An Approach for Textual Entailment Recognition based on Stacking and Voting

Zornitsa Kozareva and Andrés Montoyo

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, Spain
{zkozareva,montoyo}@dlsi.ua.es

Abstract. This paper presents a machine-learning approach for the recognition of textual entailment. For our approach we model lexical and semantic features. We study the effect of stacking and voting joint classifier combination techniques which boost the final performance of the system. In an exhaustive experimental evaluation, the performance of the developed approach is measured. The obtained results demonstrate that an ensemble of classifiers achieves higher accuracy than an individual classifier and comparable results to already existing textual entailment systems.

1 Introduction and Motivation

Textual Entailment Recognition (RTE) task captures a broad range of semantic-oriented inferences needed across many Natural Language Processing applications. This task consists in recognizing whether the meaning of one text can be inferred from another text [3]. The assertions made in an entailed sentence must be obtained from the text passage directly, or be logically derivable from it. For example the meaning of "Post International receives papers by Austin" is inferred from "Austin sells papers to Post International", so the two sentences entail each other.

In the first textual entailment challenge, a wide variety of techniques are proposed. From simple n-gram coincidence approaches [15] to probabilistic approaches that mine the web [8]. However, the majority of the systems [7], [10], [18] experiment with different threshold and parameter settings to estimate the best performance. This parameter adjustment process is related to the carrying out of numerous experiments and from another side the selected settings are dependent on the development data which can lead to incorrect reasoning for the entailment system as can be seen in [9].

For this reason, we decide to present a machine learning entailment approach that determines the result of the entailment relation in an automatic way. Machine learning techniques for named entity recognition [16], part-of-speech tagging [13] and parsing [2] among others are known to perform better than rule-based systems.

The motivations for the proposal of a textual entailment (TE) machine learning approach consists in the advantages of avoiding threshold determination, the

ability to work with large number of features, the allowance to integrate information from multiple levels of representation such as morphologic, syntactic, semantic or combination among them.

Our contribution consists in the design of lexical and semantic features that measure the similarity of two texts to determine whether the texts entail each other or not, and the creation of complementary classifiers that are combined through stacking and voting. Previous TE research did not take advantage of such approach.

In the experimental evaluation, the contribution and the limitation of the modelled attributes for the text entailment recognition are shown. We demonstrate that the classifiers' ensemble boosts the individual performance of the lexical and semantic classifiers. Additionally, a comparative study with already existing TE approaches is performed. The obtained results showed that the recognition of text entailment with machine learning based approach is possible, and yields comparable results to the already existing systems.

This paper is organized as follows: Section 2 is related to the description of the lexical and semantic attributes, Section 3 explains the combination methodology, Section 4 and 5 describe the experimental evaluation of the proposed approach and comparison with already existing systems. Finally, we conclude in Section 6 and discuss some future work directions.

2 Feature Representation

Text entailment can be due to lexical, syntactic, semantic, pragmatic variabilities, or to a combination among them. Therefore, various attributes are needed for its resolution. For our approach, we modelled lexical and semantic information by the help of text summarization [11] and similarity [12] measures.

In an exhaustive research study, we tested three machine learning algorithms (k-Nearest Neighbours, Maximum Entropy and Support Vector Machines (SVM)) together with a backward feature selection algorithm. According to the obtained results, the best performing algorithm is SVM, and the best features are the one described below:

f_1: $n-gram = \frac{c}{m}$, obtains the ratio of the consecutive word overlaps c in the entailing text T and the entailed hypothesis H, and later this ratio is normalized by the number of words m in H. According to the $n-gram$ attribute, the more common consecutive words two sentences have, the more similar they are. Therefore, the textual entailment example T: *Mount Olympus* towers up from *the center of the earth* and H: *Mount Olympus* is in *the center of the earth.* is determined correctly. First the ratio of common $n-grams$ is high both for T and H, and second most of the constituents in H are mapped into T, which shows that the hypothesis can be inferred from the text.

f_2: $LCS = \frac{LCS(T,H)}{n}$, estimates the similarity between T with length m and H with length n, by searching in-sequence matches that reflect sentence level word order. The longest common subsequence (LCS) captures sentence level structures in a natural way. In the example, T: A male rabbit is called a buck

and *a female rabbit is called a* doe, just like deer. and H: *A female rabbit is called a* buck., most of the constituents of H are mapped into T. However, the entailment relation between the two texts does not hold, because according to T, the female rabbit is called "doe" not "buck". Therefore, we incorporated a more sensitive measure – the skip-gram.

$f_{3,4}$: $skip\_gram = \frac{skip\_gram(T,H)}{C(n,skip\_gram(T,H))}$, where $skip\_gram(T,H)$ refers to the number of common skip grams (e.g. pair of words in sentence order that allow arbitrary gaps) found in T and H, $C(n, skip\_gram(T,H))$ is a combinatorial function, where n is the number of words in H. Note that in contrast to LCS, skip-grams need determinate length. Only bi and tri-skip grams are calculated, because skip-grams higher than three do not occur so often. For the texts, T: *Elizabeth Dowdeswell is* the Under Secretary General at the United Nations Offices at Nairobi and *Executive Director of the United Nations Environment Programme.* and H: *Elizabeth Dowdeswell is Executive Director of the United Nations Environment Programme.*, both skip-gram measures identify correctly that H is inferred by T, as all skip-grams of H are mapped into T.

To obtain the following attributes, we used the TreeTager part-of-speech tagger [17]. For the similarity measures, we used the WordNet::Similarity package [14].

f_5, f_6: nT, nH is 1 if there is a negation in T/H, 0 otherwise. These attributes treat only explicit negations such as "no, not, n't".

f_7: *number* identifies that "four-thousand" is the same as 4000, "more than 5" indicates "6 or more", "less than 5" indicates "4 or less". For perfect matches *number* is 1. When T and H do not have numeric expression, the attribute is 0, while partial matches have values between 0 and 1.

f_8: *Np* identifies the proper names in T and H, and then lexically matches them. Consecutive proper names that are not separated by special symbols as coma or dash are merged and treated as a single proper name for example "Mexico City" is transformed into one Np "Mexico_City". The value of *Np* is 1 for a perfect match between the proper name of T and the proper name of H, such as "London" and "London". *Np* is 0 when there are no proper names or the existing proper names are completely different. Partial matches as "Mexico_City" and "Mexico" have value 0.5, because from two words only one is completely matched.

f_9, f_{10}: *adv*, *adj* is 1 for perfect match of adverbs/adjectives between T and H, 0 when there is no adverb/adjective in one or both of the sentences, and between 0 and 1 for partial matches.

The next attributes estimate the noun/verb similarity between T and H. We did not use a word sense disambiguation (WSD) module, through which we will know the adequate word senses, however we use two different similarity measures the one of lin and the path. These measures associate different word senses to the noun/verb pairs and then estimate the WordNet similarity related to these senses. Although we did not use WSD, we obtain similarity evidence from two different similarity measures and different word senses. That allows us to capture the semantic variability in a broader way.

f_{11}, f_{12}: $n\_lin/path = \frac{\sum_{i=1}^{n} sim(T,H)_{lin/path}}{n}$, estimates the similarity $sim(T,H)_{lin/path}$ for the noun pairs in T and H according to the measure of lin/path, against the maximum noun similarity n for T and H.

f_{13}, f_{14}: $v\_lin/path = \frac{\sum_{j=1}^{v} sim(T,H)_{lin/path}}{v}$, determines the similarity $sim(T,H)_{lin/path}$ of the verbs in T and H according lin/path measure, compared to the maximum verb similarity v for T and H.

f_{15}, f_{16}: $nv\_lin/path = \frac{\sum_{i=1}^{n} sim(T,H)_{lin/path} * \sum_{j=1}^{v} sim(T,H)_{lin/path}}{n*v}$ measures the similarity of the nouns and the verbs in T and H. With this inter-syntactic information, the system is able to capture the similarity between patterns such as "X works for Y" and "Y's worker is X".

f_{17}: $units = \prod_{unit=1}^{k} \left(\frac{\sum_{i=1}^{l} sim(T,H)}{l} \right)$ represent the text similarity between T and H. Units correspond to the different constituents in a sentence, for which the noun/verb similarity is determined with attributes f_{11} and f_{12}, and the rest of the units are lexically matched.

In a ten-fold cross validation, the described attributes are divided in two complementary sets: Lex=$\{f_1,f_2,f_3,f_4,f_5,f_6,f_7\}$ and Sim=$\{f_8,f_9,f_{10},f_{11},f_{12},f_{13},f_{14},f_{15},f_{16},f_{17}\}$. These sets obtain similar results in accuracy, but they resolve different text entailment examples. The Lex set recognizes the false TE examples, while the Sim determines correctly the positive TEs. The most informative attributes for set Lex are f_2 and f_4, while for set Sim f_8 and f_{11} are determined. We denote these subsets as $bLex$ and $bSim$.

3 Combining Methodology

[6] states that an ensemble of classifiers is potentially more accurate than an individual classifier. This is one of the reasons for which we decided to study how to combine the generated classifiers. The other reason is the design of the complementary feature sets.

With the objective to combine several classifiers, the fist condition to be examined is feature set complementarity. This is needed to establish complementary classifiers that are able to resolve different examples. Their combination is beneficial, because together the classifiers will resolve more cases. To evaluate the complementarity of our feature sets, the kappa measure [1] is used. According to kappa, set Lex and Sim are complementary, and so are their subsets.

A problem that arises is how to identify which classifier gives the more precise class prediction for a given example. In our approach, this is resolved by the calculation of a confidence score that measures the distance between the predicted value of the example and the training prototypes.

3.1 Stacking

Combining classifiers with stacking can be considered as meta-learning e.g. learning about learning. For our approach stacking is applied in the following way. In

the first phase, multiple classifiers generated by the four different feature sets are produced. They represent the set of base-level classifiers. In the second phase, a meta-level classifier that combines the features of the best performing classifier together with the outputs of the base-level classifiers is learned. Thus the new meta-classifier is generated. The output of the meta-classifier is considered as the final result of the stacking method.

3.2 Majority Voting

Many techniques that aim to combine multiple evidences into a singular prediction are based on voting. Our approach uses majority voting that consists in using the generated output of the several classifiers and compares them. The final decision about the class assignment is taken regarding the class with the majority votes. Examples for which the classifiers disagree, acquire the class of the classifier with the highest performance.

4 Experimental Evaluation

After the description of the designed attributes and the combination methods, this section describes the experimental evaluation.

4.1 Data Set

There are not many textual entailment data sets, therefore for the performed experiments we use the development and test data sets provided by the Second Recognising Textual Entailment Challenge (RTE 2)[1]. The examples in these data sets have been extracted from real Information Extraction (IE), Information Retrieval (IR), Question Answering (QA) and Text Summarization (SUM) applications.

The distribution of the entailment examples is balances e.g. 50% of the examples entail each other and 50% do not. The development set consists of 800 text-hypothesis pairs, which we use as training examples. The other set of 800 text-hypothesis pairs is used for testing. The provided data sets are for the English language. Performances are evaluated with the RTE2 evaluation script[2]. According to the script, systems are ranked and compared by their accuracy scores.

4.2 Experiment 1: Single Classifier

The first carried out experiment, examines the performances of the individual feature sets. The obtained results for the development and the test data are shown in Table 1. All classifiers outperform a baseline that counts the common unigrams between T and H.

[1] http://www.pascal-network.org/Challenges/RTE2/
[2] http://www.pascal-network.org/Challenges/RTE2/Evaluation/

sets	Acc.	IE	IR	QA	SUM
devLex	56.87	49.50	55.50	51.00	71.50
devbLex	57.75	49.50	57.00	51.50	73.00
devSim	60.12	54.00	61.00	59.00	66.50
devbSim	57.13	54.50	61.00	49.50	63.50
testLex	54.25	52.00	53.50	55.50	56.00
testbLex	**56.75**	**46.00**	**55.50**	**56.50**	**69.00**
testSim	54.25	50.00	55.50	47.50	64.00
testbSim	52.38	49.00	51.50	52.00	57.00
baseline	50.50	51.00	50.00	48.50	52.50

Table 1. *Results for the individual feature sets with RTE2 data.*

As we previously described in Section 2, set *Lex* and *bLex* rely on word overlap information captured by text summarization measures therefore, the majority of the correctly resolved examples both for the development and the test data are obtained for the summarization task.

These features identify faultlessly that T: "*For the year, the Dow rose 3 percent, while the S'P 500 index added 9 percent.*" and H: "*For the year, the Dow gained 3 percent, the S'P rose 9 percent.*" infer the same meaning, because there is a high number of overlapping words in the both texts. The attributes of set *Lex* are good for entailment, because by mapping the majority of the words from the hypothesis into the text indicates that both texts have similar context and infer the same meaning. However, the attributes in *Lex* and *bLex* are sensitive to the length of the sentence, that is why they penalize texts composed of many words. Normally the hypothesis has less words than the text, and according to the attributes, the entailment relation for such sentences is most likely not to hold.

The other feature sets *Sim* and *bSim* rely on semantic information and for them textual entailment holds when T and H have many similar words. For these attributes, the entailment problem converts to a similarity problem, e.g. the more similar words two text share, the more similar meaning they reveal. The majority of the correctly resolved TE examples are for summarization. In contrast to the previous feature sets, the majority of the obtained mistakes for *Sim* and *bSim* are false positive pairs. In other words the classifiers predicted examples as true, which in reality were false. The favor of the positive class is due to the modelled similarity attributes.

Our similarity attributes reflect the similarity of the nouns/verbs according to the measure of lin/path, in respect to the maximum noun/verb similarity for a text-hypothesis pair. When all nouns/verbs have high (or low) similarity according to the measure of lin/path, the final similarity function identifies correctly the entailment relation between the two sentences. The measures fail, when the ratio of the strongly similar and dissimilar noun/verb pairs is the same, then the entailment relation is considered as positive, which is not always correct. For the pairs *"Scientists have discovered that drinking tea protects against heart*

disease by improving the function of the artery walls." and *"Tea protects from some diseases."*, the noun pairs "tea–tea" and "disease-disease" have the value of 1 due to the perfect match, the same happens to the verb "protect-protect". As all words of the hypothesis are match with those of the text and their similarity score is high, this indicates that the texts entail each other. Although other noun/verb pairs are present, the similarity measures determine the entailment relation for the sentences as correct. A disadvantage of the similarity measures comes from the WordNet[3] repository. When a word is not present in WordNet, then the similarity cannot be determined correctly. One way to overcome such obstacle is the usage of corpus statistics, the web or Latent Semantic Analysis.

The four feature sets *Lex*, *Sim*, *bLex* and *bSim*, obtained the lowest resolution for the IE task. This is acceptable for the lexical overlap sets, because they consider only continuous or insequence word coincidences. Although set *Sim* and *bSim* have an attribute that matches proper names, the experiments demonstrate that this attribute is not sensitive enough. To identify correctly that *"Former Uttar Pradesh minister Amar Mani Tripathi, his wife Madhu Mani and three others are currently in jail in connection with the killing of the poetess."* and *"Madhu Mani is married to Amar Mani Tripathi."* infer the same meaning, a named entity recognizer and entity relation extraction are needed. The named entity recognizer will identify that "Amar Mani Tripathi" and "Madhu Mani" are persons and the relations "X is the wife of Y" and "X is married to Y" describe the same event. Thus, the proper names can be weighted and the performance of the IE task can be improved.

4.3 Experiment 2: Stacking

The performance of the individual classifiers obtained similar accuracy scores, but they covered different entailment examples. Therefore, studying the way to combine the generated classifiers was a reasonable intention. The stacking experiment started with the creation of a meta-learner from a single classifier. Considering the accuracy score of the development data, the best performance of a single classifier is achieved with *Sim* feature set. This classifier is taken and selected as a base-classifier. The obtained outputs together with the predicted classes of *devSim* are used as two additional features to the initial feature set of *Sim*. In this way the meta-classifier is created. The obtained results are shown in Table 2.

The creation of a meta-classifier from the output of a single classifier does not improve the performance. After this observation is made, staging from one to two classifiers, then considering the outputs of the three and finally the outputs of the four classifiers is done. The results show that the more classifiers are incorporated, the better the performance of the meta-learner is becoming.

For the development data, stacking improved the performance of the ensemble of classifiers. In Table 1 can be seen that for QA, the individual classifiers

[3] wordnet.princeton.edu/

sets	Acc.	IE	IR	QA	SUM
dev1Classif	62.12	54.50	61.50	62.00	70.50
dev4Classif	68.63	58.50	68.00	70.50	77.50
test4Classif	54.87	54.00	56.00	50.00	59.50

Table 2. *Results for stacking with RTE2 data.*

have accuracy ranging from 49% to 59%. When stacking is applied, the QA performance is boosted to 70%. This indicates that separately the four feature sets resolve different QA examples e.g. one set identifies correctly the true TE examples while the other the negative and when they are ensembled, the performance increases.

The performance of the test data is quite different compared to the development data. This performance is influenced and related to the low coverage of the individual feature sets for the test data as can be seen in Table 1. Although for three of the four feature sets, the stacking method improved the overall accuracy of the test data, surprisingly set *bSim* only with its two attributes achieves the best result. Another reason for the performance concerns the accumulated errors by the individual classifiers, which are transmitted to stacking.

4.4 Experiment 3: Majority Voting

This experiment concerns the combination of the classifiers by majority voting. The combined classifiers are *Lex* with *Sim* and *bLex* with *bSim*. The obtained results are shown in Table 3.

sets	Acc.	IE	IR	QA	SUM
devLexSim	61.12	52.00	64.50	57.00	71.00
testLexSim	54.37	53.00	54.50	50.50	59.50
devbLexSim	62.12	54.50	61.50	62.00	70.50
testbLexSim	55.00	49.50	54.00	54.50	62.00

Table 3. *Results for voting with RTE2 data.*

For the development set, majority voting performed better than each one of the individual feature sets, but among all development runs, the stacking method achieved the best score. For the test data, stacking performed around 54.87%, due to the introduced noise of the other classifiers. Compared to it, voting reached 55.00% accuracy, but this result is not significant[4] and does not improve the final performance.

When the presence of the correct classes for the test examples is used, the voting combination of classifiers *Lex* and *Sim* (or *bLex* and *bSim* respectively)

[4] z' with 0.975% of confidence

reaches 75% accuracy. In a real application, the presence of the real class is not existing, therefore for our experiment this information is not used. The 75% accuracy score demonstrates the cooperation and the performance that can be reached by the different feature sets. An improvement for the classifier combination methods is the usage of another confidence score, such as the entropy distribution.

5 Comparative Study

The formulation and the evaluation of the proposed textual entailment approach with a single data set is insufficient to demonstrate and confirm the behavior of the approach. Therefore, we decided to present a comparative study with systems that participated in the First Recognising Textual Entailment Challenge (RTE1) [4].

For this study the attributes described in Section 2 are used and the three experiments carried out in Section 4 are performed. The new development and test data sets come from the RTE1 challenge[5]. In this data sets, textual entailment for seven NLP tasks (Comparable Documents (CD), Reading Comprehension (RC), Machine Translation (MT), Paraphrase Acquisition (PP), IR, QA, IE) has to be resolved.

sets	Acc.	CD	IE	MT	QA	RC	PP	IR
tL	51.25	74.67	50.83	48.33	39.23	48.57	56.00	35.56
tS	57.50	66.00	55.83	48.33	56.92	57.14	56.00	60.00
tbL	52.50	77.33	49.17	51.67	36.92	47.14	58.00	44.44
tbS	54.13	66.00	45.00	56.67	47.92	47.14	54.00	62.22
$\boldsymbol{tS}$	**58.13**	**62.67**	**55.83**	**52.50**	**59.23**	**58.57**	**54.00**	**61.11**
$\boldsymbol{tV}$	**57.70**	**74.67**	**50.83**	**48.33**	**56.92**	**57.14**	**56.00**	**60.00**
S_1	59.25	68.67	58.33	46.67	58.46	52.14	80.00	62.22
S_2	58.60	83.33	55.83	56.67	49.23	52.86	52.00	50.00
S_3	56.60	78.00	48.00	50.00	52.00	52.00	52.00	47.00
S_4	49.50	70.00	50.00	37.50	42.31	45.71	46.00	48.89

Table 4. *Comparative evaluation with RTE1 data.*

The obtained results of the four feature sets, together with the stacking and voting combination are shown in Table 4. In the same table are shown the performances of some system from the RTE1 challenge. With S_1 and S_2 are denoted the two best performing systems [5] and [8] for the RTE1 challenge. System S_3 [10] is an intermediate performing one, while S_4 [15] obtained the lowest accuracy score. We placed these systems, so that a general overview for the ability of the systems to recognize textual entailment can be obtained. As can be seen in Table 4, the accuracy performance varies from 49.50% to 59.25%.

Set Lex and $bLex$ resolve the majority of the comparable document examples, while set Sim and $bSim$ influenced the information retrieval performance. Stacking and voting boosted the performance of the classifiers with around 2%

[5] http://www.pascal-network.org/Challenges/RTE/

compared to the best individual set and from 1% to 5% compared to the individual *Lex*, *bLex* and *bSim* sets. Considering the achieved results from Subsections 4.3 and 4.4, and those of the new data set, it can be concluded that the combination methods can improve the performance of the individual classifiers, but are strongly related to the data sets and the outcomes of the individual classifiers.

Table 4 shows that the machine learning results are comparable with those of the other systems. Although the other approaches used more ample information, our machine learning approach outperformed thirteen from the sixteen participants and reached 58.13% accuracy. The developed feature sets confirmed that are applicable to the resolution of SUM, IR and PP.

6 Conclusions and Future Work

The main contributions of this paper consist in the proposal of lexical and semantic attributes through which textual entailment can be recognized. Additionally, we explore how stacking and voting techniques affect the performance of a TE machine-learning based approach. The experiments show that for the development set, stacking and voting outperform all single classifiers. This indicated that several distinct classifiers can be combined efficiently and boost the final performance. However, for the test data stacking and voting are outperformed by a single classifier *bSim*. This is due to the combination techniques which require a reasonable level of performance of the individual classifiers. As the performance of the individual *Lex* and *Sim* sets were particularly weak in the test, the combination scheme did not benefit.

The designed lexical features are brittle as they rely on literal matches, while the semantic features rely on conceptual matching, hence they are limited in coverage. These limitations lead to low individual performance and later the classifier combination effect was hampered. The performance of the best lexical feature set (e.g. skip-grams and LCS), achieves the best performance which suggests that H and T are quite similar with respect to their surface word choices, and that a lexical matching is sufficient for TE recognition.

To confirm the robustness of the proposed approach, we evaluate it on two textual entailment data sets and compare it to already existing TE systems. The obtained results demonstrate that the recognition of textual entailment with the proposed attributes and the combination of several classifiers yields comparable results to a probabilistic, first logic order and other approaches.

In the future we will focus on more specific TE features, as similarity does not always infer entailment. We are interested in creating back-off strategies that use lexical features at first and then lean upon the semantic ones. We want to explore more classifiers in stacking. To overcome the limitations of the presented similarity attributes, WordNet will be compared to LSA. Other attributes that handle implicit negations, named entity recognition and syntactic variabilities will be modelled.

Acknowledgements

This research is funded by the projects CICyT number TIC2003-07158-C04-01, PROFIT number FIT-340100-2004-14, and GV04B-276.

References

1. J. Cohen. A coefficient of agreement for nominal scales. *Educ. Psychol.*, 1960.
2. M. Collins and B. Roark. Incremental parsing with the perceptron algorithm. In *Proceedings of the ACL*, 2004.
3. I. Dagan and O. Glickman. Probabilistic textual entailment: Generic applied modeling of language variability. In *PASCAL Workshop on Learning Methods for Text Understanding and Mining*, 2004.
4. I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Workshop on RTE*, 2005.
5. R. Delmonte, S. Tonelli, A. P. Boniforti, A. Bristot, and E. Pianta. Venses – a linguistically-based system for semantic evaluation. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
6. T. Dietterich. Machine-learning research: Four current directions. *AI Magazine*, pages 97–136, Winter 1997.
7. A. Fowler, B. Hauser, D. Hodges, ian Niles, A. Novischi, and J. Stephan. Applying cogex to recognize textual entailment. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
8. O. Glickman, I. Dagan, and M. Koppel. Web based probabilistic textual entailment. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
9. V. Jijkoun and M. de Rijke. Recognizing textual entailment using lexical similarity. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
10. M. Kouylekov and B. Magnini. Recognizing textual entailment with edit distance algorithms. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
11. C. Lin and F. J. Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-birgam statistics. In *Proceedings of ACL*, 2004.
12. D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, 1998.
13. L. Màrquez, L. Padró, and H. Rodríguez. A machine learning approach to pos tagging. 1998.
14. S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, 2003.
15. D. Pérez and E. Alfonseca. Application of the bleu algorithm for recognising textual entailments. In *Proceedings of the PASCAL Workshop on Recognising Textual Entailment*, 2005.
16. T. K. Sang and Erik F. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, 2002.
17. H. Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, 2004.
18. M. Montes y Gomez, A. Gelbukh, and A. Lopez-Lopez. Comparison of conceptual graphs. In *Proceedings of MICAI*, pages 548–556, 2000.