STUDIA INFORMATICA
Nr 1(10) Systemy i technologie informacyjne 2008

Text classification using word sequences

Paweł Chudzian[1]
[1] Institute of Electronic Systems, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
P.Chudzian@elka.pw.edu.pl

Abstract. The article discusses the use of word sequences in text classification. As opposed to n-grams, word sequences are not of a fixed length and therefore allow the classifier to obtain flexibility necessary to operate on documents collected from various sources. Presented classifier is built upon the suffix tree structure which enables word sequences to take part in classification process. During classification, both single words and longer sequences are taken into account and have impact on the category assignment with respect to their frequency and length. The Suffix Tree Classifier and well known Naive Bayes Classifier are compared and their properties are discussed. Obtained results show that incorporating word sequences into text classification can increase accuracy and reveal some interesting relations between maximal length of used sequences and classifier's error rate.

Keywords. Text classification, text representation, generalized suffix tree

1 Introduction

The problem of automatic text processing gained much importance in recent years. This popularity is strongly correlated with development of technologies making knowledge encoded in natural language highly available in digital form. Despite all advantages such technology requires effective methods of automatic text analysis in order to take full advantage of its capacities. Demand for such solutions stimulates rapid development of text mining methods in which text classification occupies special place.

Text representation is the main problem standing on the way of development of text mining techniques. Well known methods such as the vector space model [6] result in loss of information originally encoded with natural language. Moreover term-document matrix of document collections is mostly large and sparse and in case of vector space model with global statistics such as *tf-idf* [5] it needs to be rebuild every time new document arrives. On the other hand using grammars and other Natural Language Processing usually makes impossible to take full advantage of effective machine learning methods.

Classification based on the generalized suffix tree representation integrates machine learning techniques with approach to the text representation which carries on the loss information more than the standard representation. Goal of the

representation based on generalized suffix tree is to improve the classification by taking into account information about sequences of words. Representation proposed in this paper has ability to recover some additional ,,knowledge'' (which is lost in the vector space model) not excluding information about statistics of words in document, thereby having ability to explain how do the sequences of words impact classification accuracy. Suffix tree representation also allows knowledge base to be updated easily - without full rebuild - when new document arrives.

This paper is organized as follows: in section 2 suffix tree structure is introduced as an alternative for document representation and new algorithm using word sequences is presented. Section 3 shows the results obtained using the algorithm. In section 4 conclusions are presented.

2 Suffix tree classification

The vector space model and it's enhancements fail when it comes to take into account information about mutual position of terms in text. Additionally updating term-document matrix requires full rebuild when global statistics of terms are used. In this section a new method is proposed which addresses these issues.

2.1 Suffix tree structure

Suffix tree structure is widely used for encoding sequences, to mention only data compression [2,3] or pattern matching e.g. in DNA sequences [1,8]. It is structure which stores the information on every suffix of encoded sequence. In current form it was first presented in [8].

Suffix tree is defined as follows: denote a character sequence of length N built upon the alphabet Ω as x and denote terminal character as $\$$, such that $\$$ is not included in Ω. Suffix tree is structure which stores all suffixes of the sequence $x\$$. Number of leaves in the tree is matches number of all suffixes in $x\$$ and equals $N+1$. This property is guaranteed by the presence of terminal character $\$$, which existence is crucial in order to keep the structure correct.

Suffix tree can be easily generalized by building it with multiple sequences. It is only required that each one ends with unique terminal character. Number of leaves in such tree is equal to the total length of all sequences. Amount of branches starting at the root node corresponds to size of the alphabet used to create the tree. An example of generalized suffix tree is shown in figure 2.1 where two sequences are encoded.

It is allowed to use any alphabet while creating suffix tree. In figure 2.2. suffix tree is presented with sentences as sequences and alphabet consisting of words. As one can see, there are many branches starting from the root node – there are as many branches as unique units in the alphabet, which can be large when it consists of words.

Number of leaves in generalized structure is – like in case of the standard suffix tree – proportional to the total length of all sequences and - again - each sequence must end with unique terminal unit. Fundamental property of suffix tree is

its ability to search the entire structure for particular subsequence in time linearly dependent on length of this sequence.

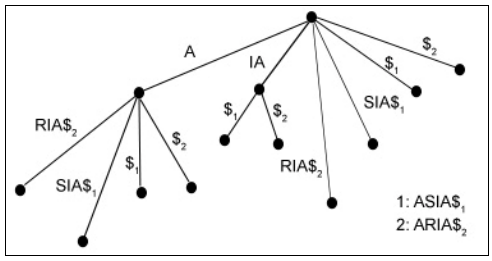

Figure 2.1. Generalized suffix tree

The suffix tree can be created with the algorithm proposed by Ukkonen [7]. It has linear time complexity and allows to update suffix tree with new sequences once the structure has been constructed. Detailed description of the algorithm can be found in [7], here it is only important to know that several tweaks can be made to improve effectiveness of the original algorithm.

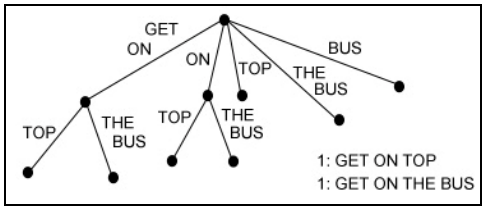

Figure 2.2. Suffix tree with words as elementary units of alphabet

2.2 Suffix tree classification

Suffix trees built upon single characters were already used in the document classification problems [4]. It performed better than standard text classification algorithms used for comparison. However it should be said that it was used in specific problem where standard algorithms are likely to fail and results on benchmark text collections were not presented.

The novelty of method proposed in this paper is that it uses words as the indivisible unit of the alphabet in order to take into account sequences of words rather than just single words. The advantage is that we consider more information than simple statistics of words occurrence.

Proposed classifier is flexible in context of category assignment. Once the suffix tree is constructed it is possible to change documents' categories without reconstruction of the entire structure. This is done by marking particular branch with document's identifier while it is being added to the structure. Figure 2.3. shows the suffix structure with two documents encoded. Each branch of the tree is marked with at least one identifier and edges encoding common terms coincide with identifiers of both documents.

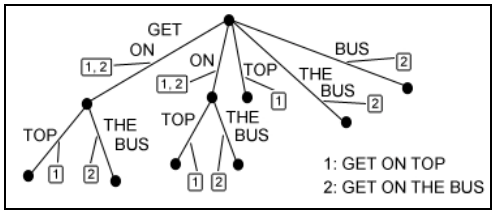

Figure 2.3. Suffix tree with branches marked with document's identifiers

The suffix tree is constructed only once for the particular document collection and there is one tree for all categories. Adding new documents to the suffix tree requires only starting the algorithm with current structure and new document as input.

Learning phase is finished once the suffix tree is constructed and information on documents linked with branches is stored. In classification phase words sequences from successive documents being classified and training examples encoded in the suffix tree are compared.

Classification of new document is done by repeating for each term in this document search for the longest sequences *s* which starts with term being currently processed and is enhanced while there is common sequence in the suffix tree. Once the longest sequence is found category scoring phase starts. Beginning from the longest sequence new reduced subsequences with the length from $|s|$ to 1 are built . Categories are then scored by function $f(s_j,C_i)$ with respect to the length of subsequence and number of documents of *i*-th category connected to *j*-th subsequence:

$$f(s_j, C_i) = \frac{1 + \left|d^{C_i}\right| \cdot f_s(|s|)}{M + N_i} \tag{1}$$

where $|d^{Ci}|$ is number of documents of category C_i containing sequence *s*, $f_s(|s|)$ is function which takes into account the length of sequence, *M* is size of the alphabet used to create the suffix tree structure and N_i is size of the alphabet from which documents of *i*-th category where derived. Function $f_s(|s|)$ was set to the sigmoid as it performed better in experiments than other functions (e.g. linear or other sigmoid functions):

$$f_s(|s|) = \frac{10}{1 + e^{-|s|+3}} \tag{2}$$

In order to take into account all found subsequences and normalize result with respect to the number of documents in each category the final score function is:

$$s(C_i, d) = P(C_i) \prod_{j=1}^{V} \prod_{k=1}^{|s|} \frac{1 + \left|d^{C_i}\right| \cdot f_s(\left|s_{jk}\right|)}{M + N_i} \tag{3}$$

where *V* is number of words in document *d*. Similarity of this function to the category probability estimate known from the bayesian classifier is not casual. First reason to choose such form of the function was to obtain a classifier similar to bayesian classifier in order to clearly see the effects of adjustment of word

sequences. Second issue was to put forward the scoring function which could enable clear interpretation of influence of word sequences on classifier's accuracy.

In the last stage of classification process a category which maximizes the score function (3) is chosen:

$$c(d) = \arg max_{C_i} s(C_i, d) \tag{4}$$

The influence of word sequences on the classifier's accuracy was examined and conclusions derived are presented in the following section.

3 Results

Experimental results were obtained using two classification algorithms: multinomial model of the Naive Bayes Classifier (hereafter NBC) and the Suffix Tree Classifier (hereafter STC). There were 2 well-known document collections used in experiments – *Reuters-21578*[1] and *20Newsgroups*[2]. The first collection consists of 21578 documents. Considerable part of documents in this set doesn't contain category information. After removing these documents and additionally articles which had empty body there were 7063 documents in the training and 2741 in the validation set left eventually. There were 117 categories and each document fell into at least one of these. Almost half of all categories were covering no more than 10 documents each and there were 2 categories which was describing approximately 65% of data set. Overall there were 9084 documents with 12219 category labels where 1512 documents were described with 2 or more categories' labels.

The second collection consisted of 19997 texts from 20 news groups and each group was considered individual category. Distribution of categories was practically uniform – each was represented by approximately 1000 documents. As there is no standard split for the training and validation sets in this collection the cross-validation (with same folds for both algorithms) was used.

3.2 Classification with word sequences

In order to evaluate influence of word sequences on classification error experiments were conducted on both document collections. Figures 3.1. and 3.2. shows classification error on Reuters and 20Newsgroups sets respectively with result obtained using NBC as reference.

Classification error of the STC was evaluated for different maximal lengths of sequence. Even for sequence of length 1 - single terms - error on the STC is less than corresponding error of NBC. Increasing maximal length of sequence causes further remission of error. One of reasons for the saturation of curve for longer sequences is shape of sigmoid function used for scoring.

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578/
[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

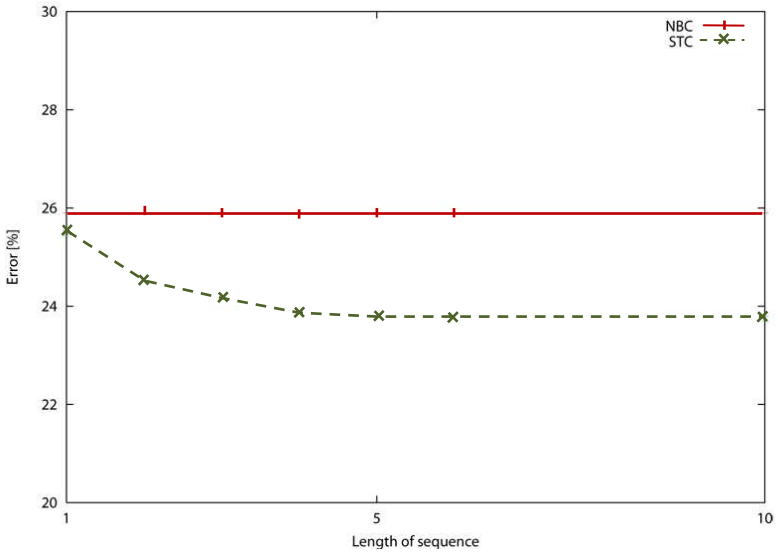

Figure 3.1. Classification error (Reuters collection)

The saturation effect in case of linear function was observed too and in addition the classifier was giving worse results for short sequences and classification processes lasted longer (in case of sigmoid function searching for sequences can be stopped when further lengthening of sequence doesn't change the score).

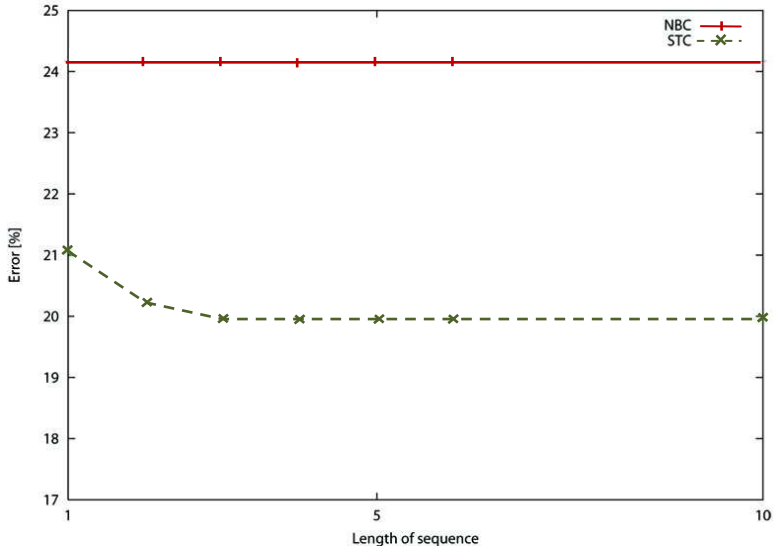

Figure 3.2. Classification error (20Newsgroups collection)

In case of 20Newsgroups collection increase of accuracy while lengthening length of sequence is not so well noticeable - effect of the saturation appears for shorter sequences. However general shape of curve remains the same and we can still observe that classifier's error decreases for longer sequences. Difference in STC and NBC errors is much more remarkable for 20Newsgroups set as the classifier built upon suffix tree gained result better by 4 percent.

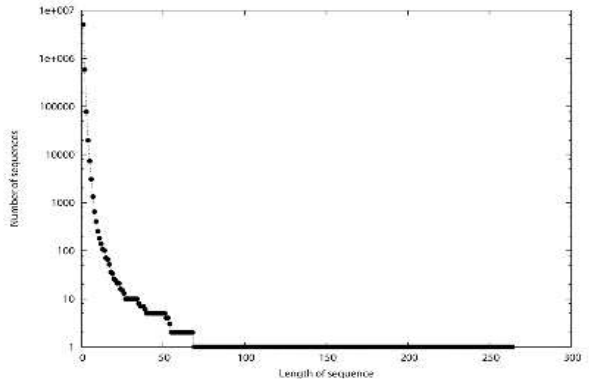

Figure 3.3. Number and length of sequence coinciding in suffix tree (Reuters)

Figures 3.1. and 3.2. show that classification accuracy is a function of length of maximal sequence. One of reasons for such relation is shown in figures 3.3. and 3.4. where number of sequences of particular length found while searching the tree are presented. Fractional values of sequences number for 20Newsgroups collection are consequence of using cross-validation.

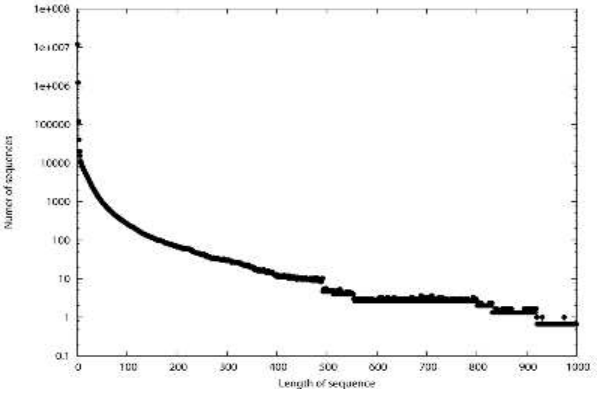

Figure 3.4. Number and length of sequence coinciding in suffix tree (20Newsgroups)

Number of single terms coinciding in the tree is approximately 10^7 for both Reuters and 20Newsgroups collections and this value is decreasing for longer sequences. However, it is still major – for sequence of length 6 number of sequences

is still bigger than number of documents in test collection. It means that statistically there is one sequence of length 6 in each test document which is coincided in the suffix tree (and respectively there are more coincidence for shorter sequences).

It is important observation that number of 2-word sequences is only 10 times smaller than number of single terms. It can be concluded that sequences of particular length are taken into account by the classifier as long as the occurrence frequency of such sequences exceeds some threshold, which in turn explains effect of the saturation of classifier's error.

For 20Newsgroups collection - unlike the Reuters set, where number of sequences of length greater than 70 was 1 - there are many common long sequences. The reason is specific character of 20Newsgroups collection, where documents with quotations of previous messages occur frequently. Alike Reuters collection some correlation between error rate and number of sequences of particular length can be noticed. Classifier's error curve saturates for such length of sequence for which curve presenting number of sequences loses its exponential character.

3.3 Suffix tree pruning

Discussion on suffix tree properties showed how large structures one have to deal with while using it in classification tasks. In order to counteract excessive growth of the suffix tree structure pruning strategy was applied. Branches representing sequences (or single terms) contained by less than defined number of documents where removed. Figure 3.5. shows how pruning strategy influences classifier's accuracy on Reuters collection.

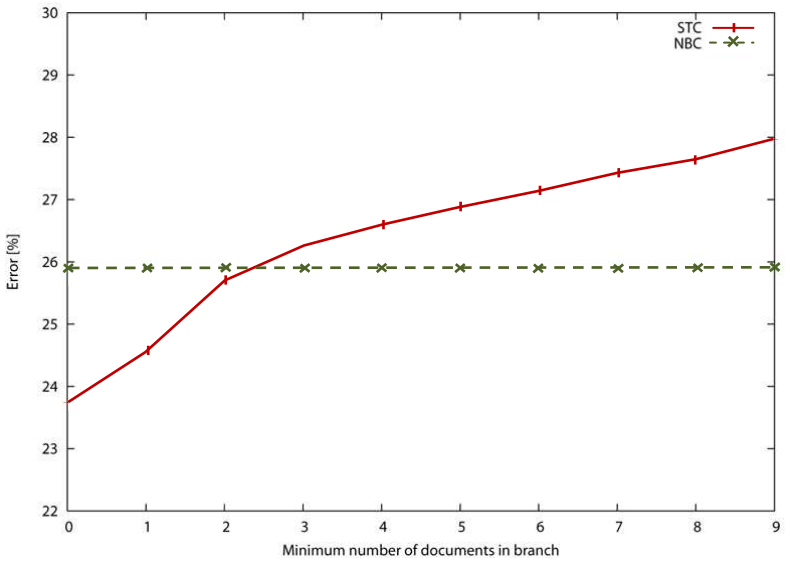

Figure 3.5. Classification error after pruning (Reuters collection)

It is apparent that classifier's error increases with threshold for minimal number of documents in the branch. In both cases error on the NBC is bigger even when branches with less than 3 documents are removed. Figure 3.6. shows that removing these branches gives tenfold reduction of the suffix tree size (number of branches). Further pruning results in increase of error rate and gained reduction of size is getting increasingly smaller.

Pruning the suffix tree for lower thresholds results in small error increase with large reduction of tree size at the same time. Observation of relation between size of the tree and classifier's error suggests that in many cases single sequences decide on category assigned to document being classified. Pruning branches connected with single document removes sequences occurring in one category and apparently have influence on assignments to this particular class. This means that in some cases decision on which category to use while labeling document is based on similarity to only one document.

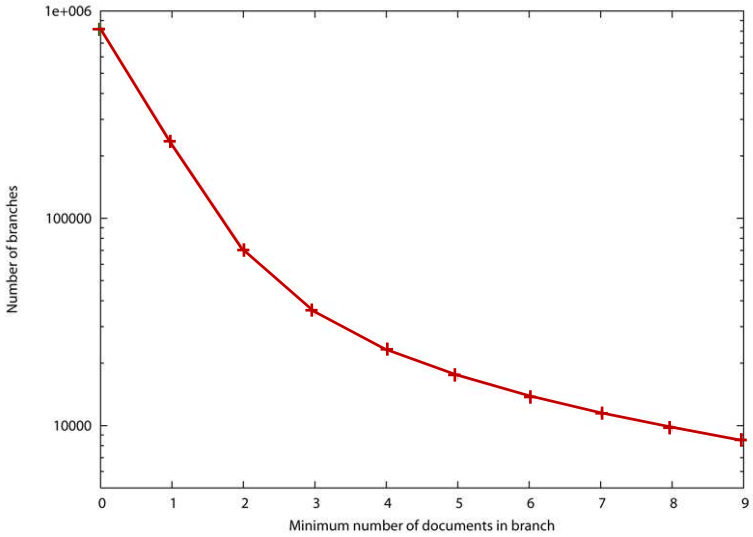

Figure 3.6. Size of the tree after pruning (Reuters collection)

On the other hand nearly logarithmic character of the relation shows that many of removed branches don't have strong influence on the category assignment thus belong to part of classifier's ,,knowledge" which is unused in the classification task.

3.3 Multi-class problem

Documents in Reuters collection may fall into more than one category, therefore experiments on multi-class categorization were conducted. Classification problem was decomposed into n binary problems (one for each category) and F-measure [9] was used to measure accuracy of classification:

$$F_1(p,r) = \frac{pr}{p+r} \tag{5}$$

where p – *precision* - is fraction of correct positive classification to all positive classification and r – *recall* – is fraction of correct positive classification to all correct classifications. Averaging of results from n independent classifiers can be done using two different approaches: *micro-averaging* or *macro-averaging.* In first statistics are calculated over categories and then averaged. In second approach statistics are calculated over classifier and averaged.

Figure 3.7. shows how F-measure changes in function of number of categories. The F-measure is decreasing with increasing number of classes considered in experiment. This is a consequence of occurrence of two categories labeling about 65% of all documents in collection. Classifiers tends to assign one of these two categories' labels to all documents. For remaining categories classifier labels documents as not adhering to them. Adding another category causes F-measure to decrease as there is another class and documents from this group will be incorrectly classified as not belonging to it.

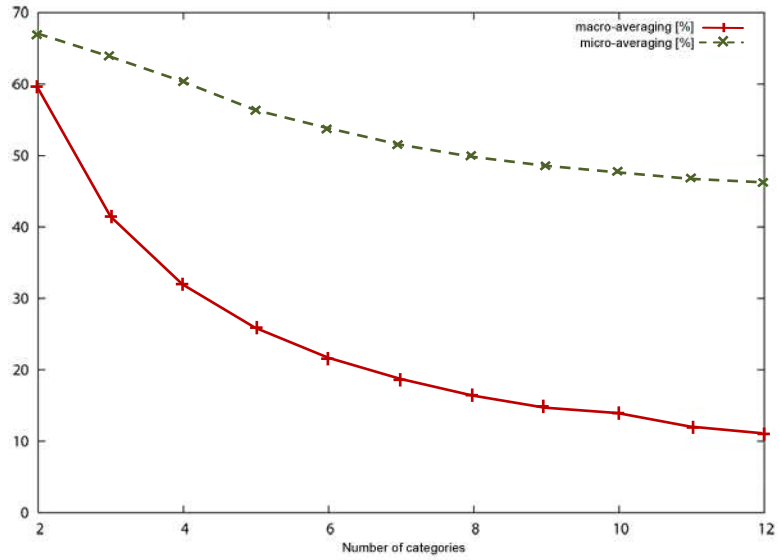

Figure 3.7. F-measure for different number of categories

4 Conclusions

Obtained results show that incorporating word sequences in classification process have great positive influence on the accuracy of such system. This influence is significant as the error rate of classifier decreases several percent while lengthening maximal sequence. Thus independence of terms in document assumed in

naïve bayesian classifier, although it often boosts effectiveness, shouldn't be overused as it is clearly untrue.

Beside accuracy improvement it is important to note the relation between length of sequences and accuracy - lengthening sequence results in increased accuracy and relation saturates only for longer sequences. The breakdown where the saturation appears means that length of sequence is correlated with both number of documents and size of the alphabet used in suffix tree creation process. However it should be mentioned that bias of size of the alphabet on linear time complexity is noticeable especially in classification phase. Pruning the tree allowed to decrease influence of size of the alphabet but also resulted in increase of error rate.

Structure of the suffix tree is easily extensible therefore new documents might be added to the tree without full rebuild. It is large improvement over vector space model representation which usually needs full rebuild of terms-documents matrix (when global statistics are used, e.g. tf-idf). Also connecting documents with branches and not particular terms allows to easily reformulate problem of classification changing only the classes of documents.

References

1. Grossi R., Italiano G. F. (1993). Suffix trees and their applications in string algorithms. In: *1st South American Workshop on String Processing*. 57-76.
2. Larsson J. (1998). The Context Trees of Block Sorting Compression. In: *Proceedings of the IEEE Data Compression Conference*. 189-198.
3. Na J. C., Apostolico A., Iliopoulos C.S., Park K. (2003). Truncated suffix trees and their application to data compression. In: *Theoretical Computer Science*. Vol. 304, No. 1-3, 87-101.
4. Pampapathi R. M., Mirkin B., Levene M. (2005). *A Suffix Tree Approach to Email Filtering*. Technical report.
5. Salton G., Buckley C. (1987). *Term Weighting Approaches In Automatic Text Retrieval*. Technical report.
6. Salton G., Wong A., Yang C. S. (1975). A Vector Space Model for Automatic Indexing. In: *Communications of the ACM*. Vol. 18, No. 11, 613-620.
7. Ukkonen E. (1995). On-Line Construction of Suffix Trees. In: *Algorithmica*. Vol. 14, No. 3, 249-260.
8. Weiner P. (1973). Linear Pattern Matching Algorithms. In: *Proceedings of 14th Annual Symposium on Switching and Automata Theory*. 1-11.
9. Yang Y., Liu X. (1999). A re-examination of text categorization methods. In: *Proceedings of {SIGIR}-99, 22nd {ACM} International Conference on Research and Development in Information Retrieval*. 42-49.