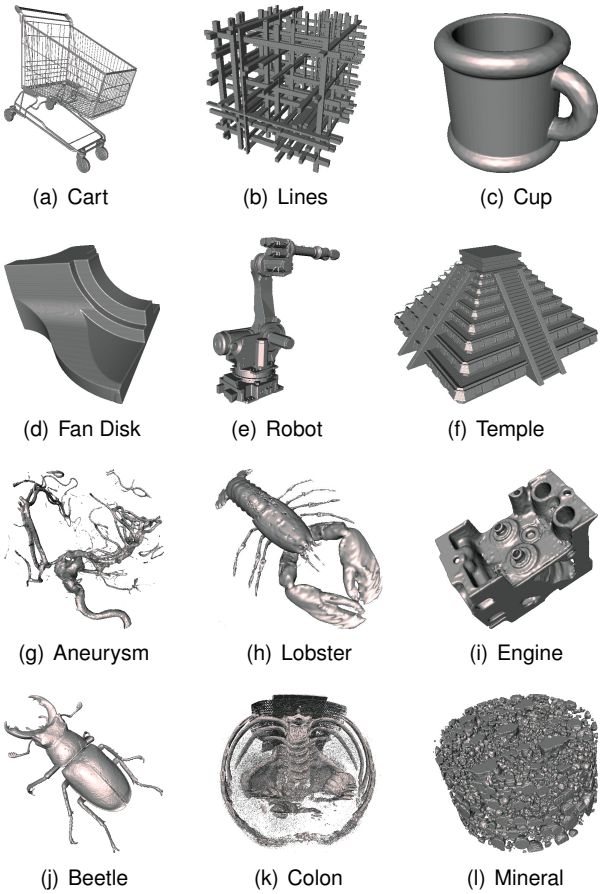


Fig. 7. Rendered images of the test datasets

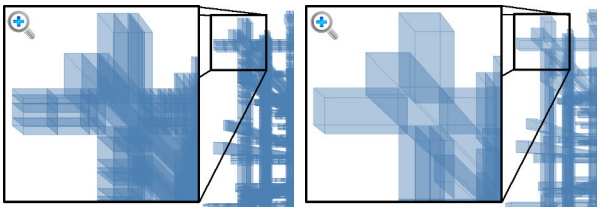


Fig. 8. Zoom in of Lines dataset. (a) XYZ-OUDB. (b) XYZ-CUDB

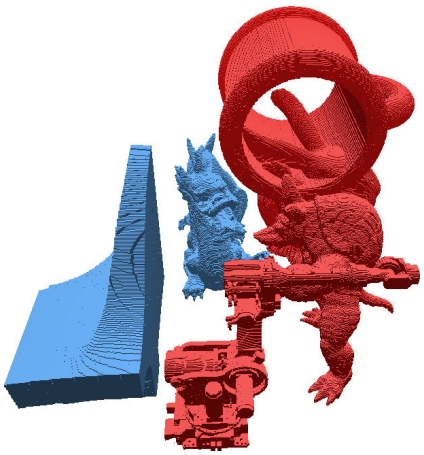


Fig. 9. Collision scene 1. Six datasets: Armadillo (18338 boxes), Bunny (23410 b.), Cup (15002 b.), Dragon (19413 b.), FanDisk (4719 b.), Robot (3832 b.). In red the objects that collide: Bunny with Cup and Armadillo, and the latter with Robot

It can be observed that computation of CCL is notably faster in CUDB than the improved version, OUDB-extended. Regarding the presented exact collision detection algorithm, although this is CPU-based, it is efficient when exact collision detection is required directly on CUDB models.

Table 4 resumes the pros and cons of CUDB with respect B-Rep and the hierarchical structures (HS) on the basis of some properties of solid representation schemes, and the CCL and collision detection performance.

The performance variability of the presented algorithms is caused by the dataset size but above all to their surface intricacy. Our method depends on the number of boxes, tightly related to the model's tortuosity (a property that represents the twist of a curve, i.e. the degree of turns or detours a model has [26], like the previous developed methods based on OPP.

Characteristics of CUDB model have been exploited in some applications as in a CUDB-based virtual porosimeter [44], which simulate mercury intrusion at increasing pressures, like the porosimeter lab. Also in a method to compute the Euler characteristic and the genus of binary volumes [14]. And a 2D version of the collision detection algorithm has been applied in a lossless simplification method to get a better shape preservation [15].

As future work, we are devising methods to compute the B-Rep model from the

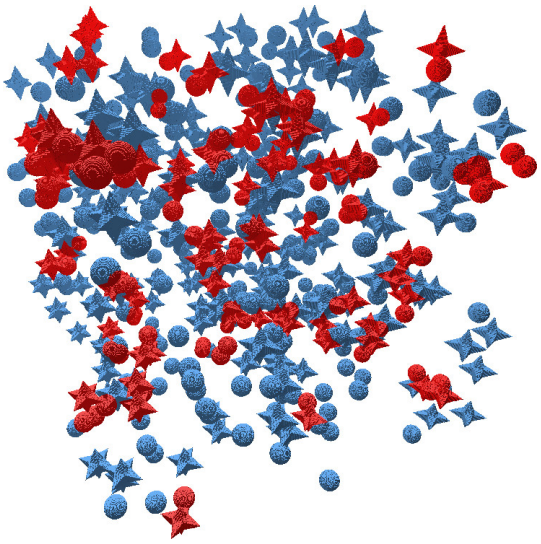


Fig. 10. Collision scene 2. 250 spheres (voxel size 50×50×50) and 250 objects of the Star dataset (70×95×70). Each sphere has 907 boxes and each Star 1280 boxes in their corresponding XYZ-CUDB representation. In red the objects that collide

Fig. 11. Collision scene 3. 2 objects of the Cart dataset. The original Cart has 100358 boxes, the rotated one 101743 boxes. Both in its XYZ-CUDB. The objects actually do not collide

CUDB-representation and to obtain the CUDB-represented complement. Another future work is a complete analysis of whether any of the six ABC-sortings in CUDB produces an optimal number of disjoint boxes that cover the OPP, otherwise, one could think of allowing overlapping boxes in order to further reduce this number.

Table 4. Comparative of CUDB vs. other representation models

	CUDB	B-Rep & HS
Accuracy	Exact representation for OPP.	Approximate representation [21].
Domain	Represents any solid.	B-Rep represents a very wide classes of objects, HS any solid [21].
Uniqueness	Six representations depending on the ordering.	Only octrees guarantees the uniqueness [21].
Storage (# of elements)	In general, OUDB has proven to be more compact than other models [42, 43], and here we have proven that CUDB is more compact than OUDB.	
CCL	In general, OUDB has proven to be faster than other models [42, 43], and here we have proven that CUDB is faster than OUDB.	
Exact collision detection	Only for static scenes	Applications in dynamic scenes and deformable objects [49].
Boolean Operations	Not yet devised methods, but can be performed via EVM.	Known operations [21].

Finally, we think that CUDB model can be used to compute structural parameters of biomaterials as those reported in [6] and [54] and new ones emerging in the bioengineering field.

References

1. **Aguilera, A. (1998).** *Orthogonal Polyhedra: Study and Application*. Ph.D. thesis, LSI-Universitat Politècnica de Catalunya.
2. **Aguilera, A. A. & Ayala, D. (1997).** Orthogonal Polyhedra as Geometric Bounds in Constructive Solid Geometry. *Fourth ACM Symposium on Solid Modeling and Applications*, ACM, pp. 56–67.
3. **Aguilera, A. A. & Ayala, D. (2001).** *Geometric Modeling*, volume 14 of *Computing Supplement*, chapter Converting Orthogonal Polyhedra from

Extreme Vertices Model to B-Rep and to Alternating Sum of Volumes. Springer-Verlag, pp. 1–28.

4. **Ayala, D., Vergara, E., & Vergés, E. (2007).** Improved skeleton computation of an encoded volume. *Proc. of Eurographics*, volume 2007, pp. 33–36.
5. **Ayala, D. & Vergés, E. (2008).** Improved virtual porosimeter. *CASEIB'08*.
6. **Ayala, D. & Vergés, E. (2009).** Structural parameters computation of a volume using alternative representations. *Proceedings of IV Iberoamerican Symposium in Computer Graphics*, DJ Editores, C.A., pp. 73–80.
7. **Ayala, D., Vergés, E., & Cruz, I. (2012).** A polyhedral approach to compute the genus of a volume dataset. *Proc. of the Int. Conf. GRAPP 2012*, SciTePress, pp. 38–47.
8. **Biedl, T. & Genç, B. (2011).** Reconstructing orthogonal polyhedra from putative vertex sets. *Computational Geometry*, Vol. 44, No. 8, pp. 409–417.
9. **Biswas, A., Bhowmick, P., Sarkar, M., & Bhattacharya, B. B. (2012).** A linear-time combinatorial algorithm to find the orthogonal hull of an object on the digital plane. *Information Sciences*, Vol. 216, pp. 176–195.
10. **Bournez, O., Maler, O., & Pnueli, A. (1999).** Orthogonal polyhedra: Representation and computation. *Hybrid Systems: Computation and Control*, pp. 46–60. LNCS 1569, Springer.
11. **Brunet, P., Juan, R., & Navazo, M. I. (1992).** Octree representations in solid modeling. *Progress in Computer Graphics*, Vol. 1, pp. 164–215.
12. **Cohen, J. D., Lin, M. C., Manocha, D., & Ponamgi, M. (1995).** I-collide: an interactive and exact collision detection system for large-scale environments. *Proc. of the 1995 Sym. on Interactive 3D graphics*, ACM, pp. 189–ff.
13. **Coming, D. S. & Staadt, O. G. (2008).** Velocity-aligned discrete oriented polytopes for dynamic collision detection. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 14, pp. 1–12.
14. **Cruz-Matías, I. & Ayala, D. (2013).** An efficient alternative to compute the genus of binary volume models. *Proc. of the Int. Conf. GRAPP 2013*, SciTePress, pp. 18–26.
15. **Cruz-Matías, I. & Ayala, D. (2014).** A new lossless orthogonal simplification method for 3D objects based on bounding structures. *Graphical Models*, Vol. 76, No. 4, pp. 181–201.
16. **Dang, T. & Maler, O. (1998).** Reachability analysis via face lifting. In *Hybrid Systems: Computation and Control*. Springer, pp. 96–109.
17. **Dillencourt, M., Samet, H., & Tamminen, M. (1992).** A general approach to connected-component labeling for arbitrary image representations. *Journal of the ACM*, Vol. 39, No. 2, pp. 253–280.
18. **Eppstein, D. & Mumford, E. (2010).** Steinitz theorems for orthogonal polyhedra. *Proceedings of the 2010 annual symposium on Computational geometry*, ACM, pp. 429–438.
19. **Esperança, C. & Samet, H. (1998).** Vertex representations and their applications in computer graphics. *The Visual Computer*, Vol. 14, pp. 240–256.
20. **Flores, J. A. M. (1999).** *Analysis and Visualization of Complex 3D Structures: a discrete boundary-based approach*. Ph.D. thesis, Ecole Nationale Supérieure des Télécommunications.
21. **Foley, J., Damm, A. V., Feiner, S., & Hughes, J. (1997).** *Computer graphics: principles and practice*. Pearson Education.
22. **Gagvani, N. & Silver, D. (2000).** Shape-based volumetric collision detection. *Proc. of the 2000 IEEE Sym. on Volume visualization*, ACM, pp. 57–61.
23. **Geleri, F., Tosun, O., & Topcuoglu, H. (2013).** Parallelizing broad phase collision detection algorithms for sampling based path planners. *Proc. of the 21st Euromicro Int. Conference on Parallel, Distributed, and Network-Based Processing*, IEEE, pp. 384–391.
24. **Greß, A. & Klein, R. (2004).** Efficient representation and extraction of 2-manifold isosurfaces using kd-trees. *Graphical Models*, Vol. 66, pp. 370–397.
25. **Grevera, G. J., Udupa, J. K., & Odhner, D. (2000).** An Order of Magnitude Faster Isosurface Rendering in Software on a PC than Using Dedicated, General Purpose Rendering Hardware. *IEEE Transactions Visualization and Computer Graphics*, Vol. 6, No. 4, pp. 335–345.
26. **Grisan, E., Foracchia, M., & Ruggeri, A. (2003).** A novel method for the automatic evaluation of retinal vessel tortuosity. *Proc. of the 25th Annual Int. Conf. of the IEEE EMBS*, volume 1, pp. 866–869.

27. **Gupta, S., Palsetia, D., Patwary, A., Mostofa, M., Agrawal, A., & Choudhary, A. (2014).** A new parallel algorithm for two-pass connected component labeling. *IEEE International Parallel & Distributed Processing Symposium Workshops (IPDPSW)*, IEEE, pp. 1355–1362.
28. **He, L.-F., Chao, Y.-Y., & Suzuki, K. (2013).** An algorithm for connected-component labeling, hole labeling and euler number computing. *Journal of Computer Science and Technology*, Vol. 28, No. 3, pp. 468–478.
29. **Hopcroft, J. & Tarjan, R. (1973).** Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, Vol. 16, No. 6, pp. 372–378.
30. **Jiménez, P., Thomas, F., & Torras, C. (2000).** 3D collision detection: A survey. *Computers & Graphics*, Vol. 25, No. 2, pp. 269–285.
31. **Kaufman, A. (1990).** *Volume Visualization*. IEEE Computer Society Press.
32. **Kaufman, A., Cohen, D., & Yagel, R. (1993).** Volume graphics. *Computer*, pp. 51–64.
33. **Lachaud, J. & Montanvert, A. (2000).** Continuous analogs of digital boundaries: A topological approach to iso-surfaces. *Graphical Models*, Vol. 62, pp. 129–164.
34. **Lee, E., Choi, Y., & Park, K. (1994).** A method of 3D object reconstruction from a series of cross-sectional images. *IEICE trans. inf and syst*, Vol. E77-D, No. 9.
35. **Lifeng, H., Xiao, Z., Bin, Y., Yun, Y., & Yuyan, C. (2014).** An efficient two-scan labeling algorithm for binary hexagonal images. *IEICE TRANSACTIONS on Information and Systems*, Vol. 97, No. 12, pp. 3244–3247.
36. **Lorensen, W. & Cline, H. (1987).** Marching cubes: A high resolution 3D surface construction algorithm. *ACM Computer Graphics*, Vol. 21, No. 4, pp. 163–169.
37. **Martínez, J., Pla, N., & Vigo, M. (2013).** Skeletal representations of orthogonal shapes. *Graphical Models*, Vol. 75, pp. 189–207.
38. **Muuss, M. J. & Butler, L. A. (1991).** *State of the Art in Computer Graphics: Visualization and Modeling*. Springer-Verlag.
39. **Park, S. C. & Choi, B. K. (2001).** Boundary extraction algorithm for cutting area detection. *Computer-Aided Design*, Vol. 33, No. 8, pp. 571–579.
40. **Quadros, W. R., Shimada, K., & Owen, S. J. (2004).** 3D discrete skeleton generation by wave propagation on PR-octree for finite element mesh sizing. *Proceedings of ACM Symposium on Solid Modeling and Applications*, pp. 327–332.
41. **Requicha, A. (1980).** Representations for rigid solids: Theory, methods and systems. *ACM Computing Surveys*, Vol. 12, No. 4, pp. 73–82.
42. **Rodríguez, J. & Ayala, D. (2003).** Fast neighborhood operations for images and volume data sets. *Computers & Graphics*, Vol. 27, pp. 931–942.
43. **Rodríguez, J., Ayala, D., & Aguilera, A. (2004).** *Geometric Modeling for Scientific Visualization*, chapter EVM: A Complete Solid Model for Surface Rendering. Springer-Verlag, pp. 259–274.
44. **Rodríguez, J., Cruz, I., Vergés, E., & Ayala, D. (2011).** A connected-component-labeling-based approach to virtual porosimetry. *Graphical Models*, Vol. 73, pp. 296–310.
45. **Rosenfeld, A. & Pfaltz, J. (1966).** Sequential operations in digital picture processing. *Journal of the ACM*, Vol. 13, No. 4, pp. 471–494.
46. **Samet, H. (1990).** *Applications of spatial data structures: Computer graphics, image processing, and GIS*. Addison-Wesley Longman Publishing.
47. **Samet, H. & Tamminen, M. (1985).** Bintrees, CSG trees, and time. *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '85, ACM, pp. 121–130.
48. **Seidl, T., Boden, B., & Fries, S. (2012).** Cc-mr–finding connected components in huge graphs with mapreduce. In *Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 458–473.
49. **Teschner, M., Kimmerle, S., Heidelberger, B., Zachmann, G., Raghupathi, L., Fuhrmann, A., Cani, M.-P., Faure, F., Magnenat-Thalmann, N., Strasser, W., et al. (2005).** Collision detection for deformable objects. *Computer graphics forum*, volume 24, Wiley Online Library, pp. 61–81.
50. **Thibault, W. & Naylor, B. (1987).** Set operations on polyhedra using binary space partitioning trees. *SIGGRAPH Comput. Graph.*, Vol. 21, No. 4, pp. 153–162.
51. **Thurfjell, L., Bengtsson, E., & Nordin, B. (1995).** A boundary approach to fast neighborhood operations on three-dimensional binary data. *CVGIP: Graphical Models and Image Processing*, Vol. 57, No. 1, pp. 13–19.

52. Udupa, J. K. & Odhner, D. (1991). Fast visualization, manipulation, and analysis of binary volumetric objects. *IEEE Computer Graphics & Applications*, pp. 53–62.

53. Udupa, J. K. & Odhner, D. (1993). Shell rendering. *IEEE Computer Graphics & Applications*, pp. 58–67.

54. Vergés, E. (2011). *Modeling, Analysis and Visualization of Porous Biomaterials*. Ph.D. thesis, LSI-Universitat Politècnica de Catalunya.

55. Vigo, M., Pla, N., Ayala, D., & Martínez, J. (2012). Efficient algorithms for boundary extraction of 2D and 3D orthogonal pseudomanifolds. *Graphical Models*, Vol. 74, pp. 61–74.

56. Vogiannou, A., Moustakas, K., Tzovaras, D., & Strintzis, M. G. (2010). Enhancing bounding volumes using support plane mappings for collision detection. *Computer Graphics Forum*, Vol. 29, No. 5, pp. 1595–1604.

57. Wilhems, J. & Gelder, A. V. (1992). Octrees for faster isosurface generation. *ACM Transactions on Graphics*, Vol. 11, No. 3, pp. 201–227.

58. Wong, W., Shih, F. Y., & Su, T. (2006). Thinning algorithms based on quadtree and octree representations. *Information Sciences*, Vol. 176, pp. 1379–1394.

59. Zachmann, G. (2002). Minimal hierarchical collision detection. *Proc. of the ACM Sym. on Virtual reality software and technology*, VRST '02, ACM, pp. 121–128.

Article received on 07/03/2016; accepted on 07/02/2017.
Corresponding author is Irving Cruz-Matías.

Design of Flat Halfband Filters With Sharp Transition and Differentiators through Constrained Quadratic Optimization

Miguel A. Platas-Garza, Johnny Rodríguez-Maldonado

Universidad Autonoma de Nuevo León, San Nicolás de los Garza, Nuevo León, Mexico

miguel.platasgrz@uanl.edu.mx, johnnyrodmal@gmail.com

Abstract. An alternative method for the design of type I Halfband FIR filters with flat magnitude and narrow transition bands is presented. The methodology shown is based on the derivation of a quadratic programming problem with inequality constraints, which represents a set of linear equations obtained from flat and ripple restrictions imposed over the frequency response of the filter. The design is based on maximally flat constraints. The obtained filters have narrow transition bands as compared to those presented in other maximally flat based designs. The proposed method is not ripple free as it does not take into account all the maximally flat restrictions. Then, control of side lobes and transition band is performed using a weighting matrix and inequality constraints as side lobes bounds. The design of type IV FIR digital differentiators through the proposed method is also shown. Examples of design, which compare the proposed method with others presented in the literature, are provided to verify the effectiveness of the proposed method.

Keywords. Halfband filters, digital differentiators, MAXFLAT constraints, weighted least square filter design.

1 Introduction

The use of halfband filters (HBF) and digital differentiators (DD) is key to perform several signal processing tasks. For example, HBF are used to perform the decimation and interpolation of signals [33, 36]; while DD are typically used when information about signal rate of change is required [21, 18].

Furthermore, maximally flat halfband filters (MFHB) and maximally lineal differentiators (MLD) have been widely studied [27, 39, 12, 29, 4, 20]. They present the maximum flatness/linearity of its frequency response around a particular frequency called flatness center.

In general, maximally flat (MAXFLAT) filters are of particular interest in applications that require smooth frequency responses or high attenuation in the stop-band [2, 24]. They are ripple-free and are of particular importance for the analysis of polynomial signals. Several designs have been proposed for the design of MAXFLAT filters since they emerged in [5]. They include FIR linear phase filters [26, 19], generalized filters [28, 40] and fractional delay filters [13, 38]. The main drawback of using MAXFLAT filters is the width of its transition band, which is superior to that presented in other designs which allow ripple, e.g., [22]. Additionally, in a MAXFLAT design, the bandwidth can only be decreased by increasing the filter length.

Several methodologies have been proposed to decrease the transition band in MAXFLAT designs preserving as much as possible the flatness of the frequency response. Some algorithms use a frequency different from zero as the flatness center to produce narrower filters [14, 15] while others increase the design flexibility by not taking into account all the MAXFLAT constraints [10, 31].

Recently, [7] presents a method for impose a prescribed cutoff frequency in a MAXFLAT design, but it was not applied to the design of halfband filters. With respect to MFHB filters, [8] presented a method for the design of MFHB filters with narrow transition bands. This method consists in convert the MLD presented in [11] into a MFHB filter using the transformation method described in

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2738

[23]. The filters obtained in [8] are designed with MAXFLAT constraints around $\omega = \pi/4$ rad. They have narrower transition bands than conventional methods. On the other hand, they are not ripple free and present the maximum ripple at $\omega = 0$.

The main purpose of this work is to obtain type 1 FIR flat HBF with linear phase and narrow transition bands, persevering as far as possible the passband flatness. The filters will be obtained by imposing a set of MAXFLAT constraints around a variable flatness center and the formulation of a convex quadratic programming problem (CQP) with inequality constraints. The application of this procedure for the design of type 4 FIR differentiators is also shown. The main differences with respect to previous work based on linear programming or CQP filter design [17, 6, 30, 16, 1, 31, 20, 34] are: (a) Design flexibility is enhanced due to a larger number of parameters. (b) It is possible to increase the accuracy of the frequency response at a particular frequency by introducing a weighting matrix. (c) The same algorithm can be used (with minor modifications) to design halfband filters or differentiators. We used QP instead of the more general convex optimization [35, 3] in order to obtain a straightforward formulation. Despite the fact that a simple and well-known optimization technique was used, it will be shown that the proposed method generates filters with better frequency characteristics than other designs which are also based on subsets of MAXFLAT constraints.

The rest of this document is structured as follows: As a brief summary Section 2 presents the conventional design of MFHB and MLD. Subsequently, in Sections 3 and 4 are presented the proposed designs for type 1 HBF and type 4 DD. Finally two design examples are shown in Section 5.

2 Summary on MAXFLAT Filter Design

In what follows we present the classical design of HBF and DD using MAXFLAT constraints. We consider type I FIR filters for the HBF and type IV FIR filters for the DD. Form more information, please refer to [32, 9].

2.1 Type I FIR MAXFLAT Halfband Filters

In the general case, a low-pass type I Halfband FIR filter of length $4N - 1$ has a frequency response given by:

$$H(\omega) = 0.5 + 2F(\omega), \quad -\pi < \omega \leq \pi, \tag{1}$$

with

$$F(\omega) = \sum_{k=1}^{N} h[2k-1] \cos\left((2k-1)\omega\right). \tag{2}$$

The function $H(\omega)$ is such that $H(\omega) + H(\omega - \pi) = 1$. The impulse response $h[k]$ of $H(\omega)$ depends only on the odd coefficients, as the even coefficients of $h[k]$ are fixed and given by:

$$\begin{aligned} h[0] &= 0.5, \\ h[2k] &= 0, \quad \text{for: } k = 1, 2, \ldots. \end{aligned} \tag{3}$$

In order to simplify the nomenclature, from here odd coefficients of $h[k]$ are represented by $f[k] = h[2k-1]$.

An example of the aspect of the frequency response of a low-pass $H(\omega)$ and a high-pass $H(\omega - \pi)$ HBF is given in the Fig. 1. The filters presented in Fig. 1 are MAXFLAT, *i.e.* their passband and stop-band fulfill a set of maximally flat constraints. Maximally flat constraints eliminate the maximum amount of derivates of $H(\omega)$ around a particular frequency $\omega = p$ called the flatness center:

$$\left.\frac{d^i H(\omega)}{d^i \omega}\right|_{\omega=p} = \begin{cases} 1 & i = 0, \\ 0 & i = 1, \ldots, L, \end{cases} \tag{4}$$

as we will consider halfband filters, eq. (4) also implies

$$\left.\frac{d^i H(\omega)}{d^i \omega}\right|_{\omega=\pi-p} = 0, \quad i = 0, 1, \ldots, L. \tag{5}$$

The conventional design of MFHB filters involves solving a system of $(N \times N)$ linear equations derived from the MAXFLAT constraints (4). This set is obtained by applying the constraints (4) to (1) with $L = N - 1$, and is given by

$$\sum_{k=1}^{N} (2k-1)^i C_{i+1}((2k-1)p) f[k] = \begin{cases} 0.25, & i = 0, \\ 0, & i = 1, 2, \ldots, L, \end{cases} \tag{6}$$

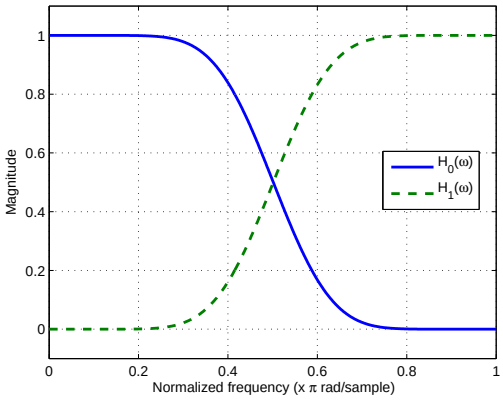


Fig. 1. Filter bank with low-pass and high-pass MFHB filters

where $C_n(\omega) = \cos(\omega)$ for odd values of n and $C_n(\omega) = \sin(\omega)$ for even values.

Usually the flatness center is $p = 0$. In this case all the odd order derivates in (6) are canceled automatically due to symmetry of $H(\omega)$. Then $L = 2(N-1)$ is chosen to obtain a $(N \times N)$ system of linear equations similar to (6), and given by

$$\sum_{k=1}^{N}(2k-1)^i f[k] = \begin{cases} 0.25, & i = 0, \\ 0, & i = 2, 4, \ldots, L. \end{cases} \quad (7)$$

Note that (7) is a Vandermonde system; then Vandermonde determinant can be used to solve (7) and find the following closed forms for the impulse response coefficients

$$f[k] = \frac{(-1)^{k-1}(2N-1)!!^2}{2^{2N}(N+k-1)!(N-k)!(2k-1)}. \quad (8)$$

Fig. 1 shows the frequency response of a bank of MFHB filters designed through this approach. The filters presented in Fig. 1 were obtained using (8) with $N = 5$.

2.2 Type IV FIR Maximally Linear Digital Differentiators

A type 4 maximally linear (MAXLIN) differentiator of length $2N$ has a frequency response given by $H(\omega) = 2F(\omega)$, with

$$F(\omega) = \sum_{k=1}^{N} h[k-1/2]\sin\left((k-1/2)\omega\right). \quad (9)$$

As for the MAXFLAT halfband case, from here we will denote $f[k] = h[k-1/2]$ as the impulse response coefficients. In this work type IV filters were preferred over type III as type IV filters are not restricted to $H(\omega) = 0$ at $\omega = \pi$. Then they are more appropriate to design full-band DDs.

If L MAXLIN constraints are imposed over $H(\omega)$ at $\omega = p$

$$\left.\frac{d^i H(\omega)}{d^i \omega}\right|_{\omega=p} = \begin{cases} p & i = 0, \\ 1 & i = 1, \\ 0 & i = 2, 3, \ldots, L. \end{cases} \quad (10)$$

Then, the following linear system of equations:

$$\sum_{k=1}^{N}(k-1/2)^i C_{i+2}((k-1/2)p) f[k] = \quad (11)$$
$$\begin{cases} \frac{p}{2}, & i = 0, \\ 0.5, & i = 1, \\ 0, & i = 2, 3, \ldots, L, \end{cases}$$

is obtained.

Considering $p = 0$, and $L = 2N-1$, (11) can be simplified to the $(N \times N)$ Vandermonde system:

$$\sum_{k=1}^{N}(k-1/2)^i f[k] = \begin{cases} 0.5, & i = 1, \\ 0, & i = 3, 5, \ldots, 2N-1; \end{cases} \quad (12)$$

with the solution expressed by

$$f[k] = \frac{(-1)^{k+1}(2N-1)!!^2}{2^{2N}(k-1/2)^2(N-k)!(N+k-1)!}. \quad (13)$$

3 Proposed Design for Type I FIR Half-band Filters

Next, we present the proposed methodology for the design of type I FIR flat halfband filters with narrow transition bands by means of quadratic programming optimization.

We consider $L \leq N-1$ MAXFLAT constraints around the flatness center $\omega = p$, where $0 \leq p \leq \frac{\pi}{2}$. An overdetermined system of linear equations

$$Af = c, \tag{14}$$

is imposed over $f[k]$ in order to meet (4), where the r-th element of the c-th column of $A \in \mathbb{R}^{N\times(L+1)}$ is

$$a_{r,c} = (2c-1)^{r-1} C_r((2c-1)p), \tag{15}$$

where $c \in \mathbb{R}^{L+1}$ and $f \in \mathbb{R}^N$ are given by:

$$c = \begin{pmatrix} 0.25 & 0 & 0 & \dots & 0 \end{pmatrix}^T, \tag{16}$$

and

$$f = \begin{pmatrix} h[1] & h[3] & h[5] & \dots & h[2N-1] \end{pmatrix}^T. \tag{17}$$

A vector with M samples of $H(\omega)$ uniformly distributed over $\omega \in [0, \frac{\pi}{2}]$ can be obtained using $H = 0.5u + Wh_o$, where $u \in R^M$ represents an unitary vector, and $W \in \mathbb{C}^{M\times N}$ has elements given by:

$$w_{r,c} = 2\cos\left((2r-1)(c-1)\Delta_\omega\right), \qquad \begin{matrix} r = 1, 2, \dots, M, \\ c = 1, 2, \dots, N. \end{matrix} \tag{18}$$

with $\Delta_\omega = \pi/2M$.

On the other hand, If M samples of an uniform sampled version of a desired frequency response $H_d(\omega)$ over $\omega \in [0, \frac{\pi}{2}]$ are defined as a vector H_d; then, the design of the halfband filter can be stated as the optimization problem: Find the set of coefficients f that minimize the energy of the difference between the ideal H_d and the proposed $H = 0.5u + Wf$ frequency responses. The problem can be stated as the following quadratic programming optimization problem with equality constraints.

Find the best f for

$$\text{Minimize:} \quad J = (F_d - Wf)^T R (F_d - Wf), \tag{19a}$$

$$\text{subject to:} \quad Af = c, \tag{19b}$$

where $F_d = H_d - 0.5u$, $R \in \Re^{M\times M}$ is a weighting matrix, and the pair (R, W) is restricted to $W^T RW > 0$ to ensure the convexity of the problem.

The solution of the QP problem presented in (19) is

$$\begin{aligned} \hat{f} \;=\; & (W^T RW)^{-1} \\ & \left[W^T RH_d - A^T (A(W^T RW)A^T)^{-1}\right] \\ & \left[A(W^T RW)^{-1} W^T RH_d - c\right] \end{aligned} \tag{20}$$

and its derivation through Lagrange multipliers is shown in the appendix.

3.1 Selection of the Weighting Matrix R

The matrix R in (19) and (20) could be used to weight certain frequencies. It is possible to use this matrix to highlight frequencies which are located far from the flatness center p. The frequencies close to $\omega = p$ could receive less weight through R because the flat constraints (4) ensure a good approximation of $H(\omega)$ to $H_d(\omega)$ around $\omega = p$ and $\omega = \pi - p$.

In Fig. 2 are shown the results obtained through the use of (20), with $N = 10$, $L = 3$, $p = 0$ and R being a diagonal matrix, with the elements of its diagonal given by the Kaiser window. Specifically, the k-th element of the diagonal of R is given by:

$$\begin{aligned} r_{k,k} \;=\; & \frac{I_0\left(\pi\alpha\sqrt{1-\left(\frac{2k}{M-1}-1\right)^2}\right)}{I_0(\pi\alpha)} + \delta_r, \\ & 0 \leq k \leq M-1, \end{aligned} \tag{21}$$

with I_0 being the zeroth order modified Bessel function of the first kind, α a non-negative real number that define the shape of the window, and $\delta_r \lll 1$ being an arbitrary positive real number which ensures the existence of R^{-1}. The effect of R over $H(\omega)$ is shown in Fig. 2 for different values of α.

Note from Fig. 2 that, for the window used, a greater α gives more weight to the frequencies near $\omega = \pi/4$ and located outside the flat bands. Then, a better approximation to $H_d(\omega)$ around $\omega = \pi/4$ is obtained.

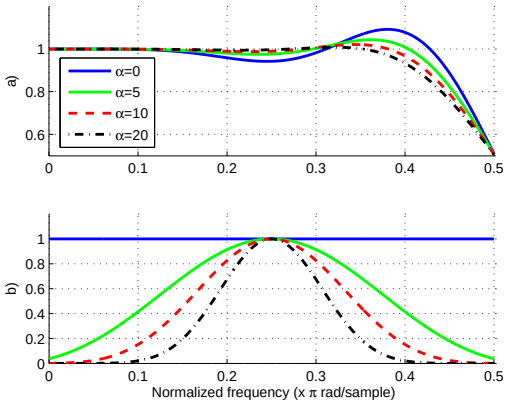


Fig. 2. The parameter R and its effect over the frequency response. (a) Frequency response and (b) The weighting function used

3.2 Side Lobe Control

The designed filters are susceptible to the presence of ripple in their bands as the design does not take into account the maximum number of flat constraints. If it is desired to minimize this ripple in certain frequency band $\omega \in [0, \omega_p]$, it is possible to add the linear box constraint

$$(0.5 + \delta_l)u \leq W_p f \leq (0.5 - \delta_l)u \tag{22}$$

on $F(\omega)$, where δ_l represents the desired level of ripple in the band $\omega \in [0, \omega_p]$ and $W_p f$ the frequency response of the filter in the same band. The matrix W_p represents a subset of the rows of the matrix W related to the frequencies within the band $\omega \in [0, \omega_p]$. The design problem is then given by the following quadratic programming problem with inequality constraints:

Find f for

$$\text{Minimize:} \quad J = (F_d - Wf)^T R(F_d - Wf), \tag{23a}$$

$$\text{subject to:} \quad Af = c, \tag{23b}$$

$$(0.5 + \delta_l)u \leq W_p f \leq (0.5 - \delta_l)u. \tag{23c}$$

The solution of (23) can be obtained using a variety of iterative algorithms found in the literature (e.g. Interior point [25, page 537] or active set [37, page 455]). In addition, there are numerous software packets available to solve such problems (e.g. Optimization toolbox for MATLAB, IBM ILOG CPLEX Optimization Studio, Gurobi, etc.).

Design approaches similar to that proposed in (23) can be consulted in [17, 6, 30, 31]. In contrast to these, the main differences of the proposed method are: Design flexibility is increased due to a larger number of design parameters $(N, L, P, R, H_d, \delta_l, \omega_p)$. Furthermore, using a criterion in the sense of weighted least squares enables a particular emphasis on specific frequencies. We also take into account inequality constraints (23c) to constrain the ripple on the passband and stop-band. Finally, note that as the desired frequency response H_d is not fixed to the ideal frequency response, any desired H_d can be used.

4 Proposed Design for Type IV FIR Digital Differentiators

The QP optimization problems shown in (19) and (23) can be used to design a type 4 DD instead of a type I HBF if the elements in (19) and (23) are redefined as follows:

— $F_d \in \mathrm{R}^M$ represents M uniformly distributed samples of the DD frequency response in $\omega \in [0, \pi]$. And, for the ideal case, it has elements f_r given by:

$$f_r = (r-1)\Delta_\omega \quad r = 1, 2, \ldots, M, \tag{24}$$

with $\Delta_\omega = \pi/M$.

— $W \in \mathrm{R}^{M \times N}$ represents a linear transformation from time to frequency domain and its elements are given by:

$$w_{r,c} = \sin\left(c(r-1)\Delta_\omega\right), \quad \begin{matrix} r = 1, 2, \ldots, M, \\ c = 1, 2, \ldots, N. \end{matrix} \tag{25}$$

— $A \in \mathrm{R}^{N \times (L+1)}$ and $c \in \mathrm{R}^{L+1}$ represent the flat constraints over the DD frequency response. A has elements:

$$a_{r,c} = c^{r-1} C_{r+1}(cp) \tag{26}$$

for $p \neq 0$, or

$$a_{r,c} = c^{2r-1} \tag{27}$$

for $p = 0$.

Where c is given by

$$c = \begin{pmatrix} \frac{p}{2} & \frac{1}{2} & 0 & \dots & 0 \end{pmatrix} \tag{28}$$

for $p \neq 0$, or

$$c = \begin{pmatrix} \frac{1}{2} & 0 & 0 & \dots & 0 \end{pmatrix} \tag{29}$$

for $p = 0$.

5 Design Examples

Next, we present two design examples. The first of them deals with the design of a flat HBF with narrow transition bands, while the second one shows the design of a full-band DD with a flatness band around $\omega = 0$. In both cases, results obtained are compared with different conventional designs.

5.1 Example 1 - Halfband Filter Design

In this example we design a HBF subject to a maximum ripple of $\delta_r = 0.005$ in the band $\omega \in [0, 0.32\pi]$. For this case, we use a length of 13 samples ($N = 5$). The rest of the used parameters are: ($L = 2$, $p = 0$, $\alpha = 0$, $H_d = u$). Under these conditions, the QP problem expressed in (23) was solved using the active set method. Specifically, we use the `quadprog` function of the Matlab optimization toolbox. It takes 9 iterations to find the local minimum solution for the algorithm; which, in view of the fact that the optimization problem is convex, is also a global solution.

Results obtained for the conventional MFHB method (obtained from eq. (8) and labeled M1), the proposed method in [8] (labeled as M2) and the proposed method of this work (solution of (23) and labeled as M3) are shown in the Table 1 and the Fig. 3. The Fig. 4 shows a zoom to the ripple in the passband of the filter. Note from Figs. 3 and 4 that the bandpass of the filter M3 was increased with respect to the classical design M1 and it is similar to M2. Furthermore, note from Fig. 4 that M3 satisfy the design bounds; the ripple of M3 was decreased by a factor of 5 with respect to the one

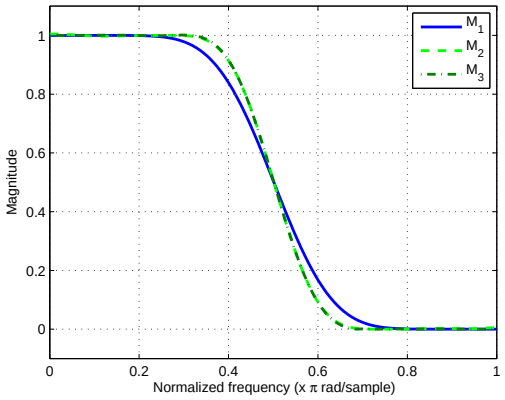


Fig. 3. Frequency responses of the halfband filters obtained from eq. (8) (labeled as M1), the design presented in [8] (labeled as M2) and the proposed method (labeled as M3)

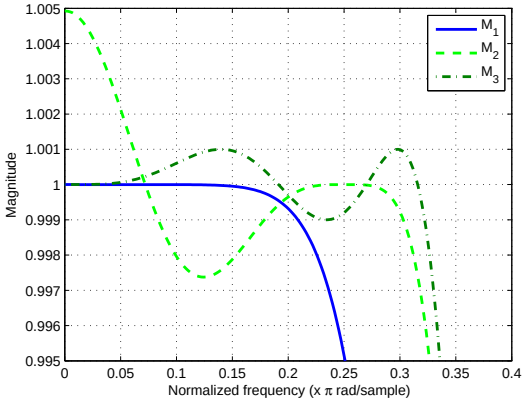


Fig. 4. Zoom to the frequency responses of the Fig. 3

obtained using M2. In addition, the ripple in M3 can be decreased if a lower δ_r is imposed and bandwidth is reduced.

5.1.1 Designs for Different Values of N

We also compare M1, M2 and M3 for different lengths. The filters obtained with $N = \{5, 10, 15, 20, 25, 30\}$ for each case (M1, M2, or M3) are shown in Fig. 5. For each filter in the M3 group we use ($L = N - 3$, $p = \pi/8$, $\alpha = 0$, $H_d = u$, $\delta_r = 1 \times 10^{-}4$).

Table 1. Impulse responses for the different HBF used. Where M1 stands for the conventional MFHB method (obtained from eq. (8), M2 for the filter obtained by applying the method shown in [8], and M3 for the approach of this work (solution of eq. (23))

Coefficients	M1	M2	M3
h_0	0.5	0.5	0.5
$h_{\pm 1}$	0.302810668945312	0.310119240690946	0.310443505472700
$h_{\pm 3}$	-0.067291259765625	-0.083278105260414	-0.084024873191551
$h_{\pm 5}$	0.017303466796875	0.032731601028004	0.032517775894448
$h_{\pm 7}$	-0.003089904785156	-0.010461516825066	-0.011146406816043
$h_{\pm 9}$	0.000267028808594	0.003349162494984	0.002209998640447

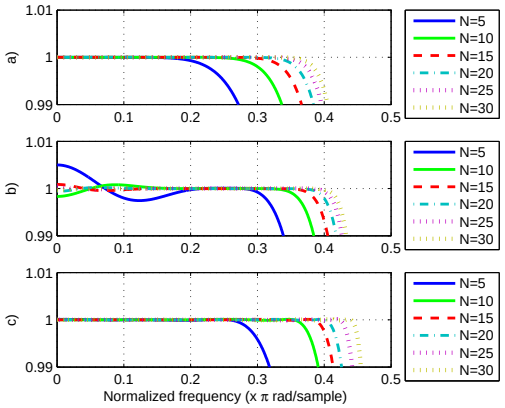


Fig. 5. Frequency responses of halfband filters obtained for different lengths from a) M1, b) M2 and c) M3. The length of each filter is related to N by the relation $4N-1$

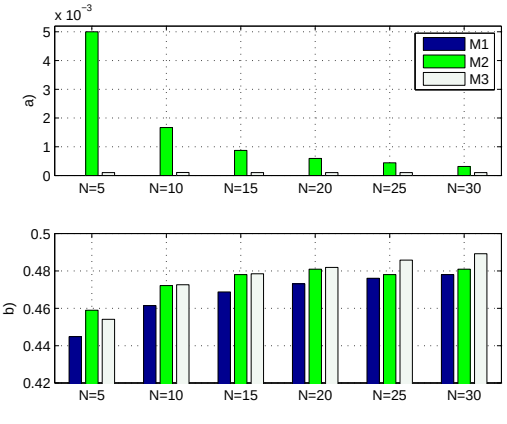


Fig. 6. Characteristics of the frequency responses of M1, M2 and M3 for different lengths. a) Maximum ripple in the pass-band and b) 3-db Bandwidth in normalized frequency. The maximum ripple of M1 is omitted because it is ripple free. The length of each filter is is related to N by the relation $4N-1$

We compare the performance of each group using the maximum ripple and the 3-dB bandwidth, this comparison is shown in Fig. 6. Note from Figs. 5 and 6.b that, under the same filter length, the bandwidth of the M3 group is in general bigger (with exception of N=5) that the one of the groups M1 and M2.

Also note form Fig. 6.a that the maximum ripple of M3 is always lower than that of M2 and within the imposed bound.

5.2 Example 2 - Full-band Digital Differentiator Design

We present the design of one full-band DD of a length equal to 34 samples $(N=17)$. The design was made through the solution of (23), using the following modified constraint (23c):

$$(1-\delta_l)Fd \leq W_p f \leq (1+\delta_l)Fd \qquad (30)$$

with $\delta_l = \pi/100$.

The Fig. 7.a shows the frequency response obtained for $p = \pi/4$ and $L = 2$; the error between the desired frequency response and the one obtained is shown in Fig. 7.b. Note from Fig. 7.b that the frequency response obtained meets the imposed constraints.

The absolute value of the error $|H(\omega) - H_d(\omega)|$ is shown in Fig. 8 for different values of p. Note that in all the cases, the error is such that (30) holds. In addition; it is possible to note from Fig. 8 that the bandwidth increases at higher p.

The comparison of the DD obtained with a DD designed from Parks & Mcmillan algorithm and a MAXLIN DD designed from (13) is presented in Figs. 9 and 10. All of the differentiators used

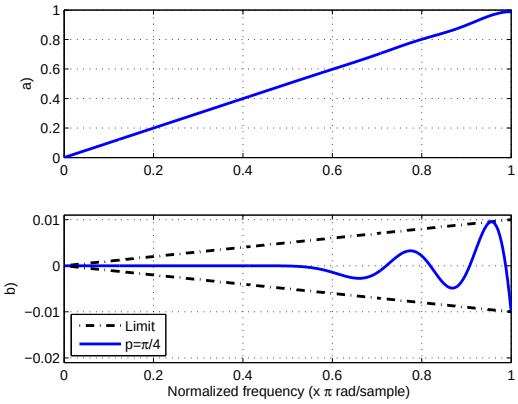


Fig. 7. Results obtained for the design of a FIR type 4 full-band differentiator. (a) Obtained frequency response, (b) Error obtained with respect to H_d

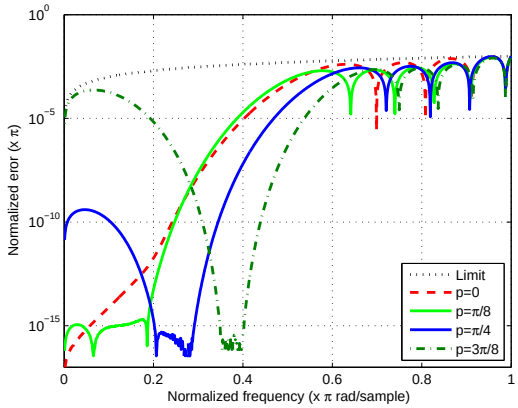


Fig. 8. Absolute value of the error $H - H_d$ obtained for several values of p for a DD design

have the same length, their impulse responses are shown in Table 3.

Note from Fig. 10 that we obtain a trade-off between the equiripable and the MAXLIN designs. The magnitude of the error in the proposed design is lower than that obtained by the equiripable differentiator at low frequencies around the flatness center.

Also, the magnitude of the error for the proposed design is lower than the one presented by the classical MAXLIN design at high frequencies due to the imposed bounds. Finally, the impulse

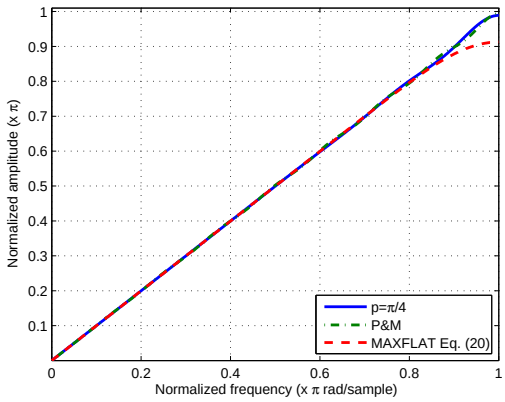


Fig. 9. Comparison of the DD obtained with $p = \pi/4$ (solid line) with other designs

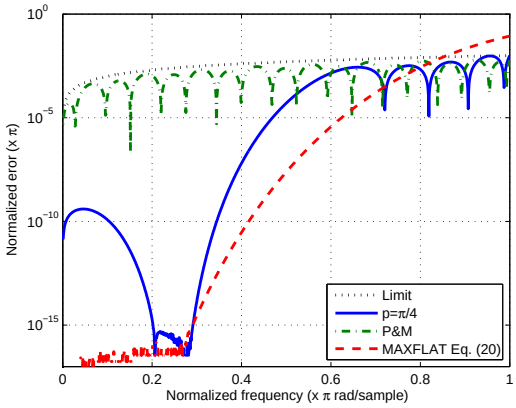


Fig. 10. Error with respect to H_d of the DD obtained with $p = \pi/4$ (solid line) and other designs

responses related to the filters used in this section are shown in Tables 2 and 3.

6 Conclusions

A design method for flat halfband filters with narrow bands via constrained optimization was presented. This method is based on the generation of one or more degrees of freedom in a MAXFLAT design. The reduced set of MAXFLAT constraints is added as a set of equality constraints to the QP problem. The ripple in the passband is bounded by linear

Table 2. Impulse responses computed via the proposed approach with $p = \{0, \frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{8}\}$

	$p = 0$	$p = \pi/8$	$p = \pi/4$	$p = 3\pi/8$
$h_{1/2}$	1.272703838491763	1.274348013359160	1.273528031915931	1.273461449991782
$h_{3/2}$	-0.142287195016303	-0.141924979204911	-0.142462524577607	-0.142079586040318
$h_{5/2}$	0.052423087839161	0.051284304032111	0.052388868323852	0.051992995508109
$h_{7/2}$	-0.026567069329830	-0.026994098090403	-0.027053722669176	-0.026873618815548
$h_{9/2}$	0.015354474001657	0.016967452128100	0.016044337311130	0.016250530090752
$h_{11/2}$	-0.010855600973637	-0.011140934198855	-0.010715801464035	-0.010560657963380
$h_{13/2}$	0.009698337795099	0.007688726157109	0.008437720054165	0.007797913925409
$h_{15/2}$	-0.008903873670370	-0.006305971304445	-0.007382522927876	-0.006342132805407
$h_{17/2}$	0.007010927210138	0.005838034604622	0.006215821019625	0.005645067912685
$h_{19/2}$	-0.004427265439342	-0.004955317932812	-0.004546053985099	-0.004697654194456
$h_{21/2}$	0.002197601711841	0.003393757921387	0.002753457699602	0.003580453593496
$h_{23/2}$	-0.000847346480109	-0.001788995561542	-0.001347479576037	-0.002282728435341
$h_{25/2}$	0.000249327969264	0.000708517283698	0.000521000323205	0.001247359342972
$h_{27/2}$	-0.000054251412312	-0.000204605386659	-0.000154118617243	-0.000537509605368
$h_{29/2}$	0.000008248148484	0.000040839335181	0.000033009418432	0.000188854663848
$h_{31/2}$	-0.000000783915838	-0.000005056684141	-0.000004595746280	-0.000044818425536
$h_{33/2}$	0.000000035119372	0.000000293670706	0.000000315545323	0.000007571855422

Table 3. Impulse responses for classical MAXLIN and the Parks & Mcmillan digital differentiators used

	Eq (13)	P&M
$h_{1/2}$	1.254655096049274	1.273366106881952
$h_{3/2}$	-0.123916552696225	-0.141600103038003
$h_{5/2}$	0.035218388661032	0.051060812829298
$h_{7/2}$	-0.012577995950369	-0.026118720110096
$h_{9/2}$	0.004710278318862	0.015856075357254
$h_{11/2}$	-0.001719906283221	-0.010665535760282
$h_{13/2}$	0.000588936265235	0.007683784426866
$h_{15/2}$	-0.000184315238564	-0.005814373497556
$h_{17/2}$	0.000051659288317	0.004569866554243
$h_{19/2}$	-0.000012724938695	-0.003703666198424
$h_{21/2}$	0.000002700589576	0.003090684240953
$h_{23/2}$	-0.000000482430463	-0.002646361955415
$h_{25/2}$	0.000000070401577	0.002332418442918
$h_{27/2}$	-0.000000008047734	-0.002176306513232
$h_{29/2}$	0.000000000675095	0.002370421782109
$h_{31/2}$	-0.000000000036925	-0.004961109469135
$h_{33/2}$	0.000000000000987	0.003665265031042

inequality constraints added to the optimization problem. The proposed methodology was applied to the design of type I FIR halfband filters and full-band type IV FIR differentiators.

With respect to the designed differentiators, only full-band differentiators were considered but in general the proposed approach can be used to design band-pass type IV and type III differentiators with some minor modifications. Design examples were presented to demonstrate the effectiveness of the proposed method to design halfband filters and full-band differentiators.

As it can be seen in the results section, it was shown that the proposed method has the advantage of producing flat halfband filters with narrower transition bands in comparison with classical MAXFLAT designs. Finally, it is important to highlight that as this method is based on quadratic programming it has some inherent limitations. For example, there are not closed expressions for the coefficients of the impulse response; and if design specifications are more demanding the feasible space will be reduced.

References

1. **Chan, S.-C., Pun, C. K., & Ho, K.-L. (2004).** New design and realization techniques for a class of perfect reconstruction two-channel fir filterbanks and wavelets bases. *Signal Processing, IEEE Transactions on*, Vol. 52, No. 7, pp. 2135–2141.
2. **Chen, Y. & Gong, Q. (2011).** Small-space microphone array fractional delay algorithm based on fir filter for cochlear implant. *Tsinghua Science & Technology*, Vol. 16, No. 1, pp. 90–94.

3. **Davidson, T. (2010).** Enriching the Art of FIR Filter Design via Convex Optimization. *IEEE Signal Processing Magazine*, Vol. 27, No. 3, pp. 89–101.
4. **de la O Serna, J. A. & Platas-Garza, M. A. (2011).** Maximally flat differentiators through wls taylor decomposition. *Digital Signal Processing*, Vol. 21, No. 2, pp. 183–194.
5. **Herrmann, O. (1971).** On the approximation problem in nonrecursive digital filter design. *Circuit Theory, IEEE Transactions on*, Vol. 18, No. 3, pp. 411–413.
6. **Horng, B.-R. & Willson Jr, A. N. (1992).** Lagrange multiplier approaches to the design of two-channel perfect-reconstruction linear-phase fir filter banks. *Signal Processing, IEEE Transactions on*, Vol. 40, No. 2, pp. 364–374.
7. **Jeon, J. & Kim, D. (2012).** Design of nonrecursive fir filters with simultaneously maxflat magnitude and prescribed cutoff frequency. *Digital Signal Processing*, Vol. 22, pp. 1085–1094.
8. **Khan, I. R. (2010).** Flat magnitude response fir halfband low/high pass digital filters with narrow transition bands. *Digital Signal Processing*, Vol. 20, No. 2, pp. 328–336.
9. **Khan, I. R. & Okuda, M. (2004).** Design of fir digital differentiators using maximal linearity constraints. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. 87, No. 8, pp. 2010–2017.
10. **Khan, I. R. & Okuda, M. (2006).** New designs of maxflat fir halfband low/high pass digital filters with narrow transition bands. *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 3, IEEE, pp. III–III.
11. **Khan, I. R. & Okuda, M. (2007).** Finite-impulse-response digital differentiators for midband frequencies based on maximal linearity constraints. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, Vol. 54, No. 3, pp. 242–246.
12. **Khan, I. R., Okuda, M., & Ohba, R. (2005).** Digital differentiators for narrow band applications in the lower to midband range. *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, IEEE, pp. 3721–3724.
13. **Koshita, S., Abe, M., & Kawamata, M. (2014).** A simple ladder realization of maximally flat allpass fractional delay filters. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, Vol. 61, No. 3, pp. 203–207.
14. **Kumar, B. & Dutta Roy, S. (1989).** Maximally linear FIR digital differentiators for high frequencies. *IEEE Transactions on Circuits and Systems*, Vol. 36, No. 6, pp. 890–893.
15. **Kumar, B., Dutta Roy, S., & Shah, H. (1992).** On the design of FIR digital differentiators which are maximally linear at the frequency pi;/p, p isin; positive integers. *IEEE Transactions on Signal Processing*, Vol. 40, No. 9, pp. 2334–2338.
16. **Lu, W.-S. & Hinamoto, T. (2003).** Optimal design of iir digital filters with robust stability using conic-quadratic-programming updates. *Signal Processing, IEEE Transactions on*, Vol. 51, No. 6, pp. 1581–1592.
17. **Medlin, G. W., Adams, J. W., & Leondes, C. (1988).** Lagrange multiplier approach to the design of fir filters for multirate applications. *Circuits and Systems, IEEE Transactions on*, Vol. 35, No. 10, pp. 1210–1219.
18. **Messina, A. (2004).** Detecting damage in beams through digital differentiator filters and continuous wavelet transforms. *Journal of Sound and Vibration*, Vol. 272, No. 1, pp. 385–412.
19. **Miller, J. (1972).** Maximally flat nonrecursive digital filters. *Electronics Letters*, Vol. 8, No. 6, pp. 157–158.
20. **Nongpiur, R. C., Shpak, D. J., & Antoniou, A. (2014).** Design of iir digital differentiators using constrained optimization. *IEEE Transactions on Signal Processing*, Vol. 62, No. 7, pp. 1729–1739.
21. **Pan, J. & Tompkins, W. J. (1985).** A real-time qrs detection algorithm. *Biomedical Engineering, IEEE Transactions on*, , No. 3, pp. 230–236.
22. **Parks, T. & McClellan, J. (1972).** Chebyshev approximation for nonrecursive digital filters with linear phase. *Circuit Theory, IEEE Transactions on*, Vol. 19, No. 2, pp. 189–194.
23. **Pei, S.-C. & Shyu, J.-J. (1991).** Relationships among digital one/half band filters, low/high order differentiators and discrete/differentiating hilbert transformers. *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, IEEE, pp. 1657–1660.
24. **Platas-Garza, M. A. & de la O Serna, J. A. (2010).** Dynamic phasor and frequency estimates through maximally flat differentiators. *Instrumentation and Measurement, IEEE Transactions on*, Vol. 59, No. 7, pp. 1803–1811.

25. **Press, W. H. (2007).** *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.
26. **Rajagpoal, L. & Dutta Roy, S. (1987).** Design of maximally-flat fir filters using the bernstein polynomial. *Circuits and Systems, IEEE Transactions on*, Vol. 34, No. 12, pp. 1587–1590.
27. **Samadi, S., Iwakura, H., & Nishihara, A. (1999).** Multiplierless and hierarchical structures for maximally flat half-band fir filters. *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, Vol. 46, No. 9, pp. 1225–1230.
28. **Samadi, S., Nishihara, A., & Iwakura, H. (1999).** Generalized half-band maximally flat fir filters. *Circuits and Systems, 1999. ISCAS'99. Proceedings of the 1999 IEEE International Symposium on*, volume 3, IEEE, pp. 279–282.
29. **Selesnick, I. W. (2002).** Maximally flat low-pass digital differentiators. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 49, No. 3, pp. 219–223.
30. **Shyu, J.-J. & Pei, S.-C. (1994).** Lagrange multiplier approach to the design of two-dimensional fir digital filters for sampling structure conversion. *Signal Processing, IEEE Transactions on*, Vol. 42, No. 10, pp. 2884–2886.
31. **Shyu, J.-J., Pei, S.-C., & Huang, Y.-D. (2009).** Least-squares design of variable maximally linear fir differentiators. *Signal Processing, IEEE Transactions on*, Vol. 57, No. 11, pp. 4568–4573.
32. **Strang, G. & Nguyen, T. (1996).** *Wavelets and filter banks*. SIAM.
33. **Tay, D. B. & Murugesan, S. (2014).** Energy optimized orthonormal wavelet filter bank with prescribed sharpness. *Digital Signal Processing*, Vol. 31, pp. 136–144.
34. **Tseng, C.-C. & Lee, S.-L. (2012).** Design of adjustable fractional order differentiator using expansion of ideal frequency response. *Signal Processing*, Vol. 92, No. 2, pp. 498–508.
35. **Tsui, K., Chan, S., & Yeung, K. (2005).** Design of FIR digital filters with prescribed flatness and peak error constraints using second-order cone programming. *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol. 52, No. 9, pp. 601–605.
36. **Vetterli, M. & Herley, C. (1992).** Wavelets and filter banks: Theory and design. *Signal Processing, IEEE Transactions on*, Vol. 40, No. 9, pp. 2207–2232.
37. **Wright, S. & Nocedal, J. (1999).** *Numerical optimization*, volume 2. Springer New York.
38. **Zhang, X. (2009).** Maxflat fractional delay iir filter design. *Signal Processing, IEEE Transactions on*, Vol. 57, No. 8, pp. 2950–2956.
39. **Zhang, X. & Amaratunga, K. (2002).** Closed-form design of maximally flat iir half-band filters. *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, Vol. 49, No. 6, pp. 409–417.
40. **Zhang, X., Kobayashi, D., Wada, T., Yoshikawa, T., & Takei, Y. (2006).** Closed-form design of generalized maxflat r -regular fir m th-band filters using waveform moments. *Signal Processing, IEEE Transactions on*, Vol. 54, No. 11, pp. 4214–4222.

Article received on 08/04/2016; accepted on 06/03/2017.
Corresponding author is Miguel A. Platas-Garza.

Identificación del campo de trabajo de un robot hexápodo utilizando optimización multiobjetivo

Josué Domínguez Guerrero[1], Roberto Sepúlveda[1], Oscar Montiel[1], Oscar Castillo[2]

[1] Instituto Politécnico Nacional, Centro de Investigación y Desarrollo de Tecnología Digital (CITEDI), Tijuana, B.C., Mexico

[2] Tijuana Institute of Technology, Tijuana, B.C., México

jdominguez@citedi.mx, rsepulvedac@ipn.mx, oross@ipn.mx, ocastillo@tectijuana.mx

Abstract. In this paper, an innovative method to find the workspace of a hexapod mobile robot is presented. Differently, to the current state of the art methods that allows to determine the working spaces for walking in straight line, the proposed method allows estimating the optimal set of configurations for walking in any viable direction. The method takes advantage of the existing similitudes between the hexapod and the Delta robot during the tripod walk; however, there are some movement restrictions between them, which were conveniently solved by handling them as a multiobjective optimization problem. Particularly, the MOEA/D algorithm was used to solve the problem.

Keywords. Hexapod robot, multobjective optimization, delta robot, evolutionary algorithms, gait.

1. Introducción

En la actualidad, la mayoría de los robots están diseñados para desplazarse en superficies lisas, niveladas o ligeramente inclinadas. Los ambientes externos con superficies irregulares e inestables como arena, nieve, lodo o grava, y los interiores como escaleras, puertas y esquinas, se consideran ambientes difíciles para el desplazamiento de robots móviles [17]. En la naturaleza existe una gran diversidad de animales de los cuales se puede imitar su locomoción para diseñar robots capaces de sortear este tipo de terrenos [10], por ejemplo, los animales hexápodos, cuya estructura y modelo de locomoción les permite evitar pequeños obstáculos y caminar por terrenos inclinados, lo que ha inspirado la creación de robots que imiten su locomoción. La habilidad de un robot hexápodo para eludir obstáculos, caminar en distintos tipos de terreno y su eficiencia energética depende en gran medida de la relación entre el largo de sus patas y su peso.

En el movimiento y desplazamiento de un robot hexápodo se tienen que controlar y coordinar al mismo tiempo sus seis patas, cada una de las cuales se representa normalmente utilizando la estructura de una cadena cinemática de tres eslabones con tres grados de libertad [3], por lo que el control de un robot hexápodo es un problema en el que se tienen que controlar 18 uniones al mismo tiempo.

El caminado de un hexápodo de acuerdo al número de patas que se encuentran en apoyo, se puede clasificar en secuencias de caminado, que pueden ser de tipo onda, tetrápodo, transición y trípode. Para crear estas secuencias de caminado se puede utilizar un generador central de patrones [3, 12, 6], redes neuronales [7], algoritmos genéticos [11] o un generador aleatorio basado en aprendizaje [5]. En el caminado cada pata puede tener dos estados: avance y recuperación.

Un robot hexápodo al considerarse como un sistema aislado sus patas se pueden mover libremente sin embargo al poner las patas sobre la superficie estas ya no se pueden mover libremente sino que estarán restringidas por la fricción entre el piso y las patas, sus movimientos se limitan de

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2739

manera similar a lo que sucede cuando una mano robótica con tres dedos sujeta un objeto uniforme [2]. Durante la etapa de avance en el caminado trípode un robot hexápodo se asemeja a un robot paralelo tipo Delta como el presentado en [18], la plataforma móvil es el cuerpo del robot y la plataforma fija es el triángulo de apoyo que forman las tres patas que se encuentran en contacto con el suelo. En [1] se presenta el diseño de una estructura de un robot hexápodo paralelo, el cual de forma similar al robot tipo Delta, su plataforma superior está fija, la longitud de cada una de sus patas varía utilizando uniones tipo pistón para mover la plataforma inferior, de este tipo de robots en [8] se presenta un análisis de su exactitud y de la simetría en el campo de trabajo; en [13], se presenta un método para resolver el problema de la cinemática directa de este tipo de robots utilizando un algoritmo genético.

Este trabajo contribuye con un método para encontrar el campo de trabajo de un robot hexápodo, utiliza la similitud existente entre un robot hexápodo y uno tipo Delta. La representación del robot hexápodo utilizada es similar a la presentada en [8, 13, 1] con la diferencia que éstos son robots estáticos, la representación de un robot hexápodo móvil como un robot paralelo también se utiliza en [15] donde se hace un análisis de la estabilidad del robot antes del caminado. El método propuesto en este trabajo permite encontrar todas las posibles combinaciones de las articulaciones de las tres patas en soporte del hexápodo que hacen que su cuerpo avance o gire manteniendo las puntas de las patas en el mismo triángulo de apoyo.

De acuerdo con la investigación realizada, no hay un trabajo donde se encuentre el campo de trabajo para un robot hexápodo móvil, las investigaciones reportadas generan solo el movimiento con patrones de caminado en línea recta [3, 12, 6, 5]. Para encontrar el campo de trabajo del robot, se requiere una solución que satisfaga de manera simultánea y óptima varios objetivos, por lo que este problema intrínsecamente es un problema de optimización multiobjetivo, en el cual se deberán minimizar los errores de los ángulos, el perímetro entre el triángulo de apoyo de la posición actual, y el de la nueva posición. La propuesta utiliza el algoritmo MOEA/D (del inglés Multiobjective Evolutionary Algorithm Based on Decomposition) propuesto en [19], cada punto en la frontera de Pareto representa una posición viable para un robot hexápodo.

Este trabajo está organizado de la siguiente manera: En la sección 2 se describe el modelo matemático de los robots hexápodos. En la sección 3 se describe la secuencia de caminado utilizado para el robot hexápodo y la formulación multiobjetivo del problema de mantener el triángulo de apoyo en la misma posición para encontrar el campo de trabajo, también se describe el algoritmo MOEA/D. Los resultados y experimentos se muestran en la sección 4. Finalmente, en la sección 5 se presentan las conclusiones.

2. Modelo cinemático de un robot hexápodo

En la Figura 1, se observan las patas del modelo del robot hexápodo utilizado las cuales están distribuidas simétricamente al eje y, que es el eje sobre el cual se desplaza el robot. En la Figura 2, se presenta el modelo cinemático utilizado para representar cada pata del robot. Considerando que el origen se encuentra en el punto de anclaje de la pata con el cuerpo, la altura z_1 es la distancia entre el origen y la superficie de contacto, cada pata de un robot hexápodo se puede analizar como una cadena cinemática de tres eslabones, la cual inicia en la unión 0 donde está sujeta la pata al cuerpo del robot, el eslabón 1 es la coxa, el eslabón 2 es el fémur y el eslabón 3 es la tibia [9, 14]. La Tabla 1 muestra los parámetros de Denavit-Hartenberg (D-H) del robot: longitud del eslabón (a_i), giro del eslabón (α_i), distancia de la unión (d_i) y ángulo de la unión (θ_i).

Tabla 1. Parámetros D-H

Eslabón	a_i	α_i	d_i	θ_i
1	L_1	$\pm 90^\circ$	0	θ_1
2	L_2	0	0	θ_2
3	L_3	0	0	θ_3

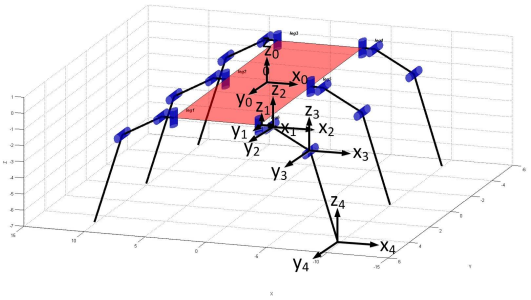


Fig. 1. Modelo del robot

La matriz de transformación homogénea que describe la traslación y rotación entre el i-ésimo y el $(i-1)$-ésimo sistema de coordenadas se representa en (1). La ubicación de la punta de la pata con respecto al origen está definida en las ecuaciones (5), (6) y (7), las cuales se obtienen utilizando las matrices de transformación homogénea definidas en (2), (3) y (4) [14]:

$$ {}^{i-1}\mathbf{T}_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & \alpha_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & \alpha_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{1} $$

$$ {}^{0}\mathbf{T}_1 = \begin{pmatrix} \cos\theta_1 & 0 & \sin\theta_1 & L_1\cos\theta_1 \\ \sin\theta_1 & 0 & -\cos\theta_1 & L_1\sin\theta_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2} $$

$$ {}^{1}\mathbf{T}_2 = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & L_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3} $$

$$ {}^{2}\mathbf{T}_3 = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & L_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4} $$

$$ x = [L_1 + L_2\cos\theta_2 + L_3\cos(\theta_2+\theta_3)]\cos\theta_1, \tag{5} $$

$$ y = [L_1 + L_2\cos\theta_2 + L_3\cos(\theta_2+\theta_3)]\sin\theta_1, \tag{6} $$

$$ z = L_2\sin\theta_2 + L_3\sin(\theta_2+\theta_3). \tag{7} $$

Resolviendo las ecuaciones de posición (5), (6) y (7), se pueden determinar los valores de los ángulos de las articulaciones de cada pata utilizando las ecuaciones (8), (9) y (10):

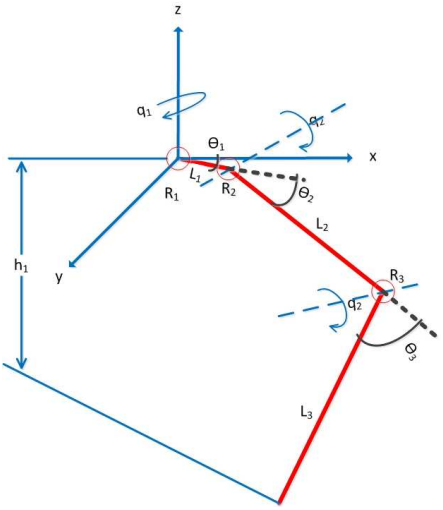


Fig. 2. Modelo cinemático de una pata

$$ \theta_1 = a\tan 2(y, x), \tag{8} $$

$$ \theta_2 = a\tan 2(c, \pm\sqrt{a^2+b^2+c^2}) - a\tan(a, b), \tag{9} $$

donde $a = 2L_2(\sqrt{x^2+y^2} - L_1)$; $b = 2zL_2$; $c = [(\sqrt{x^2+y^2} - L_1)^2 + z^2 + L_2^2 - L_3^2]$, and

$$ \theta_3 = \cos^{-1}[\frac{(\sqrt{x^2+y^2} - L_1)^2 + z^2 - L_2^2 - L_3^2)}{2L_2L_3}. \tag{10} $$

3. Identificación del campo de trabajo del robot hexápodo

El caminado trípode del robot hexápodo se dividió en seis estados a partir de la posición inicial en la cual las patas 1, 3 y 5 están en apoyo y las patas 2, 4 y 6 están levantadas. El proceso de caminado se muestra en la Figura 3.

En el primer estado las dos patas extremas del lado izquierdo 1 y 3, y la pata central 5 del lado derecho se encuentran en apoyo, estas patas se

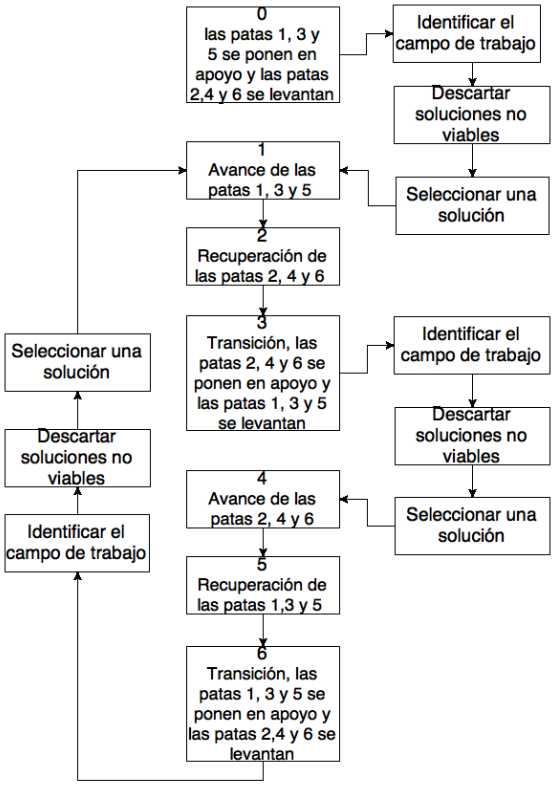


Fig. 3. Proceso de caminado del robot hexápodo

desplazan hacia atrás cambiando el valor de los ángulos $\theta_{1,1}$ y $\theta_{1,3}$ por $-\Delta\theta_1$, y el ángulo $\theta_{1,5}$ por $\Delta\theta_1$, esto provocará que el robot avance en línea recta hacia enfrente; en el segundo estado (recuperación), las dos patas extremas del lado derecho 4 y 6, y la pata central 2 del lado izquierdo se ponen en la posición inicial, a los ángulos $\theta_{1,4}$ y $\theta_{1,6}$ y $\theta_{1,2}$ se les asigna el valor cero; en el tercer estado (transición), las patas 4, 6 y 2 se ponen en apoyo, posteriormente las patas 1, 3 y 5 se levantan; en el estado cuatro las patas 4, 6 y 2 realizan el avance; en el estado cinco las patas 1, 3 y 5 se recuperan a su posición inicial; en el estado seis se realiza la transición y se vuelve a repetir el ciclo.

El primer y el cuarto estado son los que controlan el desplazamiento del hexápodo. En el avance en línea recta el ángulo $\theta_{1,i}$ de las patas del lado derecho y del lado izquierdo en apoyo deben tener la misma magnitud pero con sentido diferente. La acción de girar el cuerpo de un robot hexápodo es compleja ya que se tiene que encontrar los valores de los ángulos $\mathbf{\Theta}_{1,a}$ que permitan al robot girar manteniendo el triángulo de apoyo en la misma posición inicial. En el primer estado el movimiento del cuerpo de un robot hexápodo es controlado por los ángulos $\mathbf{\Theta}_{1,a} = \{\theta_{1,1}, \theta_{1,3}, \theta_{1,5}\}$ y en el cuarto estado es controlado por los ángulos $\mathbf{\Theta}_{1,a} = \{\theta_{1,2}, \theta_{1,4}, \theta_{1,6}\}$.

Durante la etapa de avance de un robot hexápodo con caminado trípode se forma un robot paralelo tipo Delta (Figura 4). Esta clase de robots está compuesta por una plataforma fija y una plataforma móvil conectadas por tres cadenas cinemáticas idénticas e igualmente distribuidas [16]. En el robot hexápodo la plataforma móvil es el cuerpo y la plataforma fija es el triángulo de apoyo que se forma con las tres patas en el piso.

3.1. Definición multiobjetivo del problema

Un problema de optimización multiobjetivo se define como minimizar o maximizar las funciones objetivo $\mathbf{F}(\mathbf{x})$ en el espacio objetivo Z, sujeto a restricciones de igualdad, de desigualdad y los límites de las variables de decisión $\mathbf{x} \in \Omega$. La meta en un problema de optimización multiobjetivo es encontrar el conjunto óptimo de Pareto $\mathcal{P}^*$, es decir el conjunto de soluciones $\mathbf{x} \in \Omega$ para las cuales no existe ninguna solución $\mathbf{x}' \in \Omega$ que domine $\mathbf{x}$. Una solución $\mathbf{x}'$ domina a otra solución $\mathbf{x}$ ($\mathbf{x}' \preceq \mathbf{x}$) si $\mathbf{x}'$ es parcialmente menor que $\mathbf{x}$, asumiendo que todos los objetivos se minimizarán ($\mathbf{x}'$ sera menor o igual que $\mathbf{x}$ para todos los objetivos y menor al menos que uno de los objetivos).

Para mantener el triángulo de apoyo en la etapa de avance se debe asegurar que el triángulo formado por la posición de las patas 1 (x_1^t, y_1^t, z_1^t), 3 (x_3^t, y_3^t, z_3^t) y 5 (x_5^t, y_5^t, z_5^t) en apoyo en el primer estado antes de avanzar, sea el mismo que el formado por las patas 1 $(x_1^{t+1}, y_1^{t+1}, z_1^{t+1})$, 3 $(x_3^{t+1}, y_3^{t+1}, z_3^{t+1})$ y 5 $(x_5^{t+1}, y_5^{t+1}, z_5^{t+1})$ después de avanzar. El triangulo de apoyo antes y después del avance será igual si dos de sus ángulos y su perímetro es igual. Para cumplir con lo anterior, se planteó una formulación matemática de optimización multiobjetivo, la cual se muestra en

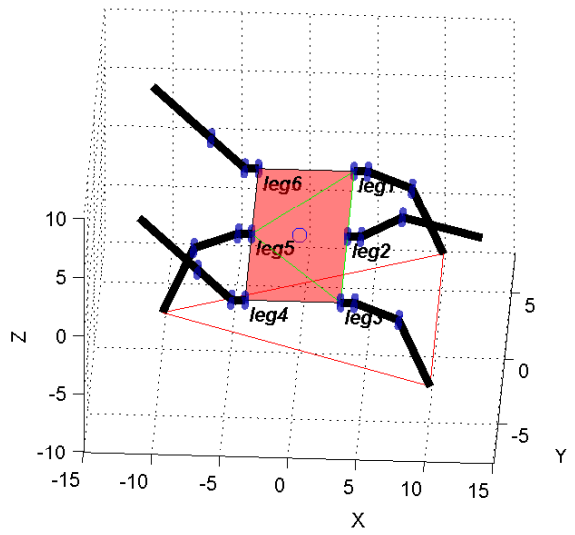


Fig. 4. Representación de un robot hexápodo como robot paralelo

la ecuación (11). El primer objetivo, es minimizar el error entre el ángulo β_1^t, y el mismo ángulo β_1^{t+1} después del avance; el segundo objetivo, es minimizar el error entre los ángulos β_5^t y β_5^{t+1}; el tercer objetivo, es minimizar el error del perímetro del triángulo de apoyo en el primer estado p^t y el perímetro del triángulo después de avanzar p^{t+1}:

Minimizar

$$\begin{aligned} f_1(\mathbf{\Theta}_{1,a}) &= |\beta_1^t - \beta_1^{t+1}|, \\ f_2(\mathbf{\Theta}_{1,a}) &= |\beta_5^t - \beta_5^{t+1}|, \\ f_3(\mathbf{\Theta}_{1,a}) &= |p^t - p^{t+1}|, \end{aligned} \tag{11}$$

sujeto a

$$-\pi/6 \leq \mathbf{\Theta}_{1,a} \leq \pi/6.$$

Los ángulos β_1^t y β_5^t son dos de los ángulos formados por el triángulo de apoyo de las patas en el tiempo t los cuales permanecen constantes. Para encontrar sus valores se utilizan las ecuaciones (12) y (13), las cuales se obtienen descomponiendo el triángulo de apoyo en tres triángulos rectángulos como se muestra en la Figura 5, y utilizando las ecuaciones de la cinemática (5) y (6) se encuentra la posición de las patas 1, 3 y 5. El perímetro p^t del triángulo de apoyo se calcula utilizando la ecuación (14). Los ángulos β_1^{t+1} y β_5^{t+1} y el perímetro p^{t+1}, después de que el cuerpo del robot se movió, se calculan utilizando las mismas ecuaciones con los valores de las patas en la nueva posición.

$$\beta_1^t = \tan^{-1}\left(\frac{x_1^t - x_5^t}{y_1^t - y_5^t}\right) + \tan^{-1}\left(\frac{x_1^t - x_3^t}{y_1^t - y_3^t}\right), \tag{12}$$

$$\beta_5^t = \sin^{-1}\left(\frac{\sin(\beta_1^t) d_1}{d_2}\right), \tag{13}$$

$$p^t = d_1 + d_2 + d_3, \tag{14}$$

en donde:

$$d_1 = \sqrt{(x_1^t - x_3^t)^2 + (y_1^t - y_3^t)^2},$$

$$d_2 = \sqrt{(x_3^t - x_5^t)^2 + (y_3^t - y_5^t)^2},$$

$$d_3 = \sqrt{(x_1^t - x_5^t)^2 + (y_1^t - y_5^t)^2}.$$

Las variables de búsqueda del problema de optimización multiobjetivo son los ángulos $\mathbf{\Theta}_{1,a}^{t+1} = \{\theta_{1,1}^{t+1}, \theta_{1,3}^{t+1}, \theta_{1,5}^{t+1}\}$, estos ángulos provocan que las patas se coloquen en una nueva posición, la cual se encuentra utilizando las ecuaciones (5), (6) y (7). Con esta nueva posición se calculan los ángulos β_1^{t+1} y β_5^{t+1} y el perímetro p^{t+1} que definen el nuevo triángulo de apoyo.

En el problema de restricción del triángulo de apoyo el conjunto óptimo de Pareto $\mathcal{P}^*$ son todas las soluciones $\mathbf{\Theta}_{1,a}^{t+1}$ en el espacio de búsqueda Ω tal que no exista ninguna solución $\mathbf{\Theta}_{1,a}^{t+1'}$ que sea dominante de $\mathbf{\Theta}_{1,a}^{t+1}$. Para que $\mathbf{\Theta}_{1,a}^{t+1'}$ domine a $\mathbf{\Theta}_{1,a}^{t+1}$, se debe cumplir $f_j(\mathbf{\Theta}_{1,a}^{t+1'}) \leq f_j(\mathbf{\Theta}_{1,a}^{t+1})$ para todo $j = 1, 2, 3$ y $f_j(\mathbf{\Theta}_{1,a}^{t+1'}) < f_j(\mathbf{\Theta}_{1,a}^{t+1})$ para al menos una $j = 1, 2, 3$.

Todos los triángulos de apoyo que cumplen esta condición, es decir todas las posiciones en las que se puede mover el hexápodo manteniendo sus patas en el mismo lugar es el campo de trabajo, el cual como se muestra en la Figura 3 se tiene que encontrar al iniciar el caminado del robot entre los

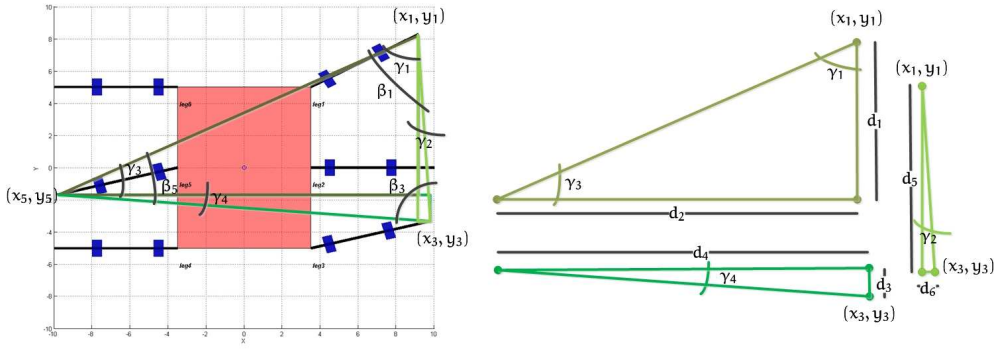


Fig. 5. Descomposición del triángulo de apoyo

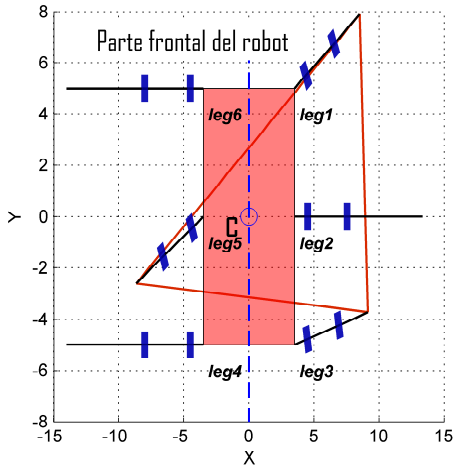


Fig. 6. Centro del robot dentro del triángulo de apoyo

estados 0 y 1, y después en la transición de los estados 3 y 4, y 6 y 1.

La condición para que el robot hexápodo no se caiga durante el caminado, es que el centro de gravedad de la parte superior del cuerpo permanezca siempre dentro del polígono de apoyo, como se muestra en la Figura 6. Esta condición se cumple mientras la distancia S_f sobre el eje x a la diagonal que se forma entre la pata central y la pata de enfrente, o la distancia S_b sobre el eje x a la diagonal que se forma entre la pata central y la pata de atrás, sean mayor que cero.

3.2. Algoritmo para optimización multiobjetivo basado en descomposición (MOEA/D

Los problemas de optimización multiobjetivo se resuelven principalmente usando métodos que asignan un valor de peso a cada objetivo de acuerdo a su importancia en el problema (métodos escalados), que lo convierten en un problema de un solo objetivo o usando algoritmos evolutivos los cuales ofrecen la ventaja de poder encontrar múltiples soluciones óptimas de Pareto en una sola corrida. También es relativamente más sencillo adaptarlos a diferentes problemas de aplicación. En optimización multiobjetivo es importante encontrar la mayor cantidad de soluciones óptimas de Pareto, pero también que éstas estén distribuidas uniformemente sobre el frente de Pareto.

Para encontrar las posiciones en las que se puede mover un robot hexápodo se utilizó el algoritmo MOEA/D propuesto en [19], el cual divide un problema de optimización multiobjetivo en N subproblemas de un solo objetivo que descompone el espacio objetivo en espacios igualmente distribuidos. La dispersión de las soluciones en el frente de Pareto es superior a otros algoritmos evolutivos de acuerdo a lo reportado en [19].

Algoritmo 1: Algoritmo MOEA/D

Entradas
MOP;
Criterio de paro;
N: número de subproblemas en MOEA/D;
N vectores de pesos: $\lambda^1, \ldots, \lambda^N$;
τ: número de vectores de pesos colindantes.
Salida
EP.
Paso 1) Inicialización:
Paso 1.1) $EP = \emptyset$
Paso 1.2) Para cada $i = 1, 2, \ldots, N$ encontrar $B(i) = \{i_1, i_2, \ldots, i_\tau\}$, donde $\lambda^{i_1}, \ldots, \lambda^{i_\tau}$ son los τ vectores mas cercanos a λ^i
Paso 1.3) Inicializar la población $x^1, \ldots, x^N$. Encontrar los vector de funciones objetivo $FV^i = F(x^i)$
Paso 1.4) Inicializar $z = (z_1, \ldots, z_m)^T$
Paso 2) Actualización:
for $i = 1$ *to* N **do**
Paso 2.1) Reproducción: Seleccionar k y l de $B(i)$, usando operadores genéticos generar y de x^k y x^l.
Paso 2.2) Mejoramiento: Mejorar la solución y para producir y'
Paso 2.3) Actualización de z:
foreach $j = 1, \ldots, m$ **do**
if $z_j < f_j(y')$ **then**
$z_j = f_j(y')$
end
end
Paso 2.4) Actualización de soluciones colindantes:
foreach $j \in B(i)$ **do**
if $g^{te}(y'|\lambda^j, z) < g^{te}(x^j|\lambda^j, z)$ **then**
$x_j = y'$; $FV^j = F(y')$
end
end
Paso 2.5) Actualización de EP:
Eliminar todos los vectores dominados por $F(y')$ de EP.
Agregar $F(y')$ a EP si no hay ningún vector en EP que lo domine.
end
Paso 3) Criterio de paro
Si se cumple el criterio de paro, terminar y devolver EP, sino ir al **Paso 2**.

Cada subproblema i tiene un vector de pesos diferente para cada función objetivo, y tiene asignado un conjunto B_i con los τ vectores de peso más cercanos. Para resolver cada subproblema se utiliza el enfoque de Tchebycheff definido en la ec. (15):

$$g^{te}(x|\lambda^j, z^*) = \max_{1 \le i \le m} \{\lambda_i^j |f_i(x) - z_i^*|\}, \qquad (15)$$

donde $\lambda^j = \lambda_1^j, \lambda_2^j, \ldots, \lambda_m^j$ es el vector de pesos del individuo j, cada subproblema se resuelve utilizando los τ vectores de pesos uniformemente distribuidos $\lambda^1, \lambda^2, \ldots, \lambda^N$ más cercanos a cada λ^j. El algoritmo MOEA/D resuelve todo los N subproblemas en una sola corrida, su funcionamiento se muestra en el Algoritmo 1.

4. Resultados experimentales

La comprobación del método propuesto se realizó utilizando la herramienta Robotics [4] de Matlab para simular un robot hexápodo y ver el movimiento de sus articulaciones. En el experimento se consideró que las patas 1, 3 y 5 están en apoyo como se muestra en la Figura 4 y los ángulos iniciales son $\theta_{1,1}^0 = \theta_{1,3}^0 = \theta_{1,5}^0 = 0$, $\theta_{2,1}^0 = \theta_{2,3}^0 = \theta_{2,5}^0 = -\pi/8$ y $\theta_{3,1}^0 = \theta_{3,3}^0 = \theta_{3,5}^0 = -\pi/4$. Las longitudes de los eslabones de las patas son $l_1 = 1$ cm., $l_2 = 3{,}5$ cm. y $l_3 = 6$ cm, el ancho del cuerpo del robot es $W = 10$ cm y el largo $L = 7$ cm.

Estas medidas se eligieron porque son las características físicas de un robot hexápodo que se tiene en el laboratorio de sistemas digitales del CITEDI-IPN. Las variables de búsqueda son los ángulos $\theta_{1,1}^1$,$\theta_{1,3}^1$ y $\theta_{1,5}^1$ todos entre $-pi/6$ y $pi/6$. Los ángulos $\theta_2 = -\pi/8$ y $\theta_3 = -\pi/4$ permanecen constantes, por lo tanto la altura de las tres patas también permanece constante $z = -6{,}8827$ cm.

En el algoritmo evolutivo se utilizó una población de 300 individuos con un vecindario $\tau = 20$, y se ejecutó durante 100 generaciones. En la Figura 7 se observa el frente de Pareto obtenido después de 100 generaciones, en esta Figura también se indica la ubicación en el espacio objetivo de las soluciones a, b, c, y d, de las cuales se muestra el movimiento que provocan en las patas del robot en la Figura 9, estas soluciones fueron elegidas aleatoriamente para ejemplificar el comportamiento del robot hexápodo. En la Tabla

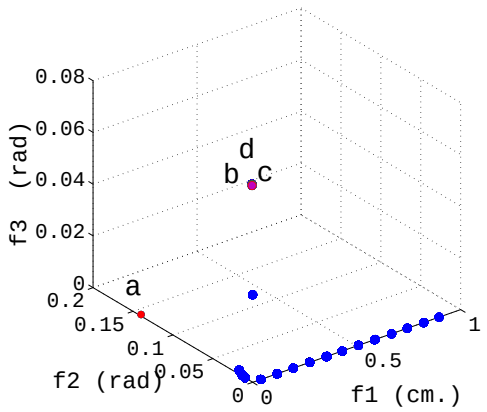
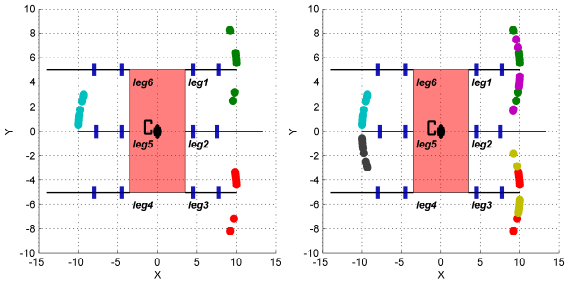


Fig. 7. Frente de Pareto obtenido después de 100 generaciones



(a) **(b)**

Fig. 8. (a) Campo de trabajo, (b) Campo de trabajo completo de robot hexápodo

2 se muestran los valores de ángulos, la posición de las tres patas en apoyo en los ejes x y y, así como los valores de las funciones objetivo de las soluciones a, b, c, y d.

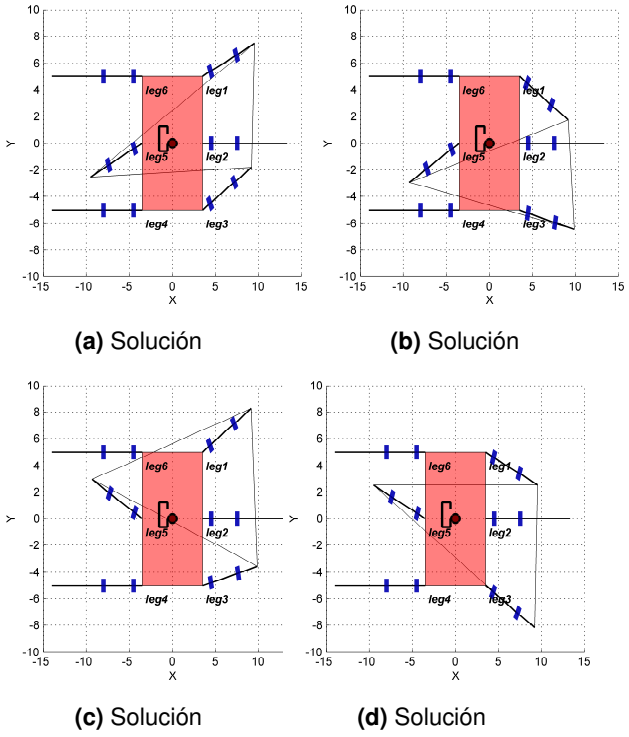
En la Figura 8a se observa el campo de trabajo del robot hexápodo obtenido por las mismas soluciones del frente de Pareto mostrado en la Figura 7, estas soluciones solo provocan que el robot se mueva en un sentido ya que el algoritmo multiobjetivo no utiliza ningún criterio para determinar la posición de la patas.

El campo de trabajo completo mostrado en la Figura 8b se obtiene creando soluciones simétricas a las obtenidas por el algoritmo y uniendo estos dos conjuntos de soluciones.

En la Figura 9 se observan cuatro soluciones representativas del conjunto óptimo de Pareto, donde se muestra la nueva posición de las tres patas (1, 3 y 5) que provocan el movimiento del robot y el triángulo de apoyo permanece en la misma posición.

En la misma figura se muestra que lo que se mueve son las patas que forman el triángulo de apoyo y no el cuerpo, ésto se debe a que en la herramienta *Robotics* de Matlab, el sistema de referencia del modelo cinemático del robot es la parte superior del cuerpo, sin embargo, en un robot real, el sistema de referencia es el piso, por lo que el cuerpo del robot es lo que se mueve.

En la Figura 9b se muestra una solución dentro del espacio de trabajo que no es viable ya que el centro geométrico del cuerpo superior del robot esta fuera del triángulo de apoyo.



(a) Solución **(b)** Solución

(c) Solución **(d)** Solución

Fig. 9. Comportamiento de un robot hexápodo para cuatro soluciones óptimas de Pareto

Tabla 2. Muestra de soluciones óptimas de Pareto

Solución	Pata	Ángulo	Posición		Función objetivo		
		θ_1	x	y	f_1	f_2	f_3
a	1	0.3883	9.544	7.472	0.0001	0.0009	0.0764
	3	0.5103	9.198	-1.81			
	5	0.3975	-9.52	-2.528			
b	1	-0.5236	9.155	1.735	0	0.014	0.0006
	3	-0.2223	9.869	-6.44			
	5	0.4682	-9.327	-2.947			
c	1	-0.3883	9.544	2.528	0.0001	0.0009	0.0764
	3	-0.5103	9.198	-8.19			
	5	-0.3975	-9.521	2.528			
d	1	-0.3883	9.544	2.528	0.0001	0.0009	0.0759
	3	-0.5069	9.209	-8.17			
	5	-0.3975	-9.521	2.528			

5. Conclusiones

En este trabajo se analizó un robot hexápodo como un robot paralelo tipo Delta, y se propuso un método innovador para obtener el campo de trabajo, ya que de acuerdo a las fuentes bibliográficas consultadas, éstas se enfocan a que el robot avance en línea recta.

El método propuesto permite encontrar los posibles movimientos que puede realizar un robot hexápodo, y tiene como propósito mantener en la misma posición sobre el piso al triángulo de apoyo después del movimiento, lo cual se planteó como un problema de optimización multiobjetivo. El método propuesto permite que un robot hexápodo se pueda mover en todas las direcciones durante la etapa de avance del caminado trípode.

Referencias

1. **Akda, M., Karagulle, H., & Malgaca, L. (2012).** An integrated approach for simulation of mechatronic systems applied to a hexapod robot. *Mathematics and Computers in Simulation*, Vol. 82, No. 5, pp. 818–835.
2. **Borras, J. & Dollar, A. M. (2014).** Analyzing dexterous hands using a parallel robots framework. *Autonomous Robots*, Vol. 36, No. 1-2, pp. 169–180.
3. **Chen, W., Ren, G., Zhang, J., & Wang, J. (2012).** Smooth transition between different gaits of a hexapod robot via a central pattern generators algorithm. *Journal of Intelligent and Robotic Systems*, Vol. 67, No. 3-4.
4. **Corke, P. I. (2011).** *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer.
5. **Erden, M. S. & Leblebicioglu, K. (2008).** Free gait generation with reinforcement learning for a six-legged robot. *Robotics and Autonomous Systems*, Vol. 56, No. 3, pp. 199–212.
6. **Fuente, L., Lones, M., Turner, A., Caves, L., Stepney, S., & Tyrrell, A. (2013).** Adaptive robotic gait control using coupled artificial signalling networks, hopf oscillators and inverse kinematics. *Evolutionary Computation (CEC), IEEE Congress on*, pp. 1435–1442.
7. **Juang, C.-F., Chang, Y.-C., & Hsiao, C.-M. (2011).** Evolving gaits of a hexapod robot by recurrent neural networks with symbiotic species-based particle swarm optimization. *Industrial Electronics, IEEE Transactions on*, Vol. 58, No. 7, pp. 3110–3119.
8. **Karimi, D. & Nategh, M. J. (2014).** Kinematic nonlinearity analysis in hexapod machine tools: Symmetry and regional accuracy of workspace. *Mechanism and Machine Theory*, Vol. 71, pp. 115–125.
9. **Mănoiu-Olaru, S. & Nţulescu, M. (2013).** Stability analysis software platform dedicated for a hexapod robot. In **Dumitrache, L.**, editor, *Advances in Intelligent Control Systems and Computer Science*, volume 187 of *Advances in Intelligent Systems and Computing*. Springer Berlin Heidelberg, pp. 143–156.

10. **Nishii, J. (2000).** Legged insects select the optimal locomotor pattern based on the energetic cost. *Biological Cybernetics*, Vol. 83, No. 5, pp. 435–442.
11. **Pap, Z., Kecskes, I., Burkus, E., Bazso, F., & Odry, P. (2010).** Optimization of the hexapod robot walking by genetic algorithm. *Intelligent Systems and Informatics (SISY), 8th International Symposium on*, pp. 121–126.
12. **Ren, G., Chen, W., Dasgupta, S., Kolodziejski, C., Worgotter, F., & Manoonpong, P. (2015).** Multiple chaotic central pattern generators with learning for legged locomotion and malfunction compensation. *Information Sciences*, Vol. 294, pp. 666–682. Innovative Applications of Artificial Neural Networks in Engineering.
13. **Rolland, L. & Chandra, R. (2016).** The forward kinematics of the 6-6 parallel manipulator using an evolutionary algorithm based on generalized generation gap with parent-centric crossover. *Robotica*, Vol. 34, pp. 1–22.
14. **Roy, S. S. & Pratihar, D. K. (2012).** Effects of turning gait parameters on energy consumption and stability of a six-legged walking robot. *Robotics and Autonomous Systems*, Vol. 60, No. 1, pp. 72–82.
15. **Rushworth, A., Cobos-Guzman, S., Axinte, D., & Raffles, M. (2015).** Pre-gait analysis using optimal parameters for a walking machine tool based on a free-leg hexapod structure. *Robotics and Autonomous Systems*, Vol. 70, pp. 36–51.
16. **Simionescu, I., Ciupitu, L., & Ionita, L. C. (2015).** Static balancing with elastic systems of DELTA parallel robots. *Mechanism and Machine Theory*, Vol. 87, pp. 150–162.
17. **Sorin, M. & Mircea, N. (2012).** Matlab simulation interface for locomotion analysis of a hexapod robot structure. *System Theory, Control and Computing (ICSTCC), 16th International Conference on*, pp. 1–6.
18. **Wang, J. & Gosselin, C. M. (1999).** Static balancing of spatial three-degree-of-freedom parallel mechanisms. *Mechanism and Machine Theory*, Vol. 34, No. 3, pp. 437–452.
19. **Zhang, Q. & Li, H. (2007).** A multi-objective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, Vol. 2007.

Article received on 09/05/2016; accepted on 07/02/2017.
Corresponding author is Josué Domínguez Guerrero.

Bridging the Gap Between Model-Based Design and Reliable Implementation of Feedback-Based Biocircuits: A Systems Inverse Problem Approach

Juan Carlos Martinez-Garcia[1], Carlos Aguilar-Ibanez[2], Alberto Soria-Lopez[1]

[1] Control Department of CINVESTAV-IPN, Mexico City, Mexico

[2] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City, Mexico

juancarlos.martinezg@gmail.com, carlosaguilari@cic.ipn.mx, soria@cinvestav.mx

Abstract. Our concern is the tuning of mathematical models describing rationally designed genetic biocircuits. Based on a deterministic lumped continuous-time approach, we propose a tuning methodology combining both exact algebraic parameter reconstruction and nonlinear parameter estimation of a given model supporting the design of a specific genetic biocircuit, *i.e.*, we bridge the gap between model-based design and implementation as the solution of a systems inverse problem. As a proof of concept, our proposal is constrained to cyclic feedback systems characterizing synthesized transcriptional networks conditioned to display sustained oscillatory behavior. Our proposed methodology is illustrated via computer–based simulations involving the tuning of a state–based model describing a well–know cyclic feedback biocircuit: the celebrated *repressilator*. Tuning in our case is conceived as a procedure to adjust the parameter values of the mathematical model taking into account for this the actual behavior observed from the corresponding synthesized biocircuit.

Keywords. Systems biology, synthetic biology, tuning of mathematical models, algebraic parameter reconstruction, observer based system identification, synthetic transcriptional networks, cyclic feedback biocircuits.

1 Motivation

This paper extends the discussion first presented in [16], following for this an approach focused on methodological issues. Our purpose is then to enlighten the limitations that are imposed by the unavoidable uncertainty on the value of the involved parameters, as far as synthetic biology designs are concerned.

The simple illustrative example that we use here in order to present our ideas leads us to conclude that no realistic synthetic biology design are possible at all, *if not parameter estimation is included in the physical realization of the designed system*. Synthetic biology asks then for hybrid designs, *i.e.*, designs involving not only the biological designed system but also a computer-based parameter estimation module.

2 Introduction

Systems biology, the integrative systems–based contemporary post-genomic vision of biology, promotes the understanding of living organisms through the tinkerer's approach, which is to say building life to understand it (and also to optimally profit from it), see for instance [7]. Recent scientific and technological progress in molecular cell biology, genetic engineering, as well as in mathematical and computational modeling of biological systems and in biological engineering methodologies, offers the opportunity to rationally design and build genetic circuits as a way to understand existing biological functionalities, and even better: as a rational way to add new prescribed functionalities to existing biological

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2740

organisms (see for instance [3, 19, 18, 9] and the references therein).

As is done in modern engineering, the current design approach followed by synthetic biology practitioners has mathematical modeling as the first step of the rational design procedure for the intended biocircuit (see for instance [17]). Going from a mathematical model to the realization of its corresponding physical system is a standard engineering approach ruling design on well–developed technological domains like electronic engineering, mechanical engineering, aerospace engineering, and computer engineering. It is then common to build highly consumed products (like cars or mobile phones) directly from computer-based design and computer-based simulation. However, as far as the physical realization of a mathematically modeled biocircuit is concerned, several factors challenge the approach followed by modern engineering. Inherent stochasticity of cellular bio–molecular systems, as well as many others factors (like technological limitations of the involved measurement processes), result in mismatch between the parameter values of the mathematically modeled system and the actual parameter values of the corresponding synthesized biological system. This mismatch moves the behavior of the synthesized system far away from the designed behavior coded by the guiding mathematical model, which reduces then the utility of mathematical modeling in biological engineering (particularly if the concerned model is constrained to be simple enough in order to be useful in terms of design and simulation).

Tuning of the mathematical model is an obvious tool when considering the minimization of the mismatch between the designed behavior and the displayed behavior. It is then necessary to reconstruct the actual parameter values of the designed system through measuring of the actual system's behavior. In this paper we propose an approach based on exact algebraic parameter reconstruction combined with nonlinear algebraic parameter estimation to find the parameter values of cyclic feedback systems describing synthesized biocircuits. As the solution of an inverse problem, the tuning procedure finds the parameter values processing available information provided by the synthesized system. We conceive our methodology as a proof of concept and we constraint our exposition to the case of transcriptional networks conditioned to display measurement available for sustained oscillatory behavior, described in mathematical terms by cyclic feedback systems. We follow a deterministic lumped continuous–time approach, and we support our conclusions via computer-based simulations involving the well-known illustrative synthetic biocircuit: *the repressilator*.

The paper is organized as follows: Section 3 discusses the problem formulation. We briefly recall what is characteristic for the cyclic feedback nonlinear dynamical systems, and its application as a modeling tool for the description of a particular class of synthesized transcriptional networks. Moreover, we formulate the inverse problem in terms of the determination of a set of real parameters tuning a ordinary differential equations based mathematical model to force it to describe the sustained oscillatory behavior of the synthesized transcriptional network. As an illustration of our proposal we limit our exposition to the tuning of a mathematical model of the well–know synthetic cyclic feedback bio–oscillator known as the *repressilator*.

We expose our tuning procedure in Section 4 and in Section 5 we illustrate it via computer–based based simulations involving the well–known synthetic biocircuit called *the repressilator*. We conclude the paper with some final remarks in Section 6.

3 Problem Formulation

In this section we recall the definition of *cyclic feedback systems*, and we restrict our exposition to the deterministic lumped continuous-time invariant framework. Then, we discuss how this particular class of coupled ordinary differential equations is applied to describe cyclic synthesized transcriptional networks with a designed modular structure. Then, we introduce our systems inverse problem, which is to say the tuning of the mathematical model which guides the conception of a modularly structured cyclic feedback synthetical transcriptional network designed to display sustained oscillations.

3.1 Cyclic Feedback Systems

A set of n coupled ordinary-differential-equations:

$$\left\{ \begin{array}{rcl} \dot{x}_1 & = & f_1(x_1, x_n), \\ \dot{x}_2 & = & f_2(x_2, x_1), \\ & \vdots & \\ \dot{x}_n & = & f_n(x_n, x_{n-1}), \end{array} \right\} \quad (1)$$

where:

$$\dot{x}_i := \frac{dx_i}{dt},$$

is called a n–th order cyclic feedback system (see for instance [11] and the references therein) if there exist real constants $\delta_i \in \{\pm 1\}$, $i = 1, 2, \ldots, n$, such that:

$$\delta_i f_i(0, v)\, v > 0, \quad \text{if} \quad \Re \ni v \neq 0$$

$$\delta_i \frac{\partial f_i(a, b)}{\partial b}\bigg|_{(0,0)} > 0.$$

Remark 1: As is obvious from the description (1), the state variable x_{i-1} drives the state variable x_i. This regulatory characteristic fixes the cyclic feedback defining property of the system.

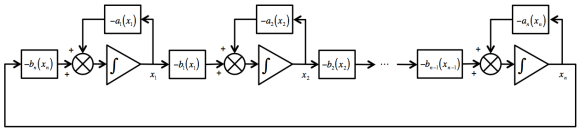


Fig. 1. A particular class of cyclic feedback systems. In this case, the rate of change of each state in the chain results from the addition of a function which depends on the state itself plus a function which depends on the contiguous downstream state

Cyclic biochemical reactions

Figure 1 shows the particular case of cyclic feedback systems described by (see for instance [2]):

$$\left\{ \begin{array}{rcl} \dot{x}_1 & = & -a_1(x_1) - b_n(x_n), \\ \dot{x}_2 & = & -a_2(x_2) - b_1(x_1), \\ & \vdots & \\ \dot{x}_n & = & -a_n(x_n) - b_{n-1}(x_{n-1}), \end{array} \right\}, \quad (2)$$

where $a_i(\cdot)$ and $b_i(\cdot)$ are continuous functions (if these functions are real constants the corresponding system is obviously linear).

Remark 2: The cyclic feedback modular structure shown in (2) has been frequently considered to describe cyclic biochemical reactions, where the end product drives the first reaction, and with the functions $a_i(\cdot)$ and $b_i(\cdot)$ usually related to enzymatic processes. The cyclic feedback modular structure (2) is particularly suited for both the description and the design of biosynthetic oscillatory systems. Notice that for biochemical reactions each state takes positive values, i.e., $x_i \in \Re^+$.

3.2 Synthesized Cyclic Feedback Transcriptional Networks

Biosynthetic oscillators based on transcriptional regulation are frequently built following the cyclic structure given by (2), with inhibition of gene expression as the basic regulatory mechanism (see for instance $[13]$). We must point out that common natural cell biochemical oscillators (*e.g.* circadian oscillators in animals and fungus), involve both transcriptional regulation and translational regulation (see for instance $[14]$).

Since the cell transcriptional machinery is easy to manipulate via standard recombinant DNA technologies, feedback-based transcriptional oscillators are preferred in cell biosynthetical constructs. We describe here the particular class of cyclic feedback systems that concerns our proposal. We focus our attention in cyclic feedback biosynthetic networks resulting from the chaining of transcriptional modules. Therefore, we first introduce the basic representation of a transcriptional module in input-output terms.

Consider x to be a gene, and let us assume that it codes the transcription factor[1] $[x]$, where $[*]$ stands for the concentration of the protein coded by $*$. We can now recall the following (see for instance [6]):

Definition 1: In input–output terms, a transcriptional module maps an excitatory input signal

[1] A transcription factor is a sequence-specific DNA-binding protein.

consisted of the concentration of a transcription factor into an output–signal consisted also of the concentration of a expressed transcription factor. If we denote z and x the genes coding the input signal and the output signal, respectively, the dynamical behavior of the transcriptional module is described in lumped time-invariant terms by the set of coupled ordinary–differential-equations given by (see Figure 2):

$$\left\{[\dot{x}] = \gamma_{r_x}[r_x] - \alpha_x[x], \quad [\dot{r_x}] = f([z]) - \alpha_{r_x}[r_x]\right\},$$

where:

— $[r_x]$ stands for the concentration of r_x, the mRNA–molecules resulting from the transcription of gene x.

— $f([z])$ denotes the excitatory signaling process promoted by the transcription factor $[z]$.

— γ_{r_x} is the constant translation rate from $mRNA$ to the transcription factor coded by x.

— α_x and α_{r_x} are the decay constant rates of $[x]$ and $[r_x]$, respectively.

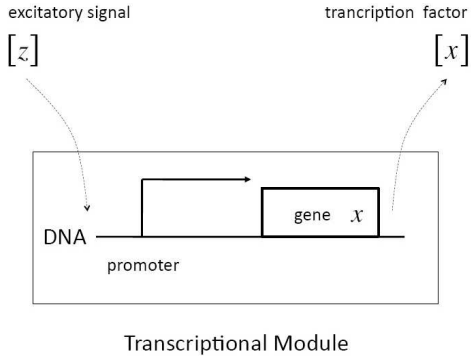


Fig. 2. Schematic representation of a transcriptional module. The excitatory signal $[z]$ binds to the promoter of gene x, which promotes changes in the expression of the corresponding transcription factor $[x]$

Remark 3: The function $f([z])$ captures the specificity of the regulatory role played by the transcription factor $[z]$, and in general is nonlinear in nature. The regulatory function comes as a result of the interaction between $[z]$ and the DNA–sequence which characterizes the promoter of the regulated gene x. Depending on the design, the regulatory process can involve inhibition or promotion through enzymatic dynamics.

We introduce now the class of cyclic feedback network of synthesized transcriptional modules that concerns us.

Definition 2: Cyclic feedback network of synthesized transcriptional modules] We define the class of n–th order cyclic feedback networks of synthesized transcriptional modules as the closed chains of transcriptional modules described as:

$$\left\{\begin{array}{rcl} [\dot{x_1}] &=& \gamma_{r_{x_1}}[r_{x_1}] - \alpha_{x_1}[x_1], \\ [\dot{r_{x_1}}] &=& f_1([x_n]) - \alpha_{r_{x_1}}[r_{x_1}], \\ [\dot{x_2}] &=& \gamma_{r_{x_2}}[r_{x_2}] - \alpha_{x_2}[x_2], \\ [\dot{r_{x_2}}] &=& f_2([x_1]) - \alpha_{r_{x_2}}[r_{x_2}], \\ & \vdots & \\ [\dot{x_n}] &=& \gamma_{r_{x_n}}[r_{x_n}] - \alpha_{x_n}[x_n], \\ [\dot{r_{x_n}}] &=& f_n([x_{n-1}]) - \alpha_{r_{x_n}}[r_{x_n}], \end{array}\right. \tag{3}$$

with:

$$\left\{[\dot{x_i}] = \gamma_{x_i}[r_{x_i}] - \alpha_{x_i}[x_i], [\dot{r_{x_i}}] = f_i([x_{i-1}]) - \alpha_{r_{x_i}}[r_{x_i}]\right\}$$

corresponding to the i–th transcriptional module, and with:

— $[r_{x_i}]$ is the concentration of r_{x_i}, the mRNA molecules transcripted from the gene that codes x_i.

— α_{x_i} and $\alpha_{r_{x_i}}$ are the decay rates of $[r_{x_i}]$ and $[x_i]$, respectively.

— γ_{x_i} is the constant translation rate from $[r_{x_i}]$ to $[x_i]$.

— $f_i([\cdot])$ is the i–th transcription regulatory function.

The simple transcriptional modular structure given by (3) has been successfully applied in the construction of actual biosynthetic oscillators, with the transcription factors x_i, $i = 1, \ldots, n$, required to display sustained oscillations through the appropriate choice of the nonlinear regulatory

functions f_i, $i = 1, \ldots, n$, and the right tuning of the parameters γ_{x_i}, α_{x_i}, and $\alpha_{r_{x_i}}$ (*i.e.* the nonlinear regulatory functions as well as the parameters, including the order of the system, are chosen in order to guarantee that system (3) has a stable limit cycle). The celebrated *repressilator* (see for instance $[8]$) is a well–known example of this family of oscillatory modular transcriptional networks governed by negative feedback, see $[6]$ and $[5]$.

In the sequel we describe our inverse problem, which concerns the tuning of the mathematical model consisting of a cyclic feedback modular transcription network from information provided by a synthesized system derived from the model.

3.3 Finding the Parameter Values

In the previous subsection we presented the class of modularly structured cyclic feedback systems intended to guide the conception and construction of biosynthetic oscillators based on transcriptional regulation.

Problem 1. Consider a synthesized transcriptional network to be given and resulting from a mathematical description given by system (3) (*i.e.* we conceive the synthesized transcriptional network to be a designed cyclic feedback system). Assume that the synthesized transcriptional network displays sustained oscillatory behavior. From the knowledge of the state variables $\{[x_1]$, $[r_{x_1}]$, $[x_2]$, $[r_{x_2}]$, ..., $[x_n]$, $[r_{x_n}]\}$, of the synthesized transcriptional network (*i.e.* the concentration of the transcription factors with their corresponding concentration of the mRNA transcripts), find a set of parameters $\{\gamma_{x_i}, \alpha_{1_{x_i}}, \alpha_{2_{x_i}}\}$, for $i = 1, 2, \ldots, n$, and the parameter values of the nonlinear transcription regulatory functions $f_i(\cdot)$, for $i = 1, 2, \ldots, n$, making a corresponding tuned mathematical model to closely describe the behavior of the synthesized transcriptional network.

Remark 4: We propose in the sequel a solution for Problem 1. We conceive our proposal as a proof–of–concept and because of that, and in order to simplify our exposition, we only consider here the case concerning the well-known synthetic bio-oscillator known as the *repressilator*.

The repressilator

In the case corresponding to the *repressilator* (see [8, 6]) $n = 3$ in (3), and the regulatory functions $f_1(\cdot)$, $f_2(\cdot)$ and $f_3(\cdot)$, correspond to inhibition actions. In the named symmetrical case it is assumed that the transcriptional regulatory functions (all of them with negative slope) satisfy:

$$f_1(p) = f_2(p) = f_2(p) = \frac{\alpha}{1 + p^m},$$

with the real parameter α and m defining the inhibitory regulatory functions. Moreover, in order to simplify all γ_{x_i} are assumed to be equal to 1, and it is also assumed that the decay rates $\alpha_{1_{x_i}}$ and $\alpha_{2_{x_i}}$ are equal to a real constant δ. Thus, the simplified symmetrical *repressilator* is described as follows:

$$\dot{x}_1 = r_{x_1} - \delta x_1, \quad \dot{r}_{x_1} = \frac{\alpha}{1 + x_3^m} - \delta r_{x_1}, \tag{4}$$

$$\dot{x}_2 = r_{x_2} - \delta x_2, \quad \dot{r}_{x_2} = \frac{\alpha}{1 + x_1^m} - \delta r_{x_2}, \tag{5}$$

$$\dot{x}_3 = r_{x_3} - \delta x_3, \quad \dot{r}_{x_3} = \frac{\alpha}{1 + x_2^m} - \delta r_{x_3}. \tag{6}$$

Applying the Mallet–Paret and Smith theorems (see [15]), it is proved in [6] that this cyclic system has only one unstable equilibrium point and displays stable oscillatory behavior when the following inequality holds:

$$\frac{\alpha^2}{\delta^2} > \sqrt[m]{\frac{\frac{4}{3}}{m - \frac{4}{3}}} \left(1 + \frac{\frac{4}{3}}{m - \frac{4}{3}}\right). \tag{7}$$

Let us now introduce our proposed tuning procedure.

4 Tuning Procedure

We consider here the symmetrical case for the repressilator (4)–(6), and we define the terms:

$$X = \begin{bmatrix} r_{x_1} \\ x_1 \\ r_{x_2} \\ x_2 \\ r_{x_3} \\ x_3 \end{bmatrix}, \quad Y = X, \quad P = \begin{bmatrix} \delta \\ m \\ \alpha \end{bmatrix}. \tag{8}$$

Our goal is to build a procedure to determine as precisely as possible the parameter vector P which tune the mathematical model (4)–(6) in order to make it to display the oscillatory behavior observed from the known state vector X.

In order to manage the algebraic manipulations of the system equations (4)–(6) we define:

$$f_1(y) = f(y_3); \quad f_2(y) = f(y_1); \quad f_3(y) = f(y_2); \tag{9}$$

where

$$f(x) = \frac{\alpha}{1 + x^m}. \tag{10}$$

Evidently, from the system equations (4)–(6) we have that the parameter δ is *algebraically identifiable* (see for instance [10]) with respect to the output vector, Y, because:

$$\dot{y}_i = y_{_{i+3}} - \delta y_i, i = \{1, 2, 3\}. \tag{11}$$

That is, δ can be reconstructed with high precision. On the other hand, once the parameter δ is available, the time-variable regulatory signals $f_i(y)$ are algebraically observable, because:

$$f_i(y) = \dot{y}_{i+3} + \delta y_{_{i+3}}, i = \{1, 2, 3\}. \tag{12}$$

It should be noticed that, if $f_i(y)$ is recovered, then α fulfills:

$$\alpha = f(y_3)(1+y_3^m) = f(y_1)(1+y_1^m) = f(y_2)(1+y_2^m). \tag{13}$$

Therefore, the parameter m can be estimated by solving the following algebraic nonlinear equation:

$$z_y(\widehat{m}) = \widehat{f}(y_3)(1 + y_3^{\widehat{m}}) - \widehat{f}(y_1)(1 + y_1^{\widehat{m}}), \tag{14}$$

where, $\widehat{f}(y_i)$ and $\widehat{m}$ are estimates of $f(y_i)$ and m, respectively.

Remark 5: From the previous analysis we conclude that the repressilator parameters can be recovered. Parameter δ can be then algebraically reconstructed with high precision, whereas parameters α and m can be estimated. It must be pointed out that our conclusion is related to the structural identifiability property introduced in $[12]$, and is derived from theoretical developments first exposed in $[10]$. Moreover, in the context of systems biology nonlinear identifiability (required for parameter reconstruction) has also been explored in $[1]$.

We show now the procedure recovering the parameter vector P with respect to the known output vector Y.

4.1 Model-based On-line Parameters Estimation

Let us introduce the following assumptions (that we require in order to ensure the solvability of our systems inverse problem):

A1 *The cyclic feedback system (4)–(6) is initialized to be strictly positive.*
A2 *The parameter vector P belongs to some neighborhood in the parameter space, such that, the system (4)–(6) exhibits sustained oscillatory behavior.*
A3 *The output vector Y is fully measurable.*

We proceed now to expose our parameter recovering procedure.

On–line algebraic recovering of δ

This procedure consist of multiplying both sides of (11) by the time variable t, and then, integrating by parts the resulting expression. Thus, we have the expression:

$$\int_0^t \sigma \dot{y}_i(\sigma) d\sigma = \int_0^t \sigma y_{i+3}(\sigma) d\sigma - \delta_i \int_0^t \sigma y_i(\sigma) d\sigma. \tag{15}$$

By integrating once the left-hand side of the above equation we obtain the following linear expression in terms of the unknown parameter δ:

$$\delta = \frac{\int_0^t \sigma y_{i+3}(\sigma) d\sigma - q_i(t)}{\int_0^t \sigma y_i(\sigma) d\sigma}, \tag{16}$$

where

$$q_i(t) = \int t \dot{y}_i(t) = t y_i - \int y_i(t). \tag{17}$$

For sake of simplicity, we introduce the following notation:

$$\overset{(m)}{\int} t^j x(t) = \int_0^t \int_0^{\sigma_1} \dots \int_0^{\sigma_{m-1}} (\sigma_m)^j x(\sigma_m) d\sigma_m d\sigma_{m-1} \dots d\sigma_1. \tag{18}$$

Remark 6: Note that the system equation (16) is not well defined at time $t = 0$. But, for any time t after a small open time interval of the form $(0, \epsilon)$ with $\epsilon > 0$, the denominator of the equation (16) is strictly positive by assumptions **A1** and **A2**. That is:

$$\int_0^t \sigma y_i(\sigma) d\sigma > 0; t > 0.$$

We focus our attention now on the computational procedure to recover the non-available regulatory signals f_i, as well as the related parameters.

Reconstructing the time variable signals $f_i(y)$ and the unknown parameters α and m

As can be seen, the implementation of the relations in (12) needs the time derivative of the signals y_{i+3} to be recovered. To overcome this issue, we introduce the following simple high–gain observer (see [4]). Assuming that the signals y_i are available, we can propose the following filter:

$$\begin{array}{l} \dot{\widehat{y}}_i = \widehat{z}_i - \frac{2k}{\xi}(\widehat{y}_i - y_i); \\ \dot{\widehat{z}}_i = -\frac{k^2}{\xi^2}(\widehat{y}_i - y_i); \end{array} \tag{19}$$

where ξ is a small positive parameter and k is a strictly positive parameter.

Proposition 1: Let $y_i(t)$ be a scalar continuous smooth function with its corresponding time derivatives satisfying $|\dot{y}| \leq n_i$. Then, the high–gain observer recovers the time derivative of $\dot{y}$, with bounded error given by:

$$|\widehat{z}_i - \dot{y}_i| \leq \frac{2\rho_i \xi}{k}; \text{for some } t^* > t. \tag{20}$$

Then, the states $\dot{y}_i$ converge to $\dot{\widehat{z}}_i$, when time elapses.

Proof: see Appendix A. □

Consequently, the time variant regulatory signal f_i can be reconstructed by:

$$\widehat{f}_i(y) = \dot{\widehat{y}}_{i+3} + \delta_i y_{i+3},$$

where $\dot{\widehat{y}}_{i+3}$ are directly obtained from the filter (19). Since all the output signals, y_{i+3}, present oscillatory behavior (which is one of our initial assumptions), this signals and their respective time derivatives are all bounded.

Now we are able to numerically solve the one variable nonlinear algebraic equation (14) and then be able to reconstruct the parameter m. To solve this equation we can apply the standard iterative Newton-Raphson method as follows:

$$\widehat{m}_{k+1} = \widehat{m}_k - \frac{z_y(\widehat{m}_k)}{z'_y(\widehat{m}_k)}, \tag{21}$$

where $z_y(m)$ is as defined in (14) and

$$z'_y(m) = \frac{\partial z_y(m)}{\partial m}.$$

Note that if $\widehat{m} \to m$, then $\widehat{\alpha}$ can be computed using the formula in (13). That is:

$$\widehat{\alpha} = f(y)(1 + y^{\widehat{m}}). \tag{22}$$

Remark 7: Note that (21) requires to sample the outputs $y_i = x_i$, for $i = \{1, 2, 3\}$, since $z_y(\widehat{m}_k) = \widehat{f}(y_3)(1 + y_3^{\widehat{m}_k}) - \widehat{f}(y_1)(1 + y_1^{\widehat{m}_k})$ (see (14)). For this we define the indexed variables $x_{ik} = x_i(t_k)$, where $t_{k+1} - t_k = T$, with the sampling time $T > 0$ fixed. We compute $\widehat{m}_k$, as long as $x_{1k} \neq x_{2k} \neq x_{3k}$.

We proceed now to illustrate our proposal with a numerical example.

5 Numerical Simulations

In this section, we test the effectivity of our proposed methodology via computer–based simulations involving our chosen cyclic feedback system, *the repressilator*.

5.1 The Simulated Repressilator

We simulate the repressilator's behavior choosing the parameter values: $m = 1.5$, $\delta = 1$, and $\alpha = \sqrt{40}$ (this selection guarantees sustained oscillatory behavior). As far as the initial conditions are arbitrary, we select them as:

$$(x1, x2, x3, r_{x_1}, r_{x_2}, r_{x_3}) = (1, 2, 3, 0.4, 0.6, 0.8).$$

5.2 Algebraic Reconstruction of the Parameter δ

The numerical implementation of our reconstructor is done by using the Matlab$^{\mathrm{TM}}$ numerical computing environment. We use the Runge–Kutta method with an integration step of 0.5×10^{-3}. The high-gain observer gains are fixed as $\xi = 0.5 \times 10^{-2}$ and $k = 1$. As already established, we need to recover the parameter δ first. In Figure 3, the reconstruction of parameter δ is shown.

As we can see, the parameter δ can be satisfactorily recovered regardless of the availability of measurements of r_{x_1}, r_{x_2}, and r_{x_3} (in fact, notice that for the current illustrative example a set of measurements is at disposal for identification).

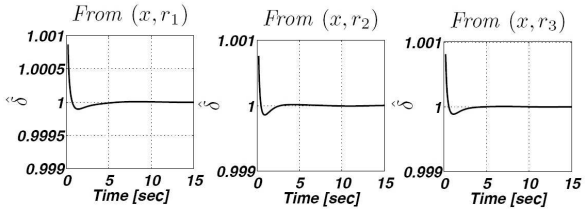


Fig. 3. Reconstruction error for the parameter δ

5.3 Estimation of the Non-available Regulatory Signals f_1, f_2 and f_3

Now, since we effectively recover δ we are able to reconstruct the non–available regulatory signals f_1, f_2 and f_3. Figure 4 shows the corresponding estimation errors for these three estimated signals. As we can see, the convergence time is very short because the high–gain observer gains were selected to be large enough.

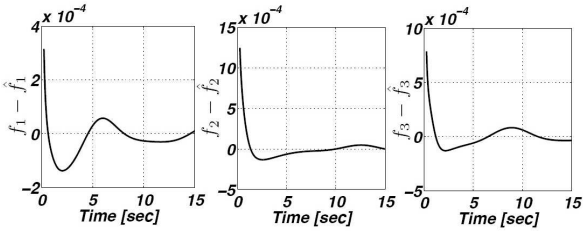


Fig. 4. Estimation errors corresponding to the estimated signals f_1, f_2, and f_3

5.4 Recovering Parameters m and α

Finally, having effectively estimated the regulatory signals $\widehat{f}_i$, the parameters m and α are effectively recovered, in that order, by solving the expression in (21) and (22) applying the Newton–Raphson method using samples of the regulatory signals $\widehat{f}_i$ and x_i every 0.5 seconds. As we can see, these parameters are in fact recovered with a very high accuracy (see Figure 5).

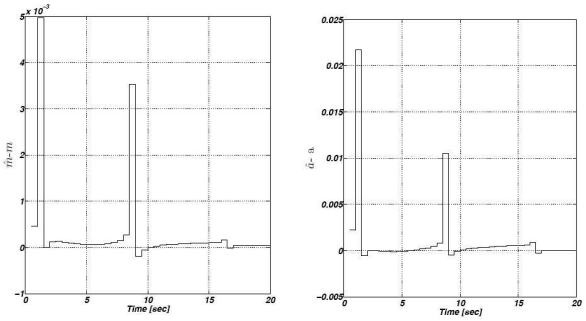


Fig. 5. Reconstruction parameters m and α

6 Conclusions

We proposed a methodology combining exact algebraic parameter reconstruction with nonlinear observed-based parameter estimation, intended to tune mathematical models guiding the design of synthetic biological circuits. We constrained our exposition to a class of cyclic modularly structured synthetic transcriptional networks which design is based on cyclic feedback systems described by deterministic lumped ordinary differential equations, and we illustrated our tuning procedure via computer–based simulations involving a low-dimensional cyclic feedback system corresponding to the celebrated synthetical transcriptional network called the *repressilator*. We focused our proposal to the minimization of the mismatch between the behavior of a synthesized biological circuit and the designed behavior coded by the mathematical model guiding the design process. The monitoring of the simulated synthesized biocircuit fed the information required to estimate the parameter values of the mathematical model.

We strongly believe that both exact algebraic parameter reconstruction and nonlinear observed-based parameter estimation are useful as a design tool for synthetic biology, since the solution of the systems inverse problem can be useful to close the gap between model–based designed biocircuits and implementations. As simple as it is, our methodology has been conceived as a *proof-of-concept* to illustrate how parameter tuning can be applied when considering the design of synthetic biocircuits when simple mathematical models are considered. It is quite obvious that realistic biocircuit design will involve the resolution of more complex systems inverse problems, particularly when phenomena like stochastically or retroactivity play a dominant role in the behavior of the involved systems.

To conclude, we must point out that the addition of identifiability properties to biocircuits may open the door to the inclusion of operational monitoring in biosynthetic networks, as a mechanism to promote functional reliability.

Acknowledgements

This research was supported by the Centro de Investigación en Computación of the Instituto Politecnico Nacional (CIC-IPN), and by the Secretaría de Investigación y Posgrado of the Instituto Politecnico Nacional (SIP-IPN), under Research Grant 20160268.

Appendix A: Proof of Proposition 1

We define the vector error as:

$$\xi_i = \left[\begin{array}{cc} y_i - \widehat{y}_i & z_i - \widehat{z}_i \end{array} \right]^T,$$

where $z_i = \dot{y}_i$. Therefore, the dynamic errors can be expressed after using (19), as follows:

$$\dot{\xi}_i = A_\epsilon \xi + \delta(t) \tag{23}$$

with

$$A_\epsilon = \left[\begin{array}{cc} -\frac{2k}{\epsilon} & 1 \\ -\frac{k^2}{\epsilon^2} & 0 \end{array} \right]; \quad \delta(t) = \left[\begin{array}{c} 0 \\ \dot{z}_i \end{array} \right]. \tag{24}$$

As can be seen, A_ϵ is a Hurwitz matrix, for all $k > 0$ and $\epsilon > 0$. Thus, error ξ_i satisfies:

$$\xi_i(t) = \left(e^{A_\epsilon t}\xi(0) + \int_0^t e^{A_\epsilon (t-s)}\delta(t)ds \right). \tag{25}$$

Since A_ϵ is exponentially stable and the signal $\dot{z}_i = \ddot{y}_i$is bounded, then the following inequality holds

$$\|\xi_i\| \leq \beta e^{-\frac{kt}{\epsilon}} \|\xi(0)\| + \frac{k\beta n_i}{\epsilon}(1 - e^{-\frac{kt}{\epsilon}}) \rightarrow \frac{k\beta n_i}{\epsilon},$$

where the constant β is as previously defined. This concludes the proof. □

References

1. **Anguelova, M. (2004).** *Nonlinear observability and identifiability: General theory and a case study of a kinetic model for S. Cerevisiae*. Chalmers University of Technology.
2. **Arcak, M. & Sontag, E. D. (2006).** Diagonal stability of a class of cyclic systems and its connection with the secant criterion. *Automatica*, Vol. 42, No. 9, pp. 1531–1537.
3. **Carlson, R. H. (2010).** *Biology is technology: the promise, peril, and new business of engineering life*, volume 2010. Harvard University Press Cambridge, MA.
4. **Dabroom, A. M. & Khalil, H. K. (1999).** Discrete-time implementation of high-gain observers for numerical differentiation. *International Journal of Control*, Vol. 72, No. 17, pp. 1523–1537.
5. **Del Vecchio, D., Ninfa, A. J., & Sontag, E. D. (2008).** Modular cell biology: retroactivity and insulation. *Molecular systems biology*, Vol. 4, No. 1, pp. 161.
6. **Del Vecchio, D. & Sontag, E. D. (2009).** Synthetic biology: A systems engineering perspective. *Control Theory and Systems Biology*, pp. 101–124.
7. **Elowitz, M. & Lim, W. A. (2010).** Build life to understand it. *Nature*, Vol. 468, No. 7326, pp. 889–890.
8. **Elowitz, M. B. & Leibler, S. (2000).** A synthetic oscillatory network of transcriptional regulators. *Nature*, Vol. 403, No. 6767, pp. 335–338.
9. **Endy, D. (2005).** Foundations for engineering biology. *Nature*, Vol. 438, No. 7067, pp. 449–453.

10. **Fliess, M. & Sira-Ramırez, H. (2004).** Reconstructeurs d'état. *Comptes Rendus Mathematique*, Vol. 338, No. 1, pp. 91–96.
11. **Gedeon, T. (1998).** *Cyclic feedback systems*, volume 637. American Mathematical Soc.
12. **Hermann, R. & Krener, A. J. (1977).** Nonlinear controllability and observability. *IEEE Transactions on automatic control*, Vol. 22, No. 5, pp. 728–740.
13. **Kim, J. & Winfree, E. (2011).** Synthetic in vitro transcriptional oscillators. *Molecular systems biology*, Vol. 7, No. 1, pp. 465.
14. **Lakin-Thomas, P. L. (2006).** Transcriptional feedback oscillators: maybe, maybe not... *Journal of biological rhythms*, Vol. 21, No. 2, pp. 83–92.
15. **Mallet-Paret, J. & Smith, H. L. (1990).** The poincaré-bendixson theorem for monotone cyclic feedback systems. *Journal of Dynamics and Differential Equations*, Vol. 2, No. 4, pp. 367–421.
16. **Martínez-García, J. C., Aguilar-Ibañez, C. F., Cabello-Sánchez, U., & Soria-López, A. (2012).** Tuning of mathematical models describing synthetic cyclic feedback biocircuits: combining exact algebraic parameter reconstruction and nonlinear parameter estimation. *20th International Symposium on Mathematical Theory of Networks and Systems (MTNS'2012)*, Melbourne, Australia.
17. **Myers, C. J. (2009).** *Engineering genetic circuits*. CRC Press.
18. **Shetty, R. P. (2008).** *Applying engineering principles to the design and construction of transcriptional devices*. Ph.D. thesis, Massachusetts Institute of Technology.
19. **Smolke, C. D. & Silver, P. A. (2011).** Informing biological design by integration of systems and synthetic biology. *Cell*, Vol. 144, No. 6, pp. 855–859.

Article received on 25/05/2016; accepted on 12/10/2016.
Corresponding author is Carlos Aguilar-Ibanez.

Estudio comparativo entre el algoritmo de Kleinberg y el algoritmo *Biased Selection* para la construcción de redes *small world*

Miguel Arcos-Argudo

Universidad Politécnica Salesiana del Ecuador,
Grupo de Investigación en Inteligencia Artificial y Tecnología de Asistencia (GI-IATa),
Ecuador

marcos@ups.edu.ec

Resumen. Actualmente las redes de tipo small world constituyen un tema muy importante en nuestro entorno, pues están presentes en varios fenómenos naturales y artificiales. Un objetivo de muchos algoritmos desarrollados para tratamiento de grafos es conseguir que un nodo cualquiera logre establecer una conexión directa con otro nodo del grafo seleccionado al azar obteniendo un "vecino lejano". Este trabajo muestra un estudio comparativo entre dos algoritmos que logran este objetivo: Kleinberg y *Biased Selection*. Se demuestra por medio de experimentos que los dos algoritmos son capaces de obtener la distribución de Kleinberg. Se concluye que el algoritmo de Kleinberg obtiene una distribución de muestreo que es directamente proporcional a la distancia Euclidiana, y *Biased Selection*, a pesar de que también obtiene una distribución directamente proporcional a la distancia Euclidiana, permite que un nodo lejano pueda ser determinado como vecino lejano con mayor frecuencia.

Palabras clave. *Biased Selection*, cadenas de Markov, grafos, Kleinberg, paseos aleatorios, *small world*.

Comparative Study between Kleinberg Algorithm and Biased Selection Algorithm for Small World Networks Construction

Abstract. Actually Small-World Networks is a very important topic, it is present in a lot of applications in our environment. A target of many algorithms is to establish methods to get that any node in a graph can establish a direct connection with a randomly "long-range neighbor". This work is comparative study between two algorithms that get this target (Kleinberg and Biased Selection), I demonstrate by my experiments that both get the Kleinberg's distribution. I conclude that the Kleinberg's algorithm distribution maintains a probability directly proportional to Euclidian distance, and Biased Selection, although also maintains a probability directly proportional to Euclidian distance, allows that a node can get a farther node as "long-range neighbor" more frequently.

Keywords. Biased selection, graph, Kleinberg, Markov chains, random walks, small worlds.

1. Introducción

En nuestro entorno existe una gran variedad de tipos de red, de diferente naturaleza y diversas estructuras. Los estudios realizados sobre redes han permitido entender varios fenómenos naturales y sociales, así como proponer soluciones a varios problemas. En 1998 Watts y Strogatz [6] introdujeron un modelo matemático de red llamado "small world", el mismo que sostiene que dada una red R en la que se encuentran los nodos a, b y c, separados por una distancia d, en donde, $d(a,b) > d(a,c) \geq 1$ y, $d(a,b) > d(c,b) \geq 1$, es posible mediante la construcción de una red *small world*, establecer un enlace unidireccional entre a y c que defina una distancia $d'(a,c) = 1$, por lo que la distancia $d'(a,b) = 1 + d(c,b) \leq d(a,b)$, es decir, se ha acercado el nodo a al nodo b, mediante un camino de longitud $d'(a,b) = d(a,b) - d(a,c) + 1$, sin embargo, esto no implica que el nodo b se ha acercado al nodo a, puesto que el nodo b no ha creado ningún enlace diferente a los que tenía inicialmente. También es evidente que $d'(a,c) \leq d(a,c)$, por lo que también el nodo a se ha acercado al nodo c, siempre y cuando $d(a,c) > 1$. Cuando se crea un enlace unidireccional entre un nodo a y un nodo c, se dice que c es un vecino lejano de a. Este concepto se utiliza para crear redes de tipo small world, y para entender diferentes fenómenos que suceden en la

redes [7]. Una manera de crear redes small world es mediante el algoritmo de Kleinberg [1, 3, 4, 13] y mediante el algoritmo de *Biased Selection* [2].

La implementación de algoritmos que permitan establecer vecinos lejanos de manera aleatoria entre los nodos de una red es un tema de gran importancia para la computación, y es de interés general debido a la variedad de áreas en las que se los puede aplicar, por ejemplo: reducción del valor promedio de consumo de energía, y conservación de la energía en redes tipo Ad-hoc [8, 9]; ayuda a la detección de inconvenientes que pueden complicar la aplicación de algoritmos y software diseñados para estudios sociales [10]; estudios sobre fenómenos de redes sociales de tipo small world [11].

Este trabajo muestra los resultados de un estudio comparativo entre los dos algoritmos luego de haber generado redes small world utilizando las dos técnicas. Se concluye que Biased Selection (BS) puede generar una small world con una distribución semejante a la de Kleinberg, pero otorgando a los nodos más alejados una mejor probabilidad de ser escogidos como vecinos lejanos que la que otorga Kleinberg.

2. Descripción de los algoritmos

Este trabajo muestra una comparativa entre el clásico algoritmo de Kleinberg y *Biased Selection*, ambos permiten crear redes small world mediante la generación de enlaces unidireccionales entre un nodo y un vecino lejano; para ello se ha trabajado sobre un grafo con la estructura de un toroide completo de dimensiones $100x100$, que es una estructura de red en forma de grilla, el grafo está conformado por diez mil nodos, todos los nodos tienen inicialmente cuatro vecinos, cada nodo i posee un identificador que es un número secuencial entre 0 y 9999, además, cada nodo i tiene una ubicación de coordenadas (x_i, y_i) dentro del grafo (Figura 1).

El Algoritmo de Kleinberg

El algoritmo de Kleinberg, para crear un enlace a un vecino lejano desde un nodo determinado, propone primeramente calcular la sumatoria del inverso en las distancias Euclidianas elevadas a la segunda potencia, desde dicho nodo a todos los demás nodos, como se muestra en (1).

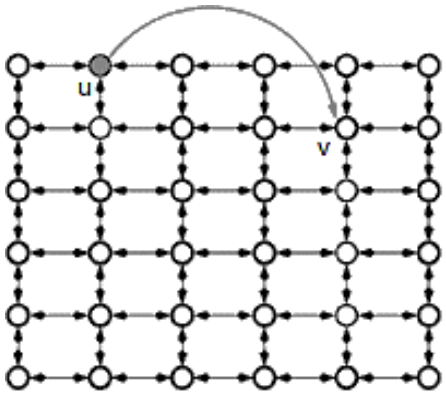


Fig. 1. Ilustración de un toroide completo

$$S = \sum_{i=0}^{m-1} \frac{1}{d(u, v_i)^2}, \quad (1)$$

donde m es el número total de nodos que tiene el toroide, y d es la distancia Euclidiana entre el nodo origen u y el nodo v_i, (si $u = v_i$, entonces $d(u, v_i) = 0$). El cálculo de la distancia Euclidiana entre dos nodos viene dado por (2):

$$d_e = \sqrt{(\min(|x_1 - x_2|, m - |x_1 - x_2|))^2 + (\min(|y_1 - y_2|, m - |y_1 - y_2|))^2}, \quad (2)$$

donde (x_1, y_1) representa la ubicación del nodo u en el toroide, y (x_2, y_2) representa la ubicación del nodo v_i en el toroide.

La figura 2. Muestra la variación de la inversa distancias Euclidiana entre el nodo 0 $(u = nodo\ 0)$ y los demás nodos del grafo, elevadas al cuadrado. En esta figura se puede apreciar un gráfico de tipo "*cola larga*".

Este algoritmo propone generar un número p de manera aleatoria con probabilidad uniforme entre 0 y S, finalmente vuelve a calcular la sumatoria del inverso de las distancias Euclidianas elevadas a la segunda potencia S' entre el nodo origen y los demás nodos pero solamente hasta que se cumpla (3):

$$S' \geq p. \quad (3)$$

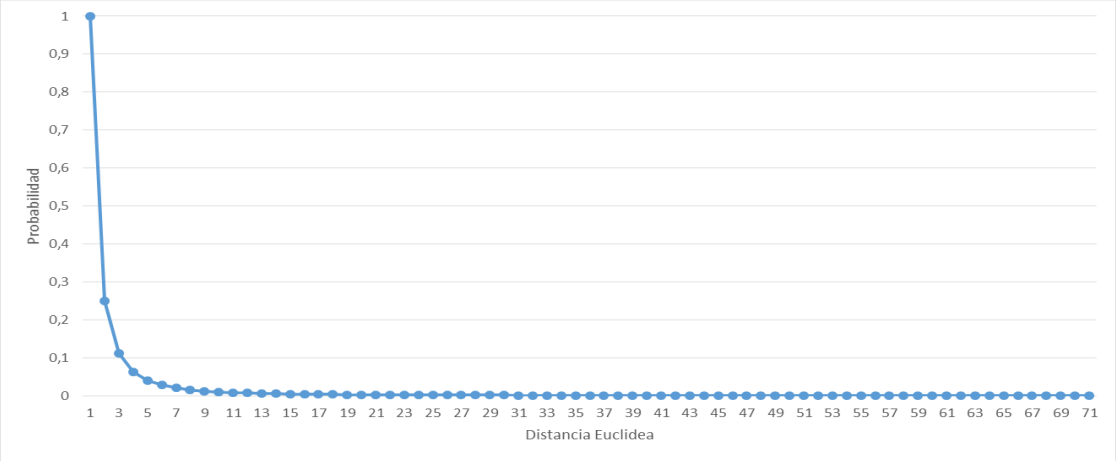


Fig. 2. Variación de la inversa de las distancias Euclidianas elevadas al cuadrado, entre el nodo 0 y los demás nodos del toroide

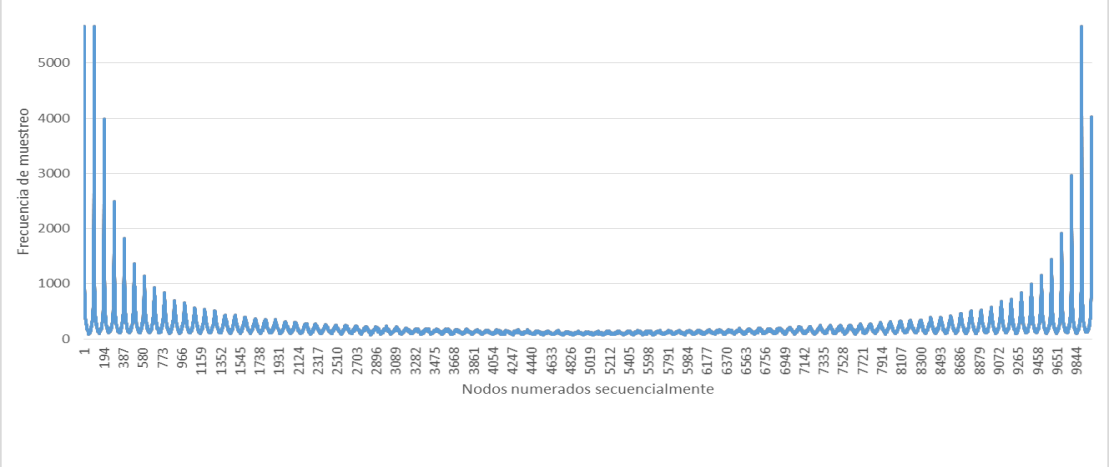


Fig. 3. Frecuencia de los vecinos lejanos escogidos para el nodo *0* ordenados por su identificador

El vecino lejano será el nodo que el algoritmo haya tomado como muestra cuando se cumpla (3). Por ejemplo, Si $S' \geq p$ cuando en la sumatoria definida en (1) sea $i = j$, entonces el valor de S' estaría definida por (4), y el vecino lejano escogido para el nodo u sería el nodo v_j [5]:

$$S' = \frac{1}{d_e(u, v_0)^2} + \frac{1}{d_e(u, v_1)^2} + \cdots + \frac{1}{d_e(u, v_j)^2}. \quad (4)$$

Se han realizado experimentos tomando 500000, 1000000 y hasta 2000000 de vecinos lejanos para el nodo 0 (nodo origen).

El algoritmo de Biased Selection

Otra manera de obtener un vecino lejano para un nodo es mediante la implementación del algoritmo *Biased Selection* (BS) [2]. Este algoritmo alcanza la distribución de probabilidad deseada cuando la distribución estacionaria del proceso de Markov es alcanzada [12].

El funcionamiento de este algoritmo se describe a continuación: se toma un nodo determinado como nodo origen (nodo u), y se obtiene dos nodos aleatorios x e y escogidos con

probabilidad uniforme distintos al nodo origen ($u \neq x \neq y$), de los cuales, inicialmente el nodo escogido como vecino lejano de u es x, la probabilidad de que el nodo y sea finalmente escogido como vecino lejano de u viene dada por la probabilidad resultante de (5):

$$P(u,y) = \frac{w(y)}{w(x) + w(y)}, \quad (5)$$

donde:

$$w(y) = d_e(u,y), \quad (6)$$

$$w(x) = d_e(u,x). \quad (7)$$

Luego se genera un número aleatorio q entre 0 y 1 de manera uniforme, el nodo escogido como vecino lejano de u es y cuando se cumple (8), en cualquier otro caso el vecino lejano de u es x; hay que tener en cuenta que BS es un algoritmo iterativo, por lo cual para establecer un vecino se debe realizar este proceso un determinado número de *rondas*:

$$q \geq P(u,y). \quad (8)$$

El número de muestras obtenido fue el mismo para los dos algoritmos, con el fin de poder hacer la comparativa correspondiente.

Se puede notar entonces que en los dos casos, un nodo v tendrá mayor posibilidad de ser escogido como vecino lejano de un nodo u, (para $u \neq v$) mientras $d_e(u,v)$ sea más pequeña, es decir, los nodos que se encuentren en el lado opuesto del toroide tendrán menor probabilidad de ser escogidos.

La Figura 3 muestra la frecuencia con la que cada nodo fue escogido como vecino lejano, ordenando los nodos según su identificador, desde 1 hasta 9999.

El objetivo de la Figura 3 es determinar los nodos que tendrían menos probabilidad de ser escogidos como vecinos lejanos del nodo 0, según la gráfica se puede ver que los nodos ubicados en la mitad de la numeración secuencial han sido muestreados menor cantidad de veces, pudiendo definir que para el nodo 0 el nodo más lejano es el que tenga la posición dentro de la grilla de $(50, 50)$, es decir el nodo cuyo identificador es 5050; al aplicar (2), se obtiene que:

$$d_e(0{,}5050) = \sqrt{5000},$$

por lo tanto, $P(0, 5050) = 0.0002$, siendo esta la probabilidad menor de todas las posibles en el experimento, esto da una pauta para determinar el número mínimo de muestras necesarias para el experimento. Determinaremos que el número de muestras sea tal que le permita al nodo con menor probabilidad ser muestreado cien veces, entonces:

$$Cantidad\ Muestras * probabilida\ menor = 100, \quad (9)$$

$$Cantidad\ Muestras = \frac{100}{probabilidad\ menor}. \quad (10)$$

Aplicando (10), el número de muestras requerido para el experimento fue de $500{,}000$, sin embargo, para dar mayor credibilidad al trabajo, se tomó una cantidad de muestras de hasta dos millones en cada experimento.

Para la ejecución de este trabajo se ha utilizado un grafo generado y cargado en memoria, dicho grafo representa un toroide completo de dimensiones *100 x 100.*

Además se puede decir que el grafo es conexo, originalmente es no dirigido, y que sus aristas no tienen peso.

3. Discusión de resultados

Para los experimentos con los dos algoritmos se ha establecido al nodo 0 como nodo origen, esto sería irrelevante, ya que en el toroide completo todos los nodos tienen igual centralidad y grado.

Algoritmo de Kleinberg: en este algoritmo prevalece por completo la probabilidad de cada nodo candidato a ser escogido como vecino largo, ya que la distribución del muestro tiene la forma de una “cola larga”, en la que los nodos escogidos como vecinos lejanos del nodo 0 se “ordenan ascendentemente” según su distancia Euclidiana al nodo origen, como se puede apreciar en la

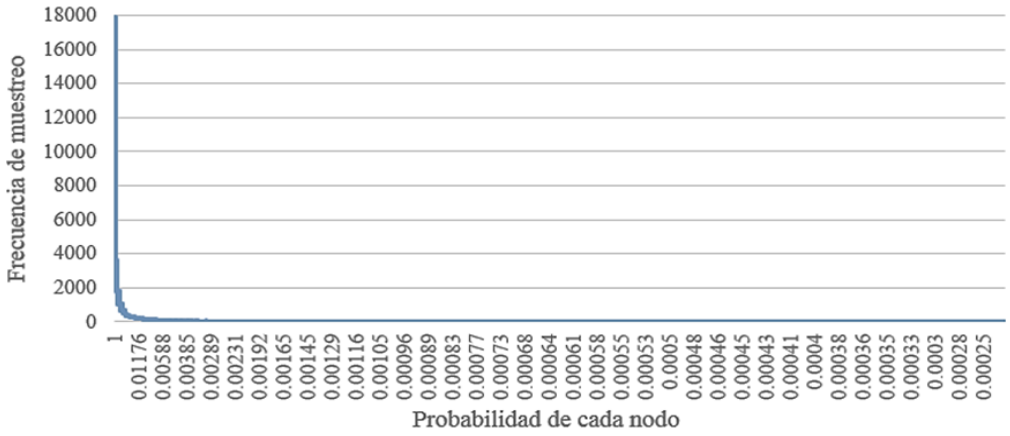


Fig. 4. Distribución de la frecuencia de elección de cada nodo como vecino lejano establecida por el algoritmo de Kleinberg

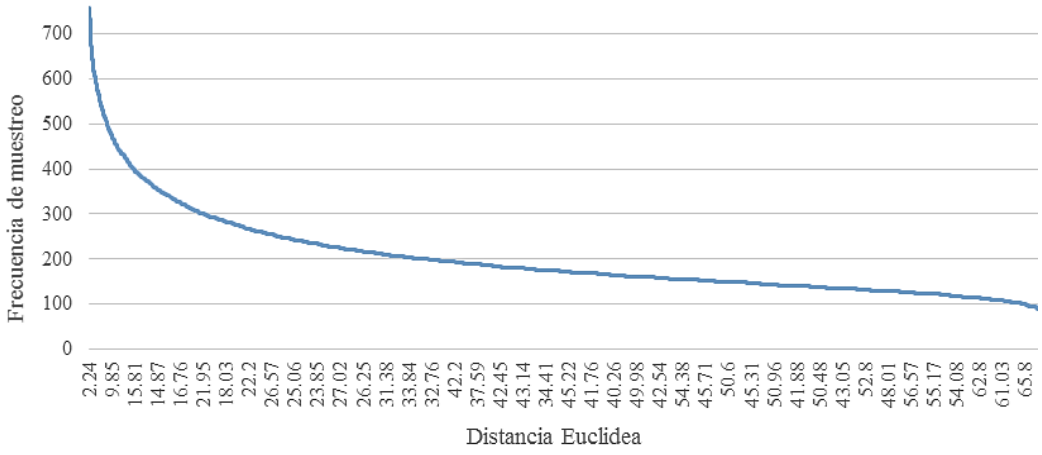


Fig. 5. Distribución del muestro generado por BS con 3 rondas

figura 4. Al comparar la figura 4 y la figura 2 se nota la similitud de las curvas, lo que corrobora esta deducción.

Algoritmo Biased Selection: Los experimentos realizados con el algoritmo BS fueron ejecutados con un número de rondas que varía de forma incremental, con ello se ha notado que mientras mayor es el número de rondas, la distribución de la frecuencia del muestreo resulta ser más parecida a la distribución de la frecuencia del muestreo de Kleinberg, a continuación se muestra los resultados más relevantes. Desde la figura 5 hasta la figura 11, se puede observar la similitud que adquiere la distribución obtenida con la distribución expuesta en la figura 4.

Desde la figura 8 hasta la figura 11, las curvas obtenidas no muestran variaciones significativamente importantes, sin embargo todas estas curvas ya han alcanzado la distribución de Kleinberg, que es semejante a la curva visualizada en la figura 2 (efecto “cola larga”).

Al graficar los resultados obtenidos luego de haber ejecutado los dos algoritmos con los que se han establecido hasta dos millones de vecinos lejanos para el nodo 0 de un toroide de tamaño $100x100$, y en el caso de BS con diferentes números de rondas, hemos podido verificar que el algoritmo BS alcanza la distribución de Kleinberg aproximadamente con un número de rondas igual a 100.

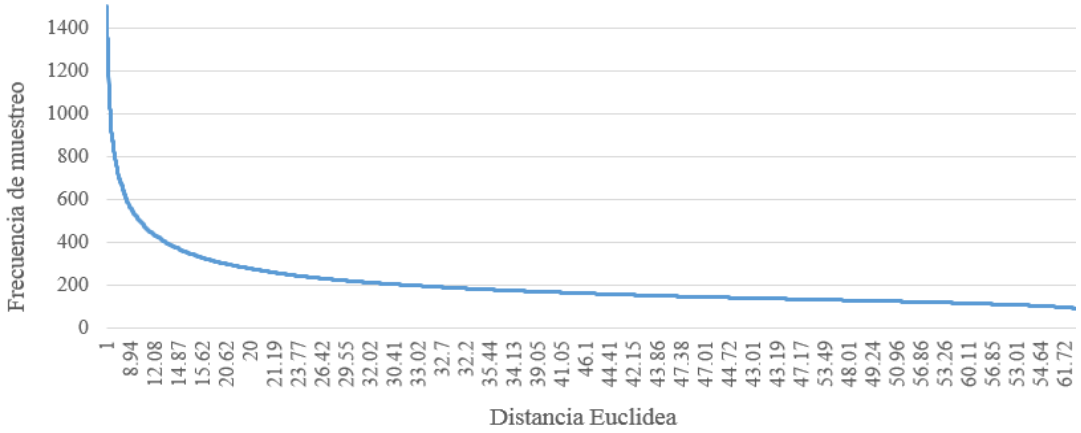


Fig. 6. Distribución del muestro generado por BS con 13 rondas

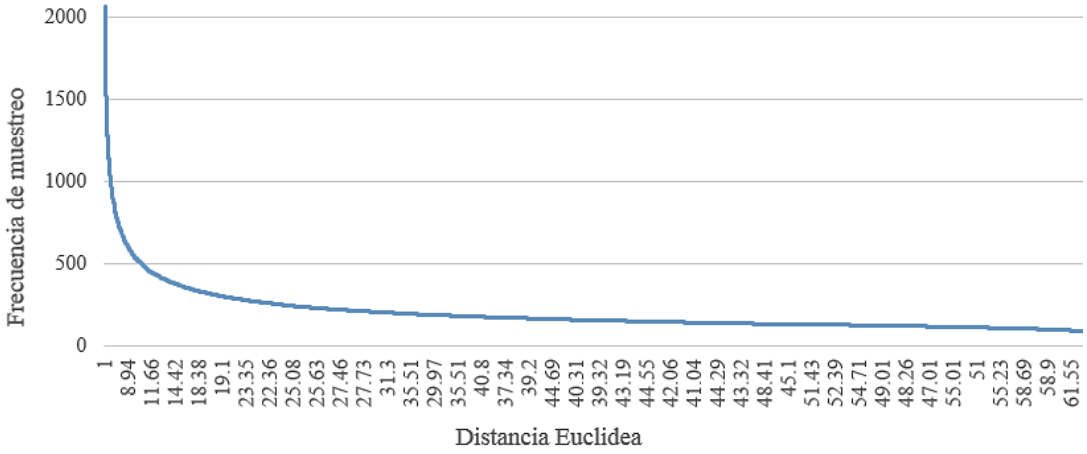


Fig. 7. Distribución del muestro generado por BS con 23 rondas

En la distribución generada por el algoritmo de Kleinberg la frecuencia de la aparición de los nodos obtenidos como muestra mantiene un orden que coincide con la ordenación ascendente de las distancias Euclidiana de dichos nodos al nodo origen, es decir, esta probabilidad se mantiene.

En la distribución generada por BS, este orden se mantiene desde los nodos vecinos que tienen una distancia Euclidiana con respecto al nodo origen igual a 1, hasta los que tienen una distancia Euclidiana con respecto al nodo origen igual a 8, los nodos que tienen distancias mayores con respecto al nodo origen empiezan a ser escogidos como vecinos lejanos de manera aleatoria, es decir, el algoritmo BS da oportunidad a que el nodo origen alcance un vecino lejano que esté ubicado a una distancia mayor y de manera aleatoria con mayor frecuencia cuando el número de rondas se incrementa. BS puede ser una mejor opción que Klienberg cuando se desea que la distribución de las frecuencias del muestreo tenga una mayor uniformidad.

En la figura 12 se puede comparar las dos distribuciones de frecuencia obtenidas, en ella hay evidencia de que la frecuencia con la que los nodos más cercanos al nodo origen han sido escogidos con el algoritmo BS es mucho menor que la frecuencia con la que los mismos nodos fueron escogidos al utilizar Kleinberg.

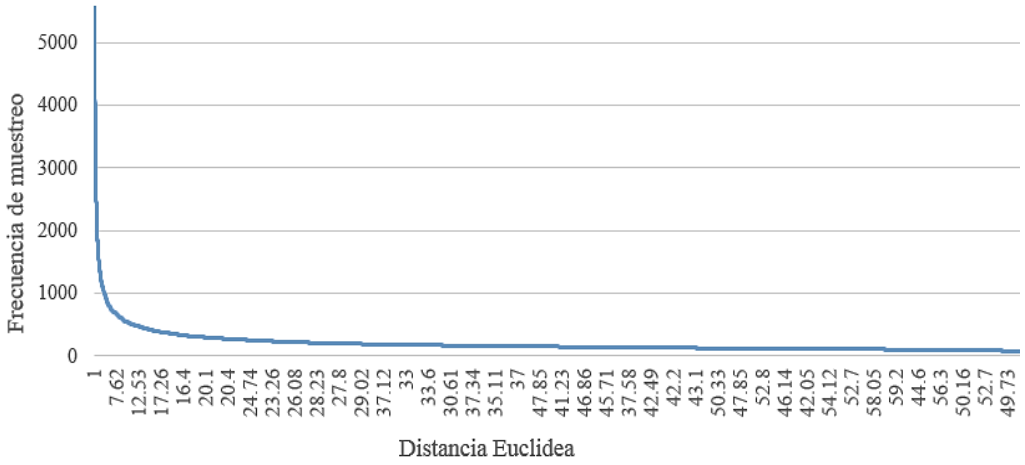


Fig. 8. Distribución del muestro generado por BS con 100 rondas

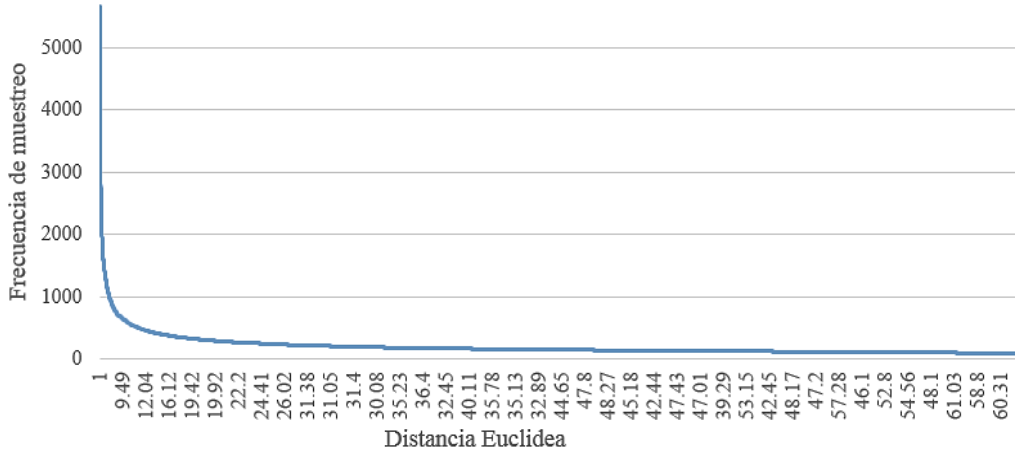


Fig. 9. Distribución del muestro generado por BS con 200 rondas

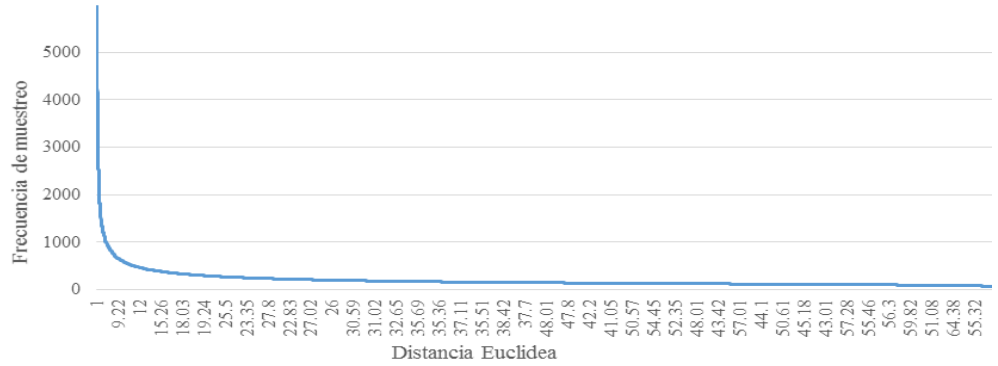


Fig. 10. Distribución del muestro generado por BS con 1000 rondas

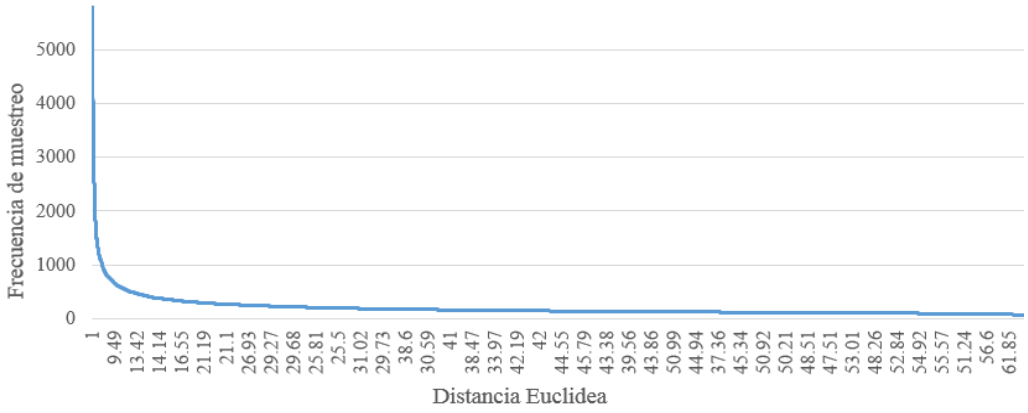


Fig. 11. Distribución del muestro generado por BS con 2000 rondas

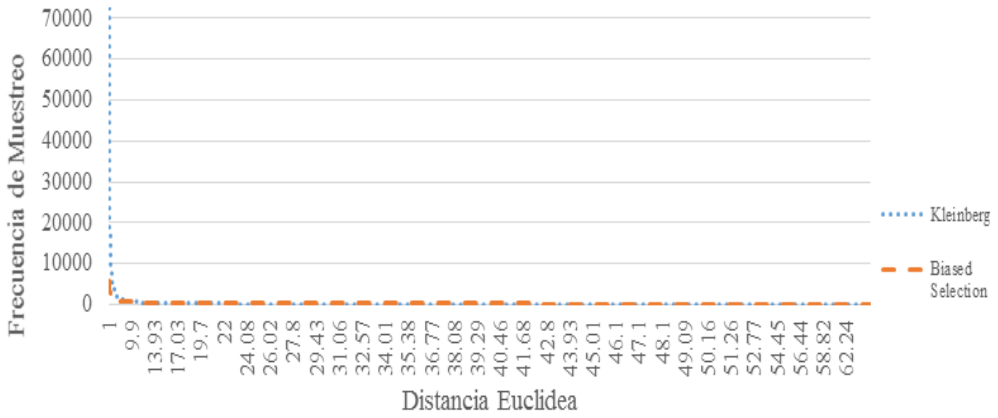


Fig. 12. Representación gráfica de las frecuencias de muestreo obtenidas por BS y por Kleinberg

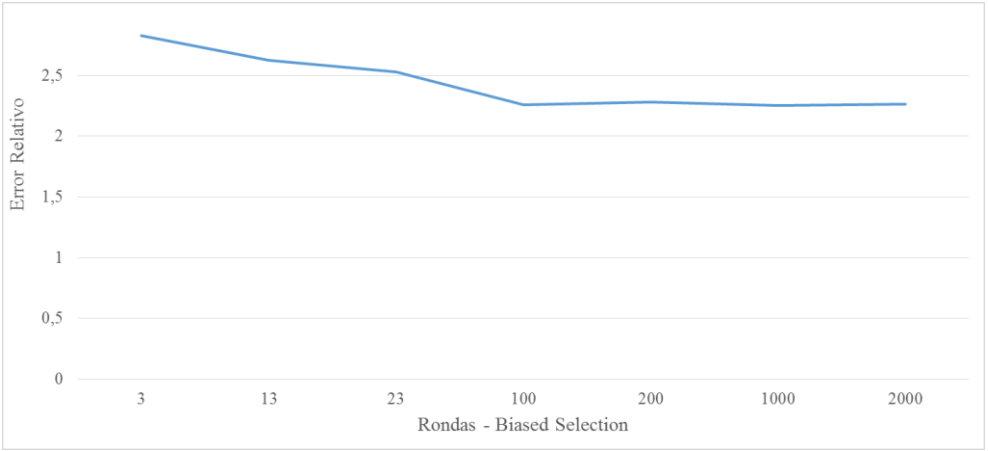


Fig. 13. Curva del error relativo de la frecuencia de muestro generado por el algoritmo BS, en relación con la frecuencia de muestro generado por el algoritmo de Kleinberg

Tabla 1. Nodos elegidos por Kleinberg como Vecinos Lejanos

Tamaño dela muestra		Identificador del nodo	Número de veces elegido
500000	Nodos más elegidos	1	17804
		99	17776
		100	17885
		1900	8867
	Nodos menos elegidos	5049	3
		5051	2
		5150	5
		5050	1
1000000	Nodos más elegidos	1	35832
		99	35559
		100	36098
		1900	36218
	Nodos menos elegidos	5049	8
		5051	5
		5150	6
		5050	4
2000000	Nodos más elegidos	1	72418
		99	72181
		100	71509
		1900	71099
	Nodos menos elegidos	5049	17
		5051	13
		5150	13
		5050	7

La figura 13 muestra la gráfica del error relativo de la selección de los vecinos lejanos escogidos por BS, comparado con la selección de los vecinos lejanos escogidos por Kleinberg, la gráfica permite observar el error relativo obtenido versus el número de rondas que se realizaron antes de escoger el vecino con BS. Se evidencia que el valor del error se estabiliza luego de 100 rondas. BS no intenta simular un muestreo idéntico al de Kleinberg con un error relativo bajo, sino crear una red small world similar a la generada por el algoritmo de Kleinberg, pero con una distribución de frecuencia de muestro con uniformidad mayor.

La Tabla 1 muestra los resultados de tres experimentos realizados con el algoritmo de Kleinberg sobre el mismo toroide, en los cuales se seleccionaron 500.000, $1'000.000$ y $2'000.000$ de vecinos lejanos para el nodo 0 (nodo origen), la tabla muestra nodos que han sido seleccionados como vecino lejano con mayor frecuencia y los que han sido seleccionados como vecinos lejanos con menor frecuencia; los nodos que han sido seleccionados una mayor cantidad de veces como vecino lejano son los más cercanos al nodo origen (nodo 0), es decir, aquellos cuatro nodos que se encuentran a un distancia Euclidiana igual a 1; y los nodos que han sido seleccionados una menor cantidad de veces como vecino lejano son aquellos que están más alejados del nodo origen, es decir aquellos que se encuentran en el lado opuesto del toroide (el nodo 5050 es el menos escogido).

Se puede evidenciar que en los tres experimentos se han escogido los mismos nodos tanto con la mayor frecuencia como con la menor frecuencia, además, mientras más vecinos lejanos se obtengan para el nodo origen, más veces se seleccionan los nodos que se encuentran a distancia Euclidiana igual a 1, mostrando un incremento de la frecuencia de muestreo muy apreciable. En cambio, a pesar de que en cada experimento se ha incrementado el tamaño de la muestra de manera considerable, los nodos menos seleccionados como vecinos lejanos experimentan un incremento en su frecuencia de muestreo demasiado pequeño.

En la Tabla 2 se puede ver que en ningún caso el nodo menos elegido por BS ha sido el nodo 5050 que es el nodo más alejado con respecto al nodo origen (nodo 0).

La Tabla 3 muestra resultados obtenidos por el algoritmo de Kleinberg y por BS en un experimento en el que se seleccionaron $2'000.000$ de vecinos lejanos para el nodo origen (nodo 0), en la primera columna se muestra el identificador del nodo, en la segunda columna se muestra la cantidad de veces que el nodo ha sido muestreado por el algoritmo de Kleinberg, en las cuatros primeras filas se muestran los nodos más elegidos, estos son los nodos que están a una distancia Euclidiana igual a

Tabla 2. Nodos elegidos en menos ocasiones como Vecino Lejano por BS

Rondas	Identificador del nodo	Número de veces elegido	Distancia Euclidiana al nodo origen
3	4859	36	63.13
	5555	36	63.64
	4642	35	62.29
	5046	34	67.94
13	5152	75	68.59
	5257	75	64.44
	5551	74	66.53
	5046	66	67.94
23	5249	75	68.59
	5246	75	66.48
	5152	74	68.59
	5046	67	67.94
100	4948	58	68.59
	5241	56	63.13
	6145	54	59.55
	4951	52	69.30
200	4845	69	65.80
	5644	69	62.23
	4752	68	67.18
	5045	67	67.27
1000	3756	56	57.49
	5847	56	63.03
	5345	54	65.07
	4948	52	68.59
2000	4152	54	63.13
	4946	52	67.21
	5254	52	66.48
	5146	44	67.21

1, es decir los cuatro nodos adyacentes al nodo origen dentro del toroide original.

Desde la fila quinta hasta la octava se encuentran los resultados de los nodos menos elegidos por Kleinberg, estos son los nodos que más alejados se encuentran con respecto al nodo origen, la distancia de cada nodo al nodo origen se muestra en la tercera columna de la Tabla 3; como se puede notar esto corresponde estrictamente con la probabilidad que el algoritmo de Kleinberg otorga a cada nodo de ser escogido como vecino lejano, a menor distancia mayor es la probabilidad, y es evidente que no puede existir una distancia menor a 1.

Desde la cuarta columna hasta la décima columna se muestran los resultados obtenidos por BS para los mismos nodos, para ejecuciones del algoritmo con 3 rondas, 13 rondas, 23 rondas, 100 rondas, 200 rondas, 1000 rondas y 2000 rondas.

Los resultados coinciden en cuanto a los nodos que han sido elegidos una mayor cantidad de veces, pues con BS los nodos que han obtenido una mayor frecuencia de elección como vecino lejano son también los que se encuentran a una distancia Euclidiana igual a 1, pero esta frecuencia es significativamente mayor con respecto a los resultados otorgados por Kleinberg, teniendo en cuenta que con los dos algoritmos se han obtenido el mismo número de muestras, lo cual implica que esa diferencia se encuentra repartida en frecuencias de muestreo de otros nodos más lejanos, lo cual implica que la distribución de la frecuencia de muestreo de BS tiene una uniformidad mayor que la frecuencia de muestreo de Kleinberg.

En cuanto a los nodos que han sido elegidos menos veces por Kleinberg, se puede notar que con BS también han sido escogidos un número mucho menor de veces, sin embargo no son precisamente los nodos menos muestreados en cada uno de los experimentos; los cuatro nodos menos elegidos por BS en cada uno de los experimentos se muestran en la Tabla 2, en ella se puede ver que los nodos menos muestreados son diferentes en cada experimento, esto evidencia que BS otorga una mayor probabilidad a los nodos más lejanos de ser elegidos como vecinos lejanos de manera aleatoria con respecto a la probabilidad otorgada por Kleinberg.

4. Conclusiones

El algoritmo *Biased Selection* puede alcanzar mediante la construcción de una red small world, una distribución de frecuencia de muestreo semejante a la que genera el algoritmo de Kleinberg. La distribución de la frecuencia de

Tabla 3. Nodos escogidos como Vecino Lejano por el algoritmo de Kleinberg para dos millones de muestras

	Identificador del nodo	Número de veces elegido con Kleinberg	Distancia Euclidiana al nodo origen	Número de veces elegido con BS						
				3 rondas	13 rondas	23 rondas	100 rondas	200 rondas	1000 rondas	2000 rondas
Nodos elegidos en más ocasiones	1	72418	1	738	1435	2045	5576	5669	5904	5726
	99	72181	1	752	1437	2024	5464	5543	5740	5678
	100	71509	1	730	1455	2065	5510	5661	5974	5796
	9900	71099	1	752	1507	2060	5522	5668	5512	5734
Nodos elegidos en menos ocasiones	5049	17	70.01	39	83	78	88	82	84	82
	5051	13	70.01	38	102	87	84	88	96	84
	5150	13	70.01	41	90	91	70	79	80	74
	5050	7	70.71	48	92	85	84	77	96	56

muestreo generada por el algoritmo de Kleinberg dentro de un toroide es directamente proporcional a la variación de las distancias Euclidiana desde todos los nodos al nodo origen ordenadas en forma ascendente. La distribución de la frecuencia de muestreo generada por BS en un toroide será más semejante a la distribución de Kleinberg a medida que se aumente el número de rondas, sin embargo existirá un número de rondas suficiente, este número dependerá de las dimensiones del toroide sobre el que se trabaje.

El algoritmo del Kleinberg mantiene la frecuencia del muestreo para todos los nodos a pesar de que se lo ejecute con tamaños diferentes de muestra dentro de un toroide, en cambio BS a pesar de que considera la distancia Euclidiana para tomar muestras, otorga una mayor aleatoriedad al momento de seleccionar un vecino lejano, lo que posibilita a los nodos más lejanos al origen ser seleccionados con una frecuencia mayor que la otorgada por Kleinberg. El tiempo de ejecución de BS es mucho menor obteniendo el mismo número de muestras en comparación con el algoritmo de Kleinberg.

Referencias

1. **Watts, D. J. & Strogatz, S.H. (1998).** Collective dynamics of “small-world” networks. *Nature*, Vol. 393, No. 6684, pp. 440–442. DOI:10.1038/30918.
2. **Barbour, A. D. & Reinert, G. (2000).** *Small worlds*. arXiv preprint cond-mat/0006001.
3. **Barrière, L., Fraigniaud, P., Kranakis, E., & Krizanc, D. (2001).** Efficient routing in networks with long range contacts*. International Symposium on Distributed Computing,* Springer Berlin Heidelberg*,* pp. 270–284. DOI: 10.1007/3-540-45414-4_19.
4. **Bertier, M., Bonnet, F., Kermarrec, A. M., Leroy, V., Peri, S., & Raynal, M. (2010).** D2HT: the best of both worlds, Integrating RPS and DHT. *European Dependable Computing Conference (EDCC),* pp. 135–144. DOI:10.1109/EDCC.2010. 25.
5. **Bonnet, F., Kermarrec, A. M., & Raynal, M. (2007).** Small-world networks: From theoretical bounds to practical systems. *International Conference on Principles of Distributed Systems,* Springer Berlin Heidelberg, pp. 372–385. DOI: 10.1007/978-3-540-77096-1_27.
6. **Sevilla, A., Mozo, A., Lorenzo, M. A., López-Presa, J. L., Manzano, P., & Anta, A. F. (2010).** Biased selection for building small-world networks. *Principles of Distributed Systems,* Springer Berlin Heidelberg, pp. 32–47. DOI: 10.1007/978-3-642-17653-1_3.
7. **Kleinberg, J. M. (2000).** Navigation in a small world. *Nature*, Vol. 406, No. 6798, pp. 845–845. DOI: 10.1038/35022643.
8. **Chen, Y. S., Chiang, L. S., Tang, C. J., & Cheng, R. S. (2015).** Power saving protocol based on rectangular grid quorums for IEEE 802.11 ad hoc networks. *11th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness* (*QSHINE*), pp. 370–374.

9. **Rainy, J. M. S. & Nesasudha, M. (2015).** Neighbor discovery in ad-hoc networks using dual band scheme. *International Conference on Electronics and Communication Systems (ICECS),* pp. 1216–1219. DOI: 10.1109/ECS.2015.7124777.
10. **Romanchukov, S. & Berestneva, E. V. (2016).** Issues of Application Design for Social Studies. *The European Proceedings of Social & Behavioural Sciences* (*EpSBS*), Vol. 7: Lifelong Wellbeing in the World, pp. 88–94. DOI: dx.doi.org-10.15405-epsbs.2016.02.12.pdf.
11. **Opsahl, T., Vernet, A., Alnuaimi, T., & George, G. (2017).** Revisiting the Small-World Phenomenon: Efficiency Variation and Classification of Small-World Networks. *Organizational Research Methods*, Vol. 20, No. 1, pp. 149–173.
12. **Sevilla, A., Mozo, A., & Anta, A. F. (2015).** Node sampling using random centrifugal walks. *Journal of Computational Science*, Vol. 11, pp. 34–45.
13. **Malkov, Y. A. & Yashunin, D. A. (2016).** *Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs*. arXiv preprint arXiv:1603.09320.

Artículo recibido el 25/11/2016; aceptado el 22/03/2017.
Autor de correspondencia es Miguel Arcos-Argudo.

Un modelo para determinar la madurez de la automatización de las pruebas del software como área de investigación y desarrollo

Edgar Serna M.[1], Raquel Martínez M.[2], Paula Andrea Tamayo O.[2]

[1] Corporación Universitaria Remington, Medellín, Antioquia, Colombia

[2] Institución Universitaria de Envigado, Envigado, Colombia

edgar.serna@uniremington.edu.co, raquel.martinez@iue.edu.co, patamayo@correo.iue.edu.co

Resumen. En este artículo se propone un modelo para determinar la madurez actual de la automatización de las pruebas como área de investigación y de desarrollo en la industria del software. Se describe el proceso y los resultados de una investigación que tiene como objetivo determinar el nivel de madurez de la automatización. Se realizó una revisión sistemática de la literatura en la que se encontraron 978 trabajos que describen modelos de madurez de las pruebas del software y/o analizan la eficacia y la eficiencia de la automatización hasta el momento. Luego de un proceso de análisis y de aplicar los criterios de inclusión/exclusión y de valoración a la calidad se extrajeron 26 trabajos. La conclusión final es que el nivel de madurez actual de la automatización de las pruebas del software es Adolescente.

Palabras clave. Modelo de madurez, calidad, ingeniería de software, ciclo de vida, industria del software.

A Model for Determining the Maturity of Automation of Software Testing as a Research and Development Area

Abstract. This article proposes a model to determine the current maturity of test automation as an area of research and development in the software industry. The process and the results of a research that aims to determine the maturity level of automation is described. A systematic review of the literature in which 978 papers were found where are described the models of maturity of software testing and/or analyze the effectiveness and efficiency of automation was performed. After a process of analysis and apply the inclusion/exclusion criteria and quality assessment 26 papers were extracted. The final conclusion is that the current maturity level of automation of software testing is teenager.

Keywords. Maturity model, quality, software engineering, life cycle, software industry.

1. Introducción

La sociedad de este siglo es software-dependiente, porque este desarrollo tecnológico tiene cada vez más un fuerte impacto en las operaciones vitales de la vida humana, tales como la medicina, la aeronáutica, la investigación espacial, las telecomunicaciones y la protección de datos, entre otras [1]. Por eso es imperativo abordar los problemas de calidad relacionados tanto con el proceso del desarrollo de software como con el producto mismo. Esta investigación se centra en el proceso y se orienta a analizar el nivel de madurez que ha alcanzado la automatización de las pruebas.

En un sentido amplio la prueba se aplica para cubrir todas las actividades relacionadas con la calidad del software, por lo que si se mejora el proceso y la aplicación de criterios de madurez se logrará un impacto positivo en la productividad de la Ingeniería de Software, y se reducirán los esfuerzos y el tiempo de producción [2].

Por otro lado, y debido a que los problemas son cada vez más complejos y a que los clientes son más exigentes sobre la calidad de los

productos, la prueba es una actividad esencial en el desarrollo de software. Es necesario probar para minimizar el riesgo de entregar sistemas con fallas y la automatización de las pruebas surgió como una alternativa para probar en menos tiempo un ámbito más amplio de sus funcionalidades. En términos generales consiste en utilizar software para ejecutar o apoyar las actividades de prueba, tales como la gestión, los casos de prueba, la ejecución y la evaluación de resultados. Sin embargo, la industria todavía la concibe como una actividad para incrementar la productividad del equipo a la vez que la calidad de los productos [3].

Otra cuestión relevante para incrementar la calidad en el desarrollo de software es utilizar modelos de madurez para apoyar el mejoramiento continuo de los procesos del ciclo de vida. Pero, aunque existen diversos modelos que cubren este ámbito de actividades, hay otros que se construyen específicamente para desarrollar la cultura de las pruebas y, por lo tanto, soportar su introducción de forma más disciplinada y programada.

Entre todas estas propuestas nadie puede discutir que la automatización de las pruebas hace parte del abanico para lograrlo. Pero a las organizaciones, que buscan introducirla en sus procesos, no les es fácil seleccionar un modelo que les ayude a comprender cuáles son las mejores prácticas de la automatización y cómo introducirlas en contexto [3]. Las organizaciones que deciden aplicar esta tecnología pueden lograr uno de dos objetivos: ahorrar tiempo y dinero en lo cotidiano de la prueba o no alcanzar este ahorro, pero sí mejorar la calidad del software más rápidamente que con las pruebas manuales [4].

Cuando la automatización alcance un adecuado nivel de madurez será posible aplicar las pruebas solamente con el toque de un botón, a la vez que seleccionar el mejor momento para hacerlo. Además, serán repetibles y utilizarán exactamente las mismas entradas en la misma secuencia de tiempo, algo que no se puede garantizar con las manuales. Pero, aunque pueda sonar sorprendente, probar es una habilidad, y para cualquier sistema existe un número astronómico de posibles casos de prueba, aunque solamente se tiene tiempo para correr un pequeño número de ellos. Sin embargo, se espera con ello se encuentre la mayoría de los defectos en el software, por lo que hay que ser hábil para seleccionar y generar los casos de prueba más importantes. En este sentido, la experiencia ha demostrado por décadas que la selección al azar no es un método eficaz, por lo que se requiere un enfoque más reflexivo para lograrlo [5].

En este trabajo se propone un modelo para determinar la madurez de la automatización de las pruebas del software, a partir de una revisión a los que se han publicado en la literatura. La propuesta se sustenta en el hecho de que los modelos analizados tienen objetivos diversos y no logran determinar el nivel alcanzado por la automatización como área de investigación y desarrollo. Esto es importante para determinar en qué nivel se encuentra en las empresas y colaborar para incrementarlo y mejorarlo para su beneficio. Se necesita conocer esa madurez, porque de nada le sirve a una organización esta tecnología si no tiene los argumentos suficientes para implementarla y obtener todos sus beneficios.

Este artículo se estructura de la siguiente forma: en la segunda parte se describe el estado del arte acerca de las pruebas y su automatización; en la tercera se analizan los trabajos relacionados con el tema tratado; en la cuarta se describe la metodología aplicada en la investigación; en la quinta se presentan los resultados; en la sexta se explica el modelo de madurez de la automatización de las pruebas propuesto, al tiempo que se aplica a los resultados para determinar el nivel de madurez de la automatización; en la parte siete se analizan los resultados de la investigación; y en la ocho se presentan las conclusiones.

2. Estado del arte

Probar es aplicar una serie de actividades con el objetivo de descubrir y/o evaluar las propiedades de cada elemento del software [6]. Estas actividades pueden incluir la planificación, la preparación, la ejecución, la presentación de informes y la gestión. Meyers [2] establece que las pruebas son el proceso de ejecución de un

programa con la intención de encontrarle errores, y Hass [7] opina que se puede considerar como una actividad de soporte, porque no tiene sentido sin los procesos de desarrollo y porque no produce nada en sí misma: si no hay nada desarrollado no hay nada que probar. Estas afirmaciones ofrecen una idea general de la definición de las pruebas de software y esencialmente conducen al objetivo general de las mismas: no se trata de encontrar todos los errores que pueda tener el sistema, sino de descubrir situaciones que podrían afectar negativamente su funcionamiento [8, 9].

Sin embargo, hay que tener en cuenta que el costo de encontrar y corregir errores puede elevarse considerablemente durante el ciclo de vida. Por lo tanto, cuanto antes se descubran los errores será mejor para controlar sus efectos, moderados o graves, en etapas posteriores.

La historia de las pruebas refleja la propia evolución del desarrollo de software que, por mucho tiempo, se focalizó en grandes programas científicos y militares y en sistemas de bases de datos, producidos en plataformas mainframes o mini-computadores. Los escenarios de prueba se escribían en papel y las pruebas se orientaban a seguir las trayectorias de los flujos de control, al cálculo de algoritmos complejos y a la manipulación de datos. Un conjunto finito de procedimientos de prueba podía probar con eficacia un sistema completo, y el proceso generalmente se iniciaba hasta el final de la programación y lo ejecutaba el personal que estuviese disponible en el momento.

Con el surgimiento del computador personal se iniciaron procesos de estandarización en toda la industria y en cómo desarrollar las aplicaciones software para operacionalizarlas bajo un sistema operativo común. La introducción de los PC dio origen a una nueva era y generó un crecimiento explosivo del desarrollo de software comercial, y las aplicaciones empezaron a competir ferozmente por la supremacía y la supervivencia. Los productos líderes empezaron a rivalizar con calidad y eficiencia para satisfacer a los usuarios y las metodologías de desarrollo evolucionaron aceleradamente.

El esfuerzo de la prueba en estas nuevas metodologías requirió un enfoque diferente para probar el diseño, porque los flujos de trabajo podían ser llamados en casi cualquier orden. Esta capacidad exigía un alto número de procedimientos para soportar un sinfín de permutaciones y combinaciones.

Más tarde, la popularidad de las aplicaciones cliente-servidor introdujo una nueva complejidad en el esfuerzo de la prueba, porque el probador ya no ejercitaba una única aplicación para un sistema individual cerrado. Esta arquitectura implicaba tres componentes separados: el servidor, el cliente y la red. Por otro lado, la conectividad inter-plataformas aumentó la posibilidad de aparición de fallas, y las pruebas se tuvieron que relacionar con el rendimiento del servidor y la red, así como con el rendimiento general y la funcionalidad del sistema a través de esos componentes. Con el uso generalizado de las aplicaciones GUI, la captura de pantallas y la reproducción de escenarios se convirtieron en una forma atractiva para probar aplicaciones. Entonces, se introdujeron herramientas automatizadas para hacerlo y poco a poco se popularizaron [2].

Aunque los escenarios de prueba y los *scripts* se seguían escribiendo en alguna aplicación de procesamiento de textos, el uso de estas herramientas se incrementó. Al ampliarse la complejidad y el esfuerzo de la prueba se requirió mayor planificación, por lo que el personal encargado de aplicarla fue obligado familiarizarse más con el sistema y a cumplir con requisitos de formación más específicos, relacionados con las plataformas y redes involucradas. Actualmente, estas herramientas han madurado y ampliado su capacidad, a la vez que aparecen otras con fortalezas y nichos específicos [8]. Además, la prueba automatizada se ha convertido cada vez más en un ejercicio de programación, aunque todavía involucra las funciones tradicionales de gestión, tales como la trazabilidad de requisitos, la planificación, el diseño y el desarrollo de escenarios y de *scripts*.

La base en todo este proceso son los casos de prueba, que se describen mediante atributos que determinan su calidad. Tal vez, el más importante sea su eficacia para detectar defectos, pero también que sea reutilizable, es decir, que se pueda modificar fácilmente para probar más de una cosa, lo que reduce el número de necesario [10]. Además, debe ser económico

para realizar, analizar y depurar errores, y ser evolutivo y emplear la cantidad mínima de esfuerzo en su aplicación. A menudo estos atributos se deben equilibrar uno con otro. Hoy se acepta que la habilidad para realizar pruebas no consiste solamente en asegurar que los casos de prueba encuentren muchos defectos, sino también en que estén bien diseñados para evitar costos y tiempos excesivos [11].

Una forma de alcanzarlo es a través de la automatización, considerada actualmente por los equipos de prueba como una opción importante [8]. Aunque algunas organizaciones han fracasado rotundamente en su esfuerzo por implementarla y han tenido que recurrir nuevamente a los procesos manuales, a otras les ha permitido producir mejor software e incrementar su calidad rápidamente. Para obtener beneficios con la automatización las pruebas deben ser cuidadosamente seleccionadas y aplicadas, porque la calidad de este proceso es independiente de la calidad de la prueba, y el hecho de que se aplique automática o manualmente no afecta ni su eficacia ni su evolución.

No importa lo inteligente que se planee la automatización o lo bien que se haga, si la prueba en sí no logra nada entonces el resultado final será una evidencia de que nada se logra al hacerlo de esa forma. Habitualmente una vez implementada es mucho más económica, porque el costo de funcionamiento es una fracción de los esfuerzos necesarios para hacerlo manualmente, sin embargo, generalmente cuesta más crearla y mantenerla. De ahí la importancia de seleccionar inteligentemente el momento para automatizar, porque será más barato ponerlo en práctica a largo plazo [4]. Además, hay que pensar en el mantenimiento, porque la sola actualización de un conjunto de pruebas automatizado puede tener un alto costo.

Para que la automatización de las pruebas sea eficaz y eficiente hay que comenzar con una buena materia prima, es decir, un buen banco de pruebas, un conjunto de pruebas hábilmente diseñado por un probador con las destrezas suficientes. Posteriormente, hay que tener la habilidad para automatizarlo de tal manera que se pueda crear y mantener a un costo razonable. En resumen, es posible que las pruebas sean de buena o de mala calidad, pero en todo caso será la habilidad del probador lo que determine la calidad total de la misma. También es posible que la automatización tenga buena o mala calidad, pero será la habilidad del automatizador lo que determine lo fácil que será agregar nuevas pruebas automatizadas, a la vez que mantenerlas y obtener los beneficios [12].

Es de amplio conocimiento que la automatización de las pruebas consiste en utilizar software para realizar o soportar todo tipo de actividades de prueba, tales como la gestión, el diseño, la ejecución y el análisis de resultados [13]. A menudo, las pruebas automatizadas se consideran como la realización de pruebas sobre secuencias de comandos en lugar de tener probadores que lo realicen manualmente [6]. Sin embargo, también es cierto que muchas tareas y actividades adicionales de prueba pueden ser apoyadas por herramientas basadas en software.

En general, la actividad de la automatización supone utilizar herramientas y, de acuerdo con Hass [7], el propósito es ejecutar el mayor número posible de actividades no-creativas, repetitivas y aburridas, además de explotar la ventaja de esas herramientas para almacenar y organizar grandes cantidades de datos. En términos generales la automatización puede ayudar a resolver: 1) el trabajo repetitivo, 2) la lentitud de las pruebas manuales, y 3) la inseguridad de las pruebas manuales. Por otro lado, la introducción de la automatización debe aumentar la productividad, porque de otra manera no compensaría el costo [7].

3. Trabajos relacionados

Diversos investigadores han presentado revisiones acerca de la automatización de las pruebas del software con diversos resultados, aunque no tienen en cuenta el análisis a la madurez del proceso como área de investigación y desarrollo. A continuación, se relacionan alguno de ellos. El objetivo de Michael Grottke [14] fue desarrollar un nuevo y ampliado modelo estadístico para predecir la fiabilidad de los programas software con base en la madurez de las pruebas. El modelo se implementa mediante un prototipo que les ayuda a las pequeñas

empresas de software a predecir la confiabilidad de sus desarrollos y a probarlos de manera fácil y eficiente, con una precisión superior a la disponible. Aunque no tuvo la aceptación suficiente se rescata la amplia revisión y comparación que realiza a los modelos previos. Analiza la automatización de las pruebas desde una perspectiva estadística, pero no se orienta a encontrar la madurez del proceso como tal, sino a analizar los modelos propuestos hasta el momento.

La idea de la investigación de Ron Swinkels [15] es averiguar lo que se puede aprender de otros modelos de mejoramiento de los procesos de prueba. Para materializarla presenta una comparación entre Test Maturity Model (TMM) y siete modelos de mejoramiento con el objetivo de extraer prácticas importantes para desarrollar otro modelo. El resultado es una descripción y comparación desde los objetivos, estructuras, áreas clave del proceso y procedimientos de evaluación de los modelos. Los resultados sirvieron para proponer un modelo propio, orientado básicamente al mejoramiento de los procesos de prueba. Aunque presenta una matriz de comparación el modelo base que utiliza no es el referente más importante para la industria, debido a que su objetivo no fue encontrar el nivel de madurez de la automatización, ni siquiera de los modelos existentes.

Shrini Kulkarni [16] da cuenta de la histórica de los modelos de madurez para las pruebas del software y su relevancia en el estado actual y futuro de la industria. También presenta algunas ideas iniciales y pensamientos en torno a un nuevo marco que refleje el estado actual de la prueba, en el que destaca la habilidad del probador y la importancia del pensamiento cognitivo y la inteligencia humana, lo que considera como una desviación de los modelos anteriores que ignoran el aspecto humano de las pruebas. El autor concluye que hay necesidad de modificar la mirada a los modelos y hacerlos más pertinentes a como está progresando el mundo en este aspecto.

Los modelos de madurez del futuro tienen que reconocer el aspecto humano de la prueba y resistir la tentación de elaborar diagramas de flujo para las actividades, reconociendo la importancia del pensamiento crítico. El autor hace un análisis comparativo a cuatro modelos de madurez, pero siempre orientado a los aspectos humanos de la prueba, no a conocer o analizar el nivel de madurez de la automatización en general.

Gustavo De Souza [17] afirma que las mejores prácticas en las pruebas de software contribuyen a mejorar la calidad y reducir el costo de los productos, mediante la reducción de los tiempos en las etapas de prueba y durante la implementación y el mantenimiento. Los modelos de madurez para el desarrollo de software se han utilizado a gran escala para aliviar estos problemas, pero todavía no tienen cuenta las actividades relacionadas con las pruebas, por lo que se hizo necesario desarrollar modelos de madurez.

El autor hace una comparación y valida la eficiencia de los modelos Test Improvement Model (TIM), Test Process Improvement (TPI) y TMM e intenta determinar su nivel de madurez. El objetivo fue encontrar una guía para ayudarles a las empresas a mejorar la calidad de sus productos. La matriz de comparación utilizada es amplia y los indicadores ajustados, pero el hecho de que solamente trabaje con tres modelos no le permite presentar una imagen amplia del nivel de madurez de los relacionados con las pruebas del software. Además, su objetivo es demostrar el nivel de madurez de la organización con respecto a la automatización, no de la automatización como área de investigación.

Muhammad Sulayman [18] presenta una revisión sistemática de la literatura para identificar y analizar los modelos y técnicas existentes que utilizan las PyMes Web. Después de aplicar los filtros establecidos seleccionó un total de 88 estudios, pero sorprendentemente una inspección más detallada reveló que solamente cuatro de ellos eran relevantes para el tema. El objetivo principal fue investigar modelos o técnicas específicas para el mejoramiento de los procesos software, pero no encontró ninguno específico a la medida para las PyMes Web. Aunque las métricas analizadas incluyen a los equipos de desarrollo y la satisfacción de los clientes, el aumento de la productividad, el cumplimiento de los estándares y la excelencia operativa general, no identifica modelos de automatización para las pruebas, en parte por el tamaño de las empresas analizadas y porque sus

limitaciones presupuestales no les permiten adentrarse en este campo. En resumen, esta investigación no pudo determinar el nivel de madurez de la automatización en las PyMes Web.

Para Christiane von Wangenheim et al. [19] el nivel de madurez de la evaluación y el mejoramiento del software, guiado por procesos basados en un modelo de capacidad/madurez, se ha consolidado en la práctica como un medio eficaz para mejorar los procedimientos de desarrollo en las organizaciones. Describe los resultados de una revisión sistemática de la literatura sobre los modelos que en la última década han evolucionado el desarrollado y la adaptación. Los resultados demuestran que existe gran variedad de modelos con tendencia a la especialización para dominios específicos, pero que la mayoría se concentra en el marco CMM/CMMI y la norma ISO/IEC 15504, con los inconvenientes subsecuentes.

Aunque los autores analizan el nivel de madurez su objetivo es mostrar una matriz comparativa entre los 29 modelos evaluados, por lo que no presentan una evaluación a la madurez de la automatización de las pruebas como área de investigación y desarrollo.

El estudio de Heiskanen, Maunumaa y Katara [3] muestra que la introducción de esta tecnología no siempre se realiza con éxito, en parte porque los procesos de prueba y de la organización no siempre se ajustan adecuadamente. En este trabajo se presenta un modelo de generación de pruebas automatizadas sobre el modelo TPI y traza un perfil de base para la introducción exitosa de la tecnología. Su objetivo se orienta a compaginar los procesos de las pruebas y los de organización y hace una comparación de algunos modelos de madurez, pero no profundiza más allá de seleccionar la estructura para comparar procedimientos, es decir, no presenta un acercamiento al nivel de madurez de los modelos analizados ni tiene en cuenta la automatización como área a la que se puede evaluar su madurez.

García, Dávila y Pessoa [20] afirman que la calidad de los productos software está fuertemente influenciada por la calidad del proceso que los genera, y que particularmente la prueba contribuye en mayor medida. En este contexto su estudio pretende encontrar qué modelos de procesos de prueba han sido definidos, adaptados o extendidos en la industria. Identificaron 23 modelos, muchos de ellos adaptados o extendidos desde Test Maturity Model Integration (TMMi) y TPI con diferentes arquitecturas, y desde la norma ISO/IEC/IEEE 29119 como enfoque arquitectónico alineado con otros modelos.

Debido a que las métricas e indicadores utilizados por estos investigadores se orientaron a determinar para cada modelo el grado de adopción, la arquitectura, el domino, las fuentes utilizadas, las tendencias y la información mínima necesaria para comprenderlo, los resultados no determinan el nivel de madurez de la automatización de las pruebas en la industria y la investigación.

En el trabajo de Furtado, Meira y Gomes [4] se lee que la práctica de las pruebas de software es una de las maneras de producir productos de calidad, y que la automatización puede ser vista como una solución para probar la mayor cantidad de software en un tiempo determinado. Por eso su objetivo fue proponer directrices utilizando un modelo de madurez para la automatización con el fin de ayudarlas a entrar gradualmente en esta práctica.

Pero también afirman que la automatización puede no ser la solución para las necesidades de todas las empresas, por lo que su introducción puede complicar más de lo necesario el proceso de pruebas. Aunque esta investigación presenta una revisión de la literatura acerca de la automatización de las pruebas, su objetivo no es el de mostrar su nivel de madurez sino justificar la propuesta de otro modelo para implementarla. Es decir, no analizan la madurez de la automatización como un tema de investigación y desarrollo de la industria del software.

4. Metodología

Se realizó una revisión sistemática de la literatura para encontrar el nivel de madurez actual de la automatización de las pruebas del software, siguiendo los procedimientos descritos por Serna M. [21]. La pregunta de investigación a responder fue: ¿Cuál es el nivel de madurez actual de la automatización de las pruebas como

área de investigación y de desarrollo en la industria del software?

Para responderla se analizaron los trabajos que cumplieran con: cubrir expresamente a la automatización como área de investigación y desarrollo; haber sido publicados entre 1990 y 2014; describir modelos de madurez; presentar análisis comparativos a la eficiencia, la eficacia, el nivel de aceptación y la proyección; y realizar observaciones a la madurez de la automatización, los modelos o los procesos de prueba del software. Los trabajos seleccionados se limitaron a artículos publicados en revistas o actas de congresos y reportes técnicos.

Se excluyó cualquier publicación que no describiera explícitamente un modelo de madurez relacionado con la automatización de las pruebas, no hiciera referencia a una matriz de comparación, no analizara el nivel de madurez del proceso de automatización, o no lo concibiera desde las perspectivas de investigación y desarrollo. La calidad de los aportes se validó con los procedimientos establecidos por el medio de publicación, es decir, la revisión por pares de las revistas y congresos como criterio principal.

Se utilizaron las bases de datos IEEEXplore, la biblioteca digital ACM, Springer, Web of Science, ScienceDirect y Wiley Interscience. Las razones para seleccionar están bases de datos es que concentran la mayor cantidad de revistas relacionados con la automatización de las pruebas del software, y a que recopilan las memorias de conferencias, simposios y publicaciones de organizaciones, normas y libros. La información es actualizada diariamente y permiten realizar búsquedas por múltiples opciones. Los términos se buscaron en español e inglés.

5. Resultados

La búsqueda se realizó entre enero y abril de 2015 y en total se recuperaron 978 trabajos. En una primera etapa se revisaron títulos y resúmenes y se descartaron los irrelevantes y/o duplicados; posteriormente se analizaron los contenidos y no se tuvieron en cuenta aquellos que no describían completamente un modelo o no analizaban la madurez de la automatización como área de investigación y desarrollo.

Este proceso arrojó 16 modelos, que se describen en la Tabla 1, y 10 reportes de análisis a la eficiencia y eficacia de los modelos y a la madurez de la automatización. Con el fin de organizarlos se clasificaron por año de publicación, para determinar la derivación subsecuente y para identificar los modelos originales. Es de anotar que, de una u otra forma, todos los trabajos analizados tienen en cuenta la automatización, aunque algunos presentan una orientación más marcada que otros.

En la Tabla 2 se aprecian los resultados del análisis a las opiniones publicadas acerca de la eficiencia y eficacia de los modelos de madurez de la automatización de las pruebas del software. Esta valoración de los modelos se resume de los trabajos recolectados en los que: 1) se describe el modelo o se analiza su aplicación en casos de la industria [10, 12, 15, 39-41]; 2) el nivel de aceptación es la ponderación a la valoración que los autores hacen de cada uno [16, 41, 43, 44]; y 3) la proyección demuestra si el modelo tiene una vida útil referente o si fue efímero su paso por la industria [10, 16, 39, 41, 43].

6. Modelo de madurez propuesto

Conocer todo el potencial de la automatización de las pruebas del software será posible en la medida en que las organizaciones aprovechen sus beneficios, y esto se logra ubicando el proceso mismo dentro de un nivel de madurez. Cuanto más maduro sea el proceso de la automatización mayor será la eficiencia y la eficacia del plan de pruebas. Aunque la mayoría de investigadores y de personas en todo el mundo están familiarizados con los niveles de madurez utilizados por Capability Maturity Model (CMM), la razón de proponer un modelo con otra escala de valoración es que CMM se centra en evaluar la evolución de los procesos de una organización, y en este trabajo los autores quisieron acercarse al estado actual de la automatización como área de investigación y desarrollo. Es decir, la idea no es presentar la

Tabla 1. Modelos de madurez para la automatización de las pruebas del software

Modelo	Año	Orientación	Niveles	Escuela de pruebas [17]
MMAST [22]	1994	Automatización	4	Factory school
TAP [5, 23]	1995	Evaluación	5	Test-driven school
TCMM [24]	1996	Capacidad	4	Quality School
TSM [25]	1996	Capacidad	3	Quality School
TMM [26, 27]	1996	Procesos	5	Context-drive school
TIM [28]	1998	Mejoramiento	5	Standard School
TOM [29]	1998	Mejoramiento	3	Standard School
TPI [30]	1999	Mejoramiento	3	Standard School
ATLM [31]	1999	Automatización	5	Factory school
MB-V^2M^2 [32]	2002	Verificación y Validación	5	Control school
CB-VVCM [33]	2005	Verificación y Validación	5	Control school
SAMM [34]	2009	Aseguramiento	4	Context-drive school
CMMI-DEV [35]	2010	Calidad	4	Quality School
TMMi [36]	2012	Actividades de prueba y desarrollo	5	Analytical school
MPT.BR [37]	2012	Mejores prácticas	5	Agile School
ISO/IEC/IEEE [38]	2013	Estándares	6	Control school

Tabla 2. Eficacia y eficiencia de la automatización de las pruebas

Modelo	Eficacia y eficiencia	Nivel de aceptación	Proyección
MMAST	Baja	Bajo	Ninguna
TAP	Baja	Medio	Poca
TCMM	Baja	Bajo	Ninguna
TSM	Baja	Bajo	Ninguna
TMM	Baja	Bajo	Ninguna
TIM	Baja	Bajo	Ninguna
TOM	Baja	Bajo	Ninguna
TPI	Media	Alta	Alta
ATLM	Baja	Bajo	Poca
MB-V^2M^2	Media	Bajo	Poca
CB-VVCM	Baja	Bajo	Poca
SAMM	Media	Medio	Alta
CMMI-DEV	Alta	Alto	Alto
TMMi	Alta	Alto	Alta
MPT.BR	Alta (Brasil)	Alto (Brasil)	Alta (Brasil)
ISO/IEC/IEEE	Media	Medio	Alta

evolución sino el estado en el que se encuentra al momento de la revisión.

El objetivo de este modelo es determinar la madurez de la automatización de las pruebas del software analizándola desde una perspectiva y con niveles comparativos a la madurez de los seres humanos, es decir: 0. Infantil, 1. Adolescente, 2. Adulto y 3. Veterano.

Nivel 0: Infantil

El proceso de automatización de las pruebas está en un nivel de madurez *Infantil* cuando necesita mucho cuidado, atención y afecto. Las características de este nivel son:

- La mayoría de las pruebas se ejecutan solamente al finalizar el desarrollo del producto, porque el plan de pruebas automatizado no está en sintonía con el ciclo de vida del desarrollo.
- La cantidad de errores detectados hace que la prueba falle y que probablemente se detenga la secuencia de comandos automatizados, porque las incoherencias entre el software bajo prueba y el marco de automatización invalida cualquier caso de prueba que se intente aplicar.
- Se generan procesos de reingeniería porque en cualquier caso hay que corregir el error o actualizar el caso de prueba y luego volver a ejecutar el plan de pruebas para encontrar el siguiente problema.
- El equipo de prueba invierte más tiempo trabajando en los casos de prueba que en su automatización.
- La mayoría de organizaciones que intentan introducir la automatización de las pruebas no tardan en retroceder al proceso manual. Lamentablemente se dan cuenta tarde de la madurez de la automatización, porque normalmente les demora entre dos y diez veces más tiempo que el proceso manual.
- Debido a que hay que cuidar, mimar y prestarle mucha atención, la automatización no se puede dejar sin vigilancia por mucho tiempo. Aquí es posible afirmar que la madurez de esta tecnología desalienta en lugar de alentar su introducción.

Nivel 1: Adolescente

El proceso de automatización se encuentra en el nivel *Adolescente* cuando es posible aplicar el plan de pruebas y el conjunto de casos de prueba y dejarlo solo y sin vigilancia durante un tiempo razonable, tal vez un par de horas o incluso una noche, pero todavía se desconfía de su responsabilidad. Las características de este nivel son:

- Un solo error en el software bajo prueba hace que falle gran cantidad de casos de prueba.
- Es importante conocer el error, pero no se necesitarán decenas de casos de prueba para demostrarlo.
- Se desperdicia mucho tiempo intentando analizar la causa de cada bloqueo a la vez que se deja de ejecutar muchas pruebas.
- Para encontrar las fallas el equipo debe solucionar los problemas y volver a correr la automatización, un ciclo que se tiene que repetir muchas veces.
- En este nivel y al igual que con los adolescentes, el proceso de automatización es sorprendentemente útil, pero se comporta de manera irresponsable y puede causar más daño que beneficio.

Nivel 2: Adulto

El proceso de pruebas automatizadas se encuentra en nivel *Adulto* cuando es digno de confianza y se puede dejar solo para que funcione sin vigilancia durante un largo tiempo, por ejemplo, durante todo un fin de semana, pero si todavía no es auto-dirigido, porque su madurez no ha llegado a ese nivel de independencia.

- Al final el proceso de prueba arroja mucha información útil.
- Aunque quizás la mayoría de casos de prueba ha fallado, los datos son diferentes y sobre todo reportan algo del software bajo prueba.
- La automatización es algo más que secuencias de comandos.

Tabla 3. Madurez del desarrollo de la automatización de las pruebas

Etapas del proceso	Nivel de madurez			
	Infantil	**Adolescente**	**Adulto**	**Veterano**
Percepción			x	
Sensibilización			x	
Decisión		X		
Implementación		X		
Apropiación	x			
Consolidación	x			
Institucionalización	x			
Externalización	x			
Nivel de madurez actual de la automatización de las pruebas		**X**		

- El equipo se concentra en encontrar y corregir los problemas reportados mientras los casos de prueba se continúan ejecutado sin intervención.
- En este nivel la automatización no requiere vigilancia extrema y aunque el proceso es responsable y se puede confiar en él todavía no es auto-dirigido, porque su madurez no ha llegado a ese nivel de independencia.

Nivel 3: Veterano

Para llegar al nivel de madurez *Veterano* la automatización de las pruebas del software tiene que haber recorrido y crecido a través de los anteriores, y en este momento es posible dejarlo para que la naturaleza siga su curso. En este nivel la automatización es totalmente gestionable, las herramientas disponibles son autónomas, los casos de prueba son repetibles y el proceso se puede dejar funcionando sin vigilancia por semanas enteras. Las características de este nivel son:

- El plan de pruebas y los casos de prueba son eficientes y eficaces y el equipo observa todo el proceso de automatización como una disciplina no como un arte.
- El proceso de la automatización utiliza la reutilización como base porque ahora es bien entendido y aplicado.
- Al final se tiene un conjunto de casos de prueba validado y maduro, y una serie de reportes que denotan la calidad y fiabilidad del producto y de la automatización de las pruebas.
- El plan de pruebas se estructura como un enfoque planificado que involucra el diseño de los casos de pruebas y el mantenimiento del plan de pruebas.
- Es posible aplicar procedimientos de gestión y de planeación estratégica a todos los aspectos de la automatización.
- La automatización de las pruebas es un proceso autónomo en cuanto a reproducción y selección de los valores de entrada y a la validación y exposición de resultados.
- La organización cuenta con un banco de pruebas automatizadas reutilizable y fácil de proyectar a cada producto bajo prueba.
- Como en el caso de los humanos, en este nivel la automatización aporta experiencia y madurez para ponerlas al servicio de los demás procesos del software.

6.1. Madurez de la automatización de las pruebas del software

Con base en los resultados presentados en las Tablas 1 y 2 y a las opiniones, reflexiones y críticas que la industria y los investigadores manifiestan acerca de los modelos de madurez y de la automatización misma, en la Tabla 3 se resumen los resultados acerca de la madurez de este proceso. En la primera columna se ubican las etapas tradicionales de un proceso de pruebas de software y en las demás los niveles del modelo de madurez de la automatización de las pruebas del software propuesto en esta investigación.

7. Análisis de resultados

Es importante observar cómo en la última década se ha incrementado el interés de los investigadores y la industria por proponer modelos para automatizar el proceso de las pruebas del software. En parte motivados por apoyar el mejoramiento de la calidad y la escalabilidad de sus productos. Pero también llama la atención que, aunque se presentan diversos modelos de madurez a la industria parece que le falta algo en este tema, y es que no los aplican rigurosamente. Una de las principales cosas que se aprende con estos modelos es que cuando se avanza a lo largo de ellos es muy importante no saltarse los niveles. Ese es el punto para que sea un modelo de madurez. Como se propone en este trabajo las personas crecen a través de cada etapa y a partir de lo que ya son en cada una. No es posible dejar de ser niño para pasar automáticamente a ser adulto, aunque se intente desesperadamente, porque si se intenta saltar adolescencia está destinado al fracaso.

Al analizar la aplicación de un modelo de automatización de pruebas y constatar que la empresa se encuentra en un nivel alto, se podría argumentar que simplemente es un caso en el que pasó de la implementación manual y aprendió en los demás niveles hasta lograr el despliegue total de la automatización. Pero la realidad en los resultados de esta investigación es que pocas empresas tienen la paciencia y los recursos para atender el proceso completo a través de todos los niveles, hasta lograr una automatización madura.

La mayoría que califica los modelos como deficientes o ineficaces es porque han intentado saltarse niveles para avanzar. Por otro lado, la industria del software ha tratado y ha invertido en mejorar la calidad de sus productos por años, pero ha sido una tarea difícil porque los clientes se han vuelto cada vez más exigentes y los sistemas software cada vez más complejos. El aseguramiento de la calidad se está convirtiendo en una condición permanente para la sobrevivencia de las empresas y actualmente es una realidad en la industria del software. A pesar de que algunos investigadores [48-50] afirman que la calidad del software ha mejorado en los últimos años, todavía está muy lejos del escenario ideal y de la madurez esperada. Para llenar este vacío la industria ha tratado de adaptar diferentes modelos de madurez a sus procesos del ciclo de vida, pero de acuerdo con los reportes analizados en esta investigación esos modelos describen prácticas de Verificación y Validación limitadas que no se centran directamente en la madurez del proceso de automatización de las pruebas. Esta madurez se puede definir como una forma de medir el nivel de capacidad que tiene una organización para gestionar los planes de pruebas de los proyectos [48], y el principal objetivo de conocerla es ayudarle a mejorar la construcción del software.

Aunque la prueba es un elemento esencial para lograr la satisfacción del cliente y una parte integral de todo el ciclo de vida del desarrollo de software, requiere velocidad, eficiencia y flexibilidad. Por eso es que el papel de las pruebas automatizadas es el de apoyarlo para eliminar tareas mecánicas, rutinarias y lentas. Debido a esta exigencia en velocidad, la gestión de los datos de prueba y de los diferentes entornos de plan de pruebas necesita ser muy eficiente y eficaces. Esta característica incrementa continuamente la demanda por una automatización madura que asegure que el proceso de pruebas ocurre sobre una base muy regular [49]. Además, que facilite la construcción de escenarios cada vez más predecibles, que requieran menos esfuerzo y que les permita a los equipos de desarrollo y de prueba obtener

información instantánea sobre la calidad del sistema en producción. Estas características no se logran de acuerdo con los resultados analizados en esta investigación.

La prueba automatizada es una estrategia fiable y para muchas empresas es la única opción para optimizar los estándares de calidad del software [50], pero de acuerdo con los resultados de esta investigación todavía se encuentra dentro de los tiempos de maduración que requiere cualquier aplicación compleja en el mundo actual.

Por otro lado, en la práctica los diferentes segmentos de aplicación de la automatización se encuentran en diferentes estados de madurez (ver Tabla 3), porque en cada etapa aparece un aprendizaje único que brinda la oportunidad de tener el poder para pasar a la etapa superior siguiente.

Pero un problema detectado es que actualmente en la industria la automatización de las pruebas se gestiona como otro proceso del desarrollo de software, y las herramientas disponibles para aplicarla se ofrecen de acuerdo con la demanda del mercado, no por una planeación real de las necesidades de la industria de un sistema escalable y mantenible para lograrlo. Esto genera algunos desafíos que es necesario afrontar para lograr que la automatización de las pruebas logre realmente su objetivo de alcanzar el nivel de madurez *Veterano*:

- El tiempo que se invierte para automatizar es muy extenso, porque la industria no tiene un adecuado nivel de madurez de sus procesos de desarrollo.
- El mantenimiento del plan de pruebas y del conjunto de casos de prueba es muy frecuente y no se adapta fácilmente a los escenarios cambiantes de los sistemas de hoy.
- Para las miles de líneas de código de la automatización no hay documentación y la que existe no es adecuada o está desactualizada.
- La rotación de recursos crea caos que cambia el enfoque completo del plan de pruebas, por lo que su automatización no está lista cuando se necesita.
- La industria actual invierte dinero, tiempo y esfuerzo solamente para hacer su trabajo, es decir, desarrollar, pero al automatizar las pruebas se encuentra con la frustración de no obtener rápidamente resultados.
- Las habilidades para automatizar las pruebas no son fáciles de adquirir, porque cada herramienta tiene un enfoque único y patentado para interactuar con las tecnologías de desarrollo. Por lo que la industria necesita capacitar diferentes equipos de prueba para atender las demandas de cada sistema que desarrolla, y debe utilizar diferentes herramientas debido a los diferentes requerimientos de los clientes en cuanto a lenguajes de programación.
- La industria tiene el problema de que toda la vida ha aplicado pruebas manuales y ha invertido bastante en capacitar a sus equipos de probadores. Pero resulta que ellos no saben automatizar el plan de pruebas simplemente porque no saben de programación.
- El otro desafío importante para que la automatización de las pruebas madure como área de investigación y de desarrollo es el costo de las herramientas y de la adquisición de las habilidades especiales para aplicarlas. Gran parte de la industria opta por la prueba manual porque a veces es más costosa su automatización que el mismo desarrollo del sistema.

De acuerdo con los resultados de esta investigación muchas empresas dedicadas al desarrollo de software ven y aplican la automatización de las pruebas como una especie de *bala de plata* que resolverá todos sus problemas de calidad, les ayudará a cumplir con todos los requisitos de los sistemas y les ahorrará mucho tiempo y esfuerzo. Este no es el caso, porque esta implementación implica una curva de aprendizaje progresivo hasta alcanzar la madurez adecuada. Eso significa que en realidad durante los primeros niveles las pruebas automatizadas pueden aumentar el esfuerzo y el costo de aplicación.

Como sucede con la generación del plan de pruebas, un área clave donde las expectativas de la automatización parecen estancadas en el tiempo porque actualmente se debe hacer de forma manual, aunque ya haya en el mercado herramientas que la apoyan, pero ninguna es automática. También es importante destacar que todavía no existe una herramienta que apoye todos los entornos de sistemas operativos y lenguajes de programación utilizados en una organización. Esto significa que la mayoría requiere un conjunto de diversas herramientas para automatizar sus pruebas, porque no hay tal cosa como una de *talla única*.

Por otra parte, las empresas deciden enfrascarse en la automatización porque piensan que la reducción de costos y de tiempos de entrega les permitirá cumplir con los plazos. Pero actualmente es poco probable que la automatización reduzca inmediatamente el esfuerzo de la prueba y ahorre el tiempo necesario, debido a que se requiere capacitación del personal para utilizar las herramientas y para aprender las diversas maneras eficaces de hacerlo. Además, la escritura de *scripts* de prueba aporta un nuevo nivel de complejidad al plan de pruebas, lo que requiere que los probadores y las empresas piensen en términos de fiabilidad y reutilización en el diseño, en lugar de simplemente en la eficacia de la prueba.

Otro asunto que la industria no comprende a cabalidad es que actualmente no todo el plan de pruebas se puede automatizar. Muchos sistemas contienen controles de terceros o *widgets* para mejorar su funcionalidad, lo que representa un problema porque es poco probable que una herramienta de automatización verifique su compatibilidad con todos estos controles, y puede que no sea capaz de manipular los caminos necesarios para probar la aplicación. Y pruebas como la comprobación de que un documento se imprime correctamente no se pueden automatizar, o tampoco es rentable para aquellos casos de prueba que solamente se ejecutan una vez.

Algo un poco más alejado de la realidad es que a menudo se espera que la automatización permitirá realizar pruebas al cien por ciento de la aplicación. La industria debe comprender que la prueba es una tarea potencialmente infinita, sin embargo, si se centra en las áreas clave del código puede mejorar considerablemente la fiabilidad del software.

En términos generales hasta el momento la automatización de las pruebas del software solamente permite: 1) incrementar la fiabilidad del sistema, 2) mejorar la calidad de las pruebas, 3) disminuir el esfuerzo que se dedica a las pruebas, y 4) reducir el cronograma del proyecto. Por todo esto es posible concluir que actualmente el nivel de madurez de la automatización de las pruebas del software como área de investigación y desarrollo es Adolescente.

8. Conclusiones

Una perspectiva que se puede aplicar en esta investigación es que en lo referente a la madurez de la automatización de las pruebas del software el vaso no está vacío, pero tampoco está lleno, solamente esta medio vacío. Esta apreciación se sustenta en los resultados analizados y a que se percibe una conciencia cada vez mayor, de quienes tienen experiencia en este campo, de que muchos esfuerzos en la automatización de las pruebas no cumplen las expectativas. Para llegar a esta conclusión la presente investigación encontró que, aunque existe gran cantidad de esfuerzo orientado al desarrollo y el mantenimiento de la automatización, es muy importante realizar un análisis costo-beneficio a cualquier intento por implementarla.

Esto significa analizar los resultados y las experiencias en diferentes empresas y métodos, porque los éxitos reportados en la literatura han sido en su mayoría en áreas en las que tenía sentido automatizar algunas pruebas en lugar de todo el plan. Además, al equipo de pruebas lo asesoraban especialistas y se les permitió el tiempo para hacerlo bien. Esto no es un denominador común a toda la industria y para todos los sistemas actualmente.

Aunque la automatización puede añadir complejidad y costos al esfuerzo de un equipo de pruebas, también puede proporcionar cierta ayuda valiosa si se cuenta con las personas adecuadas, con el entorno adecuado y si se aplica cuando tiene sentido hacerlo [11]. Luego

de realizar y presentar los resultados de esta investigación podemos concluir:

- Es importante definir el propósito de iniciar un esfuerzo de automatización de las pruebas porque, aunque existen varias categorías de herramientas para hacerlo cada una tiene su propio propósito.
- Identificar qué se desea automatizar y en qué fase del ciclo de vida implementarlo es el primer paso para desarrollar una estrategia de automatización. Solamente desear que todo sea probado más rápido no es una estrategia práctica, hay que ser específico.
- Desarrollar una estrategia de automatización es más importante que decidir qué se va a automatizar, cómo se va a hacer, cómo se mantendrán los *scripts* y cuáles serán los costos y los beneficios que se espera. Al igual que con todos los esfuerzos de prueba se debe construir una estrategia o plan de pruebas para implementarla.
- Muchas herramientas de prueba son sofisticadas y utilizan lenguajes existentes o código propietario, por lo que el esfuerzo de la automatización se convierte en una rutina manual que no es diferente del trabajo de un programador, codificando en un determinado lenguaje para escribir programas para automatizar un proceso de prueba. Hay que tratar a todo el proceso de la automatización como si fuera un esfuerzo de Ingeniería de Software, es decir, definir lo que se automatizará (requisitos); diseñar la automatización; escribir, probar e implementar los *scripts*; hacerle mantenimiento y definir su vida útil.
- Para alcanzar sus beneficios hay que observar al esfuerzo de la automatización como una inversión que requiere tiempo y recursos. Esto implica que por lo general las primeras versiones del sistema no ofrecen la recompensa esperada, y que el beneficio viene luego de correr las pruebas automatizadas en cada lanzamiento posterior. De ahí la importancia de poder ubicar rápidamente la automatización de las pruebas del software en alguno de los niveles de madurez propuestos en esta investigación.
- Debido a que la automatización realmente es otro esfuerzo de desarrollo de software, es importante que quienes realizan el trabajo posean las habilidades y destrezas necesarias. Un buen probador no significa necesariamente un buen automatizador. Los buenos probadores todavía serán necesarios para identificar y escribir casos de prueba, pero se necesita al automatizador para que tome esos casos de prueba y escriba código para su automatización. Esto no quiere decir que los probadores no puedan aprender a ser automatizadores, es sólo que esos dos roles son diferentes y las habilidades necesarias también son diferentes.

Los modelos analizados en este trabajo han sufrido cambios en el tiempo, proporcionándole a la industria nuevas versiones, en periodos cada vez más cortos. Esto dificulta su apropiación y experimentación, a la vez que la pérdida de respaldo por parte de investigadores y practicantes.

Al evaluar el proceso de automatización de las pruebas de software a partir de las etapas del modelo propuesto, y debido a que las etapas de apropiación, consolidación, institucionalización y externalización se encuentran en nivel infantil, mientras que las etapas de decisión e implementación en nivel adolecente y las de percepción y sensibilidad en nivel adulto, los autores concluyen que la madurez de la automatización, como área de investigación y desarrollo, se encuentra en un nivel Adolecente.

De acuerdo con los análisis a los resultados publicados por los investigadores y la industria, y a la aplicación del modelo de madurez propuesto, la automatización de las pruebas del software se encuentra actualmente en desarrollo y en proceso de maduración, por eso no es de esperar que en corto tiempo cumpla las promesas que viene haciendo desde hace más de dos décadas.

El trabajo para alcanzarlas debe continuar y cada vez se deberán sumar más investigadores y empresas para lograrlo. El futuro es prometedor, pero intentar proyectar lo que logrará más adelante es una lotería, y ese objetivo está por fuera de esta investigación. Lo que sí se puede

afirmar, con base en los resultados encontrados y descritos aquí, es que actualmente la madurez de la automatización de las pruebas del software está en un amplio proceso de aprendizaje y de experimentación, y que como sucede con los humanos adolescentes: se puede confiar en ella, pero no dejarla sola mucho tiempo.

Referencias

1. **Serna, M.E. (2012)**. Social control for science and technology. *Tenth Latin American and Caribbean Conference for Engineering and Technology*, pp. 1–8.
2. **Myers, J. (1979).** *The art of software testing*. John Wiley & Sons, Inc., New York.
3. **Heiskanen, H., Maunumaa, M., & Katara, M. (2012)**. A Test Process Improvement Model for Automated Test Generation. *PROFES 2012, LNCS 7343*, pp. 17–31. DOI: 10.1007/978-3-642-31063-8_3.
4. **Furtado, A., Meira, S., & Gomes, M. (2014)**. Towards a maturity model in software testing automation. *The Ninth International Conference on Software Engineering Advances*, pp. 282–285. New York, USA.
5. **Paulk, M., Weber, C., Garcia, S., Chrissis, M., & Bush, M. (1993).** *Key practices of the capability maturity model*. Technical report CMU/SEI-93-TR-25, Carnegie Mellon University.
6. **ISO/IEEE (2013).** *Part I International Standard, Software and systems engineering/software testing, concepts and definitions.* ISO/IEEE 29119, USA.
7. **Hass, A. (2008).** *A guide to advanced software testing*. Boston: Artech House.
8. **Serna, M.E. (2013).** *Functional test of software - A Constant Verification process*. Medellín: Fondo Editorial ITM.
9. **Serna, M.E. & Arango, F. (2010).** Effectiveness analysis of the set of test cases generated with the Requirements by Contracts technique. *V Congreso Colombiano de Computación*, Cartagena, Colombia. pp. 1–6.
10. **Karthikeya, S. & Rao, S. (2014).** Adopting the right software test maturity assessment model. *Cognizant,* Vol. 20, No. 20, Insights, New Jersey, USA.
11. **Anya, P. & Smith, G. (2014).** Qualitative research methods in Software Engineering. *Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS)*, Vol. 4, No. 2, pp. 1–18.
12. **Kumar, P. (2012).** *Test process improvement – Evaluation of available models*. *Maveric's Point of View*, pp. 8.
13. **ISTQB (2012).** *Standard glossary of terms used in software testing.* International Software Testing Qualifications Board. ISTQB: Munich.
14. **Grottke, M. (1999).** *Software Process Maturity Model study*. IST-1999-55017, PETS Project.
15. **Swinkels, R. (2000).** *A comparison of TMM and other Test Process Improvement Models*. 12-4-1-FP Report. Frits Philips Institute, Technische Universiteit Eindhoven. Eindhoven, Netherlands.
16. **Kulkarni, S. (2006).** Test process maturity models - Yesterday, today and tomorrow. *Proceedings 6th Annual International Software Testing Conference,* pp. 1–15, Delhi, India.
17. **De Souza, G. (2007).** *Modelo de maturidade em testes com foco em ambientes de testes hetorogêneos*. Dissertação (mestrado), Universidade Federal de Pernambuco. Pernambuco, Brasil.
18. **Sulayman, M. (2009).** A systematic literature review of software process improvement for small and medium web companies. *Communications in Computer and Information Science*, Vol. 59, pp.1–8. DOI: 10.1007/978-3-642-10619-4_1.
19. **Von Wangenheim, C., Rossa, J., Salviano, C., & Von Wangenheim, A. (2010)**. Systematic literature review of software process capability/maturity models. *Proceedings International Conference on Software Process Improvement and Capability Determination*.
20. **García, C., Dávila, A., & Pessoa, M. (2014)**. Test process models: Systematic literature review. *Communications in Computer and Information Science*, Vol. 477, pp. 84–93. DOI: 10.1007/978-3-319-13036-1_8.
21. **Serna, M.E. (2015).** Methodology for perform reliable literature reviews. *Revista Investigación Económica*.
22. **Krause, M. (1994).** Maturity model for automated software testing. *Medical Device & Diagnostic Industry Magazine*, pp. 1–10.
23. **Paulk, M., Mark, C., Weber, C., Curtis, M., & Chrissis, M. (1993).** Capability Maturity Model. *IEEE Software*, Vol. 10, No. 4, pp.18–27.
24. **Burgess, S. & Drabick, R. (1996).** *The I.T.B.G. Testing Capability Maturity Model*.
25. **Gelperin, D. (1996).** A Testability Support Model. *Proceedings Fifth International Conference on*

Software Testing, Analysis & Review, Orlando, USA.

26. Burnstein, I., Suwanassart, T., & Carlson, C. (1996). The Development of a Testing Maturity Model. *Proceedings Ninth International Quality Week Conference,* San Francisco, USA.

27. Burnstein, I., Suwanassart, T., & Carlson, C. (1996). Developing a Testing Maturity *Model*. CROSSTALK, Software Technology Support Center, Hill Air Force Base, Utah. Part I, pp. 21–24. Part II, pp. 19–26.

28. Ericson, T., Subotic, A., & Ursing, A. (1998). TIM - A Test Improvement Model. *Software Testing Verification and Reliability*, Vol. 7, No. 4, pp. 229–246.

29. **Systeme Evolutif (1998).** *Test Organization Maturity Model.*

30. Koomen, T. & Pol, M. (1999). *Test process improvement: A practical step-by-step guide to structured testing.* New York: Addison-Wesley.

31. Dustin, E., Rashka, J., & Paul, J. (1999). *Automated Software Testing - Introduction, management, and performance*. Boston: Addison-Wesley.

32. Jacobs, J. & Trienekens, J. (2002). Towards a metrics based Verification and Validation maturity model. *Proceedings 10th International Workshop on Software Technology and Engineering Practice*, Montréal, Canada. pp. 1–6. DOI: 10.1109/STEP.2002.1267622.

33. Yoon, K., Park, S., Bae, D., Chang, H., & Jung, J. (2005). A framework for the V&V capability assessment focused on the safety-criticality. *Proceedings 13th IEEE International Workshop on Software Technology and Engineering Practice,* Budapest, Hungary. pp. 17–24. DOI: 10.1109/STEP.2005.5.

34. Chandra, P. (2009). *Software assurance maturity model - A guide to building security into software development.* Version 1.0. Open Web Application Security Project (OWASP).

35. CMMI-DEV (2010). *CMMI for Development*. Version 1.3. CMU/SEI-2010-TR-033, Software Engineering Institute.

36. TMMI (2012). *Test Maturity Model Integration.* Release 1.0*. TMMi Foundation.*

37. Furtado, A., Wanderley, M., Carneiro, E., & De Farias, I. (2012). MPT.BR: A Brazilian maturity model for testing. *Proceedings 12th International Conference on Quality Software*, pp. 220–229. DOI: 10.1109/QSIC.2012.53.

38. ISO/IEC (2013). *Software and systems engineering Software testing, Part 2: Test processes*. ISO/IEC/IEEE 29119-2:2013*.*

39. Mosley, D. & Posey, B. (2002). *Just enough software test automation*. Boston: Prentice Hall.

40. Meszaros, G. (2007). *xUnit test patterns: Refactoring test code*. New York: Addison-Wesley.

41. Bhaggan, K. (2009). *Test automation in practice*. Master's Thesis Report, DSW Zorgverzekeraar, Department of ICT, Netherlands.

42. Balaraman, R. & Krishnankutty, H. (2013). *Need for a comprehensive test maturity model*. Infosys, Bangalore, India.

43. Kohlegger, M., Maier, R., & Thalmann, S. (2009). Understanding maturity models results of a structured content analysis. *Proceedings I-KNOW '09 and I-SEMANTICS '09*, pp. 51–61.

44. Wiklund, K., Eldh, S., Sundmark, D., & Lundqvist, K. (2012). Technical debt in test automation. *Proceedings Fifth International Conference on Software Testing, Verification and Validation,* pp. 887–892. DOI: 10.1109/ICST. 2012.192.

45. **Prado, D. (2003).** *Project Management in Organizations*. Belo Horizonte: Editora de Desenvolvimento Gerencial.

46. Lionbridge (2009). *Test process assessments move into the real world - A Rethinking QA white paper*. Lionbridge Technologies, Waltham, USA.

47. Alsmadi, I. (2012). *Advanced automated software testing: Frameworks for refined practice*. *Information Science Reference*. New York, USA.

48. Nasir, N. & Sahibuddin, S. (2011). Critical success factors for software projects: A comparative study. *Scientific Research and Essays*, Vol. 6, No. 10, pp. 2174–2186.

49. SGI (2010). *Modernization Clearing a pathway to success*. The Standish Group International Inc., Boston, USA.

50. Cangussu, J., DeCarlo, R., & Mathur, A. (2002). A formal model of the software test process. *IEEE Transactions on Software Engineering*, Vol. 28, No. 8, pp. 782–796. DOI: 10.1109/TSE.2002. 1027800.

Artículo recibido el 10/11/2015; aceptado el 22/02/2017.
Autor de correspondencia es Edgar Serna Montoya.

Método de asociación de datos basado en curvas B-Spline para el problema de SLAM en ambientes complejos

Alfredo Toriz Palacios[1], Abraham Sánchez López[2]

[1] Universidad Popular Autónoma del Estado de Puebla, México

[2] Benemérita Universidad Autónoma de Puebla, México

alfredo.toriz@upaep.mx, asanchez@cs.buap.mx

Resumen. En este documento se presenta una nueva propuesta para dar solución al problema de asociación de datos para SLAM destinado a la construcción de mapas de ambientes complejos. La idea principal de la que parte la propuesta, es la de utilizar curvas B-Splines como medio para describir obstáculos con geometrías complejas presentes en el área de trabajo y utilizar la información contenida en ellas para encontrar puntos característicos del ambiente que puedan ser asociados. La utilización de esta información para realizar una asociación más exacta es una de las principales contribuciones de este documento, ya que esta tiene un impacto directo sobre la localización del robot y en consecuencia en la calidad del mapa final. El problema es abordado inicialmente comparando los puntos de control que forman tanto las curvas que representan los obstáculos observados en un instante determinado, como las que representan los obstáculos almacenados en el mapa que se está construyendo, relacionando aquellos que se encuentran suficientemente cerca. A continuación, la curvatura de las B-Splines relacionadas es obtenida para extraer puntos característicos (Puntos de inflexión y esquinas) contenidos en las curvas. Finalmente, la información coincidente será utilizada para corregir la posición del robot y la de los obstáculos detectados. Numerosos experimentos han sido llevados a cabo usando información real y simulada, con la finalidad de validar los procesos y algoritmos propuestos en nuestro enfoque. Nuestro método, logra una gran precisión en la construcción de mapas de ambientes complejos, lo cual es casi imposible para las técnicas que existen actualmente.

Palabras clave. SLAM, asociación de datos, curvas B- Spline.

B-Splines Based Data Association Method for the SLAM Problem in Complex Environments

Abstract. This paper presents a new proposal to solve the problem of data association for SLAM used to build of maps of complex environments. The main idea of this proposal is to use B-spline curves as a way to describe obstacles with complex geometries found in the environment and use the information contained in them to find characteristic points that may be associated. The use of this information for a more accurate association process is one of the major contributions of this work, because a robust association has a direct impact on the localization of the robot and thus the quality of the final map. The data association problem was initially addressed by comparing the control points that form both the curves representing the obstacles observed at a given time, and those that represent the obstacles stored in the map being constructed, relating those that are close enough. Then, the curvature of the related B-spline is obtained to extract characteristic points (inflection points and corners) contained in the curves. Finally, the matching information will be used to correct the position of the robot and the detected obstacles. We carried out numerous experiments by using real and simulated information in order to validate the processes and algorithms proposed in our approach. Our method achieves a great precision in map construction of complex environments, which is nearly impossible with techniques that currently exist.

Keywords. SLAM, data association, B-Spline curves.

1. Introducción

Uno de los retos fundamentales de la robótica de hoy, es el de obtener mecanismos robustos y eficientes para modelar ambientes de creciente complejidad usando robots móviles para su exploración. Este, es conocido como el problema de localización y mapeo simultaneo (SLAM), el cual básicamente trata de la necesidad de construir un modelo del entorno (el mapa) y estimar al mismo tiempo la ubicación del robot, utilizando únicamente observaciones relativas a las características más relevantes detectadas en el ambiente por sus sensores [18].

Muchos enfoques han tratado con este problema sin éxito, ya que la mayoría de estos métodos, intentan extraer características geométricas y representar sus posiciones en el mapa [6, 9, 12, 22], o discretizar el espacio en células [2, 7, 17] y clasificar cada una como ocupada o vacía, lo que restringe su campo de aplicación a ambientes con geometrías bien definidas o de dimensiones limitadas.

En consecuencia, nuevos métodos han surgido en la literatura que tratan de lidiar con el problema de SLAM para ambientes complejos [14,15,19], siendo uno de los más recientes el presentado por Pedraza et al en [19], en donde los autores proponen la utilización de curvas B-Spline como herramienta para la descripción de obstáculos con geometrías complejas; sin embargo, esta nueva forma de representación presenta en sí mismo un nuevo reto, en donde la dificultad radica en identificar características de ambiente semejantes, de forma que estas puedan ser utilizadas para corregir los errores odométricos del robot al momento de construir el mapa. Así, aunque el enfoque presentado por Pedraza en [19] resuelve el problema de la representación de obstáculos en entornos no estructurados, su aproximación utiliza un método de asociación de datos simple, que consiste en proyectar los rayos del sensor laser desde los objetos observados en la posición odométrica estimada, hasta los obstáculos localizados contenidos en el mapa, sin importar si estos puntos son realmente coincidentes, lo que puede ocasionar inconsistencias en el mapa creado dado que cualquier asociación falsa podría invalidar el proceso entero de SLAM.

De lo anterior, la asociación de datos es sin duda el aspecto más crítico del algoritmo de construcción de mapas de ambiente y considerado por algunos el problema más difícil en esta área de la robótica [18], ya que la correcta asociación entre características censadas durante una observación y las características del mapa son esenciales para una construcción exacta del mapa.

Así, el proceso de asociación de datos ha recibido gran atención en la literatura de la robótica en recientes años [1, 3, 8, 13, 14], y muchas soluciones han sido propuestas; sin embargo, la mayoría de estas están limitadas a características simples como líneas y puntos (impidiendo su uso en ambientes complejos), geometrías simples, o a curvas simples donde los algoritmos de asociación son muy rudimentarios. Aunque estas técnicas son apropiadas y permiten en algunos casos realizar un SLAM completo, aún es necesario desarrollar e implementar nuevas estrategias que puedan ser aplicables a casos más generales sin geometrías específicas.

De lo anterior, en este documento presentamos una nueva propuesta para la asociación de datos en SLAM, destinado a la construcción de mapas de ambientes complejos basado en curvas B-Spline para la representación de los obstáculos del mapa. Nuestro enfoque propone una mejora de los actuales métodos de asociación de datos, el cual explota completamente la información contenida en la descripción paramétrica de geometrías arbitrarias. De esta forma, con la validación y creación de un modelado avanzado de mapas de entorno en forma de curvas B-Spline, las propiedades de esta representación (curvaturas, longitud de las curvas, puntos de inflexión) son usadas para obtener un método de asociación de datos robusto aplicable a cualquier situación.

2. El enfoque de SLAM basado en curvas B-Spline

La solución al problema SLAM es considerada por muchos como un requisito clave para dotar de verdadera autonomía a los robots móviles [13, 18]; sin embargo, aunque este problema ha sido objeto de importantes investigaciones en la última década, aún surgen importantes barreras que han

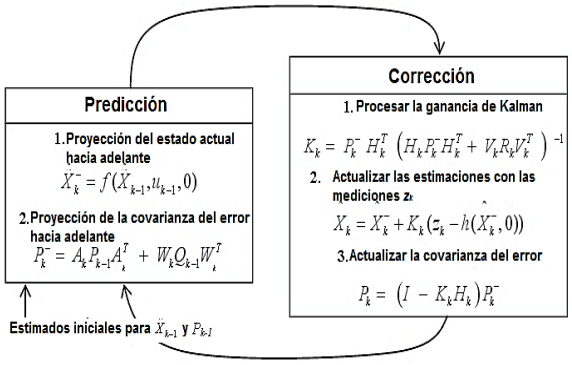


Fig. 1. Panorama completo de la operación del Filtro de Kalman Extendido

impedido que el problema tenga una solución completa, entre las que podemos mencionar las que tienen que ver con el modelado geométrico de entornos arbitrariamente complejos.

De lo anterior, Pedraza et al. Proponen en [19] un algoritmo de SLAM basado en curvas Spline como herramienta de representación del entorno. El objetivo de utilizar esta forma de representación, es el de contar con un mecanismo efectivo en el complejo problema de obtener un modelo del entorno que pueda describir el ambiente de trabajo con la mayor exactitud posible, sin hacer suposiciones ni simplificaciones peligrosas. La principal diferencia entre este enfoque y los algoritmos tradicionales de SLAM basados en características, es que no se confía en una geometría específica.

En su trabajo, los autores desarrollan técnicas y metodologías para adaptar esta novedosa forma de modelado del ambiente, en la segmentación y asociación de datos para posteriormente ser utilizados en el contexto del problema de SLAM a través de la aproximación clásica del Filtro de Kalman Extendido (EKF), mostrado en la figura 1. Las herramientas desarrolladas son descritas en esta sección.

a) Segmentación y ajuste de datos

El enfoque desarrollado por Pedraza parte del uso de un sensor laser para la adquisición de los datos del ambiente que proporciona para cada observación un conjunto de *m* puntos de datos en el espacio $\mathbb{R}^2$ ligados entre sí por la única lógica que proporciona el sensor al hacer su barrido. Para que esta información pueda ser utilizada por el método EKF de SLAM, inicialmente se deben segmentar los conjuntos de mediciones que representan los objetos físicos detectados.

La metodología presentada por Pedraza se basa en el análisis de las posiciones relativas de cada dos puntos consecutivos de datos láser, de los cuales se obtiene un conjunto inicial de m-1 vectores que conectan entre si dos puntos de datos consecutivos obtenidos por el láser:

$$p_i = d_i - d_{i-1}. \qquad (1)$$

A continuación dos comparaciones son realizadas con la intención de obtener los vectores finales de puntos que representen a los objetos en el mapa. La primera, en busca de elementos lo suficientemente cercanos como para pertenecer al mismo elemento del ambiente (Ecuación 2):

$$max(\| p_i \|, \| p_{i+1} \|) \leq \eta \cdot min(\| p_i \|, \| p_{i+1} \|). \qquad (2)$$

Y una segunda comparación, en busca de un ángulo α_i formado por dos vectores p_i y p_{i+1} (ecuación 3) para determinar si los puntos que forman los vectores están lo suficientemente alineados como para ser considerados pertenecientes al mismo objeto:

$$|\alpha_i| \leq \alpha_{\max} \Leftrightarrow \cos(\alpha_i) \geq \cos(\alpha_{\max}). \qquad (3)$$

Los autores establecen para $\eta \in [1.5, 2]$ y para $\alpha_{\max} \in [0, \pi/4]$ como valores adecuados. El proceso completo puede verse en la figura 2.

Es necesario mencionar que en esta última comparación, los autores optaron por considerar a ángulos α_i demasiado cerrados (esquinas) como un criterio de separación de características presentes en el mapa y no como un elemento distinguible del ambiente, ya que lo consideran trivial.

Finalmente, cada uno de los obstáculos obtenidos mediante el proceso de segmentación son ajustados a curvas B-Splines de grado 3 utilizando la ecuación 4 [4]:

$$D_k = C(t_k) = \sum_{i=0}^{n} N_{i,p}(t_k) X_i \quad for \quad 0 \leq k \leq m. \qquad (4)$$

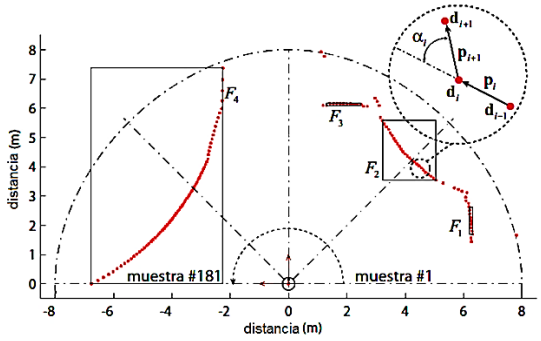


Fig. 2. Criterio de segmentación de datos brutos provenientes del sensor láser. Imagen adaptada de [19]

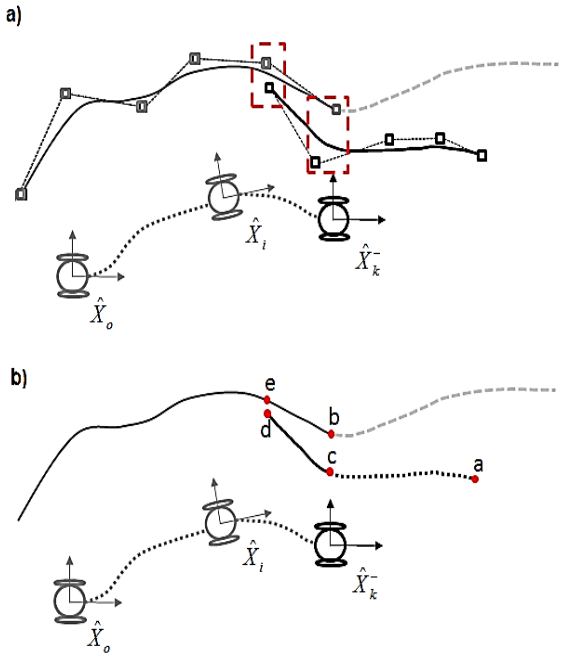


Fig.3. Concordancia entre las curvas. a) Asociación burda. b) Asociación fina

b) Asociación de datos

Una a vez que los datos han sido segmentados, el siguiente paso consiste en establecer una correspondencia entre cada uno de los m segmentos obtenidos en el instante k, y una (o ninguna) de las n características contenidas en el mapa que está siendo construido (Figura 3).

Así, el primer paso consiste en realizar una asociación denomina "burda" (Figura 3a), donde cada uno de los puntos de control de cada segmento de curva se comparan con los puntos de control de las características almacenadas en el mapa, usando el siguiente criterio:

$$min(dist(X_{m,i}, X_{o,j})) \leq d_{min} \quad \begin{cases} i = 1 \ldots n_m \\ j = 1 \ldots n_o \end{cases}. \qquad (5)$$

Así, si la distancia entre los puntos de control de la *Spline* en el mapa y los puntos de control de la *Spline* observada es mayor a cierto umbral d_{min}, la asociación de elementos es descartada; en caso contrario, estos dos objetos son asociados y un nuevo paso de correspondencia es realizado como sigue (Figura 3b):

- Uno de los extremos de la curva observada se considera como el punto a, el punto más cercano al punto a en la B-*Spline* contenida en el mapa se considera como el punto b. Si b es uno de los extremos de la B-*Spline* en el mapa, entonces el punto más cercano a b en la B-*Spline* observada se calcula y se nombra como c, si no, el punto a se asocia con el punto b.
- El proceso se repite usando el otro extremo de la B-*Spline* observada como punto de inicio (punto d en la Figura 3b). Este punto se asocia con el punto e de la B-*Spline* en el mapa. Gracias a la propiedad de las *B-Splines* acerca de la posibilidad de conocer la longitud de la curva, los segmentos eb y dc pueden ajustarse para tener la misma longitud.

Finalmente, el uso del EKF en SLAM requiere una expresión matemática que permita utilizar la asociación de datos en el contexto de la localización para predecir las mediciones que se espera obtener del sensor dado la pose del robot y el conocimiento actual del ambiente en un instante k. Así, el modelo de observación presentado por Pedraza et al. Para el caso de las curvas B-Splines se reduce a encontrar la intersección de la línea recta que forma cada rayo del láser con las curvas contenidas en el mapa y las curvas observadas por el robot (Figura 4).

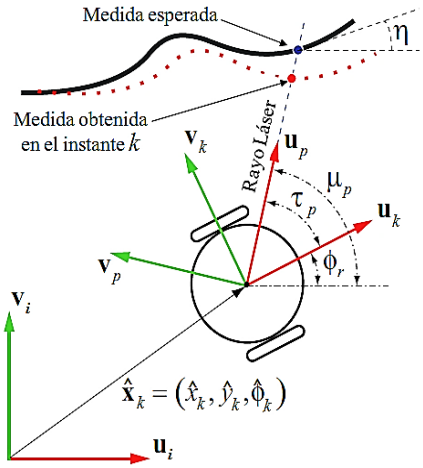


Fig. 4. Modelo de observación. Adaptado de [19]

Es importante mencionar, que este modelo de observación representa una forma riesgosa de trabajar en el método de localización, dado que las medidas esperadas no necesariamente coincidirán con las medidas observadas en el mapa, lo que puede provocar una localización incorrecta y por lo tanto un mapa de ambiente deficiente.

3. Nuestro aporte

Como se ha mencionado, uno de los aspectos crítico en el problema de SLAM es la correcta asociación datos entre las características almacenadas en el mapa que está siendo construido, y los estados subyacentes que se están observando. Para el caso de [19], aunque la forma de modelado de los obstáculos es innovadora ya que el problema de SLAM puede extenderse a cualquier tipo de ambiente de trabajo sin importar su estructura, la forma en que los autores abordan la asociación de datos puede traer consigo problemas de inconsistencia en los mapas, lo que ha motivado el desarrollo de este trabajo de investigación.

Así, aunque esta investigación toma como base la forma de representación de ambientes basada en curvas B-Spline presentada por Pedraza et. al., se desarrolla una nueva forma de tratamiento y asociación de datos que opera de forma coherente y robusta en el contexto del SLAM tomando ventaja de la información contenida en las curvas.

3.1. Administración de datos

De forma similar al método presentado en la sección 2, cuando el robot obtiene un nuevo conjunto de mediciones a través de su sistema de percepción, el primer objetivo es identificar claramente los objetos asociados con las mediciones y agruparlas en subconjuntos, para finalmente obtener las curvas B-spline que representan las porciones de los objetos detectados tan cerca como sea posible.

La segmentación de los datos brutos es realizada con el método de cluster adaptivo propuesto por Dietmayer [5], el cual permite dividir las mediciones obtenidas por el láser, usando únicamente un criterio de distancias entre dos puntos. Para explicar su funcionamiento se utiliza la figura 5, donde P_a y P_b representan dos puntos consecutivos detectados por el láser, y r_a y r_b son las distancias de estos puntos al origen de la coordenada. Dado el triángulo $O\ P_a\ P_b$, donde r_a y r_b son conocidos y α es la resolución angular del láser, podemos aplicar el teorema de cosenos para calcular la distancia entre P_a y P_b:

$$r_{ab} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(\alpha)}. \qquad (6)$$

Si la distancia r_{ab} cumple con:

$$r_{ab} \leq C_o + C_1 \min\{r_a - r_b\}, \qquad (7)$$

donde C_o representa el ajuste del ruido del sensor laser y $C_1 = \sqrt{2(1-\cos(\alpha))}$, entonces los puntos P_a y P_b formarán parte del mismo segmento de datos.

A diferencia del proceso presentado en la sección 2a, el método de Dietmayer permite mantener información relevante como esquinas y puntos de inflexión en los segmentos obtenidos, mismos que serán utilizados por el método de asociación de datos.

Una vez que los clusters son obtenidos (Figura 6 y 7), las mediciones correspondientes a cada

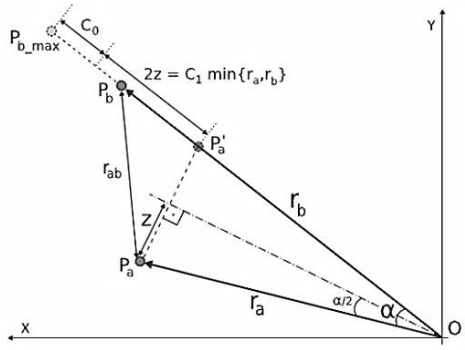


Fig. 5. Criterio de Dietmayer [5]

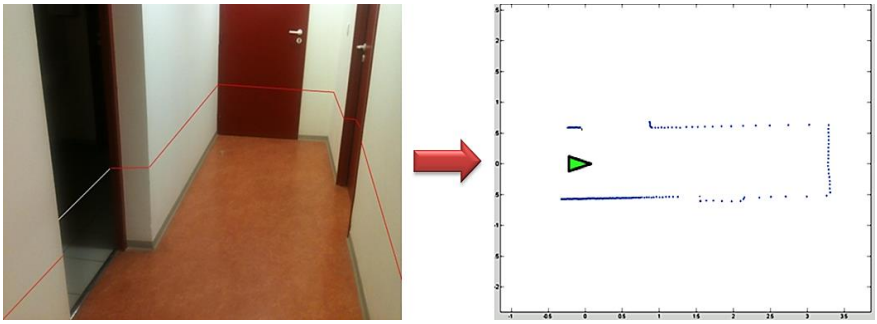


Fig. 6. Mediciones obtenidas con un sensor laser

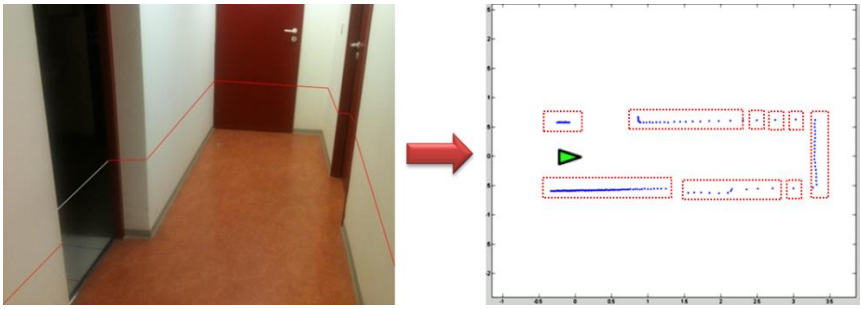


Fig. 7. Agrupación de puntos de medida obtenida con el método adaptativo propuesto por Dietmayer [5]

objeto detectado serán aproximadas utilizando curvas B-spline de grado 3 (ecuación 4).

Aunque la representación basada en B-splines reduce significativamente el ruido provocado por los errores del propio sistema de medición, un último suavizado de la curva debe ser realizado utilizando un filtro Gaussiano para garantizar que el proceso no encontrará puntos característicos falsos. Entonces, una versión evolucionada de la curva puede ser procesada como sigue:

$$D_\sigma = \{x(u,\sigma), y(u,\sigma)\}, \quad (8)$$

donde:

$$x(u,\sigma) = x(u) \otimes g(u,\sigma), \quad y(u,\sigma) = y(u) \otimes g(u,\sigma). \quad (9)$$

Aquí, $\otimes$ representa el operador de convolución y $g(u,\sigma)$ denota un filtro Gaussiano de amplitud

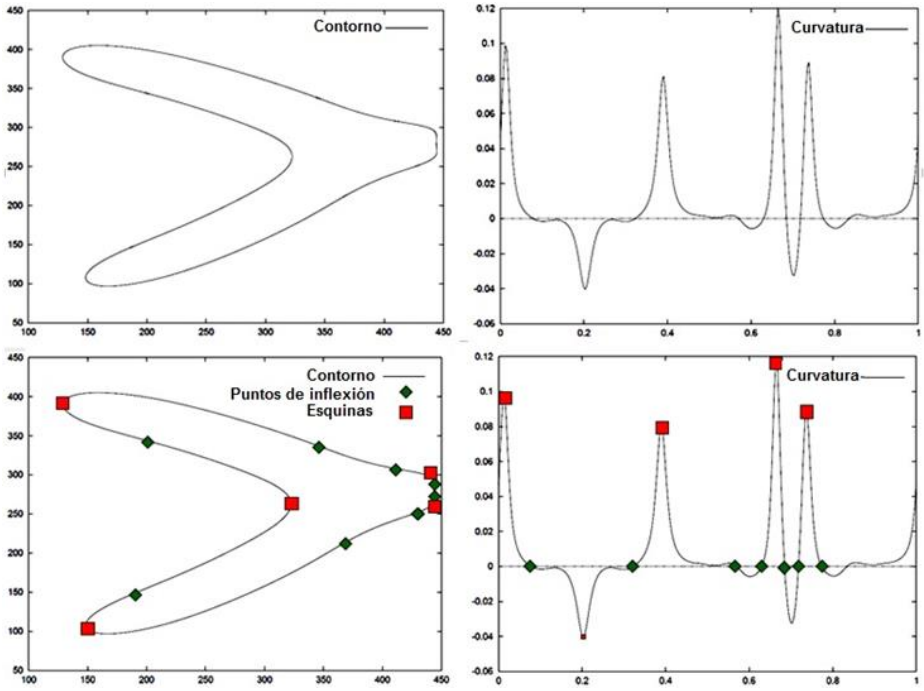


Fig. 8. Detección de puntos de inflexión y esquinas

σ; elegido de tal forma que este sólo elimina el ruido, pero no información valiosa sobre la curva, forzando el valor a ser muy bajo. Ya que nuestro proyecto trabaja normalmente con curvas abiertas, un cierto número de puntos proporcional al doble de la anchura completa en la media máxima (Full Width at Half Maximum (FWHM)) para una Gaussiana, son simétricamente compensados a ambos extremos de la curva cuando esta es suavizada.

Finalmente, con la finalidad de que el proceso de asociación pueda ejecutarse efectivamente, se debe asegurar la invariabilidad en la resolución de la curva. Así, cada curva discreta B-spline debe ser almacenada tomando puntos equidistantes en ella con una distancia ε entre cada punto:

$$dist(\sum N_p(u)X, \sum N_p(u+1)X) \cong \varepsilon . \qquad (10)$$

De lo anterior, obtenemos un vector paramétrico que contiene la B-spline.

Además del proceso descrito, una restricción sobre la longitud de la curva es aplica, esto debido a que objetos demasiado pequeños no proporcionan información suficiente y por lo tanto no es interesante incluirlos en el mapa. También, aunque nuestro método está diseñado para trabajar en ambientes estáticos, esta restricción permite en cierta forma, filtrar elementos dinámicos (personas por ejemplo).

Una vez que las B-splines han sido obtenidas y elegidas, podemos buscar característica específica contenida en las curvas que serán de gran importancia en el proceso de localización. Esencialmente, dos tipos de características serán buscados en las curvas: puntos de inflexión y esquinas.

El tipo esquina es muy común y no requiere explicación. Por otro lado, el concepto de puntos de inflexión en una forma muy general se refiere al punto sobre la curva donde esta pasa de cóncava a convexa o viceversa. El proceso para obtener ambas características en la curva está basado en el espacio escalar de curvaturas CSS [16] el cual es usado para recuperar características geométricas invariantes (Figura 8).

3.2. Búsqueda de características distinguibles en las curvas B-Splines

La búsqueda de características particulares en las curvas que definen los objetos, está completamente basada en la curvatura de la B-Spline, la cual está definida como la medida local que indica cuanto una curva se ha alejado de una línea recta; mas formalmente, la curvatura de un punto $X_u=[x_u,y_u]$ en la B-spline está definido como la cantidad igual a la inversa del radio del circulo osculador en el punto (el círculo que toca tangencialmente a la curva en el punto X_u), lo que significa que cuanto menor es el radio ρ de este círculo, más grande será la curvatura en este punto $(1/\rho)$. La fórmula para procesar la curvatura puede ser expresada como:

$$k(u,\sigma)=\frac{\dot{x}(u,\sigma)\ddot{y}(u,\sigma)-\ddot{x}(u,\sigma)\dot{y}(u,\sigma)}{(\dot{x}(u,\sigma)^2-\dot{y}(u,\sigma)^2)^{\frac{3}{2}}}. \quad (11)$$

Donde, de acuerdo con las propiedades de convolución, las derivadas de cada elemento pueden ser fácilmente calculadas ya que conocemos las formas exactas de la primera y segunda derivada del kernel Gaussiano usado *y*. Así:

$$\dot{x}(u,\sigma)=\frac{\partial}{\partial u}(x(u)\otimes g(u,\sigma))=x(u)\otimes\dot{g}(u,\sigma), \quad (12)$$

$$\ddot{x}(u,\sigma)=\frac{\partial^2}{\partial u^2}(x(u)\otimes g(u,\sigma))=x(u)\otimes\ddot{g}(u,\sigma), \quad (13)$$

$$\dot{y}(u,\sigma)=y(u)\otimes\dot{g}(u,\sigma), \quad (14)$$

$$\ddot{y}(u,\sigma)=y(u)\otimes\ddot{g}(u,\sigma). \quad (15)$$

De lo anterior, la lista de curvaturas de la B-Spline es obtenida mediante la ecuación (5), con la cual la obtención de los puntos de inflexión en la curva es realizada simplemente buscando en esta lista un cambio de signo entre dos curvaturas consecutivas. Por otro lado, la búsqueda de esquinas está basada en el trabajo de He et al. presentado en [10]. Aquí, el detector está basado en propiedades locales y globales de la curvatura (Figura 8).

Ya que nuestro sistema usa en la mayoría de los casos curvas abiertas, consideraremos como medida de seguridad y certeza solo esquinas limitadas en ambos lados por puntos de inflexión.

3.3. Asociación de datos

Con el objetivo de facilita el entendimiento del enfoque propuesto, esta sección presenta un ejemplo ilustrativo, el cual será descrito en cada fase de su desarrollo; así, en la figura 9 podemos observar un ejemplo que corresponde a la evolución de la exploración del robot hasta un instante *k+s*, donde tres objetos $B_{O,1}$, $B_{O,2}$ y $B_{O,3}$ son detectados (mostrados en rojo) dentro del rango de visión del sensor (circunferencia en línea punteada). Esta imagen subraya la posición odométrica como una aproximación a priori de la verdadera posición del robot obtenida con el modelo de movimiento, ya que errores propios de este y ruido en el sistema sensorial, llevarán al robot a una posición real diferente a la esperada. Por esta razón, los objetos $B_{O,1}$, $B_{O,2}$ y $B_{O,3}$ aparecen desplazados y no se superponen exactamente con los objetos a los que pertenecen.

De forma análoga al método de la sección 2b, la asociación de B-splines considerará inicialmente solo los puntos del polígono de control que genera las curvas. En este paso, las distancias entre los puntos de los polígonos de control de todos los objetos son obtenidas asociando las curvas observadas en la posición *k+s*, con las curvas de referencia almacenadas en el mapa, usando el criterio mostrado en la ecuación 3. Donde $X_{O,i}$ y $X_{RSL,I}$ son los puntos de control de las splines observadas y de las splines en el mapa que se está construyendo respectivamente, *dist* $(X_{RSL,i}, X_{O,i})$ representa la distancia Euclidiana entre dos puntos. Finalmente n_m y n_0 son el número de puntos de control de las spline en el mapa y de las spline observadas en la posición *k+s* respectivamente. Al final de esta primera etapa, las splies con un número mínimo μ_{min} de puntos de control relacionados, serán asociados.

En la figura 10, podemos observar como las curvas ($B_{RSL\ 1}$, $B_{O\ 2}$) and ($B_{RSL\ 2}$, $B_{O\ 3}$) han sido asociadas ya que tienen 5 y 7 puntos relacionados de sus polígonos de control respectivamente, obtenidos de la ecuación 5.

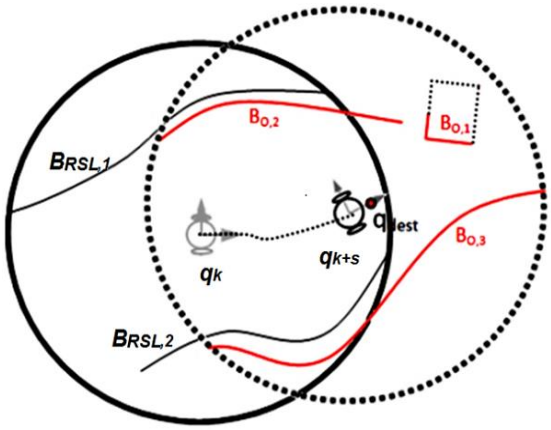


Fig. 9. Robot en la posición odometrica q_{k+s} con tres obstáculos detectados dentro de su rango de detección

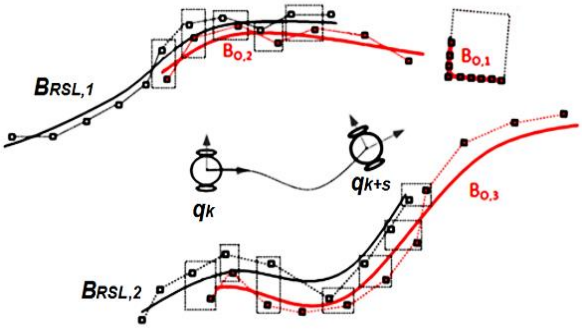


Fig. 10. Asociación áspera realizada con los puntos de control de las curvas

Si en este punto existen curvas relacionadas, el siguiente paso es realizar una asociación fina; para lo cual, se buscarán esquinas y puntos de inflexión contenidas en las curvas relacionadas; si esto ha sido posible, los elementos encontrados serán usados para realizar una asociación precisa entre cada par de curvas ($B_{RSL,i}$,$B_{O,j}$).

La información acerca del tipo de característica y de la curvatura en ese punto serán usados para evitar errores en este paso de asociación (figura 11). Si en alguna de las curvas relacionadas ningún elemento ha sido encontrado (líneas o curvas demasiado suaves), el proceso de asociación fina será ejecutado encontrando los puntos más cercanos a los finales de las curvas relacionadas como se propone en [19].

Una vez que todos los elementos están relacionados, una búsqueda de los puntos inicial y final de las curvas relacionadas es realizado. Tomando los puntos de inflexión o las esquinas más extremas como puntos iniciales contenidos en ambas curvas relacionadas, tomaremos algunos puntos continuos en la curva paramétrica hacia su final.

El número de puntos a tomar, es el máximo número de elementos que pueden ser tomados en el segmento de curva de menor longitud de las dos relacionadas desde el punto característico más extremo hacia el final de la curva.

Lo anterior puede ser visto en detalle en las curvas relacionadas ($B_{RSL,2}$,$B_{O,3}$) en la Figura 12. Aquí, observamos que el punto inicial representado por el círculo azul fue tomado eligiendo seis elementos de la curva paramétrica (mostrados en línea discontinua azul) a partir del punto de inflexión, representado por el círculo verde, al extremo inicial de la curva.

La longitud del segmento de curva $B_{O,3}$ desde el inicio de la curva al punto de inflexión es más grande que el segmento de curva $B_{RSL,2}$ desde el inicio de la curva al punto de inflexión y por lo tanto, los elementos del segmento de curva con al punto de inflexión es más grande que el segmento de curva $B_{RSL,2}$ desde el inicio de la curva al punto de inflexión y por lo tanto, los elementos del segmento de curva con menor longitud seguramente estarán contenidos en el otro de mayor longitud.

El mismo proceso es realizado en el extremo final de la curva, donde, 15 puntos de la curva paramétrica fueron tomados desde la esquina, representada con el diamante verde, dado que el segmento de curva de menor longitud desde la esquina al extremo de la curva perteneciente a $B_{RSL,2}$ contiene solo 15 puntos.

Finalmente, cuando el proceso de asociación es terminado, las curvas relacionadas tendrán una apariencia similar al de la figura 13.

4. Resultados experimentales

Aunque la asociación de datos es un problema general aplicable y necesario para cualquier algoritmo de SLAM, en esta sección hemos decidido aplicar nuestra propuesta utilizando el Filtro de Kalman Extendido para SLAM.

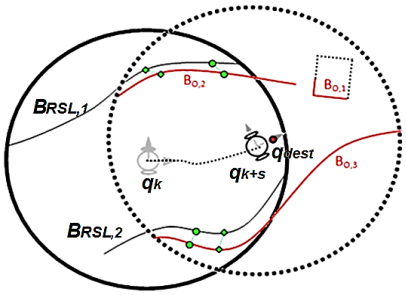


Fig. 11. Asociación de los puntos de inflexión y esquinas entre las curvas de la RSL y las curvas observadas. Los círculos verdes representan los puntos de inflexión mientras que los diamantes representan esquinas

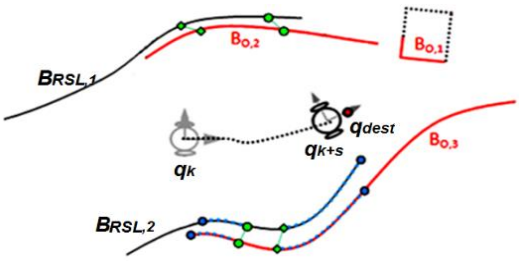


Fig. 12. Ejemplo de cómo el punto inicial y final de los segmentos de curva relacionados son encontrados

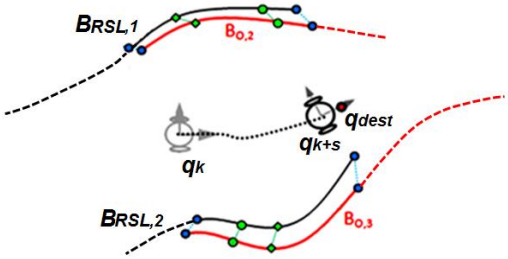


Fig. 13. Segmentos de curva relacionados con el proceso de asociación de datos descrito. Los círculos azules, representan los extremos de las curvas relacionadas mientras que los círculos y diamantes verdes representan los puntos de inflexión y las esquinas de la curva respectivamente

Lo anterior fue hecho con la finalidad de realizar una comparación de nuestro algoritmo de asociación de datos, con la propuesta presentada por Pedraza et al. en [19], ya que este es uno de los pocos trabajos que ha lidiado con el problema de representación de ambientes complejos utilizando la misma herramienta de descripción de obstáculos que se utilizó en este documento, pero con una aproximación a la asociación de datos diferente. Esta comparación tiene como objetivo verificar el impacto que la asociación de datos tiene sobre la localización del robot y por lo tanto en la calidad del mapa final.

Las pruebas hechas al método propuesto, fueron realizadas usando un robot diferencial real y simulado Pioneer P3DX equipado con un sensor laser Hokuyo URG-04lx, el cual tiene un rango de detección de 0.02 a 4 metros aproximadamente con una desviación típica (σ_L) de 1% de la medida, una resolución angular de 0.36 grados y un ángulo de escaneo de 240 grados. Además, el robot tiene un anillo de 16 sensores ultrasónicos de los cuales 6 de ellos posicionados en la parte trasera el robot son usados para obtener información del ambiente en los 120° donde el sensor laser no puede ver.

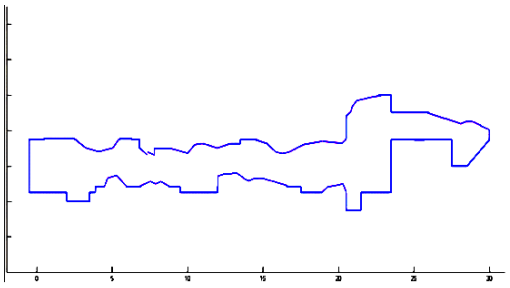


Fig. 14. Ambiente simulado LIRMM

El ambiente simulado usado para nuestras pruebas, es una adaptación de las instalaciones del Laboratorio de informática, Robótica y Micro-electrónica de Montpellier y es mostrado en la figura 14**.**

La estrategia de asociación de datos desarrollada en este documento, así como cualquier otra solución propuesta en este campo es validada usando como criterio el rendimiento computacional, calidad del mapa y la consistencia de los algoritmos.

Sin embargo, a diferencia de los métodos cuya representación del ambiente está basada en geometrías específicas y donde mucha de la información adquirida por los sensores es desperdiciada, nuestro enfoque explota la máxima cantidad de información posible proporcionada por el sensor evitando simplificaciones peligrosas.

La figura 15 muestra los resultados de experimentos de SLAM para la estrategia que hemos desarrollado.

La figura 15a muestra la trayectoria real del robot (línea continua azul), la trayectoria odométrica (línea punteada verde) y la trayectoria obtenida por el método EKF-SLAM (línea roja punteada). Por otro lado, cuando hay certeza sobre la trayectoria real del robot (como en las simulaciones) es posible realizar algunas comprobaciones para dar una idea de la calidad de los algoritmos desde el punto de vista de su consistencia.

Por esta razón, ha sido posible incluir en la figura 15b, la representación del error odométrico (línea azul) en *X*, *Y* y *Theta*, así como errores de localización (línea roja punteada), lo cual nos permite tener una idea de la calidad final del mapa, ya que esto tiene una influencia directa sobre la precisión de la localización del robot dada la estrecha relación que existe entre la localización espacial de los obstáculos y la ubicación del robot (Figura 16).

A partir de los datos obtenidos con nuestro enfoque, y tomando como referencia los errores mostrados por el método de SLAM presentado por Pedraza et al. [19], concluimos que el método de asociación de datos propuesto mejorando notablemente la calidad del mapa y los errores que este presenta como se muestra en la figura 17.

Finalmente, en esta sección presentamos experimentos con datos reales en ambientes reales, para validar los resultados presentados. En todas las pruebas se muestran los mapas obtenidos considerando sólo la información odométrica reportada por el robot y los mapas obtenidos después de aplicar el método SLAM con nuestra propuesta de asociación de datos.

La figura 18 muestra el ambiente real del corredor LIRMM. El mapa obtenido con nuestra estrategia de SLAM es mostrado en la figura 19a, mientras que el mapa odométrico obtenido utilizando solo información odométrica es mostrado en la figura 19b. En estas imágenes se puede observar el impacto que la localización del robot tiene en la calidad final del mapa (figura 19c) y el rol tan importante que la asociación de datos tiene, ya que el utilizar información con una acumulación sistemática de errores en las medidas de datos espaciales resulta en un mapa deficiente.

Considerando que nuestro enfoque ha sido creado para construir mapas de ambientes complejos, otro experimento es realizado utilizando el entorno real mostrado en la figura 20a. En este ejemplo, podemos observar que la construcción del mapa sería imposible utilizando un método clásico basado en líneas y puntos. La figura 20b muestra el mapa obtenido con nuestro enfoque.

5. Conclusiones

Se ha propuesto un método de asociación de datos basado en el análisis de las curvaturas de las curvas B-spline relacionada. Aquí, usamos las técnicas de imágenes digitales CSS [16], puntos de inflexión y técnicas de extracción de esquinas utilizadas en el campo del reconocimiento de

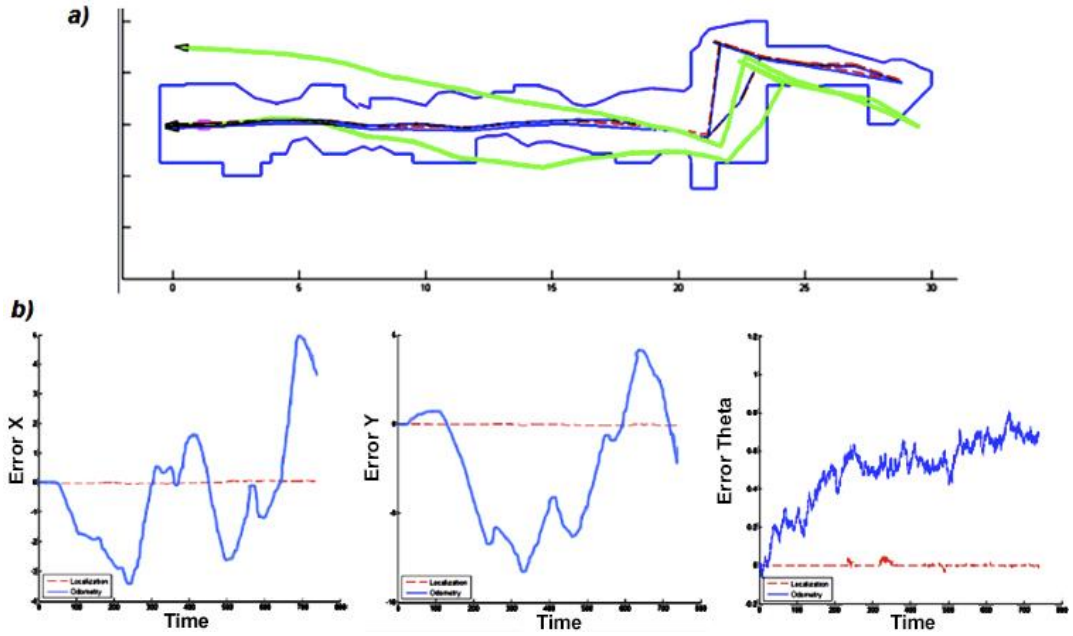


Fig. 15. Experimento de exactitud y consistencia para el EKF-SLAM basado en el enfoque de asociación de datos presentado en este documento. a) Trayectorias efectuadas por el robot con error odométrico (línea verde), robot con posición ideal (línea azul) y robot localizado utilizando el método de asociación de datos presentado en este trabajo (línea roja). b) Errores en X, Y, Theta reportados por el robot con error odométrico (línea azul), y por el robot localizado basado en el método de asociación de datos presentado en este trabajo (línea roja)

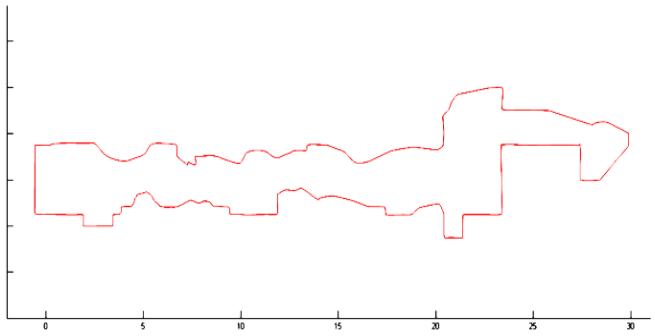


Fig.16. Mapa obtenido con el método de asociación de datos para SLAM presentado en este trabajo

patrones. Este mecanismo de asociación no sólo permite establecer una correspondencia robusta entre las observaciones realizadas por el robot y los objetos contenidos en el mapa de trabajo, sino que facilita la correspondencia paramétrica entre cada par de elementos representativos asociados.

Mostramos también una serie de pruebas en simulación del método de SLAM basado en nuestra propuesta de asociación de datos, las cuales han permitido validar nuestra aproximación. En ellas, se muestran la precisión obtenida con nuestro enfoque así como con enfoques recientemente mostrados, permitiéndonos verificar que los niveles de error muestran mejoras significativas a los obtenidos con otros métodos.

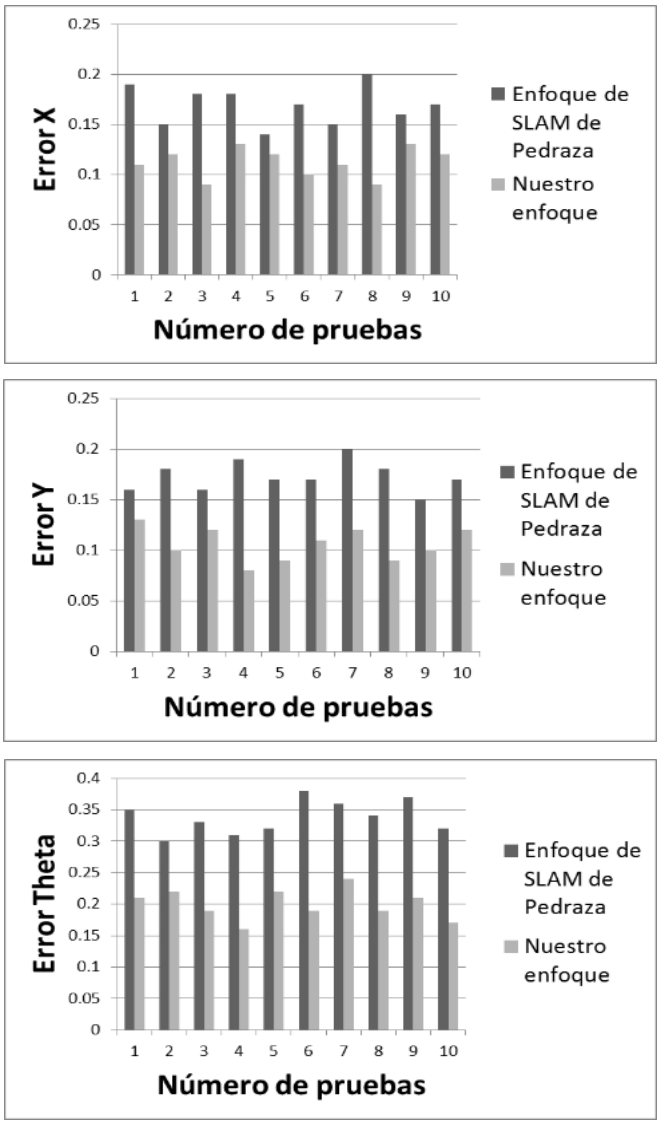


Fig. 17. Errores obtenidos con las estrategias de SLAM con asociación de datos simple [9] y SLAM con nuestro enfoque de asociación de datos

Al final, mostramos los mapas obtenidos con datos reales que permitan evaluar la aplicabilidad práctica del método propuesto, lo que nos permitió comprobar su usabilidad en ambientes reales.

El método de asociación de datos desarrollado en este trabajo propone una mejora sustancial al método de SLAM para ambientes complejos presentado en [19], no solo por manejar de forma integral la información contenida en las curvas B-Spline que representan los objetos, sino por establecer correspondencias exactas que evitan el desperdicio de tiempo en el tratamiento de asociaciones superfluas dentro de los métodos de SLAM; además, se tiene la ventaja adicional de proporcionar siempre un una localización más precisa y consistente y una estimación de mapa robusta.

Fig. 18. Ambiente corredor real usado para las pruebas

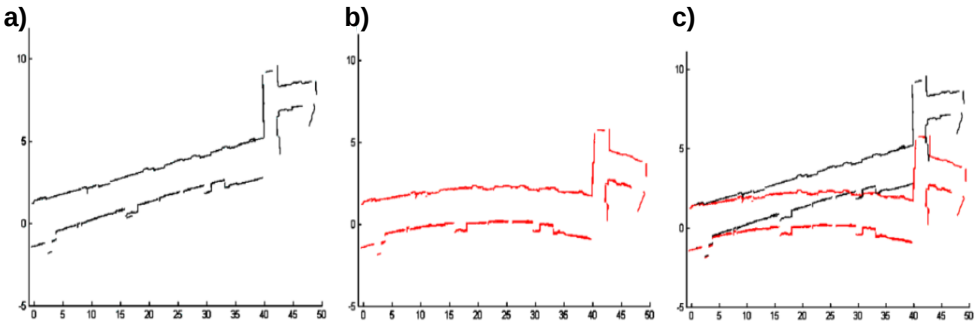


Fig. 19. a) Ambiente corredor real adquirido con el método de SLAM basado en el método de asociación de datos propuesto. b) Ambiente corredor real adquirido usando solo información odométrica. c) Diferencia entre los mapas obtenidos con el método de SLAM y con información odométrica pura.

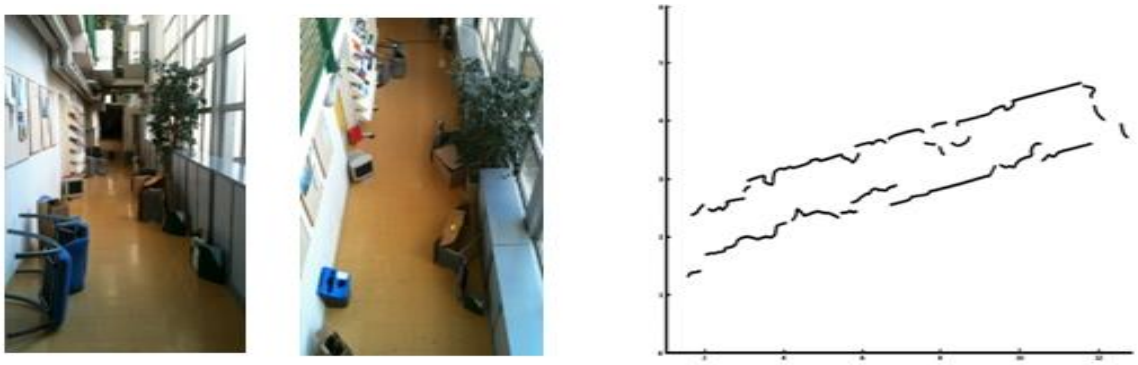


Fig. 20. a) Ambiente complejo usado para las pruebas. b) Mapa del ambiente complejo real adquirido con el método de SLAM

Referencias

1. **Beinhofer, M., Kretzschmar, H., & Burgard, W. (2013).** Deploying Artificial Landmarks to Foster Data Association in Simultaneous Localization and Mapping. *IEEE International Conference on Robotics and Automation (ICRA).* DOI: 10.1109/ICRA.2013.6631325.
2. **Brooks, A. & Bailey, T. (2009).** *HybridSLAM: Combining FastSLAM and EKF-SLAM for reliable mapping*. Algorithmic Foundation of Robotics VIII, Springer Berlin Heidelberg. pp. 647–661. DOI: 10.1007/978-3-642-00312-7_40.
3. **Cooper, J., Roy, N., Gustafson, D., & McConley, M. (2005).** *A comparison of data association techniques for Simultaneous Localization and Mapping.* Massachusetts Institute of Technology, Dept. of Aeronautics and Astronautics.
4. **De Boor, C. (1978).** *A Practical Guide to Splines*. Springer-Verlag, New York.
5. **Dietmayer, K., Sparbert, J., & Streller, D. (2001).** Model based object classification and object tracking in traffic scenes from range images. *Proceedings of the IV IEEE Intelligent Vehicles Symposium*.
6. **Dissanayake, M.W.M.G., Newman, P., Clark, H.F., Durrant-Whyte., & Csobra, M. (2001)**. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 3, pp. 229–241. DOI: 10.1109/70.938381.
7. **Grisetti, G., Stachniss, C., & Burgard, W. (2005).** Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling. *IEEE International conference on robotics and Automation.* DOI: 10.1109/ROBOT.2005.1570477.
8. **Guerra, E., Munguía, R., Bolea, Y., & Grau, A. (2013).** New validation algorithm for data association in SLAM. *ISA Transaction*, Vol. 52, No. 5, pp. 662–671. DOI: 10.1016/j.isatra.2013.04.008.
9. **Guivantand, J. & Nebot, E. (2001).** Optimization of the Simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotic and Automation*, Vol.17, No. 3, pp. 242–257. DOI: 10.1109/70.938382.
10. **He, X. & Yung, N. (2008).** Corner detector based on global and local curvature properties. *Optical Engineering.* Vol. 47, No. 5, DOI:10.1117/1.2931681.
11. **Johannsson, H., Kaess, M., Fallon, M., & Leonard, J. (2013).** Temporally scalable visual SLAM using a reduced pose graph. *IEEE International Conference on Robotics and Automation (ICRA).* DOI: 10.1109/ICRA.2013.6630556.
12. **Leonard, J.J., Durrant-Whyte, H.F., & Cox, I.J. (1992).** Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research*, Vol. 11, No. 4, pp. 286–298. DOI: 10.1109/IROS.1990.262373.
13. **Leung, C., Huang, S., & Dissanayake, G. (2008).** Active SLAM for structured environments. *IEEE Int. Conf. Robot. Automat.,* pp. 1898–1903. DOI:10.1109/ROBOT.2008.4543484.
14. **Liu, M., Huang, S., & Dissanayake, G. (2011).** Feature based SLAM using laser sensor data with maximized information usage. *IEEE International Conference on Robotics and Automation (ICRA),*

Shanghai, China. DOI: 10.1109/ICRA.2011.5980504.

15. **Liu, M., Huang, S., Dissanayake, G., & Kodagoda, S. (2010).** Towards a consistent SLAM algorithm using B-Splines to represent environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* DOI: 10.1109/IROS.2010.5649703.
16. **Mokhtarian, F. (1995).** Silhouette-based isolated object recognition through curvature scale space. *IEEE Pattern Analysis and Machine Intelligence,* Vol. 17, No. 5, pp. 539–544. DOI: 10.1109/34.391387.
17. **Nieto, J., Bailey, T., & Nebot, E. (2006).** Scan-SLAM: Combining EKF-SLAM and scan correlation. *Field and service robotics*, Springer Berlin Heidelberg, pp. 167–178. DOI: 10.1007/978-3-540-33453-8_15.
18. **Nieto, J., Guivant, J., Nebot, E., & Thrun, S. (2003).** Real time data association for fastslam. *IEEE International Conference on Robotics and Automation.* DOI: 10.1109/ROBOT.2003.1241630.
19. **Pedraza, L., Rodríguez-Losada, D., Matia, F., Dissanayake, G., & Miro, J. (2009).** Extending the Limits of Feature-Based SLAM with B-Splines*. IEEE Transactions on Robotics,* Vol. 25, No. 2, pp. 353–366. DOI: 10.1109/TRO.2009.2013496.
20. **Sun, R., Ma, S., Li, B., Wang, M., & Wang, Y. (2011).** A Simultaneous Localization and Mapping Algorithm in Complex Environments: SLASEM. *Advanced Robotics*, Vol. 25, No. 6-7, pp. 941–962. DOI: 10.1163/016918611X563373.
21. **Zhang, L. & Ghost, B. (2000).** Line segment based map building and localization using 2D laser range finder. *IEEE International conference on robotics and Automation.* DOI: 10.1109/ROBOT.2000.846410.
22. **Zhang, S., Xie, L., & Adams, M. (2004).** Gradient model based feature extraction for simultaneous localization and mapping in outdoor applications. *IEEE Control, Automation, Robotics and Vision Conference*. DOI: 10.1109/ICARCV.2004.1468864.

Artículo recibido el 15/12/2015; aceptado el 29/03/2017.
Autor de correspondencia es Alfredo Toriz Palacios.

Parameter Estimation for Chaotic Fractional Systems by Using the Locust Search Algorithm

Erik Cuevas, Jorge Gálvez, Omar Avalos

Universidad de Guadalajara, CUCEI, Departamento de Electrónica, Mexico

erik.cuevas@cucei.udg.mx

Abstract. Due to its multiple applications, parameter identification for fractional-order chaotic systems has attracted the interests of several research communities. In the identification, the parameter estimation process is transformed into a multidimensional optimization problem where fractional orders, as well as functional parameters of the chaotic system are considered the decision variables. Under this approach, the complexity of fractional-order chaotic systems tends to produce multimodal error surfaces for which their cost functions are significantly difficult to minimize. Several algorithms based on evolutionary computation principles have been successfully applied to identify the parameters of fractional-order chaotic systems. However, most of them maintain an important limitation; they frequently obtain sub-optimal results as a consequence of an inappropriate balance between exploration and exploitation in their search strategies. This paper presents an algorithm for parameter identification of fractional-order chaotic systems. In order to determine the parameters, the proposed method uses the evolutionary method called Locust Search (LS), which is based on the behavior of swarms of locusts. Different to the most of existent evolutionary algorithms, it explicitly avoids the concentration of individuals in the best positions, eliminating critical flaws such as the premature convergence to sub-optimal solutions and the limited exploration-exploitation balance. Numerical simulations have been conducted on the fractional-Order Van der Pol oscillator to show the effectiveness of the proposed scheme.

Keywords. Locust search, fractional-order systems, evolutionary computation, parameter identification, Van der Pol oscillator.

1 Introduction

A fractional order model is a system that is characterized by a fractional differential equation containing derivatives of non-integer order. Several engineering problems, such as transmission lines [1], electrical circuits [2] and control systems [3], can be more accurately described by fractional differential equations than integer order schemes. For this reason, in the last decade, the fractional order systems [4–8] have attracted the interests of several research communities.

System identification is a practical way to model a fractional order system. However, because the mathematical interpretation of fractional calculus is lightly distinct to integer calculus, it is difficult to model real fractional order systems directly based on analytic mechanisms [9]. For classical integer order system, once the maximum order of the system has been defined, the parameters of the model can be identified directly. However, for a fractional order system, because identification requires the choice of the fractional order of the operators, and the systematic parameters, the identification process of such systems is more complex than that of the integer order models [10]. Under such conditions, most of the classical identification methods cannot directly applied to identification of a fractional order systems [11].

The problem of estimating the parameters of fractional order systems has been commonly solved through the use of deterministic methods such as non-linear optimization techniques [12], input output frequency contents [13] or operational matrix [14]. These methods have been exhaustively analyzed and represent the most consolidated available tools. The interested reader in such approaches can be referred to [15] for a recent survey on the state-of-the-art.

Computación y Sistemas, Vol. 21, No. 2, 2017, pp. 369–380
ISSN 1405-5546
doi: 10.13053/CyS-21-2-2741

As an alternative to classical techniques, the problem of identification in fractional order systems has also been handled through evolutionary methods. In general, they have demonstrated, under several circumstances, to deliver better results than those based on deterministic approaches in terms of accuracy and robustness [16]. Under these methods, an individual is represented by a candidate model. Just as the evolution process unfolds, a set of evolutionary operators are applied in order to produce better individuals.

The quality of each candidate solution is evaluated through an objective function whose final result represents the affinity between the estimated model and the actual one. Some examples of these approaches used in the identification of fractional order systems involve methods such as Genetic Algorithms (GA) [17], Artificial Bee Colony (ABC) [18], Differential Evolution (DE) [19] and Particle Swarm Optimization (PSO) [20]. Although these algorithms present interesting results, they have an important limitation:

They frequently obtain sub-optimal solutions as a consequence of the limited balance between exploration and exploitation in their search strategies. This limitation is associated to their evolutionary operators employed to modify the individual positions. In such algorithms, during their operation, the position of each individual for the next iteration is updated producing an attraction towards the position of the best particle seen so-far or towards other promising individuals. Therefore, as the algorithm evolves, such behaviors cause that the entire population rapidly concentrates around the best particles, favoring the premature convergence and damaging the appropriate exploration of the search space [21, 22].

This paper presents an algorithm for parameter identification of fractional-order chaotic systems. In order to determine the parameters, the proposed method uses a novel evolutionary method called Locust Search (LS) [23, 31, 32] which is based on the behavior of swarms of locusts. In the proposed algorithm, individuals emulate a group of locusts which interact to each other based on the biological laws of the cooperative swarm.

The algorithm considers two different behaviors: solitary and social. Depending on the behavior, each individual is conducted by a set of evolutionary operators which mimics different cooperative conducts that are typically found in the swarm. Different to most of existent evolutionary algorithms, the behavioral model in the proposed approach explicitly avoids the concentration of individuals in the current best positions. Such fact allows avoiding critical flaws such as the premature convergence to sub-optimal solutions and the incorrect exploration-exploitation balance. Numerical simulations have been conducted on the fractional-Order Van der Pol oscillator to show the effectiveness of the proposed scheme.

The paper is organized as follows. In Section 2, the concepts of fractional calculus are introduced. Section 3 gives a description for the Locust Search algorithm. Section 4 gives a brief description of the fractional-order Van der Pol Oscillator. Section 5 formulates the parameter estimation problem. Section 6 shows the experimental results. Finally some conclusions are discussed in Section 7.

2 Fractional Calculus

Fractional calculus is a generalization of integration and differentiation to non-integer order fundamental operator. The differential-integral operator, denoted by ${}_aD_t^q$ takes both the fractional derivative and the fractional integral in a single expression which is defined as:

$$ {}_aD_t^q = \begin{cases} \dfrac{d^q}{dt^q}, & q > 0, \\ 1, & q = 0, \\ \displaystyle\int_a^t (d\tau)^q, & q < 0, \end{cases} \tag{1} $$

where a and t represents the operation bounds whereas $q \in \Re$. The commonly used definitions for fractional derivatives are the Grünwald-Letnikov, Riemann-Liouville [24] and Caputo [25]. According to the Grünwald-Letnikov

approximation, the fractional-order derivative of order q is defined as follows:

$$D_t^q f(t) = \lim_{h\to 0} \frac{1}{h^q} \sum_{j=0}^{\infty} (-1)^j \binom{q}{j} f(t-jh). \quad (2)$$

In the numerical calculation of fractional-order derivatives, the explicit numerical approximation of the q-th derivative at the points $kh, (k=1,2,...)$ maintains the following form [26]:

$${}_{(k-L_m/h)}D_{t_k}^q f(t) \approx h^{-q} \sum_{j=0}^{k} (-1)^j \binom{q}{j} f(t_k - j). \quad (3)$$

where L_m is the memory length $t_k = kh$, h, is the time step and $(-1)^j \binom{q}{j}$ are the binomial coefficients. For their calculation we can use the following expression:

$$c_0^{(q)} = 1, \quad c_j^{(q)} = \left(1 - \frac{1+q}{j}\right) c_{j-1}^{(q)}. \quad (4)$$

Then, the general numerical solution of the fractional differential equation is defined as follows:

$$y(t_k) = f(y(t_k), t_k) h^q - \sum_{j=1}^{k} c_j^{(q)} y(t_{k-j}). \quad (5)$$

3 Locust Search (LS) Algorithm

In the operation of LS [23], a population $\mathbf{L}^k$ $(\{\mathbf{l}_1^k, \mathbf{l}_2^k, \ldots, \mathbf{l}_N^k\})$ of N locusts (individuals) is processed from the initial stage (k=0) to a total *gen* number iterations (k= gen). Each individual $\mathbf{l}_i^k$ ($i \in [1,\ldots,N]$) symbolizes an n-dimensional vector $\{l_{i,1}^k, l_{i,2}^k, \ldots, l_{i,n}^k\}$ where each dimension represents a domain variable of the optimization problem to be solved. The set of variables represents the valid search space $\mathbf{S} = \{\mathbf{l}_i^k \in \mathbb{R}^n \,|\, lb_d \le l_{i,d}^k \le ub_d\}$, where lb_d and ub_d represents the lower and upper bounds for the d dimension, respectively. The quality of each element $\mathbf{l}_i^k$ (candidate solution) is evaluated by using the objective function $f(\mathbf{l}_i^k)$. In LS, at each iteration consists of two operators: (A) solitary and (B) social.

3.1 Solitary Operation (A)

In the solitary operation, a new location $\mathbf{p}_i$ ($i \in [1,\ldots,N]$) is generated by modifying the current element location $\mathbf{l}_i^k$ with a change of position $\Delta\mathbf{l}_i$ $(\mathbf{p}_i = \mathbf{l}_i^k + \Delta\mathbf{l}_i)$. $\Delta\mathbf{l}_i$ is the result of the individual interactions experimented by $\mathbf{l}_i^k$ as a consequence of its biological behavior. Such interactions are pairwise computed among $\mathbf{l}_i^k$ and the other N-1 individuals in the swarm. Therefore, the final force exerted between $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$ is computed by considering the following model:

$$\mathbf{s}_{ij}^m = \rho(\mathbf{l}_i^k, \mathbf{l}_j^k) \cdot s(r_{ij}) \cdot \mathbf{d}_{ij} + rand(1,-1), \quad (6)$$

where $\mathbf{d}_{ij} = (\mathbf{l}_j^k - \mathbf{l}_i^k)/r_{ij}$ is the unit-vector, pointing from $\mathbf{l}_i^k$ to $\mathbf{l}_j^k$. Furthermore, *rand*(1,-1) is an number randomly produced between 1 and -1. The factor $s(r_{ij})$ represents the social relation between $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$, which is calculated as follows:

$$s(r_{ij}) = F \cdot e^{-r_{ij}/L} - e^{-r_{ij}}. \quad (7)$$

Here, r_{ij} is the distance between $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$, F represents the strength of attraction whereas L is the attractive length factor. It is assumed that F<1 and L>1 so that repulsion is stronger in a shorter-scale, while attraction is applied in a weaker and longer-scale. $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ is a function that calculates the dominance value of the best element between $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$. In order to operate $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$, all the individuals from $\mathbf{L}^k$ $(\{\mathbf{l}_1^k, \mathbf{l}_2^k, \ldots, \mathbf{l}_N^k\})$ are arranged in terms of their fitness values. Therefore, a rank is assigned to each element, so that the best individual obtains the rank 0 (zero) whereas the worst individual receives the rank N-1. Under such conditions, the function $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ is defined as follows:

$$\rho(\mathbf{l}_i^k,\mathbf{l}_j^k)=\begin{cases} e^{-\left(5\cdot \mathrm{rank}(\mathbf{l}_i^k)/N\right)} & \text{if } \mathrm{rank}(\mathbf{l}_i^k)<\mathrm{rank}(\mathbf{l}_j^k), \\ e^{-\left(5\cdot \mathrm{rank}(\mathbf{l}_j^k)/N\right)} & \text{if } \mathrm{rank}(\mathbf{l}_i^k)>\mathrm{rank}(\mathbf{l}_j^k), \end{cases} \quad (8)$$

where rank(α) delivers the rank of the α-element. According to Eq. 8, $\rho(\mathbf{l}_i^k,\mathbf{l}_j^k)$ gives a value within [0,1]. Fig. 1 shows the behavior of $\rho(\mathbf{l}_i^k,\mathbf{l}_j^k)$ considering 100 elements. In the Figure, it is assumed that $\mathbf{l}_i^k$ represents one of the 99 individuals with ranks among 0 and 98 whereas $\mathbf{l}_j^k$ is fixed to the worst individual (rank 99).

Then, the resultant force $\mathbf{S}_i^m$ on each element $\mathbf{l}_i^k$ is computed as the superposition of all of the pairwise interactions exerted on it:

$$\mathbf{S}_i^m = \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{s}_{ij}^m . \quad (9)$$

Finally, $\Delta\mathbf{l}_i$ is assumed similar to the social force experimented by $\mathbf{l}_i^k$ as the superposition of all of the pairwise reciprocal forces. Consequently, $\Delta\mathbf{l}_i$ is represented as follows:

$$\Delta\mathbf{l}_i = \mathbf{S}_i^m. \quad (10)$$

After calculating the new locations $\mathbf{P}(\{\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_N\})$ of the population $\mathbf{L}^k(\{\mathbf{l}_1^k,\mathbf{l}_2^k,\ldots,\mathbf{l}_N^k\})$, the final locations $\mathbf{F}(\{\mathbf{f}_1,\mathbf{f}_2,\ldots,\mathbf{f}_N\})$ must be computed. This procedure can be summarized by the following formulation (in terms of a minimization problem):

$$\mathbf{f}_i=\begin{cases} \mathbf{p}_i & \text{if } f(\mathbf{p}_i)<f(\mathbf{l}_i^k), \\ \mathbf{l}_i^k & \text{otherwise.} \end{cases} \quad (11)$$

3.2 Social Operation (B)

The social operation is a discriminating operation which considers only to a subset $\mathbf{E}$ of the final positions $\mathbf{F}$ (where $\mathbf{E}\subseteq\mathbf{F}$). In the process first is necessary to order **F** in terms of their fitness values and collect the individuals in a temporal population $\mathbf{B}=\{\mathbf{b}_1,\mathbf{b}_2,\ldots,\mathbf{b}_N\}$. The individuals of **B** are arranged so that the best element is located in the first position $\mathbf{b}_1$ $\{b_{1,1},b_{1,2},\ldots,b_{1,n}\}$ whereas the worst individual is situated in the last location $\mathbf{b}_N$. Under such conditions, $\mathbf{E}$ is composed by the first *g* position of **B** (the best elements). Then, a subspace C_j is defined around each selected element $\mathbf{f}_j\in\mathbf{E}$. The size of C_j depends on the distance e_d which is determined as follows:

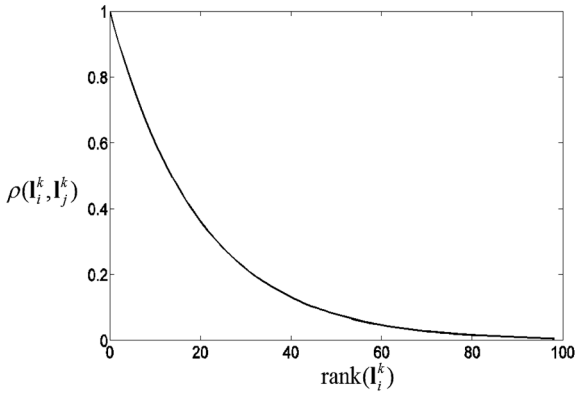


Fig. 1. Behavior of $\rho(\mathbf{l}_i^k,\mathbf{l}_j^k)$ considering 100 individuals

$$e_d=\frac{\sum_{q=1}^{n}\left(ub_q-lb_q\right)}{n}\cdot\beta, \quad (12)$$

where ub_q and lb_q are the upper and lower limits of the *q*-th dimension, n is the number of dimensions of the optimization problem, whereas $\beta\in[0,1]$ is a tuning factor. Therefore, the bounds of C_j are modeled as follows:

$$\begin{aligned} uss_j^q &= b_{j,q}+e_d, \\ lss_j^q &= b_{j,q}-e_d, \end{aligned} \quad (13)$$

where uss_j^q and lss_j^q are the upper and lower limits of the *q-th*-dimension for the subspace C_j, respectively. Once creating the subspace C_j in the neighborhood of the element $\mathbf{f}_j\in\mathbf{E}$, a set of *h* new elements ($\mathbf{M}_j^h=\{\mathbf{m}_j^1,\mathbf{m}_j^2,\ldots,\mathbf{m}_j^h\}$) are randomly produced within the limits defined by Eq. 13. Considering the *h* samples, the new

individual $\mathbf{l}_j^{k+1}$ of the next population $\mathbf{L}^{k+1}$ must be extracted. In order to select $\mathbf{l}_j^{k+1}$, the best element $\mathbf{m}_j^{best}$, in terms of fitness value from the *h* samples (where $\mathbf{m}_j^{best} \in \left[\mathbf{m}_j^1, \mathbf{m}_j^2, \ldots, \mathbf{m}_j^h\right]$), is examined. If $\mathbf{m}_j^{best}$ is better than $\mathbf{f}_j$ according to their fitness values, $\mathbf{l}_j^{k+1}$ is updated with $\mathbf{m}_j^{best}$, otherwise the position of $\mathbf{f}_j$ is assigned to $\mathbf{l}_j^{k+1}$. The elements of $\mathbf{F}$ that have not been considered by the procedure ($\mathbf{f}_w \notin \mathbf{E}$) transport their corresponding values to $\mathbf{L}^{k+1}$ without variation.

The social operation is used to exploit only favorable solutions. According to the social operation, inside each subspace C_j, *h* random samples are produced. Since the number of selected elements in each subspace is very small (typically $h < 4$), the use of this operator cannot be considered computational expensive.

4 Fractional-order Van der Pol Oscillator

The Van der Pol Oscillator model has been extensively studied as a complex example of non-linear system. It provides important models for a wide range of dynamic behaviors for several engineering applications [26, 27]. The classical integer-order Van der Pol Oscillator is described by a second-order non-linear differential equation as follows:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -\varepsilon(y_1^2(t)-1) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \tag{14}$$

where ε is a control parameter that reflects the nonlinearity degree of the system. On the other hand, the fractional-order Van der Pol Oscillator model of order *q* is defined by the following formulation [28]:

$$\begin{aligned} {}_0D_t^{q_1} y_1(t) &= y_2(t), \\ {}_0D_t^{q_2} y_2(t) &= -y_1(t) - \varepsilon(y_1^2(t)-1) y_2(t). \end{aligned} \tag{15}$$

Considering the Grünwald-Letnikov approximation (see Eq. 5), the numerical solution for the fractional-order Van der Pol Oscillator is given by:

$$\begin{aligned} y_1(t_k) &= y_2(t_{k-1}) h^{q_1} - \sum_{j=1}^{k} c_j^{(q_1)} y_1(t_{k-j}), \\ y_2(t_k) &= (-y_1(t_k) - \varepsilon(y_1^2(t_k)-1) y_2(t_{k-1})) h^{q_2} - \sum_{j=1}^{k} c_j^{(q_2)} y_2(t_{k-j}). \end{aligned} \tag{16}$$

5 Problem Formulation

In the proposed approach, the identification process is considered as a multidimensional optimization problem. In the optimization process, the parameters of a new fractional-order chaotic system FOC_E are determined by using the LS method from the operation of the original fractional-order chaotic system FOC_O.

The idea is that FOC_E presents the best possible parametric affinity with FOC_O. Under such circumstances, the original fractional-order chaotic system FOC_O can be defined as follows:

$${}_aD_t^{\mathbf{q}} Y = F(\mathbf{Y}, \mathbf{Y_0}, \boldsymbol{\theta}), \tag{17}$$

here $\mathbf{Y} = [y_1, y_2, ..., y_m]^T$ denotes the state vector of the system, $\mathbf{Y_0}$ symbolizes the initial state vector, $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_m]^T$ represents the original systematic parameter set, $\mathbf{q} = [q_1, q_2, ..., q_m]^T$ for $0 < q_i < 1 \quad (i \in [1, \ldots, m])$ corresponds to the fractional derivative orders and *F* is a generic non-linear function. On the other hand, the estimated fractional-order chaotic system FOC_E can be modeled as follows:

$${}_aD_t^{\hat{\mathbf{q}}} \hat{Y} = F(\hat{\mathbf{Y}}, \mathbf{Y}_0, \hat{\boldsymbol{\theta}}), \tag{18}$$

where $\hat{\mathbf{Y}}$, $\hat{\boldsymbol{\theta}}$ and $\hat{\mathbf{q}}$ denotes the estimated state system, the estimated systematic parameter vector and the estimated fractional orders, respectively. Since the goal is that FOC_E presents the best possible parametric affinity with FOC_O, the problem can be approached as an optimization problem described by the following formulation:

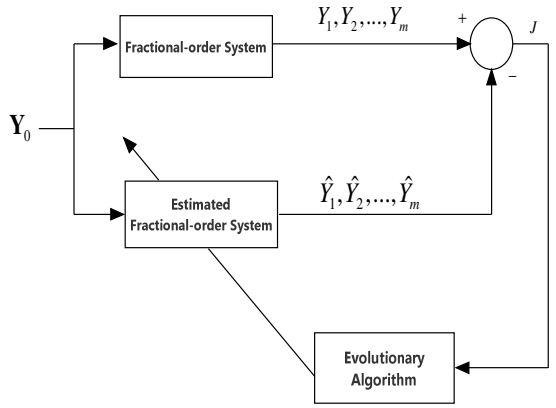


Fig. 2. Evolutionary Algorithm for Fractional-order System Parameter Estimation

$$\overline{\theta}, \overline{q} = \arg \min_{(\hat{\mathbf{Y}}, \hat{\mathbf{q}}) \in \Omega} (J(\theta, q)), \tag{19}$$

where $\overline{\theta}, \overline{q}$ denotes the best possible parametric values obtained by the optimization process, Ω symbolizes the search space admitted for parameters ($\hat{\mathbf{Y}}$ and $\hat{\mathbf{q}}$) whereas *J* represents the objective function that evaluates the parametric affinity between FOC_O and FOC_E. This affinity can be computed as follows:

$$J(\theta, q) = \frac{1}{M} \sum_{k=1}^{M} (\mathbf{Y}(k) - \hat{\mathbf{Y}}(k))^2, \tag{20}$$

where $\mathbf{Y}(k)$ and $\tilde{\mathbf{Y}}(k)$ represent the state values produced by the original and estimated systems, respectively. On the other hand, *k* denotes the sampling time point and *M* represents the length of data used for parameter estimation. According to the optimization problem formulated in Eq. 19, the parameter identification can be achieved by searching suitable values of $\hat{\mathbf{Y}}$ and $\hat{\mathbf{q}}$ within the searching space Ω, such that the objective function has been minimized.

Fig. 2 shows the graphic representation of the identification process. Since the fractional-order Van der Pol oscillator has been chosen to test the performance of the proposed approach, the fractional-order system maintain two different fractional derivative orders $\mathbf{q} = [q_1, q_2]^T$ (*m*=2) and one systematic parameter ε.

6 Experimental Results

To verify the effectiveness and robustness of the proposed approach, the fractional-order Van der Pol oscillator is chosen to test its performance. The simulations has been conducted by using MATLAB (Version 7.1, MathWorks, Natick, MA, USA) on an Intel(R) Core(TM) i7-3470 CPU, 3.2 GHz with 4 GB of RAM. In order to calculate the objective function, the number of samples is set as 300 and the step size is 0.01.

In this section, the results of the LS algorithm have been compared to those produced by the Genetic Algorithms (GA) [17], Particle Swarm Optimization (PSO) method [20], the Differential Evolution (DE) [19], and the proposed method. In all comparisons, the population has been set to 40 (*N*=40) individuals. The maximum iteration number for all functions has been set to 100. Such stop criterion has been selected to maintain compatibility to similar works reported in the literature [16].

The parameter setting for each of the algorithms in the comparison is described as follows:

1. GA: The population size has been set to 70, the crossover probability with 0.55, the mutation probability with 0.10 and number of elite individuals with 2. The roulette wheel selection and the 1-point crossover are applied.
2. PSO: In the method, $c_1 = c_2 = 2$ whereas the inertia factor (ω) is decreased linearly from 0.9 to 0.2.
3. DE: The DE/Rand/1 scheme has been employed. The parameter settings follow the instructions suggested in [30]. The crossover probability is *CR*=0.9 whereas the weighting factor is *F*=0.8.
4. In LS, *F* and L are set to 0.6 and *L*, respectively. Similarly, *g* is fixed to 20 (*N*/2), *h*=2, $\beta = 0.6$ whereas *gen* and *N* are set to 1000 and 40, respectively. Once such parameters have been experimentally determined, they are considered for all experiments in this section.

In the experiments, the fractional-order Van der Pol Oscillator to be estimated has been

Table 1. Simulation result of the algorithms GA, PSO, DE and LS

	Parameter	GA	PSO	DE	LS
BEST	ε	0.9021	0.9152	0.9632	0.9978
	$\frac{\lvert\varepsilon-1\rvert}{1}$	0.0979	0.0848	0.0368	0.0022
	q_1	1.3001	1.2810	1.2210	1.2005
	$\frac{\lvert q_1-1.2\rvert}{1.2}$	0.0834	0.0675	0.0175	0.0004
	q_2	0.8702	0.8871	0.8229	0.8011
	$\frac{\lvert q_2-0.8\rvert}{0.8}$	0.0877	0.1088	0.0286	0.0013
WORST	ε	0.1731	0.1176	0.3732	0.7198
	$\frac{\lvert\varepsilon-1\rvert}{1}$	0.8269	0.8824	0.6268	0.2802
	q_1	2.1065	0.3643	1.8532	1.3075
	$\frac{\lvert q_1-1.2\rvert}{1.2}$	0.7554	0.6964	0.5443	0.0895
	q_2	0.1219	1.7643	1.2154	0.9101
	$\frac{\lvert q_2-0.8\rvert}{0.8}$	0.8476	1.2053	0.5192	0.1376
MEAN	ε	1.2131	1.2052	1.1701	1.0186
	$\frac{\lvert\varepsilon-1\rvert}{1}$	0.2131	0.2052	0.1701	0.0186
	q_1	0.9032	1.0974	1.3421	1.2654
	$\frac{\lvert q_1-1.2\rvert}{1.2}$	0.2473	0.0855	0.1186	0.0545
	q_2	0.9052	0.7229	0.7832	0.8089
	$\frac{\lvert q_2-0.8\rvert}{0.8}$	0.1315	0.0963	0.0210	0.0111

configured such that $q_1 = 1.2$, $q_2 = 0.8$ and $\varepsilon = 1$. Similarly, the initial state has been set to [0.02- 0.2].

The statistical results of the best, the mean and the worst estimated parameters with the corresponding relative error values over 100 independent runs are shown in Table 1. From Table 1, it can be easily seen that the estimated values generated by the proposed LS algorithm are closer to the actual parameter values, which means that it is more accurate than the standard GA, PSO and DE algorithms.

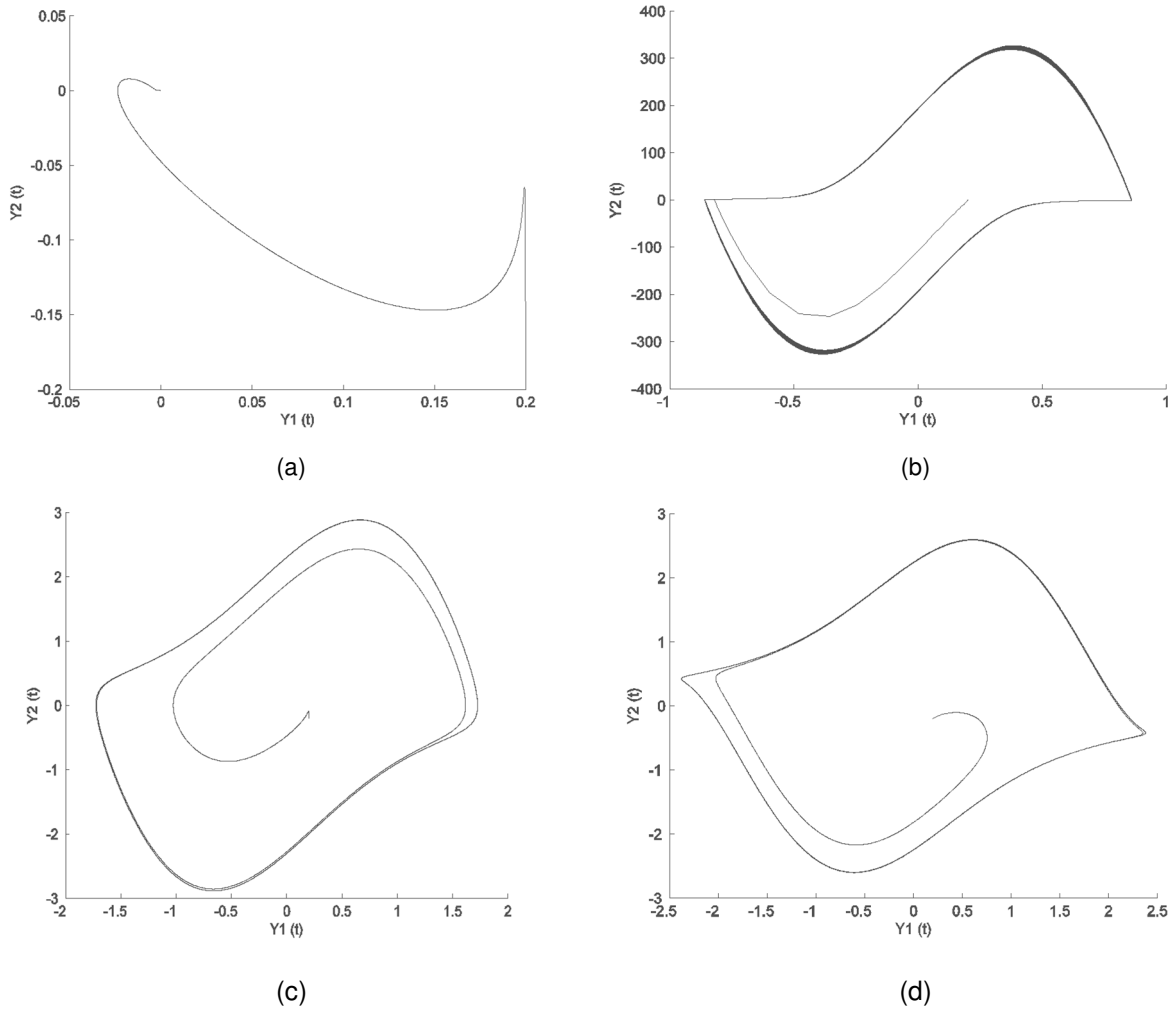


Fig. 3. Phase diagrams of the Van der Pol Oscillator by using the mean estimated parameters for (a) GA, (b) PSO, (c) DE and (d) the proposed approach

Likewise, it can also be clearly found that the relative error values obtained by the LS algorithm are all smaller than those of the standard GA, PSO and DE algorithms, which can also prove that the LS algorithm has a higher performance in terms of accuracy. Therefore, the estimated parameters can be closer to the true values than the GA, PSO and DE algorithms.

With this evidence, it can be concluded that the LS algorithm can more efficiently identify a fractional-order systems than the other algorithms used in the comparisons. Therefore, the estimated parameters can be closer to the true values than the GA, PSO and DE algorithms.

With this evidence, it can be concluded that the LS algorithm can more efficiently identify a fractional-order systems than the other algorithms used in the comparisons. In order to show the proficiency, of the proposed approach, Figure 3 presents the phase diagrams of the Van der Pol Oscillator by using the mean estimated parameters for each method.

The convergence curves of the parameters and fitness values estimated by the set of

Table 2. Average best solution obtained by each algorithm GA, PSO, DE and LS

GA	PSO	DE	LS
0.2251	0.2016	0.0982	0.0126

Table 3. *p*-values produced by Wilcoxon's test that compares LS vs GA, LS vs PSO and DE over the "average best-solution" values from Table 3

	***p*-values**
LS vs. GA	0.00021
LS vs. PSO	0.00098
LS vs. DE	0.00123

Table 4. Average Number of Function Evaluations (NFE) obtained by each algorithm GA, PSO, DE and LS

GA	PSO	DE	LS
97,378	95,366	68,446	55,933

algorithms are shown in Figures 4-6 in a single execution. From Figures 4-6, it can be clearly observed that convergence processes of the parameters and fitness values of LS algorithm are better than other algorithms. Additionally, the estimated parameter values obtained by the LS algorithm fall faster than the other algorithms.

Furthermore, Table 2 shows the average best solution obtained by each algorithm. The average best solution (ABS) expresses the average value of the best function evaluations that have been obtained from 100 independent executions. A non-parametric statistical significance test known as the Wilcoxon's rank sum test for independent samples [30, 31] has been conducted with an 5% significance level, over the "average best-solution" data of Table 2.

Table 3 reports the *p*-values produced by Wilcoxon's test for the pair-wise comparison of the "average best-solution" of two groups. Such groups are formed by LS vs. GA, LS vs. PSO and LS vs. DE. As a null hypothesis, it is assumed that there is no significant difference between mean values of the two algorithms.

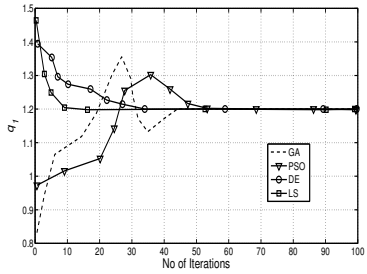


Fig. 4. Estimated parameter q_1 (fractional order).

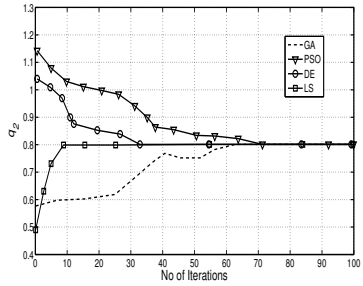


Fig. 5. Estimated parameter q_2 (fractional order).

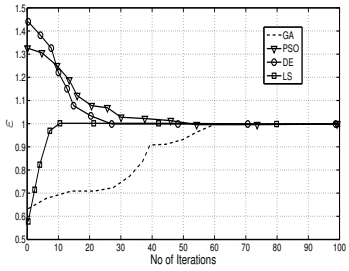


Fig. 6. Estimated systematic parameter ε .

The alternative hypothesis considers a significant difference between the "average best-solution" values of both approaches. All *p*-values reported in the table are less than 0.05 (5% significance level) which is a strong evidence against the null hypothesis, indicating that the LS results are statistically significant and that it has not occurred by coincidence (i.e. due to the normal noise contained in the process).

On the other hand, the time spent by all methods in the estimation of the parameter set is evaluated. Evolutionary methods are, in general, complex pieces of software with several operators and stochastic branches. Therefore, it is difficult to conduct a complexity analysis from a

deterministic perspective. Therefore, the number of function evaluations (*NFE*) is commonly used in order to evaluate the computational effort. Table 4 presents the *NFE* values obtained by each approach. The Table reports the averaged value considering 100 different executions. From the table, it can be seen that the proposed LS method present the best performance.

7 Conclusions

Due to its multiple applications, parameter identification for fractional-order chaotic systems has attracted the interests of several research communities. In the identification, the parameter estimation process is transformed into a multidimensional optimization problem where fractional orders, as well as functional parameters of the chaotic system are considered the decision variables. Under this approach, the complexity of fractional-order chaotic systems tends to produce multimodal error surfaces for which their cost functions are significantly difficult to minimize. Several algorithms based on evolutionary computation principles have been successfully applied to identify the parameters of fractional-order chaotic systems. However, most of them maintain an important limitation, they frequently obtain sub-optimal results as a consequence of an inappropriate balance between exploration and exploitation in their search strategies.

In this paper, an algorithm for parameter identification of fractional-order chaotic systems has been presented. In order to determine the parameters, the proposed method uses a novel evolutionary method called Locust Search (LS) [R1] which is based on the behavior of swarms of locusts. In the proposed algorithm, individuals emulate a group of locusts which interact to each other based on the biological laws of the cooperative swarm.

The algorithm considers two different behaviors: solitary and social. Depending on the behavior, each individual is conducted by a set of evolutionary operators which mimics different cooperative conducts that are typically found in the swarm. Different to most of existent evolutionary algorithms, the behavioral model in the proposed approach explicitly avoids the concentration of individuals in the current best positions. Such fact allows to avoid critical flaws such as the premature convergence to sub-optimal solutions and the incorrect exploration-exploitation balance.

In order to test the proficiency and robustness of the presented method, it has been compared to other algorithms based on evolutionary principles such as GA, PSO and DE. The comparison examines the identification of the fractional Van der Pol Oscillator. The results show a high performance of the proposed descriptor in terms of precision and robustness.

References

1. **Das, S. (2011).** *Observation of fractional calculus in physical system description*. New York: Springer; pp. 101–56. DOI:10.1007/978-3-642-20545-3_3.
2. **Arena, P., Caponetto, R., Fortuna, L., & Porto, D. (2000).** *Nonlinear Order Circuits and Systems An Introduction*. World Scientific, Singapore, Singapore.
3. **Rivero, M., Rogosin, S.V., Tenreiro-Machado, J.A., & Trujillo, J.J. (2013).** Stability of fractional order systems. *Math. Problem Eng.* DOI:/10.1155/2013/356215.
4. **Diethelm, K. (2011).** An efficient parallel algorithm for the numerical solution of fractional differential equations. *Fract. Calc. Appl. Anal.,* Vol. 14, No. 3, pp. 475–490. DOI: https://doi.org/10.2478/s13540-011-0029-1.
5. **Diethelm, K. & Ford, N.J. (2002).** Analysis of fractional differential equations. *J. Math. Anal. Appl.,* Vol. 265, No. 2, pp. 229–248. DOI: 10.1006/jmaa.2000.7194.
6. **Kilbas, A.A.A., Srivastava, H.M., & Trujillo, J.J. (2006).** *Theory and Applications of Fractional Differential Equations*. Elsevier Science, Netherlands.
7. **Podlubny, I. (1998).** *Fractional Differential Equations*. Academic Press, USA.
8. **Miller, K.S. & Ross, B. (1993).** *An Introduction to the Fractional Calculus and Fractional Differential Equations*. Wiley, New York.
9. **Wei Hu, Yongguang Yu, & Shuo Zhang (2015).** A hybrid artificial bee colony algorithm for parameter identification of uncertain fractional-order chaotic systems. *Nonlinear Dynamic*,

Vol. 82, No. 3, pp 1441–1456. DOI: 10.1007/ s11071-015-2251-6.

10. **Yongguang Yu, Han-Xiong Li, Sha Wang, & Junzhi Yu (2009).** Dynamic analysis of a fractional-order Lorenz chaotic system. *Chaos, Solitons and Fractals,* Vol. 42, No. 2, pp. 1181–1189. DOI: 10.1016/j.chaos.2009. 03.016.
11. **Poinot, T. & Trigeassou, J.C. (2004).** Identification of fractional systems using an output error technique. *Nonlinear Dyn*, Vol. 38, No. 1, pp. 133–54. DOI: 10.1007/s11071-004-3751-y.
12. **Nazarian, P., Haeri, M., & Tavazoei, M.S. (2010).** Identifiability of fractional order systems using input output frequency contents. *ISA Trans,* Vol. 49, No. 2, pp. 207–14. DOI: 10.1016/j.isatra. 2009.11.007.
13. **Saha-Ray, S. (2012).** On Haar wavelet operational matrix of general order and its application for the numerical solution of fractional Bagley Torvik equation. *App. Math Comp*, Vol. 218, No. 9, pp. 5239–5248. DOI: 10.1016/ j.amc.2011.11.007.
14. **Kerschen, G., Worden, K., Vakakis, A.F., & Golinval, J.C. (2006).** Past, present and future of nonlinear system identification in structural dynamics. *Mechanical Systems and Signal Processing,* Vol. 20, No. 3, pp. 505–592. DOI: 10.1016/j.ymssp.2005.04.008.
15. **Quaranta, G., Monti, G., & Marano, C. (2010).** Parameters identification of Van der Pol–Duffing oscillators via particle swarm optimization and differential evolution. *Mechanical Systems and Signal Processing,* Vol. 24, No. 7, pp. 2076–2095. DOI: 10.1016/j.ymssp.2010.04.006.
16. **Shengxi Zhou, Junyi Cao, & Yangquan Chen (2013).** Genetic Algorithm-Based Identification of Fractional-Order Systems. *Entropy,* Vol. 15, No. 5, pp. 1624–1642. DOI:10.3390/e15051624.
17. **Wei Hu, Yongguang Yu, & Sha Wang (2015).** Parameters Estimation of Uncertain Fractional-Order Chaotic Systems via a Modified Artificial Bee Colony Algorithm. *Entropy,* Vol. 17, pp. 692–709; DOI:10.3390/e17020692.
18. **Fei Gao, Xue-jing Lee, Feng-xia Fei, Heng-qing Tong, Yan-fang Deng, & Hua-ling Zhao (2014).** Identification time-delayed fractional order chaos with functional extrema model via differential evolution. *Expert Systems with Applications*, Vol. 41, No. 4, pp. 1601–1608. DOI: 10.1016/ j.eswa.2013.08.057.
19. **Defeng Wu, Zi Ma, Aiguo Li, & Quanmin Zhu (2011).** Identification for fractional order rational models based on particle swarm optimization. *J. of Computer Applications in Technology*, Vol. 41, No. 1/2, pp.53–59. DOI:10.1504/ IJCAT.2011. 042232.
20. **Tan, K.C., Chiam, S.C., Mamun, A.A., & Goh, C.K. (2009).** Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *European Journal of Operational Research*, Vol. 197, No. 2, pp. 701–713. DOI:10.1016/j.ejor.2008.07.025.
21. **Chen, G., Low, C.P., & Yang, Z. (2009).** Preserving and exploiting genetic diversity in evolutionary programming algorithms. *IEEE Transactions on Evolutionary Computation,* Vol. 13, No. 3, pp. 661–673. DOI: 10.1109/ TEVC.2008.2011742.
22. **Cuevas, E., González, A., Fausto, F., Zaldívar, D., & Pérez-Cisneros, M. (2015).** Multithreshold Segmentation by Using an Algorithm Based on the Behavior of Locust Swarms. *Mathematical Problems in Engineering*, DOI:10.1155/2015/ 805357.
23. **Podlubny, I. (1998).** *Fractional Differential Equations.* Acad. Press.
24. **Miller, K.S. & Ross, B. (1993).** *An Introduction to the Fractional Calculus and Fractional Differential Equations.* Wiley.
25. **Dorkal, L. (1994).** Numerical Models for the Simulation of the Fractional-Order Control Systems. *Num. Model. Simul. Fractional-Order Control Syst*.
26. **Barbosa, R.S., Machado, J.T., Vinagre, B.M., & Calderon, J. (2007)**. Analysis of the Van der Pol Oscillator Containing Derivatives of Fractional Order. *J. Vib. Control*, Vol. 13, No. 9-10, pp. 1291–1301.
27. **Cartwright, J., Eguiluz, V., Hernandez-Garcia E., & Piro, O. (1999)**. Dynamics of elastic excitable media. *J. Bifur. Chaos Appl. Sci. Eng.*, pp. 2197–2202. DOI:10.1142/S0218127499 001620.
28. **Cuevas, E., Zaldivar, D., Pérez-Cisneros, M., & Ramírez-Ortegón, M. (2011)**. Circle detection using discrete differential evolution optimization. *Pattern Analysis and Applications*, Vol. 14, No. 1, pp. 93–107. DOI: 10.1007/s10044-010-0183-9.
29. **Wilcoxon, F. (1945).** Individual comparisons by ranking methods. *Biometrics*, Vol. 1, No. 6, pp. 80–83. DOI: 10.2307/3001968.
30. **Garcia, S., Molina, D., Lozano, M., & Herrera, F. (2008).** A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: a case study on the CEC'2005 Special session on real parameter optimization. *J Heurist.* DOI: 10.1007/s10732-008-9080-4.

31. Cuevas, E., Zaldivar, D., & Perez, M. (2016). Automatic Segmentation by Using an Algorithm Based on the Behavior of Locust Swarms. *Applications of Evolutionary Computation in Image Processing and Pattern Recognition*, Vol. 100 of the series Intelligent Systems Reference Library, pp. 229–269. DOI: 10.1007/978-3-319-26462-2_10.

32. Cuevas, E., González, A., Zaldívar, D., & Pérez-Cisneros, M. (2015). An optimization algorithm based on the behavior of locust swarms. *International Journal of Bio-Inspired Computation*, Vol. 7, No. 6, pp. 402–407. DOI: 10.1504/IJBIC.2015.073178.

Article received on 02/02/2016; accepted on 07/02/2017. Corresponding author is Erik Cuevas.

Filtrado de ruido Gaussiano mediante redes neuronales pulso-acopladas

Estela Ortiz Rangel[1], Manuel Mejía-Lavalle[1], Humberto Sossa[2]

[1] Centro Nacional de Investigación y Desarrollo Tecnológico, Departamento de Ciencias Computacionales, Cuernavaca, Morelos, México

[2] Instituto Politécnico Nacional, Centro de investigación en Computación, Ciudad de México, México

{estela_or, mlavalle}@cenidet.edu.mx, humbertosossa@gmail.com

Resumen. Se describe un algoritmo llamado ICM-TM para reducir el efecto del ruido Gaussiano en imágenes monocromáticas. La operación del algoritmo se basa en el Modelo de Intersección Cortical (ICM) que es un tipo de Red Neuronal Artificial tipo Pulso-Acoplado. Una matriz de tiempos (TM) proporciona la información correspondiente a la iteración cuando la neurona correspondiente se activa por primera vez. Se establece un criterio de filtrado selectivo que combina el operador de mediana y promedio tomando como base el tiempo de activación de las neuronas. El desempeño del algoritmo propuesto se evaluó experimentalmente con ruido Gaussiano a varios niveles. Los resultados muestran la efectividad de la propuesta con respecto a los filtros mediana, Gaussiano, Sigma, Wiener y las Redes Neuronales Pulso-Acopladas tipo PCNNNI. Los resultados son representados principalmente a través del Cociente Pico Señal a Ruido (CPSR).

Palabras clave. Modelo de Intersección Cortical (ICM), ruido Gaussiano, filtro Wiener, relación pico señal a ruido (PSNR).

Using Pulse Coupled Neural Networks to Improve Image Filtering Contaminated with Gaussian Noise

Abstract. An algorithm called ICM-TM to reduce the effect of Gaussian noise in grayscale images is proposed. It is based on the operation of the well-known Intersection Cortical Model (ICM), a kind of Pulse-Coupled Artificial Neural Network. A Time Matrix (TM) provides information about the iteration when the neuron fires for first time. Each neuron corresponds to a pixel. A selective filtering criteria that combines the median and average operators using the neuron´s activation time is established. The performance of the proposed algorithm is evaluated experimentally with varying degrees of Gaussian noise. Simulation results show that the effectiveness of the method is superior to the median filter, Gaussian filter, Sigma filter, Wiener filter and to the Pulse-Coupled Neural Networks with the Null Interconnections (PCNNNI). Results are mainly provided by the parameter Peak Signal to Noise Ratio (PSNR).

Keywords. Intersection Cortical Model (ICM), Gaussian noise, Wiener filter, Peak Signal to Noise Ratio (PSNR).

1. Introducción

El ruido aditivo Gaussiano q es un modelo que permite simular la afectación aleatoria de los pixeles de una imagen con valores uniformemente distribuidos. Su función de densidad de probabilidad $p_q(x)$ viene dada en términos de la media μ y la varianza σ^2 de una variable aleatoria x. Se expresa en la siguiente ecuación [1]:

$$p_q(x) = (2\pi\sigma^2)^{-1/2} e^{-(x-\mu)^2/2\sigma^2}. \quad (1)$$

ISSN 1405-5546
doi: 10.13053/CyS-21-2-2742

Este tipo de ruido es muy común en las imágenes digitales. Debido a sus características es difícil de eliminar por completo.

El filtro Wiener $H(u,v)$ es un filtro frecuencial como se expresa en la ecuación 2 [2]:

$$H(u,v) = \frac{D^*(u,v)}{|D(u,v)|^2 + \Gamma} = \frac{1}{1 + \frac{S_w(u,v)}{S_f(u,v)}}. \quad (2)$$

Se basa en la reducción del error cuadrático medio; mejora sustancialmente la calidad de una imagen cuando ésta es afectada con ruido Gaussiano. Requiere, sin embargo del cálculo del espectro de energía de la imagen ideal $S_w(u,v)$, del espectro de energía del ruido $S_f(u,v)$, cuyo cociente se aproxima por un coeficiente Γ, y de la estimación de una función de degradación $D(u,v)$ y su conjugado $D^*(u,v)$. Algunas modificaciones a este filtro han sido propuestas con el fin de mejorar su desempeño y generalizar su utilización [3,4]. El cálculo aún depende de las estimaciones de dichos coeficientes, lo cual dificulta su aplicación.

El filtro Sigma también es efectivo en la eliminación del ruido Gaussiano; es ampliamente utilizado por su simplicidad, no obstante, su capacidad de preservar bordes aún se sigue mejorando, por ejemplo, por medio del pre-procesamiento de la imagen dividiéndola en dos componentes a las que se aplica el filtro Sigma por separado [5], lo cual mejora la preservación de bordes pero aumenta el tiempo de procesamiento del algoritmo y depende de un estimador del ruido para cada imagen.

Se han propuesto técnicas que utilizan la información de los bordes de la imagen contaminada, donde se aplican los principios de similitud [6]. No obstante, su proceso computacional es largo; su implementación en tiempo real en sistemas embebidos es difícil.

Entre las técnicas más recientes de filtrado de ruido Gaussiano se encuentran los algoritmos basados en onditas (wavelets) multidireccionales [7], que pueden remover el ruido y preservar los detalles como bordes y texturas. Tal es el caso del método de filtrado basado en la Transformada Directionlet que permite la reducción multidireccional al construir una matriz con las principales direcciones encontradas en la imagen para evitar eliminar bordes. Este algoritmo es laborioso y no toma en cuenta la información de similitud espacial entre los pixeles por lo que puede llevar a falsas detecciones.

Otras técnicas típicas de filtrado son los métodos de representación sobre-completa basados en bloques, como son el algoritmo de promedio ponderado óptimo [8], el BM3D [9] y el K-SVD [10]. Recientemente se desarrolló el SM3D-DCTNS [11] que consiste en la agrupación de bloques coincidentes con filtrado por medio de la transformada discreta coseno, el umbralado, la segmentación y un procesamiento posterior de restauración de bordes.

La técnica de filtrado mediante la integral fraccional de Alexander [9] trabaja mediante la construcción de máscaras fraccionales a partir del llamado polinomio de Alexander. Por medio de esta técnica la imagen es descompuesta a través de wavelets de los cuales sólo se filtran los componentes diagonal, vertical y horizontal. Dos de las principales desventajas de este método son la descomposición de la imagen en sub-andas así como el cálculo de coeficientes de forma experimental.

Dentro de las técnicas de inteligencia artificial también se han hecho propuestas para mejorar el filtrado de ruido Gaussiano, las más prominentes son las que emplean el aprendizaje para máquinas como el Ridgelet Suppport Vector Filter (RSVM) [12] que se basan en la generación de Diccionarios Multiescala generados mediante aprendizaje a partir de ejemplos (MRSVF y NCSR) y que permiten extraer las características sobresalientes asociadas con singularidades lineales y así evitar su desaparición. Dicho método consiste en el cálculo de una serie de matrices, la extracción de vectores columna centrales y la comparación con los diccionarios de las características estadísticas de las imágenes con ruido para después minimizar el error en un laborioso e iterativo proceso.

Una forma distinta de enfrentar el problema de eliminación del ruido Gaussiano en imágenes digitales ha surgido de la exploración experimental, tal es el caso de las Redes Neuronales Artificiales de tercera generación llamadas Redes Neuronales Pulso-Acopladas o por sus siglas en inglés PCNN (Pulse Coupled Neural Networks), las cuales han sido empleadas de modo eficiente para el procesamiento de

imágenes en diversas tareas como la segmentación, la clasificación, la identificación de imágenes, entre otras [13].

Las redes tipo PCNN son un modelo matemático, propuesto por Eckhorn. Se basa en la frecuencia de activación de las neuronas de la corteza visual de los mamíferos [14,15]. El tiempo y frecuencia de activación de las neuronas y han sido utilizados para procesar imágenes gracias al modelo computacional simplificado de PCNN propuesto por Ranganath y Kuntimad [16].

Las propiedades del modelo PCNN permiten a cada neurona procesar un pixel y relacionar su nivel de gris con el de sus vecinos. Al iterar la Red Neuronal el tiempo de activación de cada neurona se registra en una matriz. Esta información puede ser utilizada para la detección de los pixeles con ruido y la aplicación de una técnica selectiva de filtrado de ruido Gaussiano [17, 18], para la reducción del ruido Gaussiano y de sal y pimienta mezclados [19, 20].

Diversos modelos simplificados de PCNN han sido propuestos para trabajar computacionalmente con imágenes. Dos de las principales variaciones son el Intersection Cortical Model (ICM) y el Pulse-Coupled Neural Networks with the Null Interconnections (PCNNNI) [21]. Al combinarlos con el operador de mediana, promedio y morfológico o con el filtro Wiener y la matriz de tiempos (TM) se logra reducir el ruido de Sal y Pimienta y el ruido Gaussiano en imágenes digitales, no obstante la restauración de la imagen es deficiente.

Dentro de los métodos de filtrado basados en las propiedades de las PCNN que pretenden mejorar la calidad de la imagen restaurada se encuentra la propuesta de Zhang [22]. Este método utiliza un algoritmo genético adaptativo y un filtro de difusión anisotrópica para disminuir el ruido Gaussiano. No obstante sólo es útil para valores de varianza del ruido menores a 0.009 e implica gran cantidad de cálculos para completar el filtrado.

El método ICM-TM que se propone consiste en utilizar una red ICM para generar la matriz de tiempos y aplicar selectivamente los operadores mediana y promedio para suprimir el ruido Gaussiano en imágenes digitales en escala de grises, de forma que se logre una reducción significativa del ruido con un método computacional sencillo que puede ser implementado para aplicaciones con sistemas embebidos.

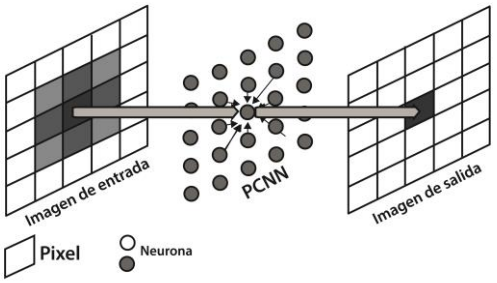


Fig. 1. Diagrama de mapeo de la red PCNN

2. Los modelos PCNN e ICM

Las Redes Neuronales Pulso-Acopladas son un sistema que permite emular efectivamente a las neuronas biológicas de la corteza visual de los mamíferos. Han sido aplicadas en variedad de dominios, especialmente en el procesamiento de imágenes para la remoción de ruido, reconocimiento de objetos, optimización, adelgazamiento, segmentación, fusión, identificación y remoción de sombras [14, 23].

En el ámbito del procesamiento de imágenes las diferencias entre las Redes Neuronales Artificiales tradicionales y las Redes Neuronales de tercera generación son evidentes tanto en su configuración como en su operación. Las principales diferencias son que a cada neurona corresponde un pixel y la imagen de salida es binaria (Fig. 1).

Las redes tipo PCNN (Fig. 2) no requieren de un entrenamiento. Su función es clasificar los pixeles por sus niveles de intensidad. En este modelo, cada pixel de la imagen se corresponde con una neurona. Su valor S_{ij} es introducido a la red por medio de una señal llamada "feeding" F_{ij}. El umbral de activación de las neuronas T_{ij} es dinámico. Además, cada neurona recibe información de sus vecinas a través de una sinapsis, lo cual se conoce como "linking" L_{ij}.

La señal *feeding* y la de *linking* ponderada por un factor de unión β se combinan para formar el

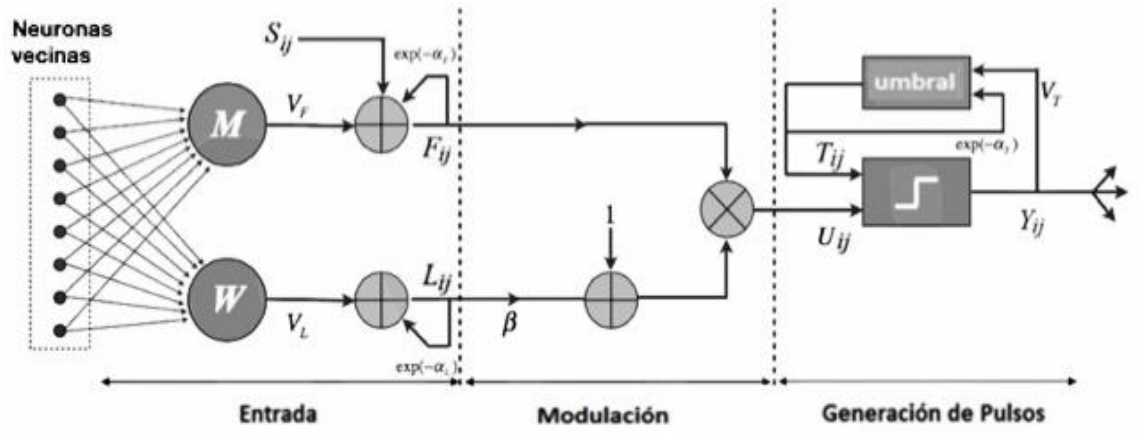


Fig. 2. Diagrama general de interconexiones de la red PCNN [14]

potencial interno de la neurona U_{ij}, que al ser comparado con el umbral produce salidas binarias Y_{ij}. Dichas señales se ven afectadas por factores de atenuación V_F, V_L, V_T y de amplificación $\alpha_F, \alpha_L, \alpha_T$; la imagen se va procesando de acuerdo al tamaño de las matrices M y W.

Estas características hacen que las neuronas correspondientes a pixeles vecinos con valores de intensidad similares se activen al mismo tiempo en ciertas regiones, a lo que se denomina activación de pulsos síncrona [23].

El modelo original de PCNN presenta algunas limitantes en la práctica debido al gran número de interconexiones entre neuronas y al establecimiento de parámetros de operación. Por este motivo se obtuvo un modelo iterativo simplificado de PCNN que permite extender el uso de estas redes en el procesamiento de imágenes. El modelo computacional iterativo ICM se describe como sigue (3)-(5) [21]:

$$F_{ij}[n] = fF_{ij}[n-1] + \sum w_{ijkl}Y_{kl}[n-1] + S_{ij} \quad (3)$$

$$T_{ij}[n] = gT_{ij}[n-1] + hY_{ij}[n-1] \quad (4)$$

$$Y_{ij}[n] = \begin{cases} 1 & if\ F_{ij}[n] > T_{ij}[n] \\ 0 & en\ otro\ caso \end{cases} \quad (5)$$

donde n es la iteración actual, w_{ijkl} es la matriz de pesos sinápticos que liga una neurona con sus vecinas y puede tener alguna de las siguientes formas:

$$w_{ijkl}(5\times 5) = \begin{bmatrix} 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.50 & 1.00 & 0 & 1.00 & 0.25 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \end{bmatrix}$$

$$w_{ijkl}(3\times 3) = \begin{bmatrix} 0.50 & 1.00 & 0.50 \\ 1.00 & 0 & 1.00 \\ 0.50 & 1.00 & 0.50 \end{bmatrix}$$

y finalmente f, g y h son coeficientes de ajuste, donde típicamente $g < 1.0$, $f < g$ y h es un valor grande, típicamente de entre 20 y 40.

ICM es un tipo de PCNN más simple (Fig. 3), donde no se considera *linking* en las neuronas y donde el *feeding* F_{ij} mantiene su salida con un factor de decaimiento dado por f.

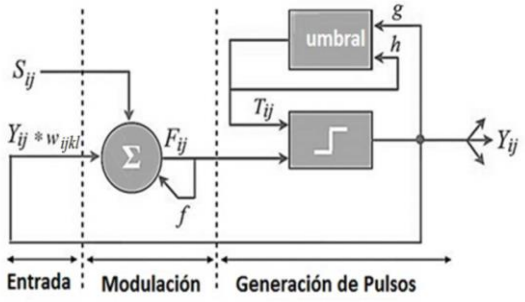


Fig. 3. Diagrama general de la red ICM [14]

La señal F_{ij} se compone por la última salida del vecindario de neuronas Y_{ij} ponderada por una matriz w_{ijkl} (generalmente Gaussiana) y por el estímulo externo de entrada S_{ij} (nivel de gris de cada pixel normalizado entre 0 y 1).

El umbral dinámico para cada pixel T_{ij} crece obedeciendo a h cuando su salida se activa y mantiene su estado previo con una atenuación dada por g, lo cual da origen a la formación de pulsos.

3. La matriz de tiempos (TM)

Las redes tipo PCNN pueden ser utilizadas para determinar la posición de los pixeles ruidosos con base en las neuronas activadas a la salida de la red por iteración. Al aplicar un operador de mediana que elimine los valores más altos y más bajos, es posible eliminar el ruido de Sal y Pimienta.

Para el filtrado del ruido Gaussiano es necesario emplear otra técnica ya que todos los pixeles han sido afectados en algún grado, por lo que se introduce la matriz de tiempos de las mismas dimensiones que la imagen a tratar. Esta matriz contiene información relacionada con la estructura espacial de la imagen, es decir, realiza un mapeo de la información espacial en una secuencia temporal [16].

La matriz de tiempos obtenida a partir de una red neuronal con interconexión nula PCNNNI [21] ha sido utilizada para detectar los pixeles ruidosos y procesada para reducir el ruido Gaussiano combinándola con otros métodos [17]. El modelo PCNNNI no considera la señal de *linking* y elimina la influencia de la matriz de pesos sinápticos, el potencial interno, umbral y salida de una neurona dependen únicamente de la intensidad del pixel correspondiente.

El método propuesto ICM-TM consiste en obtener la matriz de tiempos conservando la información de la relación espacial entre pixeles, de modo que las diferencias en la intensidad de los pixeles de la imagen originen diferencias en la secuencia de activación de sus respectivas neuronas (Fig. 4). Para almacenar el tiempo de activación de cada neurona se define una matriz M_{ij} que puede ser descrita como (6) [18].

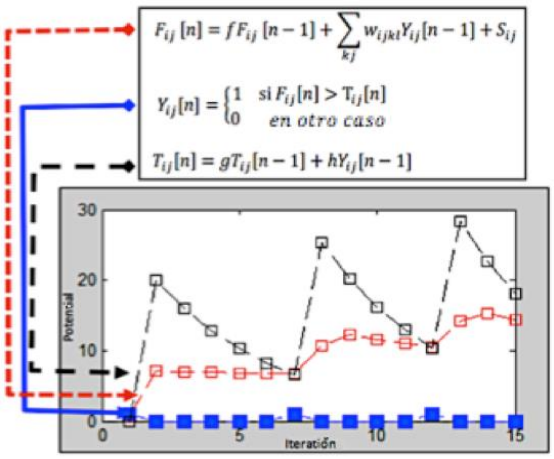


Fig. 4. Secuencias de activación de una neurona de la red ICM [14]

Fig. 5. Diagrama del algoritmo propuesto

$$M_{ij}[n] = \begin{cases} n & if\ Y_{ij}[n] = 1, \\ M_{ij}[n-1] & en\ otro\ caso. \end{cases} \quad (6)$$

El algoritmo ICM-TM mantiene el rango dinámico de la imagen y complementa el proceso de filtrado obteniendo la información de la matriz de tiempos para cualquier número de iteraciones, suavizando regiones de 5×5 pixeles con el filtro promedio para suavizar la imagen o bien conservando información de la imagen pero suprimiendo valores extremos mediante el operador de mediana.

4. Descripción del algoritmo ICM-TM propuesto

En nuestro método todas las neuronas están ligadas mutuamente del mismo modo y sus salidas Y_{ij} tienen dos estados posibles, activado

(1) y no activado (0). Este modelo es más rápido que PCNN debido a que implica menos ecuaciones y es más adaptable ya que tiene menos parámetros que ajustar. El proceso principal del algoritmo ICM-TM se lleva a cabo de manera secuencial en bloques (Fig. 5) y el algoritmo desglosado se describe en la Tabla 1.

Tabla 1. Algoritmo del método propuesto

Algoritmo ICM-TM
Inicialización:
Paso 1. Normalizar entre 0 y 1 los valores de gris de la imagen original con ruido $\boldsymbol{O_{ij}}$.
Paso 2. Inicializar los valores de las matrices $\boldsymbol{Y_{ij}}$, $\boldsymbol{S_{ij}}$, $\boldsymbol{T_{ij}}$, $\boldsymbol{F_{ij}}$ y $\boldsymbol{M_{ij}}$ en 0, sus dimensiones son las mismas que las de la imagen de entrada. Pasar los valores normalizados a la red por medio de $\boldsymbol{S_{ij}}$.
Llenado de la matriz de tiempos:
Paso 3. Calcular el valor de las matrices $\boldsymbol{F_{ij}}$, $\boldsymbol{T_{ij}}$, $\boldsymbol{Y_{ij}}$ de acuerdo a las ecuaciones 3, 4 y 5 en dicho orden.
Paso 3.a Cuando una neurona en la posición $(\boldsymbol{i,j})$ tome el valor de activada por primera vez $(\boldsymbol{Y_{ij}[n]=1})$ registrar el número de iteración actual $\boldsymbol{n}$ en la matriz de tiempos $\boldsymbol{M_{ij}[n]}$ en el lugar correspondiente al pixel procesado, omitiendo la primera iteración.
Paso 3.b Continuar iterando desde el Paso 3 hasta que todos los pixeles se hayan activado por lo menos una vez, es decir cuando todos los elementos en $\boldsymbol{M_{ij}[n]}$ sean distintos de cero.
Filtrado de la imagen:
Paso 4. Recorrer la imagen $\boldsymbol{O_{ij}}$ y la matriz $\boldsymbol{M_{ij}[n]}$ con una ventana cuadrada W de 5x5 pixeles, calcular la mediana $\boldsymbol{m_{ij}}$ dentro de $\boldsymbol{W}$ en $\boldsymbol{M_{ij}[n]}$ y aplicar el siguiente criterio:
Paso 4.a Si el tiempo de activación n de la neurona $(\boldsymbol{i,j})$ es igual a $\boldsymbol{m_{ij}}$, el valor del pixel $(\boldsymbol{i,j})$ en la imagen filtrada $\boldsymbol{Sdn_{ij}}$ será el valor resultante de aplicar el operador promedio $\boldsymbol{p_{ij}}$ sobre esta vecindad en $\boldsymbol{O_{ij}}$.
Paso 4.b Si no el valor del pixel $(\boldsymbol{i,j})$ de $\boldsymbol{Sdn_{ij}}$ será el resultante de aplicar el operador de mediana $\boldsymbol{m_{ij}}$ en $\boldsymbol{O_{ij}}$.

5. Experimentos y resultados

Las pruebas experimentales fueron realizadas con imágenes estándar en escala de grises de 255 valores de la base de datos SIPI [24], con tamaño de 512×512 pixeles.

Durante el proceso de experimentación, se introdujo a las imágenes ruido Gaussiano de media 0 y varianza entre 0.01 y 0.09.

Los otros parámetros fueron empíricamente seleccionados para la simulación.

La matriz interna de pesos formada con valores Gaussianos en función de la distancia seleccionada fue $w_{ijkl}(3\text{x}3)$. Las constantes: $f = 0.9, g = 0.8, h = 20$.

En el paso 4, el operador de mediana y el operador promedio están definidos de acuerdo a lo siguiente: si x_{ij} denota al pixel con coordenadas (i,j) en la imagen con ruido y X_{ij} denota el conjunto de pixeles en la ventana W con el vecindario $(2K+1) \times (2K+1)$ centrada en x_{ij}, entonces:

$$X_{ij} = \{x_{i-K,j-K}, \dots, x_{ij}, \dots, x_{i+K,j+K}\}. \quad (7)$$

La mediana de la ventana de la imagen se define como

$$m_{ij} = mediana\,(X_{ij}). \quad (8)$$

El promedio de la ventana de la imagen se define como

$$p_{ij} = promedio(X_{ij}). \quad (9)$$

El algoritmo ICM-TM se comparó contra filtros clásicos, el filtro de mediana de 3×3, el filtro Gaussiano , el filtro Sigma y el filtro Wiener de 3×3, y contra el PCNNNI basado en el algoritmo de Ma [21].

Se realizó la medición de los resultados obtenidos con tres métricas, las cuales están formuladas como sigue [19]:

a) $PSNR$ (Peak Signal to Noise Ratio, dB), el cual es utilizado para medir la habilidad de supresión del ruido. Mientras más grande sea su valor mejor es el efecto del filtrado:

$$PSNR = 10\,log_{10}\left(\frac{f(m,n)^2}{\frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}[f(m,n)-f'(m,n)]^2}\right). \quad (10)$$

b) MAE (Mean Absolute Error), indica la calidad del filtrado como la preservación de detalles finos, para lo que cual de ser minimizado:

$$MAE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}|f(m,n)-f'(m,n)|, \quad (11)$$

donde M y N denotan las filas y columnas de la imagen, $f(m,n)$ denota la imagen sin ruido y $f'(m,n)$ es la imagen resultante del proceso de filtrado.

c) $NMSE$ (Normalized Mean Square Error), un mejor método de filtrado debe generar un menor valor resultante de NMSE:

$$NMSE = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N}[f(m,n)-f'(m,n)]^2}{\sum_{m=1}^{M}\sum_{n=1}^{N}[f(m,n)]^2}. \quad (12)$$

De manera adicional también se midió el tiempo de cómputo, utilizando Matlab 2014 en Windows 7 Ultimate SP1, con un procesador Intel Core i7 a 3.40 GHz y 8GB de RAM.

6. Desempeño del método propuesto

Para la red ICM cada salida es distinta en cada iteración y la activación de las neuronas obedece a la relación entre los niveles de gris de la imagen con ruido Gaussiano.

En este caso de media 0 y varianza 0.01 (Fig. 6). La red puede iterarse cuantas veces se requiera. Para el caso de filtrado de ruido Gaussiano se requiere encontrar información sobre el grado de contaminación de cada pixel, por lo que se analizó la matriz de tiempos.

La información de esta matriz es numérica, por lo que se pueden distinguir los pixeles que se activan por primera vez en cada iteración y se extraen los pixeles que guardan mayor información de la imagen, discriminando a los otros para eliminar la mayor cantidad de ruido no deseado en la imagen.

Para dar un ejemplo numérico del procedimiento de filtrado considérese una matriz correspondiente a una imagen en escala de grises, contaminada con ruido Gaussiano de media 0 y varianza 0.08, de la que se tomó una ventana de 6x6 pixeles O_{ij} y su correspondiente imagen normalizada S_{ij}:

$$O_{ij} = \begin{pmatrix} 238 & 139 & 255 & 239 & 112 & 182 \\ 68 & 235 & 59 & 255 & 178 & 42 \\ 179 & 52 & 138 & 227 & 116 & 81 \\ 238 & 45 & 114 & 67 & 205 & 51 \\ 147 & 54 & 148 & 128 & 37 & 255 \\ 180 & 0 & 141 & 25 & 110 & 56 \end{pmatrix},$$

$$S_{ij} = \begin{pmatrix} 0.93 & 0.54 & 1.00 & 0.93 & 0.43 & 0.71 \\ 0.26 & 0.92 & 0.23 & 1.00 & 0.69 & 0.16 \\ 0.70 & 0.20 & 0.54 & 0.89 & 0.45 & 0.31 \\ 0.93 & 0.17 & 0.44 & 0.26 & 0.80 & 0.20 \\ 0.57 & 0.21 & 0.58 & 0.50 & 0.14 & 1.00 \\ 0.70 & 0 & 0.55 & 0.09 & 0.43 & 0.21 \end{pmatrix}.$$

Se introduce la imagen normalizada en la red ICM y se itera 10 veces, obteniéndose el umbral dinámico de dicha iteración T_{ij}, los valores en x son indistintos para este ejemplo, tal es el caso de los bordes de la imagen:

$$T_{ij} = \begin{pmatrix} 16.15 & 19.35 & 16.15 & 16.15 & 19.35 & x \\ 19.35 & 13.59 & 16.15 & 13.59 & 16.15 & x \\ 19.35 & 19.35 & 16.15 & 16.15 & 19.35 & x \\ 16.15 & 19.35 & 19.35 & 19.35 & 16.15 & x \\ 19.35 & 23.35 & 19.35 & 19.35 & 23.35 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

Luego se obtiene el potencial interno de la matriz F_{ij}:

$$F_{ij} = \begin{pmatrix} 9.31 & 8.47 & 11.48 & 11.07 & 5.92 & x \\ 6.79 & 13.76 & 8.87 & 14.04 & 9.60 & x \\ 9.71 & 8.96 & 11.15 & 13.38 & 7.92 & x \\ 11.45 & 9.09 & 10.80 & 9.51 & 10.61 & x \\ 7.09 & 6.65 & 9.20 & 8.64 & 4.18 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

Si el potencial interno de la neurona (ij) supera al valor de umbral, entonces se produce una salida en 1 para esa neurona.

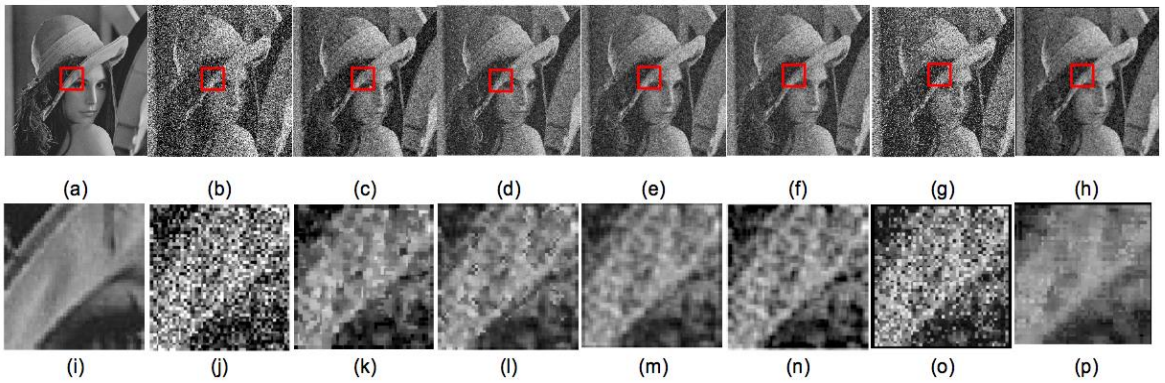


Fig. 7. Comparación entre técnicas de filtrado a) Lena original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiene*r*, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

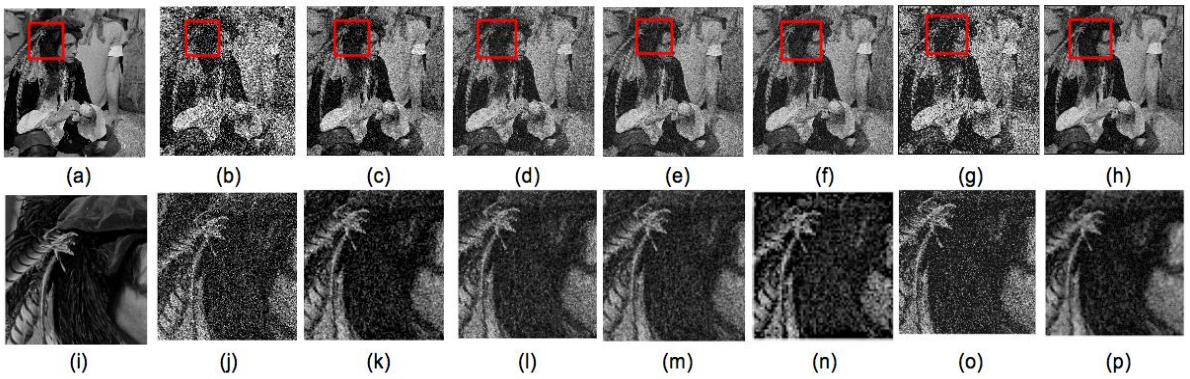


Fig. 8. Comparación entre técnicas de filtrado a) Man original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

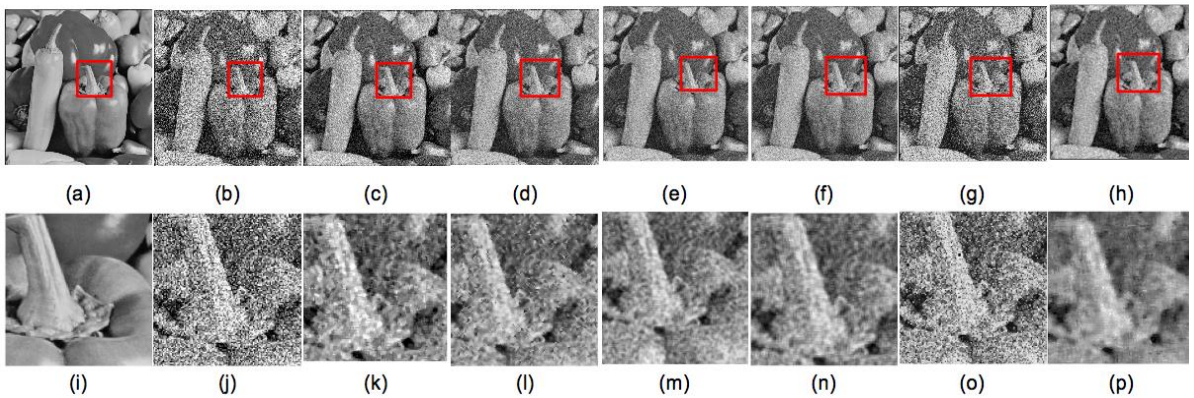


Fig. 9. Comparación entre técnicas de filtrado a) Peppers original sin ruido, b) con ruido Gaussiano var. 0.09, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

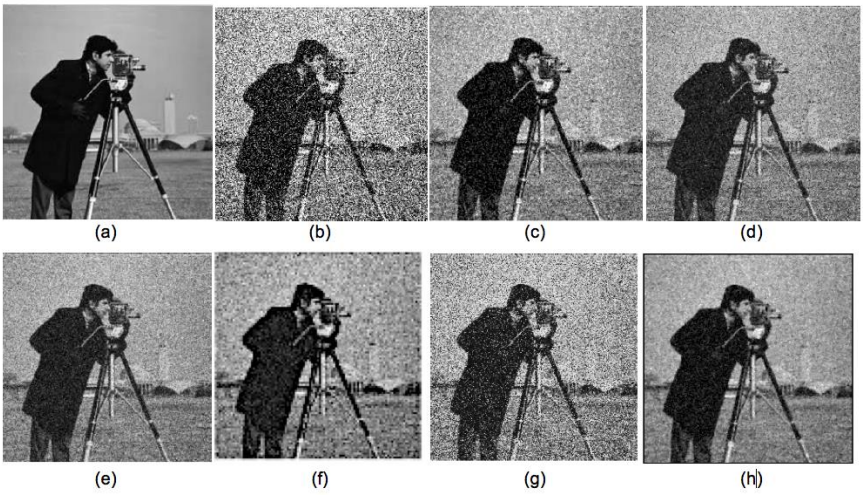


Fig. 10. Comparación entre técnicas de filtrado a) cameraman original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

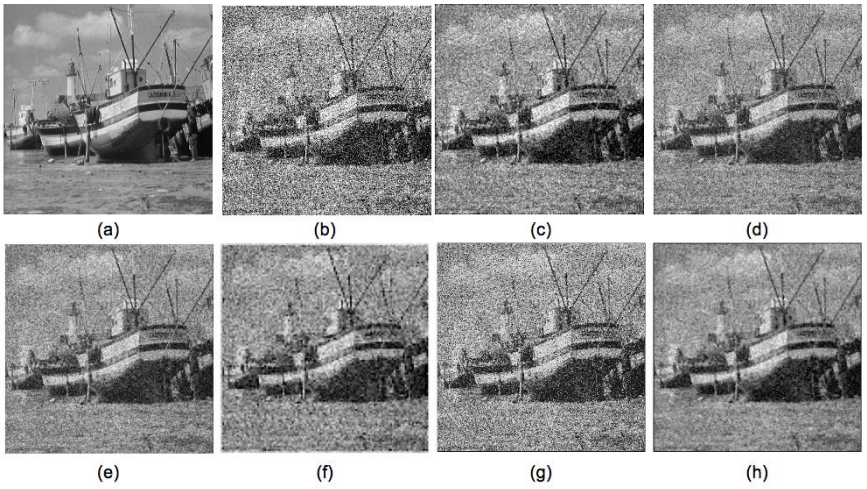


Fig. 11. Comparación entre técnicas de filtrado a) boat original sin ruido, b) con ruido Gaussiano var. 0.09, c) mediana, d) *Wiener*, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

Al terminar de procesar la imagen se tiene una matriz binaria de salidas $\mathrm{Y_{ij}}$.

$$Y_{ij} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

El número de iteración en el que sucede la primera activación de cada neurona se guarda en la matriz de tiempos final $\mathrm{M_{ij}}$:

$$M_{ij} = \begin{pmatrix} 7 & 8 & 7 & 7 & 8 & x \\ 8 & 6 & 7 & 6 & 7 & x \\ 8 & 8 & 7 & 7 & 8 & x \\ 7 & 8 & 8 & 8 & 7 & x \\ 8 & 9 & 8 & 8 & 9 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

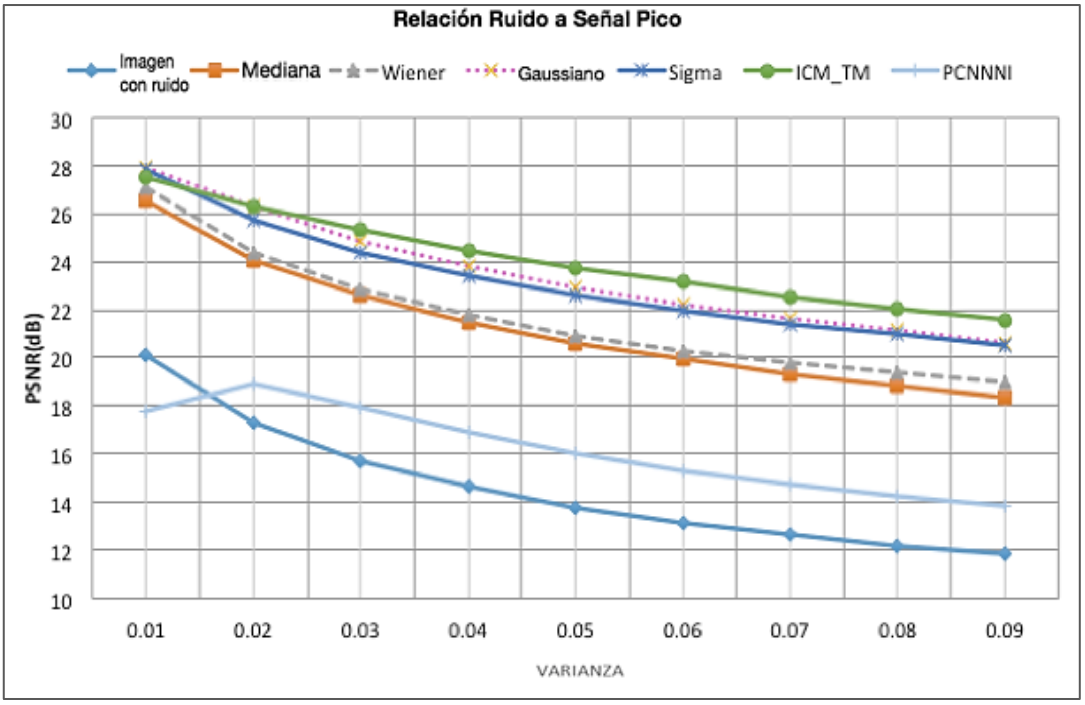


Fig. 12. Curvas de desempeño en PSNR de los filtros de ruido Gaussiano

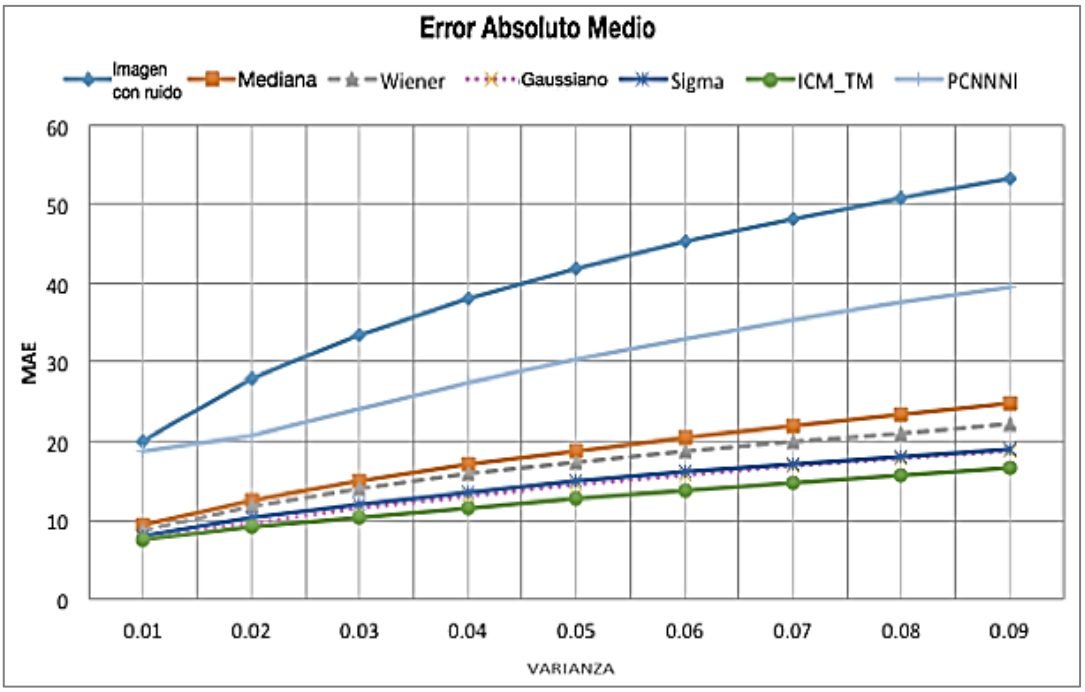


Fig. 13. Curvas de desempeño en MAE de los filtros de ruido Gaussiano

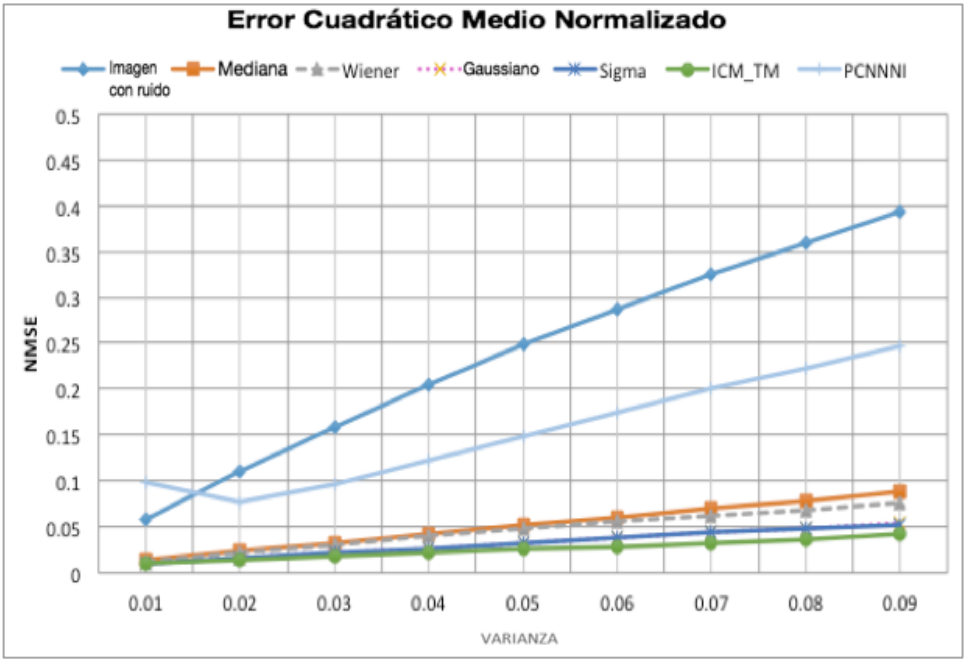


Fig. 14. Curvas de desempeño en NMSE de los filtros de ruido Gaussiano

Tabla 2. PSNR para imágenes con ruido Gaussiano de varianza 0.08

PSNR (dB)	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	ICM-TM
Lena	12.20	18.81	19.39	21.15	20.96	14.27	22.07
Baboon	11.96	17.42	18.54	19.42	19.14	13.91	19.05
Caman.	12.24	19.07	19.02	20.78	20.60	14.27	22.34
Peppers	12.16	18.79	19.32	21.05	20.86	14.21	22.30
Barbara	13.66	20.11	18.58	19.96	19.76	15.62	22.05
Cerebro	12.97	20.10	17.88	19.64	19.45	14.66	22.27
5.1.13	13.96	18.30	16.88	16.25	15.94	8.24	15.80
Lunetas	12.34	18.78	19.70	21.43	21.24	14.59	22.37
Man	12.37	19.02	18.70	20.41	20.21	14.32	21.72
Boat	12.02	18.52	19.18	20.73	20.51	14.04	21.44
Couple	13.36	19.92	18.40	19.94	19.74	15.24	21.60
Grass	12.54	15.53	16.25	16.25	15.91	13.92	13.93
House	12.07	18.83	19.57	21.24	21.04	14.14	22.12
Mountain	13.61	19.92	18.59	19.95	19.75	15.58	21.92

La etapa de filtrado selectivo consiste en recorrer las matrices calculando la mediana de los valores de la matriz de tiempos en una ventana de 5×5 denotado como $mediana\ M_{2,2}(5 \times 5)$ y comparar el valor con el correspondiente al pixel central $M_{2,2}$ de la ventana considerada, cuyos valores respectivos son:

Tabla 3. PSNR para imágenes con ruido Gaussiano de varianza 0.09

PSNR (dB)	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	ICM-TM
Lena	11.81	18.32	18.97	20.63	20.49	13.84	21.59
Baboon	11.60	17.08	18.26	19.13	18.91	13.54	18.89
Caman.	11.89	18.63	18.67	20.36	20.22	13.93	21.93
Peppers	11.79	18.32	18.96	20.61	20.47	13.84	21.85
Barbara	13.17	19.66	18.03	19.32	19.18	15.07	21.62
Cerebro	12.57	19.65	17.49	19.13	18.99	14.24	21.77
5.1.13	13.44	17.99	16.39	15.85	15.60	8.70	15.59
Lunetas	12.00	18.42	19.38	20.98	20.84	14.20	21.94
Man	11.98	18.56	18.32	19.93	19.78	13.91	21.29
Boat	11.64	18.09	18.89	20.34	20.18	13.67	21.11
Couple	12.88	19.43	17.90	19.30	19.15	14.68	21.11
Grass	12.14	15.28	16.01	16.01	15.72	13.59	13.84
House	11.73	18.26	19.18	20.76	20.62	13.81	21.66
Mountain	13.13	19.55	18.04	19.35	19.20	15.04	21.51

Tabla 4. PSNR de los filtros de ruido Gaussiano en *Lena 512x512*

Desviación estándar	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	BM3D [25]	ICM-TM
5	34.00	32.86	35.00	32.48	32.22	10.37	38.72	29.36
10	28.28	31.35	33.04	31.63	31.41	11.83	35.93	29.01
15	24.62	29.64	30.82	30.42	30.26	13.83	34.27	28.60
20	22.16	28.09	28.89	29.21	29.10	15.82	33.05	28.11
25	20.34	26.69	27.23	28.06	27.98	17.60	32.08	27.60
30	18.81	25.47	25.85	26.96	26.91	18.80	31.26	27.00
35	17.55	24.29	24.62	25.96	25.93	18.93	30.56	26.40
40	16.49	23.31	23.57	25.04	25.02	18.54	29.86	25.79
45	15.57	22.45	22.71	24.30	24.28	17.81	29.05	25.24
50	14.78	21.64	21.94	23.55	23.54	17.05	28.27	24.59

$$mediana\ M_{2,2}(5 \times 5) = 8,$$

$$M_{2,2} = 7.$$

Si estas cifras son iguales, el valor del pixel correspondiente en la imagen filtrada $Sdn_{2,2}$ será el promedio de los valores en esta ventana en la imagen ruidosa $promedio\ O_{2,2}(5 \times 5)$, en caso contrario el pixel correspondiente tomará el valor de la mediana de la vecindad de 5×5 de la imagen con ruido. Se procede a calcular la mediana de la ventana en la imagen con ruido $\text{mediana}\ O_{2,2}(5 \times 5)$ y asignarlo al pixel de la imagen filtrada $Sdn_{2,2}$. El proceso termina cuando se ha procesado la imagen completa:

$$mediana\ O_{2,2}(5 \times 5) = 138,$$

$$Sdn_{2,2} = 138.$$

Se realizó la comparación del método propuesto con los filtros de mediana, Wiener, Gaussiano, Sigma y el basado en PCNNNI para la imagen de *Lena* (Fig. 7) a la que se agregó ruido Gaussiano con varianza 0.08; se observó que la capacidad de suavizado del filtro de mediana puede ser empleado para disminuir el ruido Gaussiano que genera valores extremos en los pixeles, no obstante, no se debe aplicar este filtro de manera uniforme pues la imagen se verá afectada en detalles y bordes. Para otras regiones se debe tomar en cuenta la información presente en los pixeles por medio de un suavizado más fino como es el generado por el operador promedio.

El análisis cualitativo de la imagen ampliada hace evidente que el filtro Wiener mantiene un alto grado de ruido aunque preserva los bordes.

El filtro Gaussiano preserva bordes, disminuye el ruido al suavizar la imagen pero la imagen no es nítida.

El filtro Sigma recupera la imagen en buen grado preservando los bordes de modo claro pero el ruido aún es notorio, mientras que el método PCNNNI no elimina la mayor parte del ruido presente en la imagen. Finalmente el método propuesto muestra un mayor efecto de suavizado, los bordes mejor delimitados y el ruido menos presente.

Se repitió la prueba con distintas imágenes obteniéndose resultados similares como puede observarse para la imagen de *Man* (Fig. 8) con ruido Gaussiano de varianza 0.08 y para la imagen *Peppers* (Fig. 9) con varianza de 0.09, donde se ve en la imagen completa y el detalle ampliado que el método propuesto ICM-TM genera una mejor recuperación de la imagen con ruido.

Otros experimentos con la imagen de *cameraman* (Fig. 10) con ruido Gaussiano de varianza 0.08 y con la imagen de *boat* (Fig. 11) con varianza de 0.09 muestran que el método propuesto ICM-TM genera imágenes recuperadas visualmente con mayor calidad que otras técnicas tradicionales de filtrado y técnicas basadas en redes neuronales pulsantes.

El análisis cuantitativo de la gráfica de PSNR de las distintas técnicas comparadas cuando la varianza va de 0.01 y 0.09 (Fig. 12) permite mostrar que la supresión del ruido del método propuesto ICM-TM recupera la imagen en promedio por 9.6dB y es superior en promedio por 1.2dB a los métodos tradicionales de filtrado Sigma y Gaussiano, por 2.4dB al filtro de medina y Wiener, y por 7.8dB a la técnica basada en PCNNNI, técnica que se distingue por su desempeño no lineal.

El método propuesto muestra un buen desempeño para varianza de ruido Gaussiano de 0.02 y hasta 0.09. Si la varianza es menor a 0.02 el método propuesto recupera la imagen por 7.6dB, aunque ya no logra superar a los otros filtros.

Se determinó el MAE de las distintas técnicas (Fig. 13), la imagen con menor error es la que se obtiene por medio del método propuesto, la disminución del error es de 28 puntos.

El NMSE (Fig. 14) permite observar que el método propuesto ICM-TM tiende a minimizar esta métrica por debajo de los otros métodos Conforme la varianza del ruido aumenta el error de la imagen contaminada y la capacidad de recuperación de los filtros disminuye.

Los experimentos con distintas imágenes a las que se agregó ruido Gaussiano de varianza 0.08, muestran la capacidad de los métodos para recuperar la imagen del ruido en valores de PSNR (Tabla 2).

Se repitió el experimento con ruido Gaussiano de varianza 0.09 (Tabla 3). Los resultados en PSNR permiten observar la recuperación para cada método donde destaca la técnica ICM-TM.

Se realizaron experimentos para varianzas de ruido Gaussiano de 0.003 a 0.04 que corresponden a una desviación estándar de entre 5 y 50 (Tabla 4). Se muestra que cuando la varianza es baja el filtro Wiener es el que obtiene un mejor resultado y cuando la varianza el método propuesto ICM-TM es el de mejor desempeño; no obstante la técnica BM3D muestra superioridad en PSNR para este rango de ruido filtrado.

7. Conclusiones y trabajo futuro

Los resultados obtenidos muestran que el método de filtrado ICM-TM propuesto supera en desempeño a los filtros tradicionales como filtro Gaussiano, Sigma y Wiener en promedio por 1.2 dB según PSNR a 0.02 y por 28 puntos en MAE y por 0.2 en NMSE para varianzas superiores. Permite relacionar vecindades de pixeles y no requiere cálculos estadísticos ni complejos, del mismo modo aprovecha las características de los filtros promedio y mediana al aplicarlos selectivamente; mientras que el método BM3D supera a los métodos comparados en PSNR, para el rango de varianza dado, y es de mayor complejidad.

Por otro lado ICM-TM supera a PCNNNI [18] en promedio por 7.8dB, y requiere 80% menos tiempo de procesamiento, además no se realiza un cambio del rango dinámico de la imagen, se

considera la relación entre neuronas vecinas y se tienen que ajustar menos parámetros.

En cuanto al tiempo de procesamiento el promedio para PCNNNI fue de 160s, para el filtro de Mediana de 0.028s, para el filtro Wiener de 0.036s, para el método BM3D de 5.8s, para ICM-TM de 20s, por lo que la complejidad del método es acorde con el tiempo de procesamiento aunque éste puede ser mejorado mediante procesamiento en paralelo.

El trabajo futuro se centrará en la exploración de PCNN con variantes y la elección de los coeficientes de ajuste por medio de algoritmos evolutivos, así como en la búsqueda de un operador que permita un mejor ajuste de los valores de la imagen para preservarlos detalles, el estudio profundo de la matriz de tiempos y tomando en cuenta las frecuencias para la localización de los pixeles que guardan mayor información de la imagen discriminando aquellos que presentan mayor grado de ruido.

También se pretende estudiar el comportamiento del filtro como sistema dinámico y su desempeño sobre algunas imágenes artificiales que se enfoca a características específicas como bordes y líneas. Finalmente una vez que se haya mejorado el algoritmo se llevarán a cabo comparaciones con otras técnicas de filtrado y por medio de criterios como la similitud estructural (*ssim)*.

Agradecimientos

Los autores agradecen al CENIDET y al IPN por el apoyo económico para la realización de la presente investigación con ayuda de los siguientes fondos: SIP-IPN 20170693, y CONACYT en el marco del proyecto 65 en el marco de la convocatoria de Investigación Fronteras de la Ciencia 2015. Estela Ortiz agradece al CONACYT por la beca concedida para la realización de sus estudios de maestría.

Referencias

1. **Bovik, A. (2009).** *The Essential Guide to Image Processing*. Elsevier.
2. **González, R. C. (2002).** *Digital Image Processing*. Prentice Hall.
3. **Pratt, W. (1972).** Generalized Wiener Filtering Computation Techniques. *IEEE Transactions on Computers,* Vol. C-21, pp. 636–641.
4. **Kaur, D. (2015).** Remove Noise Effects From Degraded Document Images Using Matlab Algorithm. *International Journal of Engineering Sciences & Research Technology,* Vol. 4, No. 9, pp. 544–549.
5. **Kang, D., & Lim, H. (2013).** *Efficient noise reduction in images using directional modified sigma filter*. Springer Science Business Media New York*,* pp*. 580–592.*
6. **Panetta, K., Bao, L. & Agaian, S. (2016).** Sequence-to-Sequence Similarity Based Filter for Image Denoising. *IEEE Sensors Journal.*
7. **Liu, J., Wang, Y., Su, K., & He, W. (2016).** Image denoising with multidirectional shrinkage in directionlet domain. *Signal Processing,* Vol. 125, pp. 64–78.
8. **Guleryuz, O. G. (2007).** Weighted averaging for denoising with overcomplete dictionaries. *IEEE Trans. Image Process,* Vol. 16, pp. 3020–3034*.*
9. **Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007).** Image denoising by sparse 3D transform-domain collaborative filtering*. IEEE Trans. Image Process,* Vol. 16, pp. 2080–2095*.*
10. **Elad, M. & Aharon, M. (2006).** Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. on Image Process*, Vol. 15, pp. 3736–3745.
11. **Palacios, A. E. & Ponomaryov, V. (2016).** Image Denoising using Block Matching and Discrete Cosine Transform with Edge Restoring. *IEEE Conference Proceedings*, pp. 140–147.
12. **Kumar, A. & Singh, B. (2015).** Alexander Fractional Integral Filtering Of Wavelet Coefficients for Image Denoising. *Signal & Image Processing : An International Journal (SIPIJ),* Vol. 6, No. 3, pp. 43–54.
13. **Yang, S. & Min, W. (2013).** Image Noise Reduction via Geometric Multiscale Ridgelet Support Vector Transform and Dictionary Learning*. IEEE Transactions on Image Processing,* Vol. 22, pp. 4161–4169*.*
14. **Lindblad, T. & Kinser, J.M. (2005).** *Image processing using Pulse-Coupled Neural Networks*. Springer.

15. **Eckhorn, R., Reitboeck, H.J., & Arndt, M. (1990).** Feature linking via synchronization among distributed assemblies: simulation of results from cat cortex. *Neural Computation,* pp. 293–307.
16. **Ranganath, H.S., Kuntimad, G., & Johnson, J.L. (1995).** Pulse coupled neural networks for image processing. *Proceedings of IEEE Southeast Conference,* Raleigh, pp. 26–29.
17. **Ma, Y.D., Shi, F., & Li, L. (2003).** Gaussian noise filter based on PCNN. *Proceedings of International Conference on Neural Networks and Signal Processing,* Nanjing.
18. **Ma, Y.D., Lin, D.M., & Zhang, B.D. (2007).** A novel algorithm of image Gaussian noise filtering based on PCNN time matrix. *Proceedings of IEEE International Conference on Signal Processing and Communication,* Dubai.
19. **Lui, C. & Zhang, Z. (2008).** Sonar images de-noising based on pulse coupled neural networks. *Congress on Image and Signal Processing*, pp. 403–406.
20. **Yuan-yuan, C. & Hai-yan, L. (2011).** A new method of denoising mixed noise using Limited Grayscale Pulsed Couple Neural Network. *Cross Strait Quad-Regional Radio Science and Wireless Technology Conference,* pp. 1410–1413.
21. **Ma, Y.D., Zhan, K., & Wang, Z. (2010).** *Applications of pulse-coupled neural networks*. Springer.
22. **Zhang, D. & Nishimura, T. (2009).** Pulse Coupled Neural Network Based Anisotropic Diffusion Method for I/f Noise Reduction. *IEEE Transactions on Neural Networks.*
23. **Johnson, J. L. & Padgett M. L. (1999).** PCNN model and applications. *IEEE Transactions on Neural Networks*, Vol. 10, 480–498.
24. **http://sipi.usc.edu/database.**
25. **Matlab toolbox para BM3D.** Disponible en http://www.cs.tut.fi/~foi/GCF-BM3D/

Artículo recibido el 06/12/2016; aceptado el 26/04/2017.
Autor de correspondencia es Estela Ortiz Rangel.