Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia - Social and Behavioral Sciences 73 (2013) 101 – 108

Procedia
Social and Behavioral Sciences

The 2nd International Conference on Integrated Information

Determining unintelligible words from their textual contexts

Balázs Pintér[a,]*, Gyula Vörös[a], Zsolt Palotai[a], Zoltán Szabó[a], András Lőrincz[a]

[a]*Eötvös Loránd University, Budapest H-1117 , Hungary*

Abstract

We propose a method to determine unintelligible words based on the textual context of the word determined. As there can be many different possibilities for the word, a robust, large-scale method is needed.
The large scale makes the problem sensitive to spurious similarities of contexts: when the contexts of two, different words are similar. To reduce this effect, we induce structured sparsity on the words by formulating the task as a group Lasso problem. We compare this formulation to a k-nearest neighbor and a support vector machine based approach, and find that group Lasso outperforms both by a large margin. We achieve up to 75% of accuracy when determining the word from among 1000 words both on the Brown corpus and on the British National Corpus.
Unintelligible words are often the result of errors in Optical Character Recognition (OCR) algorithms. As the proposed method utilizes information independent from information used in OCR, we expect that a combined approach could be very successful, as OCR and the proposed method complement each other.

Selection and/or peer-review under responsibility of The 2nd International Conference on Integrated Information.

Keywords: natural language processing; structured sparse coding; word recognition; distributional hypothesis

1. Introduction

Determining unintelligible words that automated Optical Character Recognition (OCR) programs could not recognize is a hard problem. It is difficult enough to serve as a kind of Turing test: a CAPTCHA[†], used to determine whether a user is a human or a computer. The reCAPTCHA project poses this problem to millions of users every day in order to prevent automated programs from abusing online services [1].

The goal of the reCAPTCHA project is to use the tremendous human effort of solving millions of CAPTCHAs every day to help digitize books, newspapers, etc. They send the words that OCR programs could not recognize for humans to determine, in the form of CAPTCHAs.

* Corresponding author. *E-mail address:* bli@elte.hu
[†] Completely Automated Public Turing test to tell Computers and Humans Apart

Selection and/or peer-review under responsibility of The 2nd International Conference on Integrated Information.
doi:10.1016/j.sbspro.2013.02.028

In this paper, we propose a method to determine these words *automatically* by making use of their textual context. The method builds upon the tools of Natural Language Processing (NLP). Most importantly, we rely on a widely used hypothesis in NLP: the *distributional hypothesis*.

According to the distributional hypothesis, words that occur in the same contexts tend to have similar meanings [2]. The hypothesis can be exploited to determine unintelligible words, as the context of an unintelligible word will likely be most similar to contexts of its own different occurrences in contrast to contexts of other words.

The main component of the method is a distributional representation, specifically, a word-context matrix [3], where each context is labeled with the *candidate word* the context was collected for (Fig. 1). We determine the unintelligible word by formulating and solving a *classification* problem of choosing the right candidate word using the (context, label) pairs as training data.

There are exceptions to the distributional hypothesis. We call two contexts *spuriously similar* if they are similar but belong to different words. The amount of spuriously similar contexts tends to increase with the number of candidate words in the distributional representation. Thus, the more candidate words we have, the harder the learning problem is.

As the word to be determined is unintelligible, it provides no information in addition to its context (e.g., its surface form[‡]). The correct word needs to be chosen from among hundreds, maybe thousands of candidate words. This makes the learning problem considerably hard; a mechanism is needed to deal with spurious similarities of contexts.

Structured sparse coding provides such a mechanism. As we show, a structured sparsity inducing regularization, such as the group Lasso [4], is capable of diminishing the effect of spurious similarities of contexts by utilizing the structure of semantic space (Section 3).

The **contributions** of this paper are summarized as follows: (i) we approach the problem of determining unintelligible words automatically in a novel way, using tools from natural language processing and structured sparse coding. (ii) We show that inducing structured sparsity on the candidate words vastly improves performance in contrast to our baselines, a support vector machine and a k-nearest neighbor classifier. (iii) We perform large-scale evaluations where we determine words out of 1000 candidate words at once.

In the next section we review related work. Our method and results are described in Section 3 and 4. We discuss our results in Section 5 and conclude in Section 6.

2. Related work

There has been many attempts to automatically correct words in text [5,6]. For example, Hirst and Budanitsky [7] use several WordNet-based semantic relatedness measures to detect if there is a spelling variation of the target word that would fit better in the context. In [8], a supervised learning approach is used: classifiers are trained on a large English corpus to distinguish between elements of a predefined *confusion set* (i.e., words that are similar in spelling, pronunciation etc.). In [9], a string similarity function is used to generate substitute candidates for the target word, which are then sorted according to trigram statistics. Our approach differs from these in an important respect: we assume that we know nothing about the word to be corrected, *only its context*.

In information retrieval and speech recognition, unintelligible words pose a practical problem. The TREC-5 confusion track [10] studied the impact of data corruption introduced by scanning or OCR errors on retrieval performance. In the subsequent spoken document retrieval tracks [11] the errors were introduced by automatic speech recognition.

Solving natural language processing problems by structured sparsity inducing regularization has been explored in works such as [12] and [13]. [12] apply sparse hierarchical dictionary learning to learn hierarchies of topics

[‡] the form of a word as it appears in the text

from a corpora of NIPS proceedings papers. In a more recent application [13], structured sparsity was used to perform effective feature template selection on three natural language processing tasks: chunking, entity recognition, and dependency parsing.

3. The Method

We start from a list of candidate words the unintelligible word could be recognized as. For each *candidate word*, we collect a number of *contexts*. A context of a candidate word consists of the N non-stopword words preceding and following it in the text. There can be at most $2N$ words in a context.

The presented method makes use of a collection of such *contexts* arranged in a word-context matrix $\mathbf{D}$ [3]. In this matrix, each context is a column represented as a bag-of-words vector $\mathbf{v}$ of word frequencies, where v_i is the number of occurrences of the i^{th} word of the vocabulary in the context.

The unintelligible word is determined in two steps. First, we formulate an inverse problem, and compute a representation vector $\boldsymbol{\alpha}$. In the second step, a single candidate word is selected based on the weights in this vector.

To compute the representation vector $\boldsymbol{\alpha}$, the context $\boldsymbol{x} \in \mathbb{R}^m$ of the unknown word is approximated linearly with the columns of the word-context matrix $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_n] \in \mathbb{R}^{m \times n}$ called the dictionary in the terminology of sparse coding. The columns of the dictionary contain contexts, each labeled with a candidate word $l_i \in L$. Please note that multiple contexts can be, and in many cases are, tagged with the same candidate word: $l_i = l_j$ is possible. There are m words in the vocabulary, and n contexts in the dictionary.

	boot				root				...	foot			
computer	1	0	1	0	1	0	0	0		0	0	0	0
plant	0	0	0	0	0	1	3	0		0	0	0	0
shoe	0	1	0	2	0	0	0	0		1	0	2	1
⋮													
vegetable	0	0	0	1	0	1	0	1		0	1	0	0

Fig. 1. The word-context matrix $\mathbf{D}$. Each column is a context of a candidate word (e.g., *boot*, *root*). Each element D_{ij} of the matrix holds the number of occurrences of the i^{th} word of the vocabulary in the j^{th} context. For example, the word *plant* occurs three times in the 7th context, which is the 3rd context labeled with *root*.

The representation vector $\boldsymbol{\alpha}$ consists of the coefficients of a linear combination

$$x = \alpha_1 d_1 + \alpha_2 d_2 + \ldots + \alpha_n d_n . \tag{1}$$

For each unintelligible word, whose context is $\boldsymbol{x} \in \mathbb{R}^m$ a representation vector $\boldsymbol{\alpha} = [\alpha_1; \alpha_2; \ldots; \alpha_n] \in \mathbb{R}^n$ is computed.

The structured sparsity inducing regularization is introduced by organizing the contexts in $\mathbf{D}$ into groups. Each group contains the contexts annotated with a single candidate word. As only a single candidate is chosen for each unintelligible word, ideally only a single group should be active in each representation vector $\boldsymbol{\alpha}$**.** Sparsity on the groups is realized by computing $\boldsymbol{\alpha}$ with a *group Lasso* regularization [4] determined by the labels.

The groups are introduced as a family of sets $\mathcal{G} = \{G_l\}_{l \in L} \subseteq 2^{\{1,\ldots,n\}}$. There are as many sets in $\mathcal{G}$ as there are distinct candidate words in L. For each candidate word $l \in L$, there is exactly one set $G_l \in \mathcal{G}$, that contains the indices of all the columns $\boldsymbol{d}_i$ tagged with l. $\mathcal{G}$ forms a partition.

The representation vector $\boldsymbol{\alpha}$ of the word whose context is $\boldsymbol{x}$ is defined as the minimum of the loss function

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \lambda \sum_{l\in L} w_l ||\boldsymbol{\alpha}_{G_l}||_2, \tag{2}$$

where $\boldsymbol{\alpha}_{G_l} \in \mathbb{R}^{|G_l|}$ denotes the vector where only the coordinates present in the set G_l are retained.

The first term is the approximation error, the second one realizes the structured sparsity inducing regularization. Parameter $\lambda > 0$ controls the tradeoff between the two terms. The parameters $w_l > 0$ denote the weights for each group G_l.

For the sake of simplicity, we represent each candidate word with the same number of contexts: there are an equal number of columns in $\boldsymbol{D}$ for each label $l \in L$ ($|G_1| = ... = |G_{|L|}|$). The weights w_l of the groups are set to 1.

In the *second step*, a single candidate word is selected based on the weights in this vector. We utilize the group structure to condense the vector $\boldsymbol{\alpha}$ to a single word $l \in L$. We sum the weights in each group $G_l \in \mathcal{G}$, and choose the word $l^* \in L$ whose group contains the most weight,

$$l^* = \arg\max_{l\in L} \sum_i (\boldsymbol{\alpha}_{G_l})_i. \tag{3}$$

The errors introduced by spurious similarities of contexts are diminished because in the group Lasso regularization whole *groups* are selected Each group $G_l \in \mathcal{G}$ contains contexts tagged with the same candidate word l, and only a few groups can be selected. A context similar to the context of the unintelligible word $\boldsymbol{x}$ only by accident has a smaller chance to be selected. As it is in a group labeled with a candidate word that is less related in meaning to the unintelligible word than the correct candidate, it contains mainly contexts less similar to $\boldsymbol{x}$.

4. Results

The aim of the evaluations is twofold. First, we obtain the accuracy of the method in determining unintelligible words, and compare it to the baselines. Second, we examine the effect of the ratio of unintelligible words in the text on this accuracy.

In the reCAPTCHA setting, OCR algorithms cannot recognize about 20% of the words. We regard this as a worst-case scenario, as in these tasks older prints are analyzed with faded ink and yellowed pages [1]. In our simulated experiments, we set up to 20% of the words to be unintelligible in the corpus.

The minimization problem of the *group Lasso* (Eq. 2) is solved by the Sparse Learning with Efficient Projections (SLEP) package [15]. The presented method is compared to two baselines: the k-Nearest Neighbours (kNN) algorithm and a linear Support Vector Machine (SVM). We use the implementation of the scikit-learn machine learning toolkit [16].

4.1. The datasets

We demonstrate the efficiency of the presented approach on two datasets compiled from two widely used corpora, the Brown corpus and the British National Corpus [17]. They are processed as follows (Fig. 2). In the first step, the corpora are cleaned by removing stopwords and words with non-alphabetic characters, converting the remaining words to lowercase and lemmatizing them with the WordNet lemmatizer [18].

The contexts are obtained by sampling the corpus. For each *target word* (i.e., a word to be determined), we collect 50 contexts. Each context consists of the N words that precede and follow the target word in the cleaned

corpus. In accord with [19,20] and others, we use a broad context, $N = 20$.

As target words are assumed to be unintelligible, they are deleted from their contexts. Further unintelligible words are simulated by erasing them from the context. The ratio of unintelligible words is $p \in [0, 1)$. In each context, each word has a chance p to be unintelligible: if a random number $\theta \sim U[0, 1] < p$, then the word is erased from that context.

Finally, a bag-of-words vector is constructed from each context. Our final dataset consists of $(\boldsymbol{c}_i, l_i)$ pairs, where $\boldsymbol{c}_i$ is a bag-of-words context of the target word $l_i \in L$. There are 50 columns labeled with each $l \in L$.

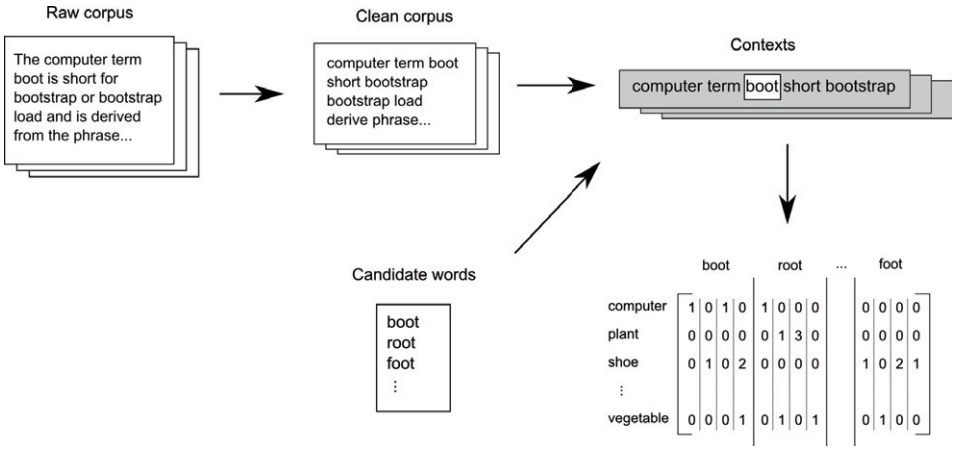

Fig. 2. Compiling a dataset. The dataset is compiled from a corpus and a list of candidate words. First, the corpus is cleaned of stopwords and words with non-alphabetic characters. Each remaining word is converted to lowercase and lemmatized. Then, for each candidate word, a number of contexts are collected from the cleaned corpus. The contexts are stored in a matrix, where each column is labeled with the candidate word the context was collected for.

4.2. Evaluations

The datasets composed from the Brown corpus and the British National Corpus (BNC) have $|L| = 1000$ distinct candidate words as labels each. We chose the 1000 most frequent words as candidate words in each dataset. There are 50 contexts labeled with each candidate word, so there are 50000 examples in each dataset.

We compare group Lasso to two baselines: kNN and a linear SVM. For the baselines, the problem of determining unintelligible words is treated as a traditional classification problem, where the candidate words are the classes.

Before evaluating the algorithms, we studied the effect of their parameters on the results. For kNN and SVM, we found $k = 40$ and $C = 1$ optimal, respectively. For group Lasso, $\lambda = 0.0005$ (Eq. 2.) was optimal. The following results were all obtained by setting the parameters to these values.

To examine the effect of the ratio of unintelligible words on the results, we remove up to 20% of the words ($p \leq 0.2$), and evaluate the accuracy. Please note that before compiling the datasets, the stopwords, that constituted approximately half of the words in the corpora (56% for the Brown corpus, and 50% for the BNC), were removed. When we evaluate the effect of removing 20% of the words, only 10% of the words is erased from the datasets in practice, as half of the 20% already fell out as stopwords during preprocessing.

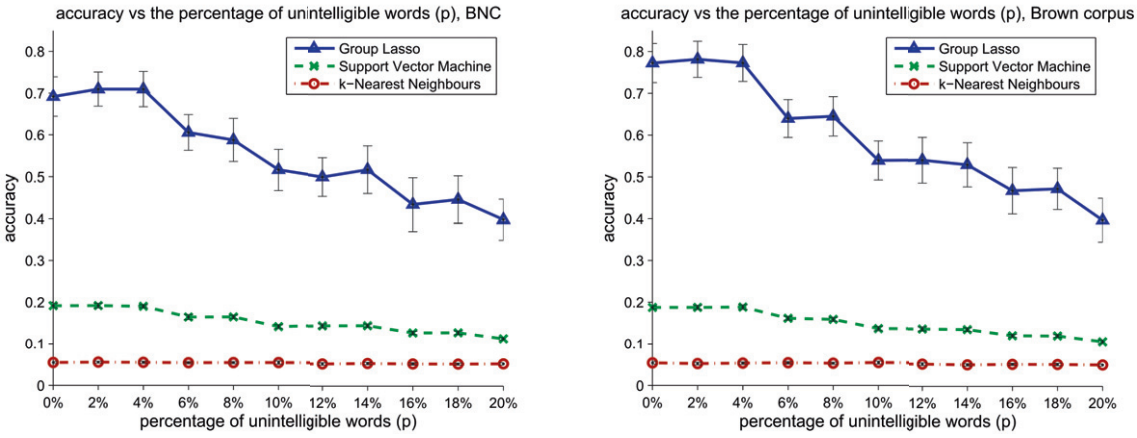

Fig. 3. The dependence of the accuracy on the percentage of unintelligible words in the text. Each data point is the mean of 30 experiments. The error bars denote the standard deviations.

Each data point is computed by random permutations cross-validation, also known as Shuffle and Split. In each experiment, the dataset is first shuffled, then split into an independent training and testing part, where the test dataset contains 100 elements. The baselines are trained on the training set and tested on the test set. For group Lasso, the labeled columns of the dictionary D are the elements of the training set, and the contexts of the unintelligible words are the elements of the test set. For each data point, we perform 30 such experiments, and report the mean and standard deviation in Fig. 3.

5. Discussion

The presented method based on group Lasso clearly outperforms the two baselines, the k-nearest neighbour classifier and the support vector machine. This may be because the sparsity inducing regularization decreases the chance of selecting contexts spuriously similar to the context of the unintelligible word (see Section 3).

To better understand how group Lasso utilizes the structure inherent in the problem, it is helpful to take a look at the representation computed by group Lasso, and one where each coordinate is computed by simple cosine similarity to the corresponding column of the dictionary:

$$\alpha_i = \frac{\langle \boldsymbol{d}_i, \boldsymbol{x} \rangle}{\|\boldsymbol{d}_i\|_2 \|\boldsymbol{x}\|_2}. \tag{4}$$

In the vector computed by group Lasso (Fig. 4 (a)), the whole group of the correct candidate word (see the approx. 951-1000$^{\text{th}}$ coordinates) is activated with positive weight, whereas in the other groups, usually only a fraction of the weights are activated, and the positive and negative weights cancel each other out. In the vector computed by cosine similarity (Fig. 4 (b)), most of the weights are activated, and the contrast is less sharp between the group of the correct candidate word and the other groups.

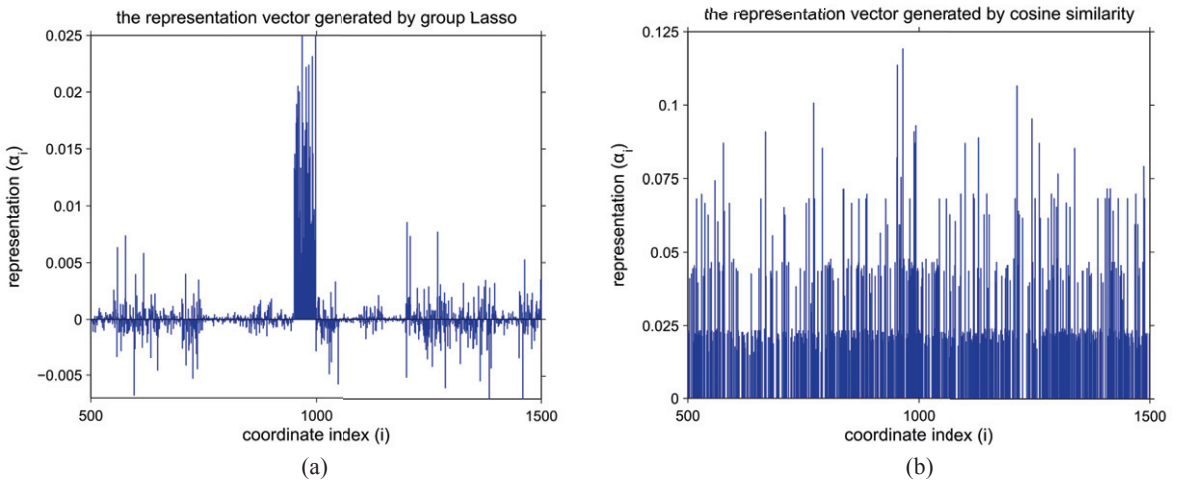

Fig. 4. A representation vector computed by group Lasso (a) and one where each coordinate is computed by cosine similarity (b). In the vector computed by group Lasso, the whole group of the correct candidate word (see the approx. 951-1000th coordinates) is activated with positive weight, whereas in the other groups, usually only a fraction of the weights are activated, and the positive and negative weights cancel each other out. In the vector computed by cosine similarity, most of the weights are activated, and the contrast is less sharp between the group of the correct candidate word and the other groups.

6. Conclusions

We proposed a method to determine unintelligible words in context. We have shown that a structured sparsity inducing regularization, more specifically, a group Lasso based solution can deal with spurious similarities of contexts and thus vastly outperform traditional classifiers like the support vector machine in this large-scale problem.

Unintelligible words are often the result of errors in Optical Character Recognition (OCR) algorithms. As the proposed method utilizes information independent from information used in OCR, we expect that a combined approach could be very successful, as OCR and the proposed method complement each other.

Acknowledgements

The research has been supported by the `European Robotic Surgery' EC FP7 grant (no.: 288233). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of other members of the consortium or the European Commission.

References

[1] Von Ahn, L., Maurer, B., McMillen, C., & Blum, M. (2008) reCAPTCHA: Human-Based Character Recognition via Web Security Measures. Science, 321, 1465 – 1468

[2] Harris, Z. (1954) Distributional structure. Word, 10 (23) 146 – 162

[3] Turney, P. D. & Pantel, P. (2010) From frequency to meaning: vector space models of semantics. Journal of Artificial Intelligence Research, 37 (1) 141 – 188

[4] Yuan, M. & Lin, Y. (2006) Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Societies, Series B, 68, 49 – 67

[5] Kukich, K. (1992) Techniques for automatically correctiong words in text. ACM Computing Surveys, 24 (2) 377 – 439

[6] Leacock, C., Chodorow, M., Gamon, M. & Tetreault, J. (2010) Automated Grammatical Error Detection for Language Learners. Synthesis Lectures on Human Language Technologies, Morgan & Claypool
[7] Hirst, G. & Budanitsky, A. (2005) Correcting real-word spelling errors by restoring lexical cohesion. Natural Language Engineering, 11 (1) 87 – 111
[8] Carlson, A. J., Rosen, J. & Roth, D. (2001) Scaling up context sensitive text correction. In Innovative Applications of Artificial Intelligence (pp. 45 – 50)
[9] Islam, A. & Inkpen, D. (2009) Real-word spelling correction using Google Web IT 3-grams. In Empirical Methods in Natural Language Processing (vol. 3, pp. 1241 – 1249)
[10] Kantor, P. B. & Voorhees, E. M. (2000) The TREC-5 Confusion track: comparing retrieval methods for scanned text. Information Retrieval, 2, 165 – 176
[11] Garofolo, J. S., Auzanne, C. G. P. & Voorhees, E. M. (2000) The TREC spoken document retrieval track: a success story. In RIAO-2000 (pp. 1 – 20)
[12] Jenatton, R., Mairal, J., Obozinski, G. & Bach, F. (2011) Proximal methods for hierarchical sparse coding. Journal of Machine Learning Research, 12, 2297 – 2334
[13] Martins., A. F. T., Smith, N. A., Aguiar, P. M. Q. & Figueiredo, M. A. T. (2011) Structured sparsity in structured prediction. In Conference on Empirical Methods on Natural Language Processing (pp. 1500 – 1511)
[14] Tibishirani, R. (1994) Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society, Series B, 58, 267 – 288
[15] Liu, J., Ji, S. & Ye, J. (2009) SLEP: sparse learning with efficient projections. Arizona State University.
[16] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. et. al. (2011) Scikit-learn: Machine learning in Pyton. Journal of Machine Learning Research, 12, 2825 – 2830
[17] The British National Corpus, version 2 (BNC World). 2001. Distributed by Oxford University Computing Services on behalf of the BNC Consortium. URL: http://www.natcorp.ox.ac.uk/
[18] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D. & Miller, K. (1993) Five papers on WordNet. Technical Report CLS-Rep-43, Cognitive Science Laboratory, Princeton University
[19] Lee, Y. K., Ng, H. T. (2002) An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In Conference on Empirical Methods on Natural Language Processing (pp. 41 – 48)
[20] Schutze, H. (1998) Automatic word sense discrimination. Computational Linguistics, 24 (1) 97 – 123