Taxonomy-Based Feature Extraction for Document Classification, Clustering and Semantic Analysis

Sattar Seifollahi[1,2] and Massimo Piccardi[1]

[1] University of Technology Sydney, NSW, Australia
[2] Capital Markets Cooperative Research Centre (CMCRC),
Sydney, NSW, Australia

Abstract. Extracting meaningful features from documents can prove critical for a variety of tasks such as classification, clustering and semantic analysis. However, traditional approaches to document feature extraction mainly rely on first-order word statistics that are very high dimensional and do not capture well the semantic of the documents. For this reason, in this paper we present a novel approach that extracts document features based on a combination of a constructed word taxonomy and a word embedding in vector space. The feature extraction consists of three main steps: first, a word embedding technique is used to map all the words in the vocabulary onto a vector space. Second, the words in the vocabulary are organised into a hierarchy of clusters (word clusters) by using k-means hierarchically. Lastly, the individual documents are projected onto the word clusters based on a predefined set of keywords, leading to a compact representation as a mixture of keywords. The extracted features can be used for a number of tasks including document classification and clustering as well as semantic analysis of the documents generated by specific individuals over time. For the experiments, we have employed a dataset of transcripts of phone calls between claim managers and clients collected by the Transport Accident Commission of the Victorian Government. The experimental results show that the proposed approach has been capable of achieving comparable or higher accuracy than conventional feature extraction approaches and with a much more compact representation.

1 Introduction

The recent years have witnessed an incessant growth in the creation of digital text, from the increasing number of organisational documents and workflows to the large amounts of messages continuously generated on social media. As an example, the number of tweets generated on the popular Twitter platform is estimated to have reached over 200 billion per year. The immediate challenge stemming from such a huge growth in textual data is how to understand their contents in effective and efficient ways.

Document classification (or categorisation) is likely the most widespread automated task on textual data, with a vast array of applications such as sentiment analysis, ad targeting, spam detection, client relationships, risk assessment and medical diagnosis. A class is a subset of documents which are, in some sense, similar to one another and different from those of other classes, and the goal is to assign text documents to a predefined set of classes. Document classification has been extensively studied, especially since the emergence of the Internet where documents are typically created by an unverified variety of authors and with little metadata.

A major issue for effective document classification is the extraction of appropriate features and document representations. Techniques using the bag-of-words (BoW) model are the most widespread [2], with an early reference in a linguistic context dating back to 1954 [13]. In this model, a text (such as a sentence or a document) is represented as the bag or multiset of its words, disregarding the word order but retaining the word counts. The entire vocabulary used in the document corpus is considered as the feature space, and each document is represented by the vector of its word frequencies. Given that most documents only utilize a small subset of the available words, such feature vectors tend to be very sparse and unnecessarily high-dimensional. Such a high dimensionality can be regarded as an instance of the *curse of dimensionality* [12], making it difficult for clustering and classification algorithms to perform effectively. In addition, the BoW model ignores the linguistic interaction between words and does not account for word ordering [7], while the meaning of natural languages deeply depends on them. It is also very common to reweigh these vectors to somehow reflect the “discriminative power” of each word. The term frequency-inverse document frequency (tf-idf) approach is the most popular weighting scheme for the BoW model [23]. The tf-idf weighting increases the importance of words that appear rarely in the corpus, assuming that they would be more discriminative in any ensuing clustering and classification tasks [4, 10, 22].

A promising solution for mollifying the BoW flaws is to exploit an ontology (an organised set of concepts in the domain area) [9, 11, 14]. This can both lead to a dramatic decrease in the number of features and help incorporate semantic knowledge [7, 9, 11]. We refer the reader to [3] for further details on ontology learning and applications. However, how to best exploit ontologies to represent documents is still an open problem. Most of the proposed techniques only employ existing semantic lexical databases [9, 11] such as WordNet to organise the documents’ words and determine similarities. However, if the ontology does not suit the collection of documents, the extracted features may result in poor performance, not only in classification, but also in clustering, semantic analysis and other tasks. An example of challenging text data are the phone call transcripts used for the experiments in this paper (3.1). The transcripts originate from phone conversations between claim managers and clients of an accident support agency, and reflect a very specialised terminology and a diversity of transcription styles. These documents represent a challenge for existing, general-purpose ontologies and suggest exploring dedicated approaches for effectively incorporating semantic aspects in the document features.

Recently, text analytics have been extensively benefiting from the adoption of word "embeddings" that map words from their intrinsic categorical values to vector spaces [19, 21]. These models take into input a large corpus of documents and embed each distinct word in a vector space of low-to-moderate dimensionality (typically, a few hundred dimensions, compared to the $\approx 10^5 - 10^6$ of BoW). Word embeddings overcome the typical drawbacks of the BoW model, including the issue of high dimensionality and the dismissal of the word's order in the text [18]. These models have been widely used in the literature for document clustering and classification [1, 15–18, 25, 27–30]. For these reasons, in this paper we propose leveraging a dedicated taxonomy (an instance of an ontology) built upon a word embedding space. This approach had been originally proposed for document clustering in [24], where it had demonstrated its effectiveness at grouping documents into consistent clusters. In this paper, we extend it to classification and semantic analysis to demonstrate its effectiveness also for these tasks. We also leverage sets of predefined "words of interest" for the specific organisation to greatly decrease the number of features to a very manageable size, equal to the size of the set of predefined words. The feature extraction algorithm uses a three-step process: first, a word embedding technique is used to convert each distinct word to a vector of $|W|$ dimensions. Second, the word vectors are partitioned into a hierarchy of clusters, simply referred to as "word clusters", via a hierarchical clustering algorithm. Third, the individual documents are projected onto a space of predefined words whose size is equal to the size of this set. Our ultimate goal is to develop a versatile feature extraction technique with a strong focus on dimensionality reduction that can assist with a variety of tasks, from phone call classification and clustering to the monitoring of progress of individual clients.

2 Methodology

This section describes the main components of our methodology, namely i) the approach for generating the hierarchy of word clusters, ii) the approach for extracting taxonomy-based features based on a set of predefined words, and iii) a comparison approach that generates the features directly from the clusters in the hierarchy.

2.1 Hierarchy of word clusters

In this subsection, we describe a method for partitioning words into a hierarchy of clusters. In the literature, there exist two main lines of methods for clustering, namely partitional and hierarchical algorithms (please refer to [20] for further discussion on clustering techniques and their convergence properties). The k-means algorithm and its variants are the most broadly used partitional algorithms [2, 8, 23, 24, 26]. In turn, hierarchical algorithms can be divided into two main categories, namely agglomerative and divisive. Agglomerative algorithms generally perform better than divisive algorithms, and often "better" than single-layer algorithms such as k-means [20]. These algorithms exploit a bottom-up approach, i.e. the clustering produced at each layer of the hierarchy merges similar clusters from the previous layer. However, [26] showed that *bisecting k-means* can produce clusters that are both better than those of standard k-means and as good as (or better than) those produced by agglomerative hierarchical clustering. For these reasons, in this paper we use a method similar to bisecting k-means with two modifications; 1) we use spherical k-means instead of standard k-means, and 2) each cluster is split into two sub-clusters only if the number of its elements exceeds a predefined threshold. Figure 1 illustrates the hierarchical word clusters. For details and steps of the algorithm, we refer readers to the recent paper [24].

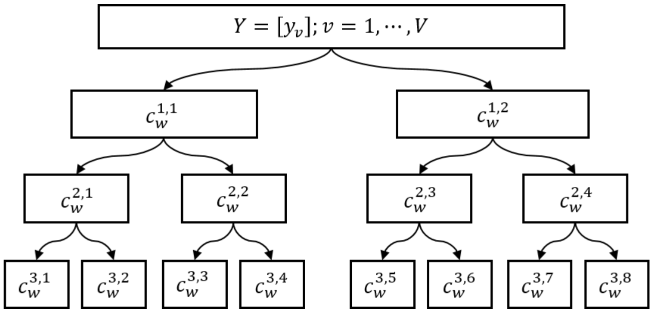


Fig. 1. Three layers of the hierarchy of word clusters.

2.2 Taxonomy-augmented features given a set of predefined words

A taxonomy can play a key role in document clustering by reducing the number of features from typically thousands to a few tens only. In addition, the feature reduction process benefits from the taxonomy's semantic relations between words. In [24], the authors presented an approach for taxonomy-based feature extraction and proved its usefulness for document clustering. In this approach, the individual documents are projected to a space of predefined words, with the feature dimensionality equal to the number of such words. We believe that the approach can also prove useful for other tasks such as document classification and semantic analysis, where features are extracted from the documents (e.g., phone call transcripts) that the individuals produce over time to monitor their progress and tailor interventions.

The approach is briefly recapped here for ease of reference; for further details please refer to [24]. To describe the process precisely, let us assume that $D = [d_1, \ldots, d_M]$ is the document matrix, where $d_i \in \mathbb{R}^N$ and N is the number of the predefined words. With these positions, the steps for generating the document features are described by Algorithm 1.

Algorithm 1 Taxonomy-augmented features given a set of predefined words

```
Input: Hierarchy of word clusters $C_w$, set of predefined words $S = \{w_1, \dots, w_N\}$
Output: Documents $D \in \mathbb{R}^{M \times N}$

1. Set $D = [d_1, \dots, d_M] = 0$, where $d_i \in \mathbb{R}^N$ and $N$ is the size of the set of predefined
   words. Set $W_s = [w_s^1, \dots, w_s^L] = \big[[w_s^{1,1}, w_s^{1,2}], \dots, [w_s^{L,1}, \dots, w_s^{L,2^L}]\big]$, and $w_s^{l,j} = \emptyset$.
2. For each word $w$ in $S$ :
   2.1 For level $l = 1, \dots, L$ :
       2.11 Find index of cluster in $C_w^l$, j, such that $w \in C_w^{l,j}$.
       2.12 Set $w_s^{l,j} = w_s^{l,j} \cup w$.
3. For each document $d_i, i = 1, \dots, M$ :
   3.1 For each word $w$ in document $d_i$ :
       3.11 For level $l = 1, \dots, L$ :
            3.111 Find cluster index in $C_w^l$, i.e. $j$ such that $w \in C_w^{l,j}$.
            3.112 Retrieve all words of $w_s^{l,j}$ with corresponding indices in $S$, $I(l, j)$.
            3.113 Set $d_i^{I(l,j)} = d_i^{I(l,j)} + x_{i,w}$.
```

2.3 Taxonomy-augmented features given the hierarchy of word clusters

In this section, we review another method from [24] solely for the sake of comparison. The approach is largely similar to that introduced in the previous subsection; however, it differs in its final stage where the documents are projected directly onto the word clusters rather than the set of predefined words. We note that this approach is not suitable for what we call semantic analysis in this paper since the features are not representative of any "words of interest" for the domain experts. We briefly describe the algorithm's steps here and refer the interested readers to [24] for further details. Let us assume that $D = [d_1, \dots, d_M]$ are M documents in word space. Each document can be further noted as $d_i = [d_i^1, d_i^2, ..., d_i^L]$, where L is the number of levels in the hierarchy of word clusters and $d_i^l = [d_i^{l,1}, d_i^{l,2}, ..., d_i^{l,2^l}]$. For simplicity, we store D in a two-dimensional matrix of size $M \times N$, where N is the overall number of clusters in all levels, i.e. $N = \sum_{l=1}^{L} 2^l$.

3 Experiments

3.1 Datasets

To test and compare the proposed approach, we have carried out experiments with textual data from an accident support agency, the Transport Accident Commission (TAC) of the Victorian Government in Australia. The data consist of phone calls between the agency's clients and its claim managers, annotated by the managers into transcripts. Phone calls for single clients take place over time, so the data are suitable for monitoring the clients' progress, such as return to work and physical and emotional recovery. The phone calls typically cover a

Algorithm 2 Taxonomy-augmented features

```
Input: Hierarchy of word clusters $C_w$
Output: Document matrix $D \in \mathbb{R}^{M \times N}$

1. Set $D = [d_1, \dots, d_M] = 0$ where $d_i \in \mathbb{R}^N$, $N = \sum_{l=1}^{L} 2^l$, and $L$ is the number of
   layers in the hierarchy of word clusters.
2. For each document $d_i, i = 1, \dots, M$:
  2.1. For each word $w$ in document $d_i$:
    2.1.1. For level $l = 1, \dots, L$:
      2.1.1.1. Find cluster index in $C_w^l$, $j$, such that $w \in C_w^{l,j}$.
      2.1.1.2. Set $d_i^{l,j} = d_i^{l,j} + x_{iw}$.
  2.2. For $l = 1 \dots L$:
    2.2.1. Normalize $d_i^l$ to one, i.e. $\|d_i^l\| = 1$.
3. Remove any features from $D$ that have zero value across all documents.
```

wide range of topics: for example, some are related to the client's health and recovery, while others are related to payments and compensation.

To conduct experiments on classification, we have merged all the phone call transcripts of each individual client into single documents and created four datasets (D1-4) based on the number of words per document and the number of documents itself. Datasets D1 and D2 consist, respectively, of 2,000 and 5,000 short-length documents, ranging from 20 words to 100 words per document. The TAC experts have indicated that such short-length documents are likely to come from clients with fewer phone calls overall and more rapid recovery. Datasets D3 and D4 are similar, but with larger-size documents ranging from 100 to 5,000 words each. The four datasets have been manually annotated by the TAC experts into a binary classification problem using labels "MH" and "NO MH" (label "MH" is used by the TAC to identify clients with a variety of mental health issues). All datasets are balanced in terms of number of samples per class.

The following preprocessing steps have been applied to each phone call before its use in the experiments: 1) removal of numbers, punctuation, symbols and "stopwords"; 2) replacement of synonyms and misspelled words with the base and actual words; 3) removal of sparse terms (keeping 95 % sparsity or less) and infrequently occurring words; 4) removal of uninformative words such as people names and addresses.

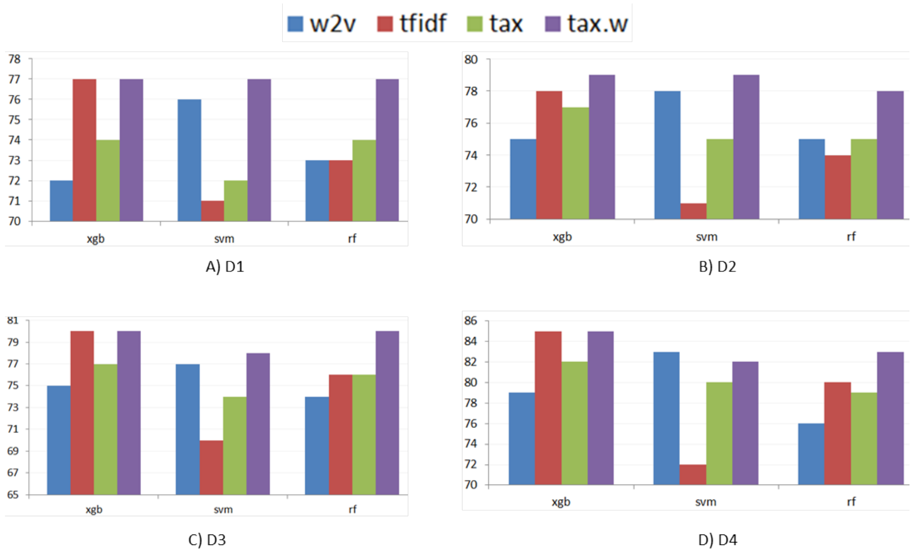
3.2 Experimental set-up

To learn the word embeddings, we have used the following settings: the dimensionality was set to 100; the context window size was set to 12; and the number of training epochs was set to 1,000. In Algorithm 1, we have set $N = 100$.

For the experiments on document classification we have employed two document features as the baselines for comparison: 1) the well-known tf-idf and 2) doc2vec which is based on the average of word embeddings. We have also used Algorithm 2 as another method for comparison. As word embeddings, we have used GloVe due to its reported strong performance in a variety of tasks [21]. For classification, we have compared three popular classifiers: 1) eXtreme Gradient Boosting (XGBoost) [6], 2) a support vector machine (SVM) with a radial basis function kernel, and 3) a random forest. We have used 10-fold cross-validation to report the accuracy: first, the dataset is randomly shuffled and split into 10 folds; then, in turn, each fold is used as the test set and the remaining nine as training set. All codes, including the packages for the classifiers, have been implemented in the R language in a Windows environment.

3.3 Experimental results on document classification

Figure 2 shows the average accuracies from the 10-fold cross-validation for datasets D1-4. Notations “xgb”, “svm” and “rf” stand for classifiers XGBoost, SVM with RBF kernel and random forest, respectively. Notations “w2v”, “tfidf”, “tax” and “tax.w” stand for feature extraction models doc2vec, BoW with tf-idf weighting, taxonomy-augmented algorithm 2 and taxonomy-augmented algorithm 1, respectively.



A) D1

B) D2

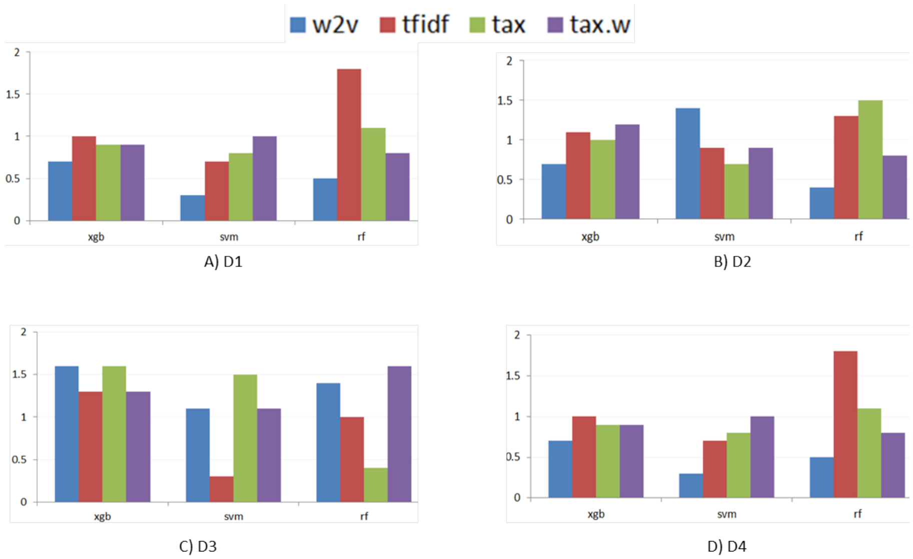
C) D3

D) D4

Fig. 2. Average accuracy from 10-fold cross-validation. The horizontal axis maps the classifier and the coloured bars represent the feature vectors.

Based on Figure 2, the model based on predefined words, “tax.w”, performs better than the others in most cases, followed by “tax”. The “tfidf” model performs very similarly to “tax.w” using “xgb”; however, it fails to produce accurate predictions with the other two classifiers, in particular with “svm”. Amongst the classification methods, “xgb” performs the best, followed by “rf” and “svm”, respectively.

In Figure 3 we also use the average of the absolute deviations (AVDEV) to explore the variability of the 10-fold cross-validation accuracies around their means. Figure 3 shows that the AVDEV for all methods are mostly in a very similar range, with the exception of “tfidf” using “rf” in D1 and D4 where the AVDEV are, undesirably, much higher.



A) D1

B) D2

C) D3

D) D4

Fig. 3. Average of the absolute deviations (AVDEV) of accuracies from their mean. Horizontal axis shows classification methods and vertical axis is the AVDEV value.

We have not conducted a formal complexity analysis for the methods. However, we have noted that classification takes a very similar time with the different feature vectors, with the exception of “tfidf” that is considerably slower. If we include the time for feature extraction, “w2v” becomes the fastest, followed by “tax”, “tax.w” and “tfidf”, respectively. Model “tfidf” is much slower due to its much larger dimensionality, in particular in conjunction with “svm” classification.

3.4 Experimental results on document clustering

The performance of a document clustering approach can be well described using two complementary measures: connectivity and silhouette [5]. The connectivity captures the “degree of connectedness” of the clusters and it is measured in terms of how many nearest neighbors of any given sample belong to other clusters; its value ranges between 0 and infinity and should be minimized. The silhouette measures the compactness and separation of the clusters and ranges between -1 (poorly-clustered observations) and 1 (well-clustered observations). To compute these measures,clValid, a popular R package for cluster validation [5] has been used.

Figure 4 shows the connectivity and silhouette measures for a dataset called PCall, similar to D4 but with 8,000 samples. In this figure, "tfidf" stands for the BoW model, "w2v" for doc2vec, and "tax.1" and "tax.2" for the models from Algorithms 2 and 1, respectively. In turn, "km" and "pam" stand for two clustering algorithms, k-means++ and PAM, respectively. These results show that the taxonomy-augmented models, "tax.1" and "tax.2", have performed better than the other two models for a large majority of cluster numbers. Among the conventional models, the performance of BoW is generally better than that of doc2vec. It is noted that BoW is the most time-consuming due to its large number of features. Based on the experiments, it is approximately 10 times slower than the other models.

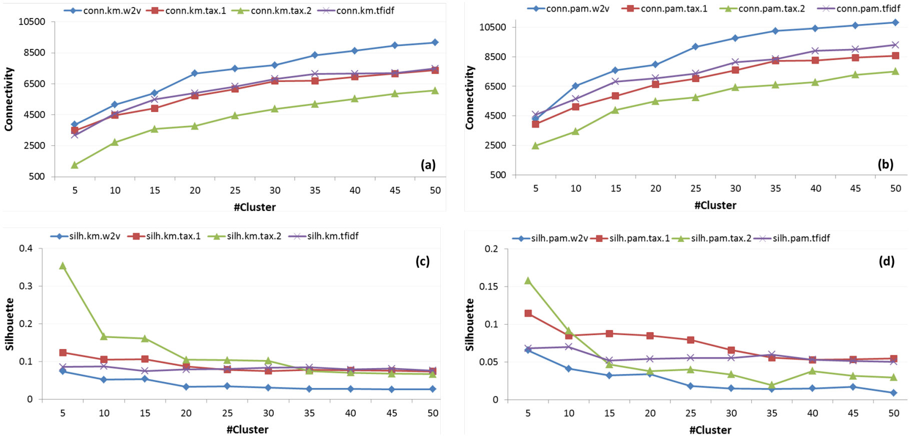


Fig. 4. Connectivity and silhouette measures of all models for the PCalls dataset.

3.5 Semantic analysis

In this section, we illustrate the use of the taxonomy-augmented features of Algorithm 1 for the "semantic analysis" of TAC clients. The data for each client consist of several phone calls, where each phone call has been represented by a set of predefined words using Algorithm 1. In this section, we use $S = \{family, pain, provider, recovery, stress, upset\}$ as predefined words. In this way, each phone call is represented as a distribution over chosen words, which in turn are represented as distributions over the entire vocabulary. The concept looks very similar to that of *topic modelling*, where each document is represented as a distribution over topics and each topic is a distribution over the vocabulary. However, the two models differ notably in that the topics are latent variables without an a-priori semantic, whereas our predefined words are observed and can be chosen by experts.

Figure 5 shows the semantic analysis of two clients as plots of the six chosen features along successive phone calls. The first client recorded a total of 48 phone calls over 38 months and the second client 14 over 29 months (a few phone calls have been removed because the number of words after preprocessing had fallen below a minimum set threshold). This figure should be regarded just as an illustrative example as any set of words or reasonable size can be used in this analysis.

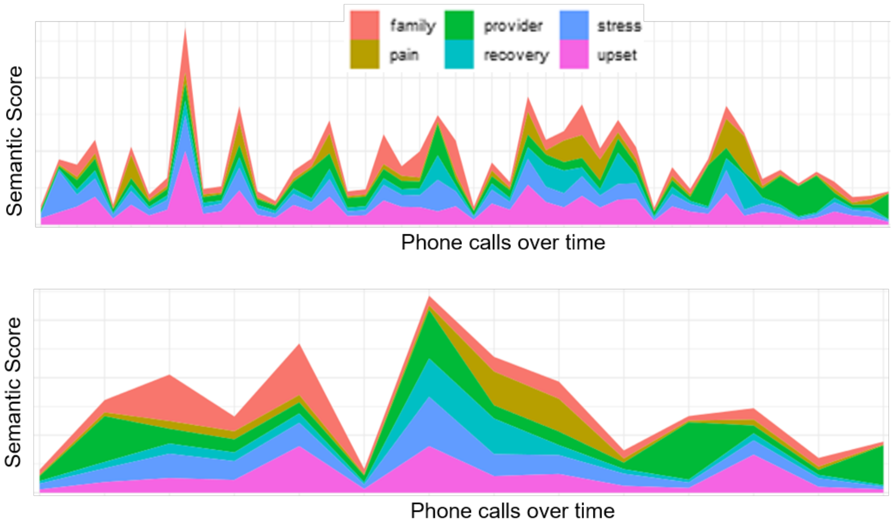


Fig. 5. Evolution of the chosen features in the phone calls of two randomly-selected clients. The semantic scores are computed by Algorithm 1.

4 Conclusion

In this paper, we have presented the application of a taxonomy-augmented feature extraction approach to a variety of important document tasks such as classification, clustering, and semantic analysis. The approach addresses two urgent challenges of conventional document representations, namely 1) their large number of features, and 2) the dismissal of the word ordering in the formation of the features. By amending these two shortcomings, the proposed model has proved able to provide a more compact and semantically-meaningful document representation and improve the tasks' performance.

In an original set of experiments on document classification (3.3), we have compared the proposed approach with two well-known methods, BoW/tf-idf and doc2vec, over four phone call datasets from an accident support agency. The results have shown that the proposed models have achieved better average accuracy in the large majority of cases, while their absolute deviations from the averages have kept almost the same. These improvements confirm the results obtained by the proposed models in experiments on document clustering (3.4). In addition (3.5), we have illustrated the usefulness of the proposed feature vector for the monitoring of individual progress over time.

Acknowledgement. This project has been funded by the Capital Markets Cooperative Research Centre and the Transport Accident Commission of Victoria. Acknowledgements and thanks to our industry partners David Attwood (Lead Operational Management and Data Research) and Bernie Kruger (Business Intelligence and Data Science Lead). This research has received ethics approval from University of Technology Sydney (UTS HREC REF NO. ETH16-0968).

References

1. Eissa M. Alshari, Azreen Azman, Shyamala Doraisamy, Norwati Mustapha, and Mustafa Alkeshr. Improvement of sentiment analysis based on clustering of Word2Vec features. In *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*, 2017.
2. D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *In H. Gabow (Ed.) Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms [SODA07]*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
3. M.N. Asim, M. Wasim, Khan M.U.G., W. Mahmood, and H.M. Abbasi. A survey of ontology learning techniques and applications. *Database*, 2018.
4. A. Bagirov, S. Seifollahi, M. Piccardi, E. Zare, and B. Kruger. SMGKM: An efficient incremental algorithm for clustering document collections. *CICLing 2018*, 2018.
5. G. Brock, V. Pihur, S. Datta, and S. Datta. clValid: An R package for cluster validation. *Journal of Statistical Software*, 25:1–22, 2008.
6. T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785 – 794, 2016.
7. Y. Cheng. Ontology-based fuzzy semantic clustering. In *Proceedings - 3rd International Conference on Convergence and Hybrid Information Technology, ICCIT 2008*, volume 2, pages 128–133, 2008.
8. S. Dhillon, J. Fan, and Y. Guan. Efficient clustering of very large document collections. In C. Kamath V. Kumar R. Grossman and R. Namburu, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, Oxford, 2001.
9. A. Elsayed, H.M.O. Mokhtar, and O. Ismail. Ontology based document clustering using Mapreduce. *International Journal of Database Management Systems*, 7(2):1–12, 2015.
10. U. Erra, S. Senatore, F. Minnella, and G. Caggianese. Approximate tf-idf based on topic extraction from massive message stream using the gpu. *Information Sciences*, 292:143–161, 2015.

11. S. Fodeh, B. Punch, and P.-N. Tan. On ontology-driven document clustering using core semantic features. *Knowledge and Information Systems*, 28(2):395–421, 2011.
12. J.H. Friedman. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1):55–77, 1997.
13. Z.S. Harris. Distributional Structure. *Word*, 10 (2-3):146 – 162, 1954.
14. A. Hotho, S. Staab, and G. Stumme. Ontologies improve text document clustering. In *Third IEEE International Conference on Data Mining*, pages 541–544, 2003.
15. J. Kim, F. Rousseau, and M. Vazirgiannis. Convolutional sentence kernel from word embeddings for short text categorization. In *Proceedings EMNLP 2015*, number September, pages 775–780, 2015.
16. M.J. Kusner, Y. Sun, N.I. Kolkin, and K.Q. Weinberger. From word embeddings to document distances. In *Proceedings - ICML*, volume 37, pages 957–966, 2015.
17. L. Lenc and P. Král. Word embeddings for multi-label document classification. In *Proceedings of Recent Advances in Natural Language Processing*, pages 431–437, 2017.
18. J. Lilleberg, Y. Zhu, and Y. Zhang. Support vector machines and word2vec for text classification with semantic features. *Proceedings of IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, pages 136–140, 2015.
19. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *Arxiv*, pages 1–12, 2013.
20. B. Moseley and J.R. Wang. Approximation bounds for hierarchical clustering: Average linkage, bisecting K-means, and local search. Number Nips, pages 3097–3106, 2017.
21. J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings EMNLP 2014*, pages 1532–1543, 2014.
22. C. Qimin, G. Qiao, W. Yongliang, and W. Xianghua. Text clustering using vsm with feature clusters. *Neural Computing and Applications*, 26(4):995–1003, 2015.
23. S. Seifollahi, A. Bagirov, R. Layton, and I. Gondal. Optimization based clustering algorithms for authorship analysis of phishing emails. *Neural Processing Letters*, 46(2), 2017.
24. S. Seifollahi, M. Piccardi, E.Z. Borzeshi, and B. Kruger. Taxonomy-augmented features for document clustering. *Australian Conference on Data Mining*, 2018.
25. Roger Alan Stein, Patricia A. Jaques, and João Francisco Valiati. An analysis of hierarchical text classification using word embeddings. *Information Sciences*, 2019.
26. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 1–2, 2000.
27. D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings ACL*, pages 1555–1565, 2014.
28. P. Wang, B. Xu, J. Xu, G. Tian, C.L. Liu, and H. Hao. Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. *Neurocomputing*, 174:806–814, 2016.
29. D. Zhang, H. Xu, Z. Su, and Y. Xu. Chinese comments sentiment classification based on word2vec and SVMperf. *Expert Systems with Applications*, 42(4):1857–1863, 2015.
30. Lei Zhu, Guijun Wang, and Xiancun Zou. A Study of Chinese Document Representation and Classification with Word2vec. *Proceedings - 2016 9th International Symposium on Computational Intelligence and Design, ISCID 2016*, 1:298–302, 2017.