Using Cellular Automata for Improving KNN Based Spam Filtering

Fatiha Barigou, Bouziane Beldjilali, and Baghdad Atmani
Computer Science laboratory, University of Oran, Algeria

Abstract: *As rapid growth over the Internet nowadays, electronic mail (e-mails) has become a popular communication tool. However, junk mail also, known as spam has increasingly become a part of life for users as well as internet service providers. To address this problem, many solutions have been proposed in the last decade. Currently, content-based anti-spam filtering methods are an important issue; the spam filtering is considered as a special case of binary text categorization. Many machine learning techniques have been developed and applied to classify email as spam or non-spam. In this paper, we proposed an enhanced K-Nearest Neighbours (KNN) method called Cellular Automaton Combined with KNN (CA-KNN) for spam filtering. In our proposed method, a cellular automaton is used to identify which instances in training set should be selected to classify a new e-mail; CA-KNN selects the nearest neighbours not from the whole training set, but only from a reduced subset selected by a cellular automaton.*

Keywords: *Spam e-mail filtering, machine learning, KNN, cellular automata, instance selection.*

Received September 27, 2011; accepted August 18, 2012; published online April 4, 2013

1. Introduction

The problem of undesired electronic messages called spam is nowadays a serious issue. Spam can be defined as unsolicited (or junk) email for a recipient or any email that the user does not want to have in his inbox. Spam is a big problem because of the large amount of shared resources it consumes including storage space, bandwidth, processing time and the resulting loss in productivity.

To solve the spam problem, several spam filtering systems have been proposed by both academic community and the industry, ranging from simple blacklisting [7] to advanced text classification [3, 9, 15, 18]. Among them, approaches which use machine learning algorithms to classify e-mails have achieved more success [11]. Indeed, machine learning techniques are widely used in automated text classification, including spam filtering (see [24] for a survey of techniques used).

The K-Nearest Neighbours (KNN) rule [12] is known to be one of best state of the art classifiers used for text classification. Many studies that have used the Reuters corpus [17, 26, 27] suggest that KNN and Support Vector Machines (SVM) outperform other methods like Linear least square fit, Naïve Bayes (NB) and Neural networks. The idea of KNN can be explained as follows: given a test document to be classified, the algorithm searches for the k nearest neighbours among the pre-classified training documents based on some similarity measure, and ranks those k neighbours based on their similarity scores, the categories of the k nearest neighbours are used to predict the category of the test document by using the ranked scores of each as the weight of the candidate categories, if more than one neighbour belong to the same category then the sum of their scores is used as the weight of that category, the category with the highest score is assigned to the test document.

KNN is robust and placed among the top algorithms. It was recommended previously as a practical approach. In 1999, Yang [26, 27] considered KNN as one of recommended approaches among more than ten approaches to text categorization, and in 2002, Sebastiani [23] recommended it, since it is simple and comparable to the best approach SVM. In addition, to its good performance, it is very easy to understand and to implement. However, the practicality will be lost when the KNN algorithm is applied to text categorization with high dimension; it has certain limitations:

- *Huge Memory:* It requires storing the whole training set.
- *High Computation Cost:* It has to explore the entire training set in order to classify a new document.
- *Low Tolerance to Noise:* Because it uses all data as relevant even when the training set contains noise or unbalanced data.

Our aim is to further improve the performance of KNN algorithm by adopting a new strategy called CA-KNN. We propose the use of the cellular automaton CASI [4] for two purposes; first, to encode training documents into a boolean representation, and secondly to search quickly a subset of relevant instances from

training set to be used by KNN algorithm. With this model we can face time and storage requirements and unbalanced data of KNN algorithm. In this paper, we examined CA-KNN for the task of e-mail spam filtering.

The rest of the paper is organized as follows: Section 2 will describe the related work about spam mail filtering. Section 3 outlines the principle of the cellular automaton CASI. Section 4 is devoted to the description of the proposed method. The experimental study to evaluate this approach is presented in section 5 and the work is concluded in section 6.

2. Content-Based Anti Spam Filtering

Numerous spam filtering strategies with varying degrees of efficiency have been proposed and developed (see [16, 24] for a survey of techniques used). However, despite this increasing development, the number of spam messages continues to increase.

The first published work on statistical spam filtering was proposed by Sahami *et al.* [19]. They proposed a Baysian networks to filter spam email using bag of words representation and binary weighting. However, the training and test data were not large enough in the experiment and the data are not publicly available.

Androutsopoulos *et al.* [2] constructed the Lingspam corpus and used it to compare a multinomial Naïve Bayes classifier and a variant of KNN classifier. Both methods achieved very high classification accuracy. Carreras and Marquez [8] used AdaBoost algorithm and improved results on the same corpus. Compared with two learning algorithms, the induction Decision Trees (DT) and NB, the method clearly outperformed than the above two learning algorithms.

In experiments by Sakkis *et al.* [20], a classifier combination method is proposed. They combined a NB and kNN by stacking and found that the ensemble improved performance over any of the classifiers separately.

Schneider [22] performed experiments with two statistical classifier models multivariate Bernoulli model and a multinomial model. To select the words within the vocabulary, different feature selection measures were used. Experiments obtained very high filtering rates higher than 95 %.

SVM was proposed for the first time by Drucker *et al*. [13] to filter spam emails. Comparisons with various classification techniques like Boosting tree, Ripper and Rocchio show that SVM is the best.

Extensive tests have been performed comparing different configurations of different classifiers. For example, comparisons were made between NB, maximum entropy, KNN, SVM and AdaBoost [28]. The results showed that SVM, maximum entropy and AdaBoost classifiers were much more effective than KNN and even the popular NB classifier. El-halees [14] compared several supervised machine learning for filtering spam e-mail from mixed Arabic and English messages. The experiments suggested that words in Arabic messages should be stemmed before applying classifier.

Cormack and Lynam [10] tested the real-world spam filtering tools SpamAssassin, Bogofilter, SpamProbe and CRM114 against the LingSpam corpus. They found that real spam filters were in general unable to classify the Lingspam messages correctly.

Barigou *et al*. [5] proposed a new symbolic induction approach based on cellular automata to filter spam they tried to examine the impact of various features selection algorithms, stemming and stop words removal on the performance of the cellular automaton classifier. Experiments show a very high quality of prediction when using stemming and feature selection with information gain function. A performance improvement is observed over the reported results of NB and KNN [2].

A combination of NB classifier and cellular automaton is proposed in [6]. Majority voting provided better results compared to individual classifiers.

Santos *et al*. [21] have explored the use of semantics in spam filtering by representing e-mails with a recently introduced Information Retrieval model: the enhanced Topic-based vector space model. This model used a semantic ontology to deal with synonymy. Based upon this representation, they apply several well-known machine-learning models (NB, KNN, SVM and DT) and show that the proposed method can detect the internal semantics of spam messages. Experiments on Lingspam corpora show that this approach provides high percentages of spam detection whilst keeping the number of misclassified legitimate messages low.

3. Cellular Automaton CASI

The cellular automaton CASI [4] is a boolean inference engine. It is organized into cells where each cell is connected only with its neighbours. All cells obey in parallel to the same rule called local transition function, which results in an overall transformation of the system. It uses a knowledge base in the form of two layers of finite automata. The first layer, called CelFact, represents the fact base and the second layer, called CelRule, represents the rule base. In each layer, the content of a cell determines whether and how it participates in each inference step: at every step, a cell can be active or passive, can take part in the inference or not. The states of cells are composed of three parts: EF, IF, SF, and ER, IR, SR which are the input, internal and output parts of the CelFact cells, and the CelRule cells, respectively.

Any cell "i" in the CelFact layer with input EF (i) = 1 is regarded as representing an established fact. If EF

(i) = 0, the represented fact has to be established. Any cell "j" of the CelRule layer with input ER (j) = 0 is regarded as a candidate rule. When ER (j) = 1, the rule should not take part in the inference.The neighbourhood of cells is defined by two incidence matrices called R_E, R_S respectively. They represent the input respectively output relation of the facts and are used in forward chaining.

- The input relation, noted iR_Ej, is formulated as follows: If (fact i∈Premise of rule j) then $iR_Ej = 1$ else $iR_Ej = 0$.
- The output relation, noted iR_Sj, is formulated as follows: If (fact i∈Conclusion of rule j) then $iR_Sj = 1$ else $iR_Sj = 0$. The cellular automaton dynamics implemented as a cycle of an inference engine made up of two local transition functions *δfact* and *δrule*, where *δfact* corresponds to the evaluation, selection and filtering phases and *δrule* corresponds to the execution phase.
- The transition function *δfact* is defined as:

$$(EF, IF, SF, ER, IR, SR) \rightarrow (EF, IF, EF, ER + (R_E^T \times EF), IR, SR)$$

- The transition function δrule is defined as:

$$(EF, IF, SF, ER, IR, SR) \rightarrow (EF + (R_S \times ER), IF, SF, ER, IR, \overline{ER})$$

We have made changes at this automaton:

1. We have defined three layers of finite automata instead of two to represent training set.
2. We have brought changes in the two transition functions *δfact* and *δrule* in order to extract relevant documents.

The details of this model will be represented in section 4.

4. Proposed Approach

In this section, we present our approach to enhance KNN classification. First of all we give the following definitions and notations:

- *Definition 1*: Training documents set $D = \{(d_i, c_i)\ i = 1, 2, ..., N\}$, where d_i is a training document and c_i is a label category[1]∈{0,1}. Each document $d_i \in D$ is associated with a class label c_i, which indicates whether d_i, belongs to the target class C ($c_i = 1$) or not ($c_i = 0$).
- *Definition 2*: Vocabulary of terms, $V = \{t_j;\ j = 1, 2, ..., M\}$, where t_j is a term (in our case this is the stem).
- *Definition 3*: Documents representation as vectors each document d_i is represented as a vector $\vec{d}_i = (w_{i1}, w_{i2}, ..., w_{iM})$, where w_{ij} is the weight of term t_j in document d_i.

[1] We consider only binary classification.

As we have declared in introduction, KNN technique is very simple, highly efficient and effective in the field of classification. But its computation cost is very expensive especially for text categorization where the number of documents and features is large. The algorithm is very slow because it has to exhaustively match all the training documents against the test document to find its k-nearest neighbours. The implementation requires storing the complete training set, and classification takes time proportional to the size of the training data times the dimension of the feature vectors. Also, for unbalanced corpora, the classification performance often decreases.

For all these reasons, we propose a new approach named Cellular Automaton combined with K-Nearest Neighbours (CA-KNN), which is based on the notion of retrieving a subset of relevant documents from training set to classify a new instance. The selection of this subset will be done by the cellular automaton CASI. Our findings are:

- The number of terms in common between the document to classify and the one of training set is an interesting parameter to locate relevant documents that will be involved in the calculation of closest neighbors.
- Discarding every document in the training corpus not sharing any words with the document to classify doesn't increase error rate and should be used as a filter before measuring the similarity between the training document and the new one.
- We know that the similarity between documents is sensitive to the "number" of terms in common. As this number increases as the documents become more similar.

In this research, we propose to keep only relevant documents from training set for classifying new instances. We show that excluding irrelevant documents from training set when classifying a new document improves significantly the KNN classifier. The main challenge here is how to estimate that a document is relevant or non-relevant.

We consider that a training document is irrelevant when the total number of vocabulary terms in common with the new document is below a threshold. Our hypothesis is that type of documents represents noise; they are not useful training instances because they decrease the classification accuracy.

We use cellular automaton CASI [4] for two purposes: first to represent the training set and secondly to extract relevant documents which must be involved in classifying a new instance.

As shown in Figure 1 bellow, the proposed approach consists of two processes:

- Instance representation.
- Instance selection and classification.

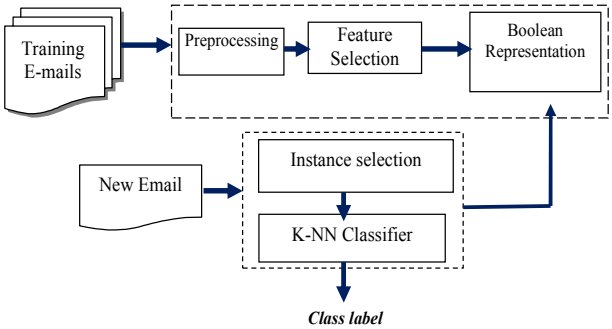

Figure 1. CA-KNN diagram.

4.1. Instance Representation

We propose an alternative strategy of encoding documents; training documents are encoded into a Boolean representation (by using the cellular automaton CASI). Before that, the training set is first preprocessed; we extracted word tokens from the data, removed stop words and used a variant of the Porter[2] algorithm for stemming. Since too many terms are usually extracted, some of them should be selected as features. Many schemes of selecting features were already proposed [25]. In this work, to reduce vocabulary size we kept only features selected by the information gain function.

Once the index is built and reduced, we obtain a document-by-word matrix $A\ (N \times M)$ like that shown in Table1. Column $M + 1$ contains the class label of documents. The ith–document d_i is represented by the characteristic vector $\vec{d_i} = (w_{i1}, w_{i2}, \ldots, w_{iM}, c_i)$. In this paper, we deal with a binary weighting. $(w_{ij}) = 1$ if the term $t_j \in V$ is present one or more times in d_i, 0 otherwise. After that, we proceed with the proposed strategy of encoding documents. To illustrate this encoding, let us consider the training set D represented in Table 1. $D = \{d_1, d_2, d_3, d_4, d_5, d_6\}$; V = {differential, extract, index, matrix} and $C = \{0, 1\}$.

Table 1. Vector representation.

	Differential	Extract	Index	Matrix	Class = c
d1	0	1	1	1	1
d2	0	1	0	0	0
d3	0	1	1	0	1
d4	1	0	0	1	0
d5	1	0	0	0	0
d6	1	0	0	1	0

The cellular automaton CASI has been reviewed and some changes have been established in order to represent the training documents. We have defined three layers instead of two. Table 2 shows the three layers modelling the training set:

1. *CEL-TERM* layer is planned to hold all the terms of vocabulary V. Initially, all the cells inputs are passive (ET = 0).

Table 2. The three layers used for documents encoding.

	Word	ET	IT	ST
CEL-TERM	Differential	0	1	0
	Extract	0	1	0
	Index	0	1	0
	Matrix	0	1	0
	ID-Doc	**ED**	**ID**	**SD**
CEL-DOC	d1	0	1	0
	d2	0	0	0
	d3	0	1	0
	d4	0	0	0
	d5	0	0	0
	d6	0	0	0
	Rule	**ER**	**IR**	**SR**
CEL-RULE	R1	0	1	1
	R2	0	1	1
	R3	0	1	1
	R4	0	1	1

2. *CEL-DOC* layer is intended to hold the identifiers of training documents. Initially, all the cells inputs are passive (ED = 0). The ID allows us to distinguish between documents in the class "c" and those who are not (ID = 1 if the document is classified c; 0 otherwise).
3. *CEL-RULE* layer represents the presence of the term in the training documents. For each term $t_j \in V$ we associate a rule R_j which tell us where the term occurs. According to example of Table 1, we have four rules:

   R_1: If (term = "differential") then d_4, d_5, d_6
   R_2: If (term = "extract") then d_1, d_2, d_3
   R_3: If (term = "index") then d_1, d_3
   R_4: If (term = "matrix") then d_1, d_4, d_6

For example the rule R_1 indicates that the word "*differential*" could be retrieved from documents d_4, d_5 and d_6. The terms are linked to their documents by Input Matrix (IM) and Output Matrix (OM): The IM matrix is $M \times M$ of dimension, while OM matrix is of dimension $N \times M$. For example, the term "*differential*", which is the input of rule R_1 in IM matrix is found in documents d_4, d_5 and d_6, which are the output of R_1 in OM matrix as shown in Table 3.

1. *Input relation: IM (t, r)*
   $\forall t \in \{ t_j \backslash t_j \in$ *CEL-TERM*; $j = 1 \ldots M\}$
   $\forall r \in \{ R_j \backslash R_j \in$ *CEL-RULE*; $j = 1 \ldots M\}$
   If (t is a premise of r) then (IM (t, r) =1).
2. *Output relation: OM (d, r)*
   $\forall d \in \{ d_i \backslash d_i \in$ *CEL-DOC*; $i = 1 \ldots N\}$; $\forall r \in \{ R_j \backslash R_j \in$ *CEL-RULE*; $j = 1 \ldots M\}$:
   If (d is in conclusion of r) then (OM (d, r) =1)

We can observe that the concatenation of OM matrix with ID vector of CEL-DOC layer corresponds to the document-by-word matrix using a binary weighting.

[2] http://tartarus.org/~martin/PorterStemmer/.

Table 3. Input and output matrices.

	Word	R1	R2	R3	R4
IM	Differential	1	0	0	0
	Extract	0	1	0	0
	Index	0	0	1	0
	Matrix	0	0	0	1
	ID-Doc	**R1**	**R2**	**R3**	**R4**
OM	d1	0	1	1	1
	d2	0	1	0	0
	d3	0	1	1	0
	d4	1	0	0	1
	d5	1	0	0	0
	d6	1	0	0	1

4.2. Instance Selection and Classification

In this section we describe in detail our algorithm for instance selection and classification. Before applying our instance selection algorithm, we will assume that the Boolean representation of training documents has been created. We will only operate on this representation to determine a subset of relevant documents using algorithm 1.

Let consider q the new unlabelled document to classify and *TNT (q)* the total number of vocabulary terms *(∈V)* and found in q.

- We define a threshold $T(\eta, q)$ to be $\left\lceil \frac{TNT(q)}{\eta} \right\rceil + 1$, where $\eta \geq 2$.
- For a training document d_i we define $TC(d_i)$ to be its total number of terms in common with q.
- A document d_i is relevant for classification if it satisfies the following condition: $TC(d_i) \geq T(\eta, q)$.
- $\delta fact$ is defined as follows:

$$(ET, IT, ST, ER, IR, SR) \rightarrow (ET, IT, ET, ER + (IM^T \times ET), IR, SR)$$

- $\delta rule$ is defined as follows:

$$(ED, ID, SD, ER, IR, SR) \rightarrow (ED + (OM \times ER), ID, SD, ER, IR, \overline{ER})$$

Any training documents whose *TC* is lower than the threshold $T(\eta, q)$ is rejected and can't be used in the classification process.

Algorithm1: Instance Selection Algorithm
1. *Parameter* η
2. *Input: New unlabelled document (q), CEL-TERM, CEL-RULE, CEL-DOC, IM and OM*
3. *Output:* $E \subset D$ *a subset of relevant documents*
4. $A = \varnothing, E = \varnothing$
5. *Calculate* $T(\eta, q)$
6. *Initialize CEL-DOC*
7. *For each term* $t_j \in V \cap q$ *do*
8. $ET(t_j) = 1$
9. *End for each*
10. *Apply* $\delta fact \bullet \delta rule$
11. *For each* d_i *in CEL-DOC do*
12. *If* $ED(d_i) = 1$ *then*
    12.1. $A = A \cup \{d_i\}$
    12.2. *Calculate* $TC(d_i) = And(OM(d_i), ET^T)$
13. *End if*
14. *End for each*
15. *For each* $d_i \in A$ *do*
16. *If* $TC(d_i) \geq T(\eta, q)$ *then*
    16.1. $E = E \cup \{d_i\}$
17. *End if*
18. *End for each*
19. *Output the subset E*

To classify a new instance q, CA-KNN operates in three steps. During the first step, CA-KNN initializes the *CEL-TERM* layer by activating the *ET* state for each cell corresponding to the terms found in q and starts the inference by applying the global transition $\delta fact \bullet \delta rule$ to select documents containing at least one term in common with q see equation 1. We obtain a reduced set of training documents called A:

$$A = \{d_i \in D; where\ TC(d_i) \geq 1\} \quad (1)$$

In the second step, and to further reduce the training set, CA-KNN searches within A for the subset of documents satisfying the condition $TC(d_i) \geq T(\eta, q)$. These documents constitute the set E see equation 2:

$$E = \{d_i \in A; where\ \left|(ET)^T\ and\ OM\ (d_i)\right| \geq T\ (n, q)\} \quad (2)$$

The set E is obtained by calculating for each d_i in A, the total number of active cells of the output logic operator AND of the $(ET)^T$ with $OM(d_i)$.

In the third step, CA-KNN uses KNN algorithm with subset E as training set to classify the new document. To illustrate this algorithm, let us consider the test document q with the vocabulary terms: {index, extract}. In CEL-TERM layer, only ET (index) and ET (extract) states will be set to 1 as shows in Figure 2-a. After, applying $\delta fact \bullet \delta rule$ the retrieved documents are: d_1, d_2 and d_3 their ED state is activated. Documents d_3, d_4 and d_6 are ignored as shows in Figure 2-b. In the second step, we fixed $\eta=2$ and the threshold will be $\left\lceil \frac{TNT(q)}{2} \right\rceil + 1$ (in this case it is equal to $\left\lceil \frac{2}{2} \right\rceil + 1$) the classification will be done only with d_1 and d_3. Document d_2 is also, discarded from the set of training documents as shows in Figure 2-c.

a. Initialization

Term		ET	IT	ST
1	Differential	0	1	0
2	Extract	1	1	0
3	Index	1	1	0
4	Matrix	0	1	0
		CEL-TERM		

b. Inference ($\delta fact \bullet \delta rule$):

Document		ED	ID	SD
1	d_1	1	1	0
2	d_2	1	0	0
3	d_3	1	1	0
4	d_4	0	0	0
5	d_5	0	0	0
6	d_6	0	0	0
		CEL-DOC		

A={d_1, d_2, d_3}

c. From A keep only those satisfying $TC(d_i) \geq 2$:

$$\left\langle \begin{array}{l} TC(d_1) = OM(d_1)\ AND\ (ET)^t = 2 \\ TC(d_2) = OM(d_2)\ AND\ (ET)^t = 1 \\ TC(d_3) = OM(d_3)\ AND\ (ET)^t = 2 \end{array} \right\rangle$$

E={d1, d3}

Figure 2. Instance selection example.

5. Experimental Evaluation

To evaluate the proposed approach for spam filtering, we have undertaken experiments on the Lingspam corpus. Lingspam is freely available and has been used in many studies [1, 2, 5, 6, 20, 21]. It consists of 2412 legitimate emails and 481 spam in the total dataset; the specialty of this corpus is larger legitimate examples than spam examples. Spam e-mails represent only 16% of the whole dataset. So, the class of legitimate e-mails is larger than the class of spam e-mail.

5.1. Performance Measures

We measured several indicators of classification performance: the recall of class spam (SR), precision class spam (SP), the F-measure of class spam (F1) and finally the accuracy (A) shown in equations 3 to 6.

Let TN the number of legitimate emails classified as legitimate (true negatives), TP the number of spam emails classified as spam (true positives), FP the number of legitimate emails classified as Spam (False Positives) and FN the number of spam emails classified as legitimate (false negatives), then we have:

$$SP = \frac{TP}{TP + FP} \tag{3}$$

$$SR = \frac{TP}{TP + FN} \tag{4}$$

$$F1 = \frac{2 \times SP \times SR}{SP + SR} \tag{5}$$

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

Furthermore, we measured the True Positive Ratio (TPR) shown in equation 7, the False Positive Ratio (FPR) shown in equation 8 and the total cost ratio (TCR)[3], shown in equation 9. Greater TCR values indicate better performance:

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

$$TCR = \frac{TP + FN}{FP + FN} \tag{9}$$

5.2. Experimental Results and Discussion

We performed a k-fold cross validation with k = 10. The dataset was split 10 times into 10 different sets of learning sets (90% of the total dataset) and testing sets (10% of the total data).

We used the Euclidean distance for searching k neighbours see equation 10 and majority voting to determine the class of the new email see equation 11:

$$D_E(q, d_i) = \sqrt{\sum_{t \in V} (w_t(q) - w_t(d_i))^2}; \tag{10}$$

$$w_t(q) \text{ is the weight of term } t \in V \text{ in the new instance } q;$$
$$w_t(d_i): \text{is the weight of term } t \in V \text{ in document } d_i \in D$$

$$c(q) = Arg\,Max\left(\sum_{d_i \in kNN(q)} y(c_k, d_i)\right)_{c_k \in \{spam, legitimate\}} \tag{11}$$

$$y(c_k, d_i) = \begin{cases} 1 & d_i \in c_k \\ 0 & d_i \notin c_k \end{cases}$$

Where *kNN(q)* denote the set of nearest neighbours of new instance *q*.

The results mainly discuss the effectiveness of the proposed approach; Figures 3, 4, 5 and 6 show the results of the precision, recall, F-measure and accuracy when applying CA-KNN on Lingspam data.

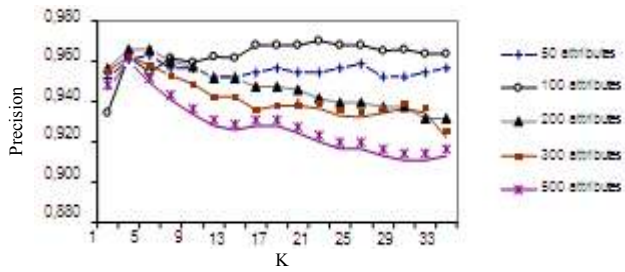

Figure 3. CA-KNN's precision as a function of number of selected attributes and threshold K.

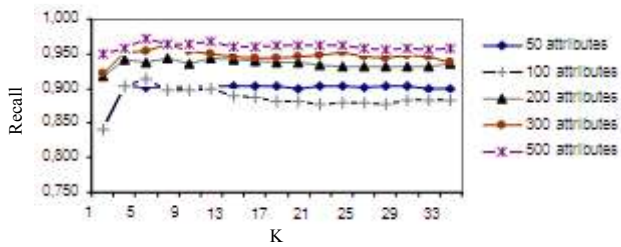

Figure 4. CA-KNN's recall as a function of number of selected attributes and threshold K.

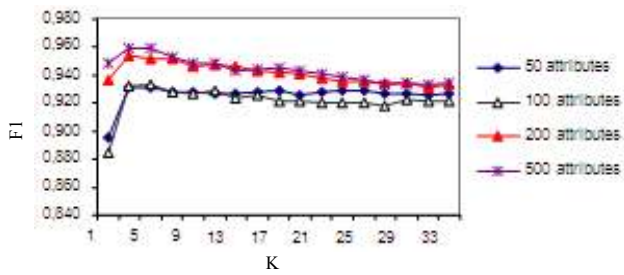

Figure 5. CA-KNN's F1-measure as a function of number of selected attributes and threshold K.

The results of performance measures depend on the number of selected term-attributes and the threshold k. As we can see, except precision which has its best values when the number of selected term-features is 100 and k = 23, recall, accuracy and F1 measures have the best results with lower values of k (between 3 to 7) and higher values of term-attributes. When k increases, there is a decrease on recall, accuracy and F1.

[3]We consider TCR with only λ =1.

Accuracy with varying vocabulary size

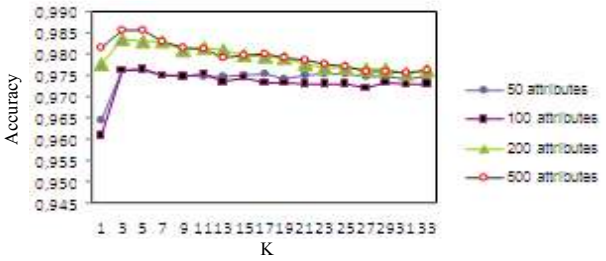

Figure 6. CA-KNN's accuracy measure as a function of number of selected attributes and threshold K.

Figures 5 and 6 show a result of 98,55% score for accuracy and 96,0% score of F-measure. These values are achieved when k takes 5 and attributes number takes 500. The reason for these results is the way how CA-KNN works; a reduced number of neighbors is sufficient to make decisions because CA-KNN searches the k neighbors from a subset of relevant documents.

We do not need to increase the number of neighbors to improve decision; on the contrary, using high values of k can harm the effectiveness of classification because more documents with low similarity scores are involved, low similarity score indicates that test and train documents are probably of different categories, and they may confuse the classifier decision.

We have also, implemented the traditional KNN approach to compare it with CA-KNN (Figures 7-9).

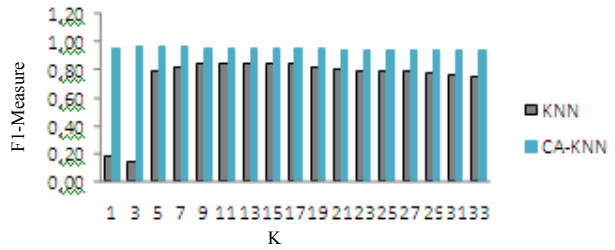

Figure 7. The F1-Measure by varying the K value from 1 to 33.

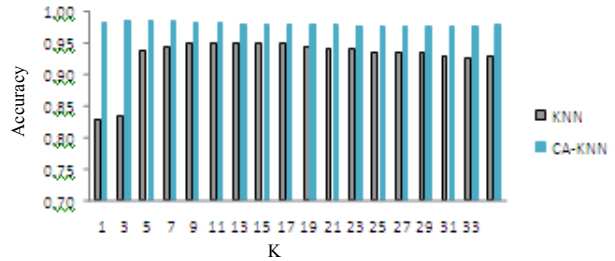

Figure 8. The accuracy of CA-KNN and KNN by varying the K value from 1 to 33.

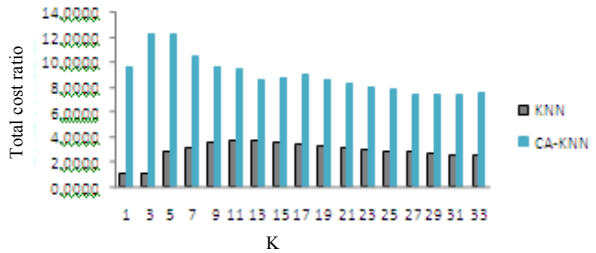

Figure 9. The total cost ratio of CA-KNN and KNN by varying the K value from 1 to 33.

Table 4 depicts the best configurations for performance accuracy of CA-KNN compared with traditional KNN. From the results, it is shown that our proposed method outperforms the traditional KNN with greater accuracy.

Table 4. Performance of KNN and CA-KNN with the best configuration on Lingspam data.

	K	M	SR(%)	SP(%)	F1(%)	A(%)	TCR
KNN	13	50	73,39	98,88	84,25	95,05	3,64
CA-KNN	5	500	97,1	94,9	95,99	98,55	12,59

5.3. Comparison with Previous Work

To evaluate the contribution of the proposed algorithm to spam filtering, we compare our approach with the best reported results of both real-world solutions and academic approaches cited in section 2 using the Lingspam corpora see Table 5 below.

Table 5. Comparison with published works.

Model	A (%)	TPR	FPR
[1]			
$\lambda = 1$	97,06	0,83	0,08
$\lambda = 9$	96,33	0,78	0,08
$\lambda = 99$	94,19	0,65	0
[20]			
K = 5, $\lambda = 1$ and m = 100	98,06	0,92	0,01
K = 3, $\lambda = 9$ and m = 200	97,20	0,84	0,01
K = 7, $\lambda = 1$ and m = 300	84,89	0,90	0,16
K = 3, $\lambda = 9$ and m = 100	97,30	0,85	0,01
[22]			
Bernoulli	98,00	0,89	0,01
mv-MI	98,86	0,96	0,01
mn-MI	98,06	0,93	0,01
dmn-MI	85,52	0,17	0,01
tf-MI	98,79	0,96	0,01
dtf-MI	98,48	0,95	0,01
[10]			
SpamAssassin	84,1	0,04	0
BogoFilter	90,1	0,40	0
SpamProb	94,8	0,69	0
CRM 114	81,5	88,8	0,45
[21]			
Bayesian Network	99,26	0,97	0,00
DT: Random forest N = 10	98,72	0,94	0,00
CA-KNN K = 5 m = 500	98,55	0,97	0,00

Table 5 shows the best results obtained using the CA-KNN technique alongside those previously published [1, 10, 20, 21, 22] and reported in [21]. The parameter k is the neighbourhood size for the KNN, λ determines the strictness of the criteria for classifying an e-mail as spam, m is the number of attributes. The results indicate improved performance when classifying with CA-KNN.

6. Conclusions

In this paper a new method called CA-KNN is proposed to improve the KNN's classification performance. CA-KNN needs not searching the K nearest neighbours from all training set based on cellular automaton CASI. Therefore, the searching scope is reduced and only a subset of relevant

documents is selected to participate in the classification of a new instance.

By using a cellular model to represent training documents and retrieve relevant documents for classification, we have shown that our proposed method not only improves the classification accuracy but also, a performance improvement is observed over traditional KNN and other published works.

The main contributions in this paper are summarized as follows:

- We propose and develop an improved KNN algorithm which is better than classical KNN algorithm while improving the storage memory and classification performance.
- Based on the cellular automaton CASI [4], we propose for the first time a cellular inference technique which allows us to search very quickly from training set only relevant documents and skip the others which are not important for classification.
- We improve storage memory by organizing the training set into a cellular structure.
- We improve performance classification and face the problem of noisy or unbalance data by selecting only relevant document from training set to classify the new document.
- Finally, we evaluate this algorithm for the e-mail spam filtering.

Although, the findings are interesting and encouraging, many issues may be studied in future work. We must also conduct a detailed comparison of this approach with other learning algorithms used in spam filtering and considering other corpus such as Spam Assassin and evaluation criteria such as weighted accuracy and weighted error.

References

[1] Androutsopoulos I. and Koutsias J., "An Evaluation of Naive Bayesian Networks," *in Proceedings of Machine Learning in the New Information Age*, Barcelona, Spain, pp. 9-17, 2000.

[2] Androutsopoulos I., Paliouras G., Karkaletsis V., Sakkis G., Spyropoulos C., and Stamatopoulos P., "Learning to Filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach," *in Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*, Lyon, France, pp. 1-13, 2000.

[3] Apté G., Damerau F., and Weiss S., "Automated Learning of Decision Rules for Text Categorization," *ACM Transactions on Information Systems*, vol. 12, no. 3, pp. 233-251, 1994.

[4] Atmani B. and Beldjilali B., "Knowledge Discovery in Database: Induction Graph and Cellular Automaton," *Computing and Informatics Journal*, vol. 26, no. 2, pp. 171-197, 2007.

[5] Barigou N. and Barigou F., "A Boolean Model for Spam Detection," *in Proceedings of the International Conference on Communication, Computing and Control Application*, Tunisia, pp. 450-455, 2011.

[6] Barigou F., Barigou N., and Atmani B., "Combining Classifiers for Spam Detection," *International Conference on Networked Digital Technologies*, vol. 293, no. 3, pp.78-89, Dubai, UAE, 2012.

[7] Carpinter J. and Hunt R., "Tightening the Net: A Review of Current and Next Generation Spam Filtering Tools," *Computers and Security*, vol. 25, no. 8, pp. 566-578, 2006.

[8] Carreras X. and Marquez L., "Boosting Trees for Anti-Spam Email Filtering," *in Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing*, Bulgaria, pp. 58-64, 2001.

[9] Cohen W., "Learning Rules That Classify E-Mail," *in Proceedings of AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, California, USA, pp. 18-25, 1996.

[10] Cormack G. and Lynam T., "Online Supervised Spam Filter Evaluation," *ACM Transactions on Information Systems*, vol. 25, no. 3, pp. 1-31, 2007.

[11] Cormack G., "Email Spam Filtering: A Systematic Review," *Foundations and Trends in Information Retrieval*, vol. 1, no. 4, pp. 335-455, 2008.

[12] Cover T. and Hart P., "Nearest Neighbor Pattern Classification," *IEEE Transactions Information Theory*, vol. 13, no. 1, pp. 21-27, 1967.

[13] Drucker H., Wu D., and Vapnik V., "Support Vector Machines for Spam Categorization" *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1048-1054, 1999.

[14] El-halees A., "Filtering Spam E-Mail from Mixed Arabic and English Messages: A Comparison of Machine Learning Techniques," *the International Arab Journal of Information Technology*, vol. 6, no. 1, pp. 52-59, 2009.

[15] Gee K., "Using Latent Semantic Indexing to Filter Spam," *in Proceedings of ACM Symposium on Applied Computing, Data Mining Track*, Melbourne, Florida, USA, pp. 460-464, 2003.

[16] Guzella T. and Caminhas W., "A Review of Machine Learning Approaches to Spam Filtering," *Expert Systems with Applications*, vol. 36, no.7, pp. 10206-10222, 2009.

[17] Joachims T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *in Proceedings of the 10th European Conference on Machine Learning*, Berlin, Germany, vol. 1398, pp. 137-142, 1998.

[18] Katirai H., "Filtering Junk E-Mail: A Performance Comparison between Genetic Programming and Naive Bayes," *Technical Report*, University of Waterloo, 1999.

[19] Sahami M., Dumais S., Heckerman D., and Horvitz E., "A Bayesian Approach to Filtering Junk E-Mail," *Technical Report*, Learning for Text Categorization Workshop, Stanford University, UK, 1998.

[20] Sakkis G., Androutsopoulos I., Paliouras G., and Karkaletsis V., "Stacking Classifiers for Anti-Spam Filtering of E-mail," *in Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing*, Pittsburgh, USA, pp. 44-50, 2001.

[21] Santos I., Laorden C., Sanz B., and Bringas P., "Enhanced Topic-Based Vector Space Model for Semantics-Aware Spam Filtering," *Expert Systems With Applications*, vol. 39, no. 1, pp. 437-449, 2012.

[22] Schneider K., "A Comparison of Event Models for Naive Bayes Anti-Spam E-Mail Filtering," *in Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, Stroudsburg, USA, pp. 307-314, 2003.

[23] Sebastiani F., "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1-47, 2002.

[24] Upasana P. and Chakraverty S., "A Review of Text Classification Approaches for E-Mail Management," *Journal of Engineering and Technology*, vol. 3, no. 2, pp. 137-144, 2011.

[25] Yang Y. and Pedersen J., "A Comparative Study on Feature Selection in Text Categorization," *in Proceedings of the 14th International Conference on Machine Learning*, Nashville, USA, pp. 412-420, 1997.

[26] Yang Y., "An Evaluation of Statistical Approaches to Text Categorization," *Information Retrieval*, vol. 1, no. 1-2, pp. 67-88, 1999.

[27] Yang Y. and Liu X., "A Re-Examination of Text Categorization Methods," *in Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in IR*, New York, USA, pp. 42-49, 1999.

[28] Zhang L., Zhu J., and Yao T., "An Evaluation of Statistical Spam Filtering Techniques," *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 4, pp. 243-269, 2004.

Fatiha Barigou is a computer science teacher in the Department of Computer Science at Oran University (Algeria). She earned her Master of Science Degree in 1998 from Oran University. She is currently a PhD candidate in the Computer Science Department at the same university. Her research interests focus on text mining, information extraction, and information retrieval areas.

Bouziane Beldjilali received his PhD degree in computer science from the University of Oran (Algeria) in 1996. He is a professor in the Computer Science Department at the University of Oran. His research interests include formal specifications, knowledge management, databases, artificial intelligence, and automatic learning.

Baghdad Atmani received his Master of Science Degree in 1996 from the Department of Computer Science in Oran (Algeria). He is currently a PhD candidate in the Computer Science Department at the University of Oran. His research interests include knowledge discovery in databases, data mining, feature selection, neural networks, and cellular automata.