IMPROVING K-NEAREST NEIGHBOR EFFICIENCY FOR TEXT CATEGORIZATION

*F. Barigou**

Abstract: With the increasing use of the Internet and electronic documents, automatic text categorization becomes imperative. Many classification methods have been applied to text categorization. The k-nearest neighbors (k-NN) is known to be one of the best state of the art classifiers when used for text categorization. However, k-NN suffers from limitations such as high computation, low tolerance to noise, and its dependency to the parameter k and distance function. In this paper, we first survey some improvements algorithms proposed in the literature to face those shortcomings. And second, we discuss an approach to improve k-NN efficiency without degrading the performance of classification. Experimental results on the 20Newsgroup and Reuters corpora show that the proposed approach increases the performance of k-NN and reduces the time classification.

Key words: *text categorization, k-nearest neighbors, cellular automaton, efficiency*

Received: July 30, 2014 **DOI:** 10.14311/NNW.2016.26.003
Revised and accepted: March 5, 2015

1. Introduction

Text categorization (TC), the activity of labeling texts with predefined thematic categories, is a task which began in the early 60s [22]. In the field of information retrieval, TC becomes more and more important due to the increased documents available in electronic format and the need to access them in a flexible manner. TC is used in all applications requiring document management or document routing, for example, spam email filtering, documents indexing or web page classification [24, 27]. Formally, automatic text categorization is a supervised learning task, defined as identifying the class labels for new documents based on a training set of labeled documents [27]. A wide variety of machine learning techniques has been designed for text categorization, including decision trees [1], k-NN (k-Nearest Neighbors) [32], Bayesian probabilistic methods [11], neural networks [20], Support Vector machines (SVM) [19], boosting methods [12], and many other machine learning. Among them, the k-NN algorithm has shown great potential in text categorization. k-NN [6], is a very simple instance-based learning algorithm. Despite its simplicity, it can offer very good performance. Its major disadvantage is that k-NN requires more time for classifying objects when a large number of training examples are

*Fatiha Barigou, Computer Laboratory of Oran, Department of Computer Science, University of Oran 1, Ahmed Benbella, Algeria, PB 1524, EL M'Naouer, Es Senia, Oran, E-mail: fatbarigou@gmail.com

given. Several approaches have been suggested for improving k-NN algorithm [16]. Advanced storage structures, such as kd-trees, R-trees or ball-trees, are proposed by [21] to speed up the k-NN computation. Genetic algorithm is combined with k-NN in [29] to improve performance. Another successful technique known as instance selection is also proposed to face simultaneously, the efficiency storage and noise of k-NN. The authors in [13] gave a complete taxonomy of instance selection methods and presented an empirical study for analyzing those methods in terms of accuracy, data reduction and efficiency. However, the evaluation was done only on structured data sets from UCI repository. Motivated by these facts, through this paper, we discuss a new approach to improve k-NN speed without degrading the performance of text categorization.

We propose an original solution to overcome one of the major drawbacks of k-NN method, which is the cost classification in a text categorization task where we handle thousands of documents or even thousands of thousands. The principle of this method is as follows: when a new instance should be classified; instead of involving all learning instances to retrieve the k-neighbors which will increase the computing time, a selection of a smaller subset of instances is first performed.

Unlike other methods [25, 14, 18], our instance selection approach is dynamic in the sense that is applied each time a classification of a new instance is needed. We adopt the approach that we have already studied within the context of spam filtering application [5], this time; it will be around k-NN-based text categorization. We propose using the Boolean model of CASI [2], with some revisions to represent the training documents and use its inference engine to select relevant documents that will participate in the search of neighbors. In our previous work [5], the performance of the proposed approach was partially investigated; it was applied to spam filtering, and only accuracy classification was investigated. Our concern was to show that reducing the size of training instances will not degrade the performance of classification. Experiments performed on Ling-Spam corpus showed that the proposed approach achieves better classification accuracy compared to other published works in the field of spam filtering. In this work, we focus on k-NN efficiency. The main purpose of this paper is to show how the new approach will also significantly speed up the process of text categorization without degrading the performance of classification. Experiments on Reuters and 20NewsGroups showed that the proposed method is competitive in terms of predictive performance, while accelerating the time of classification.

The rest of the paper is organized as follows. The next section reviews some of the pertinent literature on k-NN improvements for text categorization. In Section 3, the new improvement of k-NN is presented and discussed. The results of experiments carried out on the proposed classifier, as well as on k-NN, are presented in Section 4. Discussions of results are given in Section 5. Lastly, the paper is concluded in Section 6.

2. k-NN improvements for TC

The k-nearest neighbor algorithm differs from other learning methods because no model is induced from the training examples. The data remains as they are; they are simply stored in memory. To decide the class of a unknown document, the

algorithm looks for its k-nearest neighbors and predicts the most frequent class of those k-nearest neighbors. Therefore, the method uses two parameters; the number k and a similarity function to compare the new instance with training instances.

This process has a few inherent problems:

- To determine the class of the unknown document, k-NN has to compare it with all the training documents. As a result, its efficiency will degrade when using a large repository [3].
- k-NN classification performance depends on choosing similarity function [10] and the appropriate number of neighbors, k [7, 3]. The optimal value of the parameter k is chosen by many tests. Nevertheless, this procedure is improper in some applications.
- k-NN classifier is noise tolerant since it uses all training data as relevant, even when training documents contain noise or unbalanced data [28, 37].

After examining some research papers that have tackled the k-NN improvement for text categorization, we consider useful to review, in this section, some of the pertinent literature on k-NN based text categorization improvements pinpointing their idea, their advantages, and their drawbacks.

2.1 Improving k-NN by neighborhood size and similarity function

In the traditional k-NN algorithm, the value of k is fixed beforehand. If it is large, big classes will overwhelm little ones. On the other hand, if k is small, the advantage of k-NN algorithm will not be exhibited. To be less dependent on the choice of k, an improved k-NN algorithm is proposed by [3]. The new algorithm uses different numbers of nearest neighbors for distinct categories, rather than a fixed number across all categories. More neighbors will be used for deciding whether a test document should be classified into a category, which has more examples in the training set. Experiments on Chinese text categorization show that their method is less sensitive to the parameter k than the traditional one. This approach concentrates on neighborhood size; however, it needs more computation.

To reduce unnecessary processing of cross-validation to find the optimal value of k, the approach proposed by [7] stops when the best value is found. The excessive processing of cross-validation is then reduced and therefore, time and space used for classification are also reduced. Unfortunately, the method is evaluated on small data set.

The authors of reference [10] are tempted to see if the use of measures other than cosine similarity can improve the performance of k-NN classification. They propose replacing the classical cosine similarity with a KL divergence based similarity measure. They make use of the relevance measures recently popularized in language modeling based document retrieval research to find the nearest neighbors [17]. Although experiments on Reuters Corpus Volume I (RCV1) and the 20 Newsgroups data set show that the new measure improves the classification results compared to the classical approach based on the cosine similarity measure, the approach needs more computation time.

2.2 Improving k-NN speed by reducing the number of training documents

As we said before, the traditional k-NN based text classification algorithm used all training documents for classification; it requires a high storage memory and a high degree of calculation complexity. To deal with these problems several improvements are proposed. A new technique known as the generalized instance set (GIS) algorithm is proposed by unifying the strengths of k-NN and linear classifiers [18]. The main idea is to construct a set of generalized instances (GI) that should replace the original training examples. Nevertheless, some drawbacks still exist. For example, it is hard to choose an appropriate number of clusters and the order in which positive instances are selected to construct local generalized instances. Extensive experiments were conducted on two large-scale document corpora, namely the Ohsumed collection and the Reuters-21578. All experimental results show that GIS outperforms traditional k-NN and Rocchio.

To speed up text categorization, a training-corpus pruning method is discussed in [14]. By using this technique the size of training corpus could be reduced while classification performance can be kept at a comparable level to that of without training-document pruning. Their technique is based on the fact that boundary documents of each class are more important in classification than inner documents. To reduce the size of training documents and therefore, enhance the classification efficiency they decide to discard inner documents and noisy documents. Experiments on Reuters show an improvement in the classification speed, but a degradation of less than 3% micro averaging performance is observed.

Through experiments performed on 20 NewsGroups corpus, [28] show that excluding outliers from the training data significantly improves k-NN classifier. At the training stage, the authors calculate the centre of each category and then form new categories by discarding outliers. They observed that training documents that are far away from the centre of its training category reduce the accuracy of classification. They consider them as noisy data and decide to discard them from training documents. The proposed method obtained 9.93% improvement over the original Centroid-based classification but needs more computation during the learning phase.

Another work is proposed by [36]. They used k-means algorithm to cluster each category and considered the cluster centres as the representative points. These centres become the new training samples, and a weight value is introduced for the new documents in different categories, which can indicate the different importance of each document. Experiments on Chinese texts confirmed the effectiveness of this algorithm but there are also some limitations in this algorithm, for example, how to determine the parameter k-value when clustering.

Another work similar to [18] is the work of [25] who proposed a Generalized Cluster Centroid based Classifier (GCCC) by integrating two classifiers the K-nearest-neighbors and the Rocchio. The proposed method mainly focuses on two points; one point is that clustering algorithm is used to strengthen the expressiveness of the Rocchio model; another one is that they employ the improved Rocchio model to speed up the categorization process of k-NN. Extensive experiments conducted on both English and Chinese corpora show that GCCC has a better

categorization ability than Rocchio, k-NN and SVM. However, it must be noted that the modeling stage is more time-consuming than k-NN and Rocchio.

Recently, Du and Chen [8] have proposed an effective strategy to accelerate k-NN classification. Their idea is based on a simple principle; usually, near points in space are also near when they are projected into a direction, which means that distant points in the projection direction are distant in the original space. Using this strategy, most of the irrelevant points can be removed when searching for the k-nearest neighbors of a query point, which greatly decreases the computation cost. Experimental results show that the proposed strategy extremely improves the time performance of the standard k-NN, with little degradation in precision. Specifically, it is superior in applications that have large and high-dimensional data sets.

2.3 Improving k-NN by sampling and weighting neighbors

K-nearest neighbor suffers from inductive biases. For examples, it takes the assumption that training data are evenly distributed among all categories. For unbalanced text corpora, the majority class tends to have more samples in the k-neighbors set for each test document. With a traditional k-NN, the new document tends to be assigned the majority class label. As a result, the big category tends to have high classification accuracy, while the other the minority class tends to have low classification accuracy. To deal with this problem some researchers proposed the sampling strategies [37]. However, the removal of training documents in large categories may lose some important information and tend to reduce the classification accuracy.

Tan [30] proposed Neighbour-Weighted k-Nearest neighbors for unbalanced text categorization problems. Instead of balancing the training data, his algorithm assigns a big weight for neighbors from small class, and assigns a little weight for neighbors contained in large category. To deal with unbalanced text corpora, this author proposed in [31] the DragPushing technique as a refinement strategy to enhance the performance of k-NN. He suggests a weight vector for each class and uses training errors for successively refining the k-NN classifier by dragging and pushing documents from these weighted vectors. The experiments on three benchmark evaluation collections showed that the proposed method could make a significant difference on the performance of the k-NN classifier and gave better performance than other five commonly used methods (winnow, C4.5, NB, centroid and k-NN).

2.4 Improving k-NN by advanced storage structures

To reduce document similarity computing, the authors in [32] used the SS-tree structure to index training documents. This technique is an improvement over k-NN in terms of speed. The leaves of the tree contain relevant information and internal nodes are used to guide efficient search through leaves. The method reduces the computation time because it does not need to search k-nearest neighbors in all training documents on the tree. Experiments on Reuters confirm the speeding. However, the performance of classification is not given. [23] proposed and developed

an improved k-NN algorithm which is faster than the classical k-NN algorithm while improving significantly the accuracy. He imposed a top-k buffer technique that can skip looking inside each and every training document and also each and every word in a document, which improves the performance of the algorithm. The author also proposed an improved decision rule to identify a class from k-nearest neighbour space by maintaining the classical k-NN property (majority votes) with penalizing the training samples which are far from the test sample, which can avoid any biasness from large dominating class in a data set and improves the accuracy.

2.5 Discussion

As pointed by Sebastiani in his survey on text categorization; k-NN has been applied to that field since the early days of its research [27] and it is shown to be one of the most effective methods on Reuters corpus. In this paper, we discussed the shortcomings of k-NN when applied to text categorization and gave an overview of improved techniques. According to our findings, we can draw the following conclusions:

- k-NN is a simple technique for text categorization, but its main drawbacks are its slowness during the classification and its dependency on both similarity function [10] and the parameter k [7, 3].
- Being a lazy learning method, k-NN is excluded from many applications such as online text categorization for a large deposit [15]. One way to improve its efficiency is to find some representatives to describe the whole training data for classification [8, 28, 14, 18]. Another way is to use fast structures to index training documents and then speed up searching k nearest neighbors [32].
- Various improvement techniques were proposed in the literature, but clear conclusion about comparison of those techniques is still difficult because the published results are not directly comparable, because different performance measures and different data collection with different sizes are used.

3. k-NN based on cellular automaton

As we have said in introduction section, text categorization with k-NN algorithm is known to be computationally expensive as it needs to compare the new instance with all training documents. To reduce this complexity, it was proposed previously the use of the cellular automaton CASI to represent the training documents and to select those to be used by k-NN algorithm [5]. In that work, the performance of the proposed approach was partially investigated. The approach was applied to spam filtering, and only accuracy classification was investigated. Our concern was to show that reducing the size of training instances will not degrade the performance of classification. Experiments performed on Ling-Spam corpus[1] showed better classification accuracy compared to other published works in the field of spam filtering. In this work, we focus on k-NN efficiency. The main purpose of this paper is to

[1] `http://www.csmining.org/index.php/ling-spam-datasets.html`

show that the proposed approach can also significantly speed up the process of text classification without degrading the performance of classification. The idea of this method is as follows: instead of engaging all training instances for the research of k-neighbors which will increase the computing time, selecting a reduced subset of instances is first performed. We consider two contributions. First, we extend the approach for text categorization and not just spam filtering. And secondly, we describe the results of extensive experiments using two different document collections. We evaluate the method in two ways: the predictive performance and the time of classification. The proposed system consists of three modules; a module to preprocess training documents, a module to create the Boolean text representation based on cellular automaton and a module to select training documents that should be used by k-NN classifier.

3.1 Cellular automaton CASI

Training documents will be modeled according to the principle adopted by the cellular automaton CASI [2], to reduce the complexity of storage and the response time during their use. Before giving more details, we will first recall the principles of the cellular automaton CASI (for details see [2]).

A cellular automaton is a grid of cells, which change their state in discrete steps. After each step, the state of each cell is modified according to those of its neighbors before that step [33]. The cells are updated in a synchronous way and the transitions are carried out, in theory, simultaneously [26]. Some of the key concepts for cellular automata are vicinity, parallelism, determinism, homogeneity, discritization and transition function.

CASI (Cellular Automata for Symbolic Induction) is a cellular method proposed by [2] as a model to represent and optimize induction graphs generated from learning examples. It is composed of three modules: COG (Cellular Optimization and Generation), CIE (Cellular Inference Engine), and CV (Cellular Validation). In this work, we are interested by the CIE component; a cellular automaton that is made of two finite arbitrary long layers of finite state machines (cells) that are all identical. CIE, which is the core of the CASI machine, simulates the running of the basic cycle of an inference engine by using two finite layers of finite automata to represent knowledge. A first layer, called CELFACT, represents the fact base, and a second layer, called CELRULE, represents the rule base. Each cell, at time $t+1$, depends only on the state of its neighbors and its state at time t. In each layer, the content of a cell determines whether and how it participates in each inference step: at each step, a cell can be active (1) or passive (0), that is to say, it participates or not in the inference. The states of cells are composed of three parts: EF, IF and SF, and ER, IR and SR which are the input, internal state and output parts of the CELFACT cells, and of the CELRULE cells, respectively. The internal state of a CELFACT cell indicates the fact role: IF = 0 corresponds to a fact of the form node of the graph (Ni), IF = 1 corresponds to a fact of the form attribute=value. In a CELRULE cell, IR can be used as a probability coefficient. We will not use it in this work. To illustrate the architecture and operating principle of the CIE module, we consider the portion of the graph, taken from the article [4], obtained using the partitions $P0 = \{N0\}$, $P1 = \{N1, N2\}$, $P2 = \{N3, N4\}$, $P3 = \{N5\}$ (see Fig. 1).

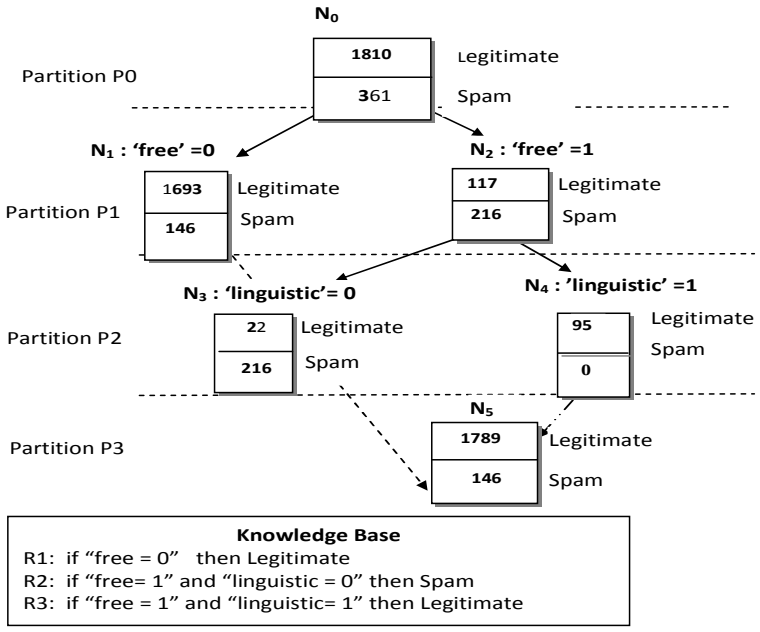

Fig. 1 *Induction graph composed of four partitions P0, P1 P2, P3.*

During the learning phase, the Sipina method produces a graph. From this graph, a set of rules is inferred. They are in the form of "if premise then conclusion". For example, in the graph of Fig. 1, we have the rule *if the term 'free' absent (= 0) in the email then the email is legitimate* (majority class of the node N1 in partition P1).

In the cellular automaton CASI, this set of rules is modeled as follows:

- A Boolean facts base, **CELFACT**, contains all the premises and conclusions facts of such rules (e.g. "free" = 0; "free" = 1; class = legitimate; class = spam ...).
- A Boolen Rule-based, **CELRULE**, contains all the rules.
- An input matrix, **RE**, memorizes premises of the rules.
- An output matrix, **RS**, memorizes conclusions of the rules.

Forward chaining will allow the model to move from initial configuration to the next configurations G(1), G(2) ... G(i). The inference stops after stabilization with a final configuration. At this step, the construction of cellular model is complete. Fig. 2 presents the final configuration corresponding to the example of Fig. 1. Three rules, represented by CELRULE layer are deduced from the graph. The premises and conclusions of these rules are stored in CELFACT layer. The vicinity is introduced by the notion of the incidence matrix. In the input matrix, RE (respectively output matrix, RS), is stored the premises (respectively the conclusions) of each rule. The rule R1, for example, has as a premise "free = 0", and as a conclusion "class = Legitimate". Interaction between these two layers is done by $\delta fact$ and $\delta rule$.

	CELRULE		
	ER	IR	SR
R1	0	1	0
R2	0	1	0
R3	0	1	0

	CELFACT		
	EF	IF	SF
"free=0"	0	1	0
"free=1"	0	1	0
"linguistic=0"	0	1	0
"linguistic=1"	0	1	0
N3:class=Spam	0	1	0
N5:class=legitimate	0	1	0

	RE Input Matrix		
	R1	R2	R3
"free=0"	1	0	0
"free=1"	0	1	1
"linguistic=0"	0	1	0
"linguistic=1"	0	0	1
N3:class=Spam	0	0	0
N5:class=legitimate	0	0	0

	RS Output Matrix		
	R1	R2	R3
"free=0"	0	0	0
"free=1"	0	0	0
"linguistic=0"	0	0	0
"linguistic=1"	0	0	0
N3:class=Spam	0	1	0
N5:class=legitimate	1	0	1

Fig. 2 *CELRULE, CELFACT Input and Output incidence matrices of Fig. 1.*

- The input relation, noted iRj, is formulated as follows: if (fact i $\in$ Premise of rule j) then $iREj = 1$ else $iREj = 0$.
- The output relation, noted $iRSj$, is formulated as follows: if (fact i $\in$ Conclusion of rule j) then $iRSj = 1$ else $iRSj = 0$.

The cellular automaton dynamics implements the CIE module as a cycle of an inference engine made up of two local transitions functions $\delta fact$ and $\delta rule$, where $\delta fact$ corresponds to the evaluation, selection and filtering phases, and $\delta rule$ corresponds to the execution phase.

1. The transition function $\delta fact$ is defined as

$$\delta\text{fact}(\mathbf{EF}, \mathbf{IF}, \mathbf{SF}, \mathbf{ER}, \mathbf{IR}, \mathbf{SR}) \rightarrow (\mathbf{EF}, \mathbf{IF}, \mathbf{EF}, \mathbf{ER} + (\mathbf{RE}^T \times \mathbf{EF}), \mathbf{SR}). \tag{1}$$

2. The transition function $\delta rule$ is defined as

$$\delta\text{rule}(\mathbf{EF}, \mathbf{IF}, \mathbf{SF}, \mathbf{ER}, \mathbf{IR}, \mathbf{SR}) \rightarrow (\mathbf{EF} + (\mathbf{RS} \times \mathbf{ER}), \mathbf{IF}, \mathbf{SF}, \mathbf{ER}, \mathbf{IR}, \overline{\mathbf{ER}}). \tag{2}$$

3.2 Preprocessing

The first step in the process of constructing a classifier is to produce from samples of texts of labeled document a format appropriate for the classification algorithms. We used the vector space model (VSM). We establish an initial list of terms by performing a segmentation of text into words, eliminate stop words using a pre-defined stop list and apply the Porter algorithm to perform stemming of the different retained words. Since the number of terms after this preprocessing phase is very high, and to reduce the computational cost and improves the classification performance, we select those that best represent the documents and remove fewer informative and noisy ones. We use the gain information [35] as a feature selection method.

Once the Vocabulary (V) is built and reduced, we obtain a document-by-word matrix ($N \times M$), where N is the number of training documents ($D = d_1, d_2, \ldots, d_N$) and M is the number of terms that were selected to represent vocabulary. Each document d_i is represented by the characteristic vector $\overrightarrow{d_i} = (w_{i1}, w_{i2}, \ldots, w_{iM})$ where w_{ij} is a TFIDF[2] weighting.

3.3 Boolean representation

The vector space model representation is transformed into a Boolean representation according to the following principle:

- The terms of the index are used to create the premise of the rule.
- The training documents are used to create the conclusion of the rule.

Thus, this modeling will produce the cellular rules CR_i as

$$CR_i : \textit{if premise then conclusion.}$$

To represent documents with the Boolean model some changes have been established at CIE component. We have defined three layers instead of two as follows:

- The first layer called CELRULE represents the set of rules.
- The second layer called CELTERM represents the terms of index.
- The third layer called CELDOC represents training documents.

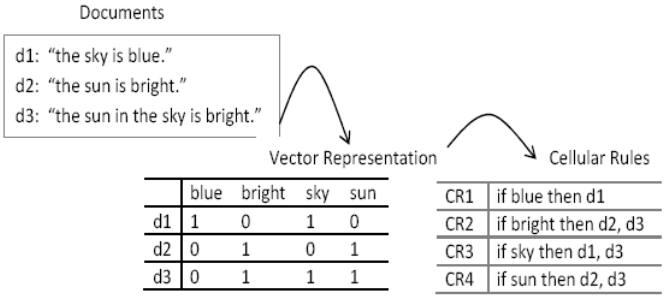

Fig. 3 *From documents to cellular rules.*

To clarify the idea of that work, we consider the documents d_1, d_2 and d_3 of Fig. 3. For this illustration, a binary weighing is used; if an index term occurs in the document, its value in the vector representation is one, otherwise it is zero. Using the Boolean representation we have four rules where each one is associated with an index term and tell us documents that contain that term. For example the rule CR_2 indicates that the term *bright* is found in d_2 and d_3. Fig. 4 shows the 3 layers modeling the cellular rules of Fig. 3.

[2]TFIDF= Term Frequency * Inverse Document Frequency

	CELRULE		
	ER	IR	SR
CR1	0	1	1
CR2	0	1	1
CR3	0	1	1
CR4	0	1	1

	CELTERM		
	ET	IT	ST
blue	0	1	0
bright	0	1	0
sky	0	1	0
sun	0	1	0

	CELDOC		
	ED	ID	SD
d1	0	1	0
d2	0	1	0
d3	0	1	0

Fig. 4 *The three layers used for documents modelling.*

	IM			
	CR1	CR2	CR3	CR4
blue	1	0	0	0
bright	0	1	0	0
sky	0	0	1	0
sun	0	0	0	1

	OM			
	CR1	CR2	CR3	CR4
d1	1	0	1	0
d2	0	1	0	1
d3	0	1	1	1

Fig. 5 *Input and output matrices.*

The neighborhood of cells is defined by two incidence matrices called **IM**, **OM** respectively. They represent the input respectively output relation and are used in forward chaining during instance selection. The **IM** matrix is $M \times M$ of dimension, while **OM** matrix is of dimension $N \times M$ (see Fig. 5). Matrices are constructed as follows:

- Input relation: **IM**
  $\forall t \in \{t_j \setminus t_j \in \text{CELTERM}; j = 1 \dots M\}$
  $\forall R \in \{\text{CR}_j \setminus \text{CR}_j \in \text{CELRULE}; j = 1 \dots M\}$
  IF (t premisse of R) THEN $\mathbf{IM}(t, R) = 1$ otherwise 0

- Output relation: **OM**
  $\forall d \in \{d_i \setminus d_i \in \text{CELDOC}; i = 1 \dots N\}$
  $\forall R \in \{\text{CR}_j \setminus \text{CR}_j \in \text{CELRULE}; j = 1 \dots M\}$
  IF (d conclusion of R) THEN $\mathbf{OM}(d, R) = 1$ otherwise 0

3.4 Instance selection

Prior to the classification of a new instance, we use the Boolean inference engine of CASI automaton to determine among all the training documents those which are relevant to take part in the classification of a new unlabelled instance. This process, called instance selection, allow us to determine the contribution of each training document for classification of a new one. This selection derives from the next hypothesis: *The learning document that has a larger number of common terms with the new unlabelled document to be classified is more relevant, it will have more impact on the classification performance.*

Before describing the selection process, we will first state the following definitions:

Definition 1. We consider Q the new unlabelled document to be classified and $N(Q)$ the number of different indexing terms found in Q.

Definition 2. We define $|d \cap Q|$ as the total number of terms in common with Q and a training document d.

Definition 3. Training document d is relevant for classification if it satisfies the following condition: $|d \cap Q| \geq T(\eta, Q)$ where $T(\eta, Q) = \lceil N(Q)/\eta \rceil + 1$ and $\eta \geq 2$.

For example, with $\eta = 2$, the selected documents are those sharing with Q at least half the number of terms of Q.

Instance selection is done in three steps. First, the $TERM$ layer is initiated by activating the **ET** states corresponding to the indexing terms belonging to the new instance Q. Then in the second step, the Boolean inference is performed by applying the global Boolean function $\delta\,\text{fact} \bullet \delta\,\text{rule}$ (cf. Eq. (3) and Eq. (4)). This will allow us to retain only instances sharing at least one common term with Q:

$$\delta\,\text{fact}(\mathbf{ET}, \mathbf{ST}, \mathbf{ER}, \mathbf{SR}) \rightarrow (\mathbf{ET}, \mathbf{ET}, \mathbf{ER} + (\mathbf{IM}^{\mathrm{T}} \times \mathbf{ET}), \mathbf{SR}), \tag{3}$$

$$\delta\,\text{rule}(\mathbf{ED}, \mathbf{SD}, \mathbf{ER}, \mathbf{SR}) \rightarrow (\mathbf{ED} + (\mathbf{OM} \times \mathbf{ER}), \mathbf{SD}, \mathbf{ER}, \overline{\mathbf{ER}}). \tag{4}$$

After applying these two functions, we obtain a new reduced set of training documents. We call this set $\mathcal{A}$. This set consists of all documents whose **ED** states in the DOC layer become active after execution of $\delta\,\text{fact} \bullet \delta\,\text{fact}$.

In the next step, we calculate for each instance d_i belonging to $\mathcal{A}$, the number of common terms with Q by applying the logical operator AND between the **ED** vector of the DOC layer and the vector $\mathbf{OM}(d_i)$. Finally, the threshold $T(\eta, Q)$ is applied to further reduce the set $\mathcal{A}$ and get the set E. Once the subset $\mathcal{E}$ is calculated, it is used by the k-NN classifier instead of the entire training set $\mathcal{D}$. If the obtained set is less than the number of k neighbor's size, we consider the set $\mathcal{E}$ as the set of neighbors with $k = |\mathcal{E}|$.

It is interesting to note in the algorithm 1 that the k nearest neighbour's classification is done only with the subset $\mathcal{E}$ which is supposed to be representative, and at the same time it consists of a very small number of instances compared to the initial set $\mathcal{D}$.

3.5 k-NN classification

To classify a new document Q, the k-NN ranks the documents neighbors among the training documents, and uses the class labels of k most similarity neighbors to predict the class of Q. To measure the similarity S, we make use of the cosine function as follows:

$$S(Q, d_i) = \frac{\sum_{t \in V} w(Q, t) \times w(t, d_i)}{\sqrt{\sum_{t \in V} w(Q, t)^2} \sqrt{\sum_{t \in V} w(t, d_i)^2}}. \tag{5}$$

Here, $w(t, Q)$ and $w(t, d_i)$ are the weights of term t in Q and d_i respectively. The score of classes of these neighbors are calculated using the similarity of each neighbor to Q as

$$\text{score}(c_j) = \sum_{\substack{d_i \in k\text{-NN}(Q), \\ c_j \in \mathcal{C}}} S(Q, d_i) \times y(d_i, c_j), \tag{6}$$

where k-NN(Q) indicates the set of K-nearest neighbors of document Q, $\mathcal{C}$ stands for the set of different classes and $y(d_i, c_j)$ stands for the classification for document

d_i with respect to class c_j, that is

$$y(d_i, c_j) = \begin{cases} 1 & \text{if } d_i \text{ is of class } c_j, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Consequently, the decision rule in k-NN classification can be written as

$$c(Q) = \arg\max_{c_j \in \mathcal{C}} (\text{score}(c_j)). \tag{8}$$

As illustrated in Fig. 6, in our approach, the k-NN classifier ranks the documents neighbors among the reduced training documents (see Algorithm 1).

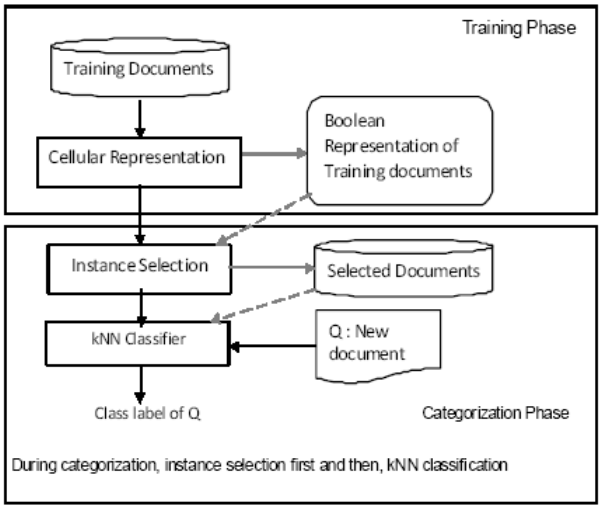

Fig. 6 *Principle of the approach.*

In our approach, to classify a new document, first, we reduce the set of training documents and after that we apply k-NN classification. The proposed approach allows gaining compression, which leads to improved time classification. The main computation with k-NN classification is the on-line scoring of training documents given a test document Q to find its k nearest neighbors. computing cosine similarities between the document Q and training documents can be done in $O(NM)$; where N is the number of training documents, and M is the number of index terms. The sorting of the N similarities takes $O(N \log(N))$. Accordingly the total running time is $O(T(N \log(N) + NM))$.

In our case, we use a Boolean inference to select from training base only relevant instances. We handle Boolean matrices and Boolean vectors where multiplication is replaced by the logical AND operator, and the addition operation is replaced by the logical OR operator. So we have a very low complexity for calculating the product of $\mathbf{IM}^{\mathrm{T}}$ with $\mathbf{ET}$ and $\mathbf{OM}$ with $\mathbf{ER}$ (considering that AND and OR are in constant time operations). The instance classification with the reduced training documents is executed in $O(kM|\mathcal{E}|^2)$. Since $|\mathcal{E}| \ll |\mathcal{D}|$, this time will be very small. In addition, if the size of the set $\mathcal{E}$ is less than k, the complexity is of the order of $O(k)$, which is negligible compared to $O(T(N \log(N) + NM))$.

Algorithm 1 Classification with our approach $(\mathcal{E}, Q, \mathcal{C}, k, S)$.

Require: $\mathcal{E}$: set of selected training documents, Q: new unlabelled document, $\mathcal{C}$: set of categories, k: neighbor's number, S: cosine similarity measure.
Ensure: $N_k(Q)$ the k-neighbors of instance Q
  if $|\mathcal{E}| \leq k$ **then**
    $N_k(Q) \leftarrow \mathcal{E}$
  else
    for document d in $\mathcal{E}$ **do**
      Calculate $S(d, Q)$
    end for
    Sort and select the k neighbors
    $N_k(Q) \leftarrow \arg\max_{d \in \mathcal{E}} S(d, Q)$
  end if
  for all category c_k in $\mathcal{C}$ **do**
    Compute its score in the set $N_k(Q)$
  end for
  Assign to Q the category with the best score

4. Experimental study and results

The performance of the approach discussed in this paper has been tested with two different data sets downloaded from the web site `http://web.ist.itl.pt/~acardoso/datasets`. The first data set is the *Reuters-21578* data set that consists of documents collected from the Reuters newswire. We used the $R8$ version with 8 categories. The categories are also not evenly distributed; some categories have few documents while others may have many documents. The second one is a version of the 20 Newsgroups data set that contains 18, 828 documents and 20 categories that are almost evenly distributed over the documents.

4.1 Performance measures

For evaluation performance over categories, we used macro-averaging. The proposed approach is evaluated along two axes: the predictive performance (i.e. the rate of correct classification in terms of precision (P), recall (R) and F-measure (F)), and time classification (T). For this approach, we consider the time classification of a new unlabelled document as the sum of time of instance selection and the time of classification of this document.

In the case of Reuters corpus, experiments were performed with 70% of the training documents (5483 documents) and 30% (2192 documents) for the test. In the case of 20Newsgroups corpus, experiments were performed with 80% of the training documents (16899 documents) and 20% (1929 documents) for the test.

4.2 Empirical results

We studied the performance of classification by varying the number of terms and the parameter η. The results indicate that the best performance is obtained when $\eta = 5$

with 600 terms in the case of *Reuters* and 700 terms in the case of *20NewsGroups*. Tab. I and Tab. II include the results of the k-NN classification performance on unseen documents of the Reuters and 20Newsgroups corpora by using the standard k-NN and the proposed approach with $\eta = 5$.

k	**R**[%]	**P**[%]	**F1**[%]	**T**[%]	**Technique**
1	86.48	84.42	85.89	483	Proposed
	83.23	83.12	83.24	1392	k-NN
5	88.66	89.10	88.88	492	Proposed
	83.06	87.00	84.98	1442	k-NN
10	86.82	91.13	88.92	510	Proposed
	82.99	86.27	84.60	1743	k-NN
21	89.23	92.27	90.73	532	Proposed
	83.38	88.75	85.98	1748	k-NN
30	89.48	93.28	91.34	570	Proposed
	85.12	92.17	88.50	1753	k-NN
60	88.58	92.72	90.61	572	Proposed
	85.12	93.02	88.90	1768	k-NN
100	80.81	91.88	85.99	575	Proposed
	83.76	93.66	88.43	1788	
200	77.65	90.68	83.66	588	Proposed
	76.24	92.35	83.53	1830	k-NN

Tab. I *Results of classification of 2191 Reuters documents with* $\eta = 5$ *by varying the neighborhood* k*.*

The new results appear to be significantly better; from Tab. I and Tab. II we disclaim two results:

- The first result relates to the effectiveness of the approach. The quality of k-NN classification becomes better after instance selection when compared to that of the original input. The best results are obtained when $k = 30$ in the case of Reuters corpus and $k = 5$ in the case of *20NewsGroups* corpus. We obtained a macro-F1 measure equal to 91.34% for the first corpus and 78.83% for the second corpus.
- The second result is a reduction of the time of classification obtained through the drastic reduction of training instances.

Fig. 7 and Fig. 8 indicate that the differences between results before and after applying the proposed approach, are significant enough. For example, we observe in Fig. 7, when k is equal to 1; the approach needs only 1037 seconds (approximately 17.28 minutes) to classify 1929 documents of *20NewsGroup*[3] but the k-NN classifier requires 5275 seconds (87.92 minutes) a difference of 70.64 minutes.

[3]We point out that this time includes the time of instance selection and time of classification

k	R[%]	P[%]	F1[%]	T[%]	Technique
1	78.70	78.76	78.73	1037	Proposed
	75.82	76.10	75.96	5275	k-NN
5	78.66	78.99	78.83	1097	Proposed
	76.76	77.13	76.95	5424	k-NN
10	78.18	78.93	78.55	2040	Proposed
	75.78	76.45	76.12	5585	k-NN
18	75.40	75.76	75.58	2044	Proposed
	72.77	73.27	73.02	5724	k-NN
30	76.06	77.47	76.76	2058	Proposed
	72.16	72.99	72.57	6450	k-NN
60	73.91	74.96	74.43	2062	Proposed
	71.19	71.76	71.48	6734	k-NN
100	74.86	77.56	76.19	2084	Proposed
	72.08	73.15	72.61	8356	k-NN
200	74.04	77.21	75.59	2095	Proposed
	71.20	72.80	71.99	10063	k-NN

Tab. II *Results of classification of 1929 documents from 20NewsGroups with* $\eta = 5$ *by varying the neighborhood* k.

Test document	No. of training documents Proposed	k-NN	Time of classification [μs] Proposed	k-NN
001900	39	5483	45651.27	734752.99
001746	2190	5483	365763.90	770345.89
001038	2790	5483	479076.70	764007.28
000380	79	5483	48350.38	773454.72

Tab. III *Time classification of test documents.*

These interesting results are obtained through the reduction of the training set before the search of the k nearest neighbors. When $\eta = 5$, we have noticed that the average reduction is equal to 88% over *20NewsGroups*, and 73% on *Reuters*.

To see better the impact of this reduction on the time cost we give in Tab. III, some results obtained during the classification of *Reuters* documents. Consider the document labeled 001900 of Tab. III, which is a test document from the *Reuters* database, we see clearly that reducing the number of training documents is very significant. While the traditional method uses the entire training set (5483 instances) to classify this document, the proposed approach will retrieve the k nearest neighbors from the reduced subset composed of only 39 documents instead of doing it from the initial set composed of 5483 documents. And as mentioned in Section 3.4,

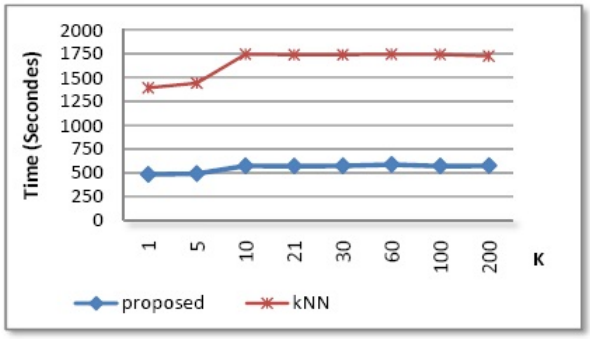

Fig. 7 *Time classification of 2191 test documents of* Reuters *by varying k.*

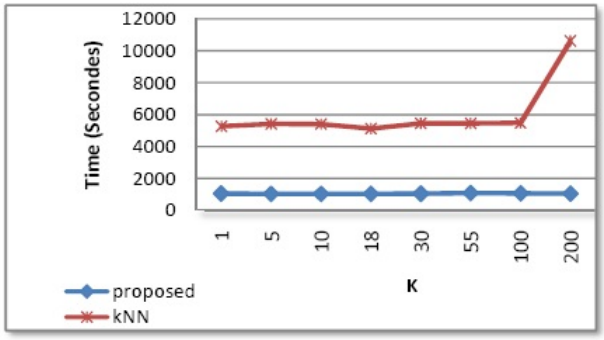

Fig. 8 *Time classification of 1929 test documents of* 20NewsGroups *by varying k.*

this subset of training documents shares at least $T(\eta, Q)$ terms with this new document. This reduction explains why the proposed method took 1/16 less time than conventional k-NN to categorize this document.

5. Discussion of results

In this work, we have focused on retrieving documents whose similarities are very close to the new unlabelled document Q. The proposed approach aims to avoid computing cosine scores for all training documents in the collection; as a result, it allows decreasing time classification while preserving classification accuracy.

Consider the Eq. (5) of Section 3.5, the idea of cosine similarity between two documents is based on the number of terms (or words) occurring in both documents. This function will give a minimum value of 0 for each document, which does not share any terms with Q and a maximum value of 1 to those documents that share all the terms with Q. On the other hand, to classify a new unlabelled document Q, its k closest neighbors $d^1, d^2, \ldots, d^k$ are found and a majority vote is conducted to assign the most common class to Q. That is, the class of Q is determined by the Eq. (8) given in Section 3.5. Nevertheless, as we know, this equation is affected

by the sensitivity of the selection of the neighborhood size k. With a big value of k the classification performance will degrade with the introduction of the outliers from other classes. So, we want a means by which we can pick off only those documents that share with Q a higher number of terms and exclude the others. To handle this problem we use the Definition 3 set out in Section 3.4. The proposed approach is intended to select the set of documents $\{d\}$ whose terms in common with the new unlabelled document $(d \cap Q)$ is greater than $T(\eta, Q)$. For example, if $\eta = 5$ and $N(Q)$ is equal to 30 then $T(\eta, Q)$ is equal to 7, so all training documents whose $(d \cap Q)$ is less than 7 are removed and thus avoiding computing their cosine similarities.

As regards reducing the training set, it is clear the proposed approach eliminates all documents which do not satisfy the Definition 3 of Section 3.4. Consequently, the set of training documents for which we compute cosine similarities is reduced.

As regards accuracy of classification, we use for classification only documents that share many terms with Q, that is, we just compute cosine similarities for documents containing $T(\eta, Q)$ terms of Q.

The experiments illustrate that the technique we employ reduces significantly the time classification without any loss of information. In fact, it seems to have the effect of noise reduction since classification accuracy becomes better after instance selection when compared to that of the original data.

6. Conclusions

In this paper, we investigate the efficiency of the k-NN algorithm for TC. We resumed the approach proposed in [5] to study its contribution point of view efficiency in the context of text categorization. Unlike the k-NN method that involves the whole corpus of training to search the nearest neighbors, the idea of this solution is based on the relevance of training documents to classify new ones. We do not need to use the entire training set for classifying a new instance but only a subset satisfying the condition of relevance. Using the cellular automaton model of *CASI* to represent the training documents and to select relevant documents, we have shown that the proposed approach not only improves classification accuracy but also accelerates the time of classification. Experiments on *Reuters* and *20News-Groups* showed that the proposed method is competitive in terms of predictive performance, while selecting a minimum of instances.

The model used for representing and selecting relevant documents is based on words (stems). We consider a word as the only representative of a unique meaning. It is assumed that there is a $1:1$ relation between words and meaning. However, in reality, a word can have several meanings, and a sense can be expressed in different words. Words clearly do not correspond directly to concepts. Some words are used for more than one concept, e.g., "bank" as a financial institution and "bank" as part of a river. Therefore, the use of words to represent the contents of documents poses two problems:

- The semantic ambiguity of words implies that non-relevant training documents that share the same words with the one we want to classify will be selected and this may increase the noise.

- The words disparity refers to lexically different words but with a common sense. This means that documents, yet relevant, not sharing words with the new document are not selected. And this may increase the silence.

Much work has been proposed to overcome the limits of this representation. For example, the authors in [9] propose the use of WordNet ontology to enrich the vector space model representation for classifying documents. The experiments showed significant improvements in the performance of classification. Motivated by these results, as future work, we plan to enrich the Boolean model of CASI with the concepts of WordNet.

References

[1] APPAVU S., RAJARAM R. Knowledge-based system for text classification using ID6NB algorithm. *Knowledge-Based Systems*. 2009, 1(22), pp. 1–7, doi: 10.1016/j.knosys.2008.04.006.

[2] ATMANI B., BELDJILALI B. Knowledge Discovery in Database: Induction Graph and Cellular Automaton. *Computing and Informatics Journal*. 2007, 26(2), pp. 171–197. Available from: http://www.cai.sk/ojs/index.php/cai/article/view/306.

[3] BAOLI L., SHIWEN Y., QIN L. An Improved k-Nearest Neighbor Algorithm for Text Categorization. In: *Proc. of the 20th International Conference on Computer Processing of Oriental Languages*, Shenyang, China. 2003, pp. 1–7. Available from: http://arxiv.org/abs/cs.CL/0306099.

[4] BARIGOU, N., BARIGOU, F., ATMANI, B. A Boolean model for spam detection. *Proceedings of the International Conference on Communication, Computing and Control Applications*, Hammamat, Tunisia. 2011, pp. 450–455, doi: 10.1109/CCCA.2011.6031517.

[5] BARIGOU F., BELDJILALI B., ATAMNI B. Using cellular automata for improving k-NN based spam filtering. *The International Arab Journal of Information Technology (IAJIT)*. 2014, 11(4), pp. 345–353. Available from: ccis2k.org/iajit/PDF/vol.11,no.4/4647.pdf.

[6] COVER T., HART P. Nearest Neighbor pattern classification. *IEEE Transactions Information Theory*. 1967, 13(1), pp. 21–27, doi: 10.1109/TIT.1967.1053964.

[7] DADHANIA S., DHOBI J. Improved k-NN Algorithm by Optimizing Cross-validation. *International Journal of Engineering Research and Technology*. 2012, 1(3), pp. 1–6. Available from: http://www.ijert.org/browse/volume-1-2012/may-2012-edition#vi1.

[8] DU M., CHEN X. Accelerated k-nearest neighbors algorithm based on principal component analysis for text categorization. *Journal of Zhejiang University SCIENCE C*. 2013, 14(6), pp. 407–416, doi: 10.1631/jzus.C1200303.

[9] ELBERRICHI Z., RAHMOUN A., BENTALAAH M. Using WordNet for text categorization. *The International Arab Journal of Information Technology (IAJIT)*. 2008, 5(1), pp. 16–24. Available from: http://ccis2k.org/iajit/?option=com_content&task=view&id=276.

[10] ERKAN G., HASSAN A., DIAO Q., RADEV D. *Improved Nearest Neighbor Methods For Text Classification*. University of Michigan, 2011. Department of electrical Engineering and Computer Science. Technical Report CSE-TR-576-11. Available from: https://www.eecs.umich.edu/techreports/cse/2011/CSE-TR-576-11.pdf.

[11] FRANK E., BOUCKAERT R. Naive Bayes for text classification with unbalanced classes. In: *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Vol. 4213 of the series Lecture Notes in Computer Science. Berlin, Germany, 2006, pp. 503–510, doi: 10.1007/11871637_49.

[12] FREUD Y., SHAPIRE R. Experiments with a new boosting algorithm. In: *Proceeding of the 13th international conference on machine learning*. Morgan Kaufmann, 1996, pp. 148–156.

[13] GARCIA S., DERRAC D., CANO J., HERRERA F. Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study. *IEEE Transactions on pattern analysis and machine intelligence*. 2012, 34(3), pp. 417–435,doi: 10.1109/TPAMI.2011.142.

[14] GUAN J., ZHOU S. Pruning Training Corpus to Speed up Text Classification. In: *13th International Conference, DEXA'02*, Aix-en-Provence, France. Berlin, Heidelberg: Springer, 2002, pp. 831–840, doi: 10.1007/3-540-46146-9_82.

[15] GUO G. *k*-NN Model-Based Approach in Classification. In: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, Vol. 2888 of the series Lecture Notes in Computer Science. Catania, Sicily, Italy, 2003, pp. 986–996, doi: 10.1007/978-3-540-39964-3_62.

[16] JIANG L., CAI Z., WANG D., JIANG Jiang S. Survey of Improving K-Nearest-Neighbor for Classification. In: *4th International Conference on Fuzzy Systems and Knowledge Discovery*. Haikou: IEEE, 2007, vol.1, pp. 679–683,doi: 10.1109/FSKD.2007.552.

[17] LAFFERTY J., ZHAI C.: Document language models, query models, and risk minimization for information retrieval, *In Proceedings of the 24th Annual International Conference on Research and Development in Information Retrieval*, New Orleans, Louisiana, USA, 2001, pp. 111–119, doi: 10.1145/383952.383970.

[18] LAM W., HO C. Using a generalized instance set for automatic text categorization. In: *Proceedings of the 21st Annual International Conference on Research and Development in Information Retrieval*, Melbourne, Australia. 1998, pp. 81–89, doi: 10.1145/290941.290961.

[19] LEE K., KAGEUR K. Virtual relevant documents in text categorization with support vector machines. *Information Processing and Management*. 2007, 43(4), pp. 902–913, doi: 10.1016/j.ipm.2006.08.010.

[20] LI C., PARK S. Combination of modified BPNN algorithms and an efficient feature selection method for text categorization. *Information Processing and Management*. 2009, 45(3), pp. 329–340, doi: 10.1016/j.ipm.2008.09.004.

[21] LIU T., MOORE A., GRAY A. New Algorithms for Efficient High Dimensional Non-Parametric Classification. *Journal of Machine Learning Research*. 2006, vol. 7, pp. 1135–1158. Available from: http://www.jmlr.org/papers/volume7/liu06a/liu06a.pdf.

[22] MANNING C., RAGHAVAN P., SCHUTZE H. *Introduction to Information Retrieval*. New York, USA: Cambridge University Press, 2008.

[23] MIAH M. Improved k-NN Algorithm for Text Classification. In: *Proceedings of the 2009 International Conference on Data Mining*, Las Vegas, USA. Las Vegas: CSREA Press, 2009, pp. 434-440.

[24] NIHARIKA S., SNEHA LATHA V., LAVANYA D. A survey on text categorization. *International Journal of Computer Trends and Technology*. 2012, 3(1), pp. 39–45. Available from: http://www.ijcttjournal.org/Volume3/issue-1/IJCTT-V3I1P108.pdf.

[25] PANG G., JIANG S. A generalized cluster centroid based classifier for text categorization. *Information Processing and Management*. 2013, 49(2), pp. 576–586, doi: 10.1016/j.ipm.2012.10.003.

[26] SCHONFISCH B., ROOS, A. Synchronous and Asynchronous Updating in Cellular Automata. *Biosystems*. 1999, 51(3), pp. 123–143. Available from: http://www.ncbi.nlm.nih.gov/pubmed/10530753.

[27] SEBASTIANI F.: Machine learning in automated text categorization. *ACM computing surveys*. 2002, 34(1), pp. 1–47, doi: 10.1145/505282.505283.

[28] SHIN K., AJITH A., SANG YONG H. Improving *k*-NN Text Categorization by Removing Outliers from Training Set. In: *7th International Conference CICLing*, Mexico City, Mexico. 2006, pp. 563–566, doi: 10.1007/11671299_58.

[29] SUGUNA N., THANUSHKODI K. An Improved k-NN Classification using Genetic Algorithm. *International Journal of Computer Science Issues*. 2010, 7(4-2), pp. 18–21. Available from: http://www.ijcsi.org/papers/7-4-2-18-21.pdf.

[30] TAN S. Neighbor-weighted K-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*. 2005, 28(4), pp. 667–671, doi: 10.1016/j.eswa.2004.12.023.

[31] TAN S. An effective refinement strategy for k-NN text classifier. *Expert Systems with Applications*. 2006, 30(2), pp. 290–298, doi: 10.1016/j.eswa.2005.07.019.

[32] WANG Y., WANG Z. A Fast k-NN Algorithm Applied to Web Text Categorization. In: *Proceeding of the 6th international conference on machine learning and cybernetics*, Hong Kong. 2007, pp. 19–22, doi: 10.1109/ICMLC.2007.4370742.

[33] WOLFRAM, S. *Cellular Automata and Complexity*. Perseus Books Group, 2002.

[34] XUESONG Y., WEI L., WEI C., WENJING L., CAN Z., QINGHUA W., HAMMIN L. Weighted K-Nearest Neighbor Classification Algorithm Based on Genetic Algorithm. *TELKOMNIKA*. 2013, 11(10), pp. 6173–6178, doi: 10.11591/telkomnika.v11i10.2534.

[35] YANG Y., PEDERSEN J. O. A comparative study on feature selection in text categorization. In: *In Proceedings of ICML-97, 14th International Conference on Machine Learning*. Nashville, USA: Morgan Kaufmann, 1997, pp. 412–420.

[36] ZHOU Y., LI Y., XIA S. An Improved k-NN Text Classification Algorithm Based on Clustering. *Journal of Computers*. 2009, 4(3), pp. 230–237, doi: 10.4304/jcp.4.3.230-237

[37] ZHANG J., MANI I. k-NN Approach to unbalanced data distributions: a case study involving information extraction. In: *Proc. International Conference Machine Learning (ICML'2003), Workshop Learning from Imbalanced Data Sets*, Washington DC, 2003.