Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition Letters 25 (2004) 1819–1831

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Adaptive anti-spam filtering for agglutinative languages: a special case for Turkish

Levent Özgür, Tunga Güngör [*], Fikret Gürgen

Department of Computer Engineering, Boğaziçi University, İstanbul 34342, Turkey

Received 25 December 2003; received in revised form 11 May 2004; accepted 23 July 2004
Available online 27 August 2004

Abstract

We propose anti-spam filtering methods for agglutinative languages in general and for Turkish in particular. The methods are dynamic and are based on Artificial Neural Networks (ANN) and Bayesian Networks. The developed algorithms are user-specific and adapt themselves with the characteristics of the incoming e-mails. The algorithms have two main components. The first one deals with the morphology of the words and the second one classifies the e-mails by using the roots of the words extracted by the morphological analysis. Two ANN structures, single layer perceptron and multi-layer perceptron, are considered and the inputs to the networks are determined using binary model and probabilistic model. Similarly, for Bayesian classification, three different approaches are employed: binary model, probabilistic model, and advanced probabilistic model. In the experiments, a total of 750 e-mails (410 spam and 340 normal) were used and a success rate of about 90% was achieved.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Anti-spam filtering; Artificial neural network; Bayesian network; Agglutinative language; Morphology

1. Introduction

In the past few years, internet technology has affected our daily communication style in a radical way. The electronic mail (e-mail) concept is used extensively for communication nowadays. This technology makes it possible to communicate with many people simultaneously in a very easy and cheap way. But, many e-mails are received by users without their desire. *Spam mail* (or, *junk mail* or *bulk mail*) is the general name used to denote these types of e-mail. Spam mails, by definition, are the electronic messages posted blindly to thousands of recipients usually for advertisement.

As time goes on, much more percentage of the e-mails is treated as spam and this increases the seriousness of the problem. In 1998, Canor and

[*] Corresponding author. Tel.: +90 212 3597094; fax: +90 212 2872461.
E-mail address: gungort@boun.edu.tr (T. Güngör).

0167-8655/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2004.07.004

LaMacchia found that about 10% of the incoming e-mails to the network was spam. As a result of another study, American Online (AOL) has stated that it has received 1.8 million spam mails until precautions have been taken (NISCC Quarterly Review, January–March, 2003). According to a research performed by Symantec in 2002, 63% of the people receive over 50 spam mails per week while 37% of them receive over 1000; 65% waste at least 10 min for spam messages daily while 24% waste over 20 min (http://www.turk.internet.com).

Many methods have been proposed to solve this problem, but they are not completely satisfactory. We can group them into two broad categories: static methods and dynamic methods. Static methods base their spam mail identification on a predefined address list. For instance, the mail server "hotmail" allows a person to receive an e-mail only if his/her address is one of the recipient addresses; otherwise the server treats the e-mail as spam. Needless to say, most spam mails pass this test and some important mails are treated as spam. Also, some servers try to collect addresses which are reported as *spammers* (people who send spam messages) and treat the e-mails coming from them as spam. However, spammers are all aware of most of these methods. All these solutions lack the dynamic structure of the problem, which highly limits their effectiveness.

Some more complex approaches are dynamic in nature. They take the contents of the e-mails into consideration and adapt their spam filtering decisions with respect to these contents. Most of them use general text categorization techniques by implementing machine learning methods. Naive Bayesian algorithms have been used by training a classifier on manually categorized normal and spam mails (e.g. Androutsopoulos et al., 2000; McCallum and Nigam, 1998; Sanchez et al., 2002). Cross-validation was performed for decreasing random variation. Some important issues like increasing training size, lemmatization, and stop-lists were covered in these studies and successful results were obtained for filtering English spam mails. A detailed study on probabilistic model of information retrieval in document classification has also been performed with different complexity measures (Jones et al., 2000a,b).

Rule based systems were used for automatic discovery of classification patterns (e.g. Apte et al., 1994). Some specific rule learning methods (ripper, tf-idf) were also discussed and remarkable results were achieved (e.g. Cohen, 1996). For instance, it was understood that much of the learning took place with around only 200 examples in filtering English spam mails. Neural networks were employed for web content filtering, which is a special domain of text categorization (Lee et al., 2002).

Lewis proposed a model of feature selection and extraction for categorization (Lewis, 1992; Lewis and Croft, 1990). The optimal feature set size for word-based indexing was found to be 10–15 (which was surprisingly low) despite the large training sets. Dagan proposed a model for text categorization based on mistake driven learning (Dagan et al., 1997). This method used words in a probabilistic model for the formation of feature vectors rather than binary model. This was done to tolerate the length variation of documents and significant success was achieved.

The aim of this paper is to propose dynamic anti-spam filtering methods for Turkish that are based on Artificial Neural Network (ANN) and Bayesian Network. We develop several algorithms by changing the topologies of the networks and by adjusting some parameters, and we compare their success rates. The research in this paper is directed towards agglutinative languages in general and Turkish in particular. As already mentioned, dynamic methods are currently being applied to languages like English with quite success. However, there are no studies on Turkish, which has a highly complex morphological structure. In this study, we take the special characteristics of Turkish into account, propose solutions for them, and develop adaptive anti-spam filtering algorithms for the language.

In order to classify e-mails, first a data set containing examples of spam mails and normal mails is compiled. The content of each e-mail example is analyzed, some features of the text are identified, and these features are mapped to a vector space. For instance, the features used for classification may be some selected words, which is the case here. These words are represented with a vector whose each element corresponds to a particular word and indicates whether that word occurs or not in the text or the number of times it occurs. Provided that sufficient number of e-mails are analyzed in this manner, the results will be an indication of how probable these words appear in spam mails and in normal mails. Then a new e-mail can be classified as spam or normal according to these statistics.

In the classification process for agglutinative languages, it is obvious that the features cannot be the surface forms of the words. Different surface forms correspond to a single root form and since they all denote the same concept, the root form should be used as the classifier. Thus, the root words in an e-mail message must be identified before processing it with the learning modules.

The paper is organized as follows: Section 2 discusses the morphological analysis of the e-mails and the data set used in this research. In Sections 3 and 4, we explain the algorithms based on ANN and Bayesian Network, respectively, developed for the classification of e-mails. Section 5 covers the details and the results of the experiments. The last section is for the conclusions.

2. Morphology module and data set

2.1. Morphology module

The anti-spam filtering system described in this paper is composed of two modules: Morphology Module (MM) and Learning Module (LM). A Turkish morphological analysis program was prepared based on a research on Turkish morphology (Güngör, 1995). Turkish is an agglutinative language—given a word in its root form, we can derive a new word by adding an affix (usually a suffix) to this root form, then derive another word by adding another affix to this new word, and so on. This iteration process may continue several levels. A single word in an agglutinative language may correspond to a phrase made up of several words in a non-agglutinative language. This makes the language more complex than non-agglutinative languages like English (Kornfilt, 1997; Lewis, 2002).

Besides the morphological complexity, another important issue that must be dealt with is the use of special Turkish characters (‘ç’,‘ğ’,‘ı’,‘ö’,‘ş’,‘ü’). These characters may not be supported in some keyboards and in message transfer protocols; thus people frequently use their “English-versions” (‘c’,‘g’,‘i’,‘o’,‘s’,‘u’, respectively) instead of the correct forms. In this study, all possible variations in a given word are examined until an acceptable Turkish word is found. For example, if the word *kıtabım* (the correct form is *kıtabım* (*my book*)) appears in a message, first the word as it appears is sent to the MM. When the parse fails, the words *kıtabım*, *kitabım* (*my book*), and *kıtabım* are formed and parsed. When the parse succeeds for one of these surface forms, it is accepted as the correct word. In this case, the MM signals success for *kıtabım* (*my book*) and outputs *kitap* (*book*) as the root.

2.2. Success rate and time complexity

Success rate refers to the ratio of the words that can successfully be parsed by the MM over all the words. We have observed that the module can find the roots of the words in any text file with a success over 90%. Most of the words that cannot be parsed are proper nouns and misspelled words. For time complexity, actual CPU timings (in seconds) of the program were collected. This was found to be approximately 6 s for an average length e-mail. The reason of this long processing time is considering all the possible variations of special Turkish characters, as explained in the previous subsection. In order to decrease the processing time, the variations in each word were restricted to the most probable ones by exploiting the morphological nature of the language. In this way, the processing time was reduced to about one second for an average length e-mail.

2.3. Data set

The spam and normal mail examples were collected under different e-mail addresses. The numbers of spam mails and normal mails used in this study were 410 and 340, respectively. The size of the data set may seem small; the main reason is that there does not exist a publicly available set of compiled messages for Turkish and the data set was built during this study. However, as will be seen in Section 5, it is sufficient for a text categorization problem having only two opposite output classes. Also, in our models we used cross-validation technique (data are shuffled and different parts are used for training and test in each iteration) which enables to obtain sound results from the data set (Cohen, 1995).

From these e-mails, two files were created, one containing all the spam mails and the other containing all the normal mails. The mails were then parsed by the MM and the root forms of the words in the mail contents were identified. The module outputed two files to be processed by the LM, containing the roots of all the words in the spam mails and the normal mails. In the future, we plan to add a Microsoft Outlook interface to the system which will get the mail contents automatically and run the anti-spam filtering algorithm.

3. Artificial neural network classification

Artificial Neural Network (ANN) is a machine learning algorithm that is inspired by the way biological nervous systems, such as the brain, process information. There are two phases in ANN as in many other learning algorithms: in the training phase, a training data set is used to determine the weight parameters that define the neural model. The weight of each neuron or each inter-neuron connection is calculated during this phase. In the test phase, the trained neural model is used to process real test patterns and yield classification results. Knowledge is acquired by the network through these weights and the values of inputs (e.g. Bishop, 1995).

Although our task here is classifying the e-mails into just two classes, the data (e-mail contents) are noisy (a word may appear in both a spam mail and a normal mail) and not well-defined (an e-mail can be considered as spam by some people and as normal by others). The representation of the data needs high dimensionality and more importantly, there is a high variation in the number of active features in different instances. Thus we considered ANN as an appropriate environment. ANN is robust enough to fit a wide range of domains and can model any high degree exponential problem using multi-layer perceptrons (Dagan et al., 1997; Lee et al., 2002). In this research, the standard implementation of ANN was used subject to two improvements. The first is using momentum function while calculating error, which yields more stable results. The second is weight decaying, in which the effect of weight errors decays as the iteration goes on, which generally makes the results more robust to noise (Bishop, 1995). We considered two types of ANN: Single Layer Perceptron (SLP) and Multi-layer Perceptron (MLP).

3.1. Determination of the feature vector

In order to determine the root words that will serve as the features in the classification process, the concept of *mutual information* was used (e.g. Androutsopoulos et al., 2000). The feature vector can be defined as a group of critical words that are used in classification of e-mails as spam or normal. For the formulation of the feature vector, first the candidate words are identified. In this study, the candidates for these critical words were all the words in the training data. The following formula was used in order to obtain a mutual information score for each candidate word W:

$$\mathrm{MI}(W) = \sum_{w\in\{0,1\},c\in\{\text{spam},\text{normal}\}} P(W=w,C=c) \times \log\frac{P(W=w,C=c)}{P(W=w)P(C=c)} \tag{1}$$

where C denotes the class (spam or normal), $P(W=w,C=c)$ is the probability that the word W occurs ($W=1$) or does not occur ($W=0$) in spam ($C=\text{spam}$) or normal ($C=\text{normal}$) mails, $P(W=w)$ is the probability that the word W occurs or not in all e-mails, and $P(C=c)$ is the probability that an e-mail is spam or normal. The probabilities were obtained from the example e-mails in the training set.

A number of words with the highest mutual information scores were selected to form the feature vector. We refer to this number as the *feature vector size*. The algorithms were executed with different feature vector sizes. The highest valued words are most probably the words occurring frequently in one class of e-mails and not so much in the other. So, these selected words are said to be the best classifiers for spam and normal mails.

3.2. Values of input nodes in the ANN

After the feature vector size and the root words that form the feature vector are identified, it is necessary to determine the range of values that each element of the vector can take. Each vector element corresponds to an input node in the ANN. A message was represented by a vector $X = (x_1, x_2, \ldots, x_n)$, where n is the feature vector size and x_i, $1 \leqslant i \leqslant n$, denotes the value of the ith word in the feature vector for that message. There are basically two main approaches in the literature for value assignments and both were considered in this research: *binary model* and *probabilistic model* (e.g. Jones et al., 2000a,b). In the binary model, the problem is whether the word occurs in the text or not:

$$x_i = \begin{cases} 1, & \text{if } i\text{th word of the feature vector occurs in the e-mail} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

This formula does not take the length of the text and the number of occurrences of the word into consideration. Thus, when the text is very short or very long, or when a word occurs several times in an e-mail, the results may not reflect the real situation. On the other hand, the probabilistic model takes these factors into consideration and uses the following estimation:

$$x_i = \frac{\text{Number of occurrences of } i\text{th word of the feature vector in the e-mail}}{\text{Number of all words in the e-mail}} \quad (3)$$

4. Bayesian classification

Bayesian classifier (filter) is a fundamental statistical approach in machine learning (e.g. Mitchell, 1997). It uses the probability and the costs that accompany decisions, so it is based on quantifying the tradeoffs between various classification decisions. Although it is limited in expressive power, it has a surprisingly good performance in a wide variety of domains. The standard Naive Bayesian model is based on the assumption that the inputs are independent of each other. Bayesian classification has been successfully applied to text and document classification in many studies (Androutsopoulos et al., 2000; McCallum and Nigam, 1998). The success of this simple model especially in problems involving two decision classes comes from the fact that the output class is estimated just from the sign of the function estimation and this can be handled using simple mechanisms rather than complex ones.

In our study, the Bayesian approach was used as an alternative to the ANN approach in order to compare their performances for mail classification in an agglutinative language. Given an e-mail, the feature vector $X = (x_1, x_2, \ldots, x_n)$ corresponding to the e-mail was formed in the same way as ANN. Let C_1 denote the class of spam mails and C_2 the class of normal mails. By making use of the assumption that the words in the feature vector are independent of each other and by considering that the prior probability that an e-mail belongs to a class is the same for both classes, the conditional probability that the message belongs to class C_i, $i = 1, 2$, can be written as follows after some common transformations (e.g. Gama, 2000):

$$P(C_i|X) \approx \sum_{j=1}^{n} \log(P(x_j|C_i)) \quad (4)$$

In fact, this formula does not compute a probability, but rather a score that is sufficient to find the most suitable class for the input. In other words, given the feature vector X, we decide that the e-mail is spam if $P(C_1|X) > P(C_2|X)$ and normal if $P(C_2|X) > P(C_1|X)$.

4.1. Bayesian models

Eq. (4) is the general form for a Bayesian classifier. The probabilities on the right-hand side of the equation are estimated from the training data. The general approach is trying to fit a well-known distribution (usually Gaussian distribution) to the data. However, it was shown that Bayesian classifier exhibited limited performance in some domains and this was due to the additional use of the unwarranted Gaussian assumption, and not to the method itself (Domingos and Pazzani, 1997; John and Langley, 1995). In addition, fitting a distribution to the data is appropriate only when the attributes are continuous. In the case of spam mail detection, we have discrete attributes (a word occurs in an e-mail or not, or the number of times it occurs) and thus we did not use a distribution for the data. Instead, we employed summation and computed the conditional probability (score) of each class by summing up the individual probabilities (Gama, 2000).

In this study, we used three different models of Bayesian classification: binary model, probabilistic model, and advanced probabilistic model. These can be regarded as "semi-original" models based mainly on classical methods of discrete Bayesian filtering. As in the case of ANN, cross-validation technique was used in these models.

The binary model is based on whether a word occurs or not in an e-mail. The score for an e-mail with a feature vector X belonging to class C_i, $i = 1, 2$, was calculated by the formula:

$$P(C_i|X) = \sum_{j=1}^{n} \begin{cases} cP_{ij}, & \text{if } j\text{th word of feature vector occurs in e-mail} \\ -P_{ij}, & \text{otherwise} \end{cases} \quad (5)$$

P_{ij} was obtained by dividing the number of e-mails in class C_i containing the jth word by the total number of e-mails in class C_i. For instance, if a word in an e-mail X occurs frequently in e-mails of class C_i, then it will have a large positive effect on $P(C_i|X)$. Likewise, if a word occurs frequently in e-mails of class C_i but does not occur in the e-mail X, then it will have a large negative effect on $P(C_i|X)$. In e-mails, occurrence of an input word usually indicates a stronger idea for classifying e-mails than a non-occurrence of that word. c is the *coefficient level* which indicates the level of this strength. In this study, c was taken between 1 and 41 in different runs. The best results occur when c is between 10 and 30, and the success tends to decrease beyond the point of 41.

In the probabilistic model, the number of occurrences of a word is taken into account. In this case, the score for an e-mail with a feature vector X belonging to class C_i, $i = 1, 2$, was calculated as follows:

$$P(C_i|X) = \sum_{j=1}^{n} \begin{cases} c(P_{ij}H_j), & \text{if } j\text{th word of feature vector occurs in e-mail} \\ -P_{ij}, & \text{otherwise} \end{cases} \quad (6)$$

P_{ij} is the total number of occurrences of the jth word in all e-mails in class C_i divided by the total number of e-mails in class C_i. H_j is the number of occurrences of the jth word in this e-mail.

In the advanced probabilistic model, the length of the e-mail is also considered. The formula for the score of an e-mail with a feature vector X belonging to class C_i, $i = 1, 2$, was calculated as follows:

$$P(C_i|X) = \sum_{j=1}^{n} \begin{cases} c(P_{ij}H_j)/\text{counter}, & \text{if } j\text{th word of feature vector occurs in e-mail} \\ -P_{ij}/\text{counter}, & \text{otherwise} \end{cases} \tag{7}$$

P_{ij}, c, and H_j are the same as in the previous model. *Counter* is the total number of words in that e-mail.

5. Experiments and results

We have performed five experiments by using the methods that were discussed in the previous sections. In an experiment, we first determined the method (ANN or Bayesian Network) to be used and the parameters of the method. If ANN was used in the experiment, the program was executed with three different feature vector sizes (10, 40, and 70) and the success rate was obtained as a function of vector size. For each vector size (i.e. number of inputs), six runs were performed to train and test the network. In each run, 5/6 of the e-mails in the data set (about 340 spam and 280 normal mails) were used for training and the rest for testing. In this way, different parts of the whole data set were treated as training and test examples for each run. The average of the results in the six runs were then taken as the final results for that network. The learning coefficient was taken as 0.1 for ANNs.

If Bayesian network was used in the experiment, then the program was executed with a feature vector size of 70 only and with five different coefficient levels: 1, 11, 21, 31, and 41. The other vector sizes 10 and 40 were not used since the ANN algorithms gave the best results with 70 inputs. The success rate was obtained as a function of coefficient level.

The overall results of the algorithms are summarized in Table 1. The "success rate" column in the table indicates the score of the best case for an algorithm. The best case score is found by averaging the success rate of the algorithm with normal test data and the success rate with spam test data for each feature vector size (in the case of ANN methods) or for each coefficient level (in the case of Bayesian methods), and then taking the highest average. For instance, the best case for the first algorithm in Table 1 occurs with a feature vector size of 40. The success rates for normal and spam test mails are 90% and 68%, respectively, with 40 inputs (see Fig. 1), and thus the best case score is 79%.

All the algorithms were implemented in Visual C++ 6.0 under Windows XP, on an Intel Pentium PC running at 1.6 GHz with 256 MB RAM.

Table 1
Success rates and time complexities of algorithms

Algorithm	Success rate	Average time (s)
SLP (binary model + Turkish words + spam words as candidate)	79	1.5
SLP (probabilistic model + Turkish words + spam words as candidate)	81	4
MLP (probabilistic model + 150 hidden nodes + Turkish words + spam words as candidate)	83	19
MLP (probabilistic model + 250 hidden nodes + Turkish words + spam words as candidate)	83	>100
SLP (probabilistic model + all words (Turkish and non-Turkish) + spam words as candidate)	85	45
SLP (probabilistic model + all words (Turkish and non-Turkish) + all words as candidate (spam and normal))	83	46
SLP (probabilistic model + all words (Turkish and non-Turkish) + all words as candidate (spam and normal, spam more weighted))	86	46
Bayesian (binary model)	89	46
Bayesian (probabilistic model)	86	46
Bayesian (advanced probabilistic model)	84	46

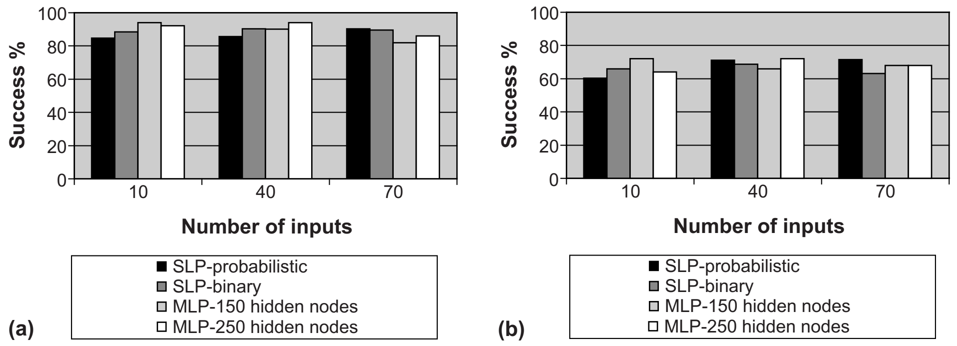

Fig. 1. Results of ANN algorithms. Success of method with: (a) normal test data; (b) spam test data.

5.1. Experiment 1

In this experiment, the success of ANN structure in mail classification task was measured. For SLP, the two input value assignment models explained in Section 3.2 were used. The probabilistic model seemed to achieve more successful results; so with MLP, only the probabilistic model was considered. In addition, MLP was implemented with different numbers of hidden layer nodes. Thus, four different methods were implemented: SLP with binary model, SLP with probabilistic model, MLP with probabilistic model and 150 hidden layer nodes, and MLP with probabilistic model and 250 hidden layer nodes.

The inputs were selected as the words occurring frequently in spam mails and seldom in normal mails by using a special form of Eq. (1):

$$\mathrm{MI}(W) = P(W=1, C=\mathrm{spam}) \times \log \frac{P(W=1, C=\mathrm{spam})}{P(W=1)P(C=\mathrm{spam})} \quad (8)$$

The words with the highest mutual information scores were selected. In this way, the words that are used frequently in spam mails were biased. The intuition behind this idea is that such words are good candidates for filtering. Also, only Turkish words were sent to the LM. This means that when the root form of a word could not be found by the MM, the word was not included in the learning process.

The results are shown in Fig. 1. From this experiment, we conclude that the optimal number of inputs is between 40 and 70. With 10 inputs, success rates are quite low when compared with other input numbers. This indicates that 10 words are not enough to classify e-mails. On the other hand, it seems that the success rates for 40 and 70 inputs are almost the same on the average.

Considering the best cases of these four algorithms, we can conclude that the probabilistic model achieves slightly better performance (81%) than the binary model (79%), as expected (Jones et al., 2000a,b). When compared with SLP algorithms, MLP algorithms give slightly better success rates only with very large hidden layer node numbers. This means high time complexities which yield inefficient results for such type of a filtering application. We plan to design a final product for end-users in the future, so time complexity is an important measure. Thus, with these results, SLP with probabilistic model was selected to be the most efficient among all.

5.2. Experiment 2

Although SLP with probabilistic model seemed to be the best of the ANN models, the success rates of the filter were not as high as expected, especially in spam test runs. To increase these rates, we have developed some more models to work together with SLP. The first of these models used the non-Turkish words in the e-mails in addition to Turkish words. The surface form of a word was used when a root form of the word could not be found by the MM. As shown in Fig. 2, with this modification, the success rate of the filter has reached to 85%, which was previously about 81%.

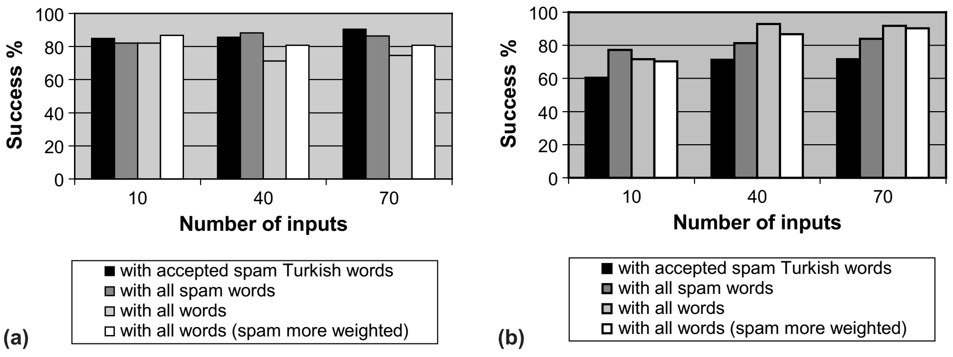

Fig. 2. Results of SLP algorithms. Success of method with: (a) normal test data; (b) spam test data.

The main reason of the improvement on success is the fact that some non-Turkish words (*http*, *www*, *com*, etc.) are frequently used in Turkish spam mails while not so much used in normal mails. Therefore, they can be said to be better classifiers than most of the Turkish words in the e-mails.

We also implemented two other models by improving the selection of the candidates for the feature vector. In the first experiment, most of the words in the feature vector were those occurring frequently in spam mails and much less in normal mails. In the second model of the current experiment, we changed this policy by also including words that occur frequently in normal mails and much less in spam mails (see Eq. (1)). In the analysis of spam mails, this approach showed better results than the previous one; but, as a surprise, a decline of success was observed with normal mails. The best case success rate was 83%. In the third model, which is an improvement of the second model, the probabilities of spam words were multiplied empirically by 3/2. In this case, an increase in success was observed and this model has showed the best performance (86%) among all of the ANN models.

5.3. Experiment 3

In this experiment, the performance of the Bayesian classification method was analyzed. The three models described in Section 4.1 were implemented. As in the previous experiment, both Turkish and non-Turkish words were sent to the LM and both spam and normal words were considered while forming the feature vector. The results are shown in Fig. 3. As can be seen in the figure, the Bayesian approach seems more successful than ANN, especially with binary model (89% total).

The reason for very low success rates with $c = 1$ in spam test mails is the fact that most of the input vector consists of spam words which occur frequently in spam mails and much less in normal mails. Although an e-mail is said to be spam, only

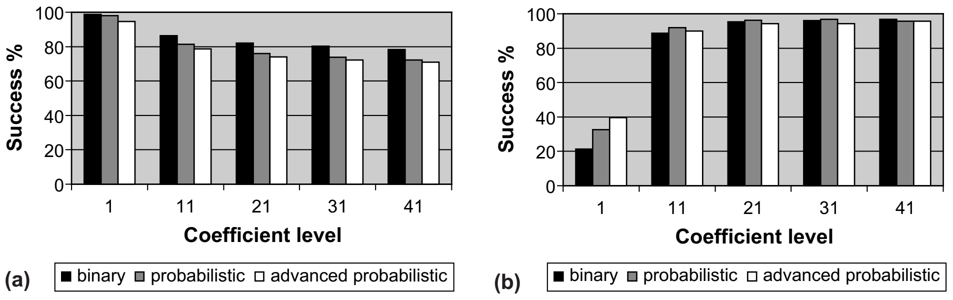

Fig. 3. Results of Bayesian algorithms. Success of method with: (a) normal test data; (b) spam test data.

a small portion of these spam words occur and when $c = 1$, the other spam words that do not occur in the e-mail usually constitute the majority over the spam words in that e-mail. For that reason, most of the spam test mails were concluded as normal with $c = 1$. This is also the reason of the decrease in the success rate of normal test data and the increase of spam test data as the coefficient level increases.

The overall results of Bayesian filter are more successful than ANN filter. This is in accordance with the results in previous studies (e.g. Billsus and Pazzani, 1996; Cestnik, 1990; Clark and Niblett, 1989; Kononenko, 1990; Langley et al., 1992; Sahami, 1996). In all of these, Bayesian algorithms showed better performance than (or at least the same performance as) other more sophisticated and complex algorithms (Domingos and Pazzani, 1997; McCallum and Nigam, 1998). As stated previously, for classification tasks where there are a few decision classes, Bayesian algorithms which base their decisions on the conditional probabilities for classes are more appropriate than ANN algorithms that try to learn an emerging pattern. We also like to note that although classification accuracy is high in Bayesian method, this method is not good at regression (determining the exact value instead of determining the class corresponding to a range of values) and more detailed methods like ANN behave better in regression problems (Bishop, 1995).

Among the three Bayesian models, the binary model shows the best performance and the advanced probabilistic model the worst. This seems interesting since the simplest model gives the best results. In order to understand the reason of this behaviour, we performed other experiments and tested the models using a wide range of feature vector sizes from 10 to 500. The binary model is the best one in the case of a small number of inputs, but it is outweighed by the probabilistic model as the number of inputs increases. After a vector size of 100, the success rate of the binary model decreases steadily. The same decline occurs (but more slowly) in probabilistic model after about 150 inputs and beyond this point it remains more successful than the binary model. (Past 300–400 inputs, the success rates of all three models drop drastically.) The same result was obtained in some studies which involve comparison of Naive Bayesian classification models (McCallum and Nigam, 1998; Sanchez et al., 2002). In another study related to information retrieval where only the probabilistic model was used for document classification, it was stated that increasing the complexity of the model (e.g. incorporating also the number of occurrences of words) results in better performance (Jones et al., 2000a,b). Thus we conclude that the binary model, despite its simplicity relative to other models, is the most successful one in the case of simple domains and small dimensionality (e.g. a small feature vector size). For categorization problems which consist of diverse subject matters with overlapping vocabularies and which include high dimensionality, more complex approaches like probabilistic and advanced probabilistic models yield better results.

When we compare the success rates of the probabilistic model and the advanced probabilistic model, we observe that the former exhibits better performance regardless of the number of features. This may seem counterintuitive since the advanced probabilistic model can be regarded as a more powerful model. There exist no studies using such a model for text categorization in the literature. We argue that the decline of success in the advanced probabilistic model is related to the nature of the problem at hand. The spam mail detection problem with only two opposite output classes and with a small text length can be considered as a relatively simple categorization problem which does not necessitate sophisticated methods. The advanced probabilistic model may be expected to give much more successful results for more complicated tasks in text categorization.

5.4. Experiment 4

Motivated by an interest in understanding the learning rate of the filtering process, we performed an experiment by using probabilistic SLP type of ANN with a feature vector size of 70. In this experiment, the network was trained and tested with different numbers of e-mail. Initially, we used 140 e-mails selected randomly from all the e-mails by preserving the ratio of normal and spam mails. The execution was done in the same way as other

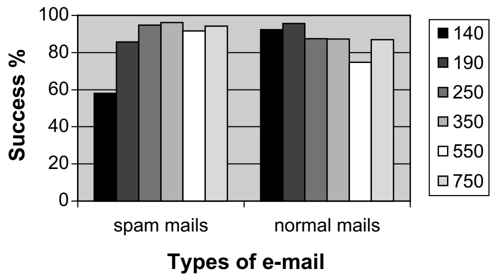

Fig. 4. Results of SLP algorithm with different numbers of e-mail.

ANN executions: Six runs were performed, a different part of the data set containing 5/6 of the e-mails in the set was used for training in each run, and the results were averaged. The process was repeated by increasing the size of the data set.

The results are shown in Fig. 4. We observed that significant success could be obtained with a small number of e-mails (90% success with only 190 e-mails). Thus, we understand that much of the learning takes place with about 100–200 e-mails which means a rapid learning rate.

5.5. Experiment 5

In order to understand the importance of the MM, we have designed an experiment where the MM was excluded and only the LM was used. Similar to the previous experiment, the model in this experiment was SLP with probabilistic model.

Fig. 5 shows the results of the implementation. As can be seen in the figure, with 70 inputs (the best case), 83% success (90% with spam mails and 76% with normal mails) was obtained. This rate does not seem so bad, but it is lower than the previous tests. As discussed before, 86% success was achieved with the same data set using MM module with LM. Thus, we can conclude that MM is really an essential module for spam filtering in agglutinative languages.

5.6. Time complexities

Table 1 shows the average time (in seconds) of each algorithm with both training and test phases. We see that SLP method where only Turkish words are considered is the fastest method with 1.5 s (binary model) and 4 s (probabilistic model). However, as discussed in the experiments, the success rate of SLP with Turkish words only is far below than that of SLP with all words.

When the non-Turkish words are also included, the processing time increases substantially. But, we should note that these experiments were run using 750 e-mails. For a user-specific application, the number of e-mails may be much less. Also, even with this large number of e-mails, the time is below one minute, which is a reasonable time for a user application.

6. Conclusions

In this paper, we proposed several methods for anti-spam filtering in agglutinative languages. We implemented algorithms based on these methods for Turkish and compared their success rates and time complexities. As far as studies on Turkish language are considered, the filter developed in this research is the first one for filtering Turkish spam mails.

We dealt with two important phases in this study. In the first one, a morphological analysis program was prepared based on a previous research. The analyzer extracted the root forms of the words in 750 e-mails with a success of over 90% and a speed of 3000 words/min. In the second phase, these root forms were used by the LM for the classification of e-mails as spam or normal. ANN and Bayesian filters were employed for this purpose, which gave about 90% success in filtering spam mails. The experiments have shown that

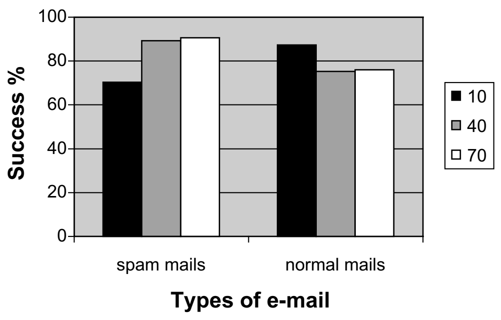

Fig. 5. Results of SLP algorithm without MM.

some non-Turkish words that occur frequently in spam mails (*http*, *www*, *com*, etc.) were better classifiers than most of the Turkish words. We also observed that most of the words in the feature vectors were spam words (e.g. *reklam* (*advertisement*), *promosyon* (*promotion*), *tel* (*phone*)) that appear seldom in normal mails. Some of the few normal words in the feature vectors were private names like the name of the recipient or the names of the recipient's friends. This seems as a useful decision criterion for a user-specific filter; an e-mail including such names is probably not spam. We also examined the role of MM in filtering by designing an experiment in which only LM was used. The decline in the success rate has shown its importance for agglutinative languages.

In future studies, we plan to improve the algorithms by taking the attachments of the e-mails (pictures, text files, etc.) into consideration. The existence and the type of such attachments seem to give important information for detecting spam mails. Also, since anti-spam filtering is a subproblem of text categorization, the scope of this research can be extended and the methods developed can be used for automatic classification of Turkish texts.

Acknowledgment

This research was supported by the Boğaziçi University Research Fund, Grant no. 04A101.

References

Androutsopoulos, I., Koutsias, J., Chandrinos, K.V., Paliouras, G., Spyropoulos, D., 2000. An evaluation of Naive Bayesian anti-spam filtering. In: Potamias, G., Moustakis, V., van Someren, M. (Eds.), Proceedings of Workshop on Machine Learning in the New Information Age, Barcelona. pp. 9–17.

Apte, C., Damerau, F., Weiss, S.M., 1994. Automated learning of decision rules for text categorization. ACM Trans. Inf. Syst. 12 (3), 233–251.

Billsus, D., Pazzani, M., 1996. Learning probabilistic user models. In: Jameson, A., Paris, C., Tasso, C. (Eds.), Proceedings of International Conference on User Modeling, Chia Laguna, Italy.

Bishop, C., 1995. Neural Networks for Pattern Recognition. Oxford University, New York.

Cestnik, B., 1990. Estimating probabilities: a crucial task in machine learning. In: Aiello, L. (Ed.), Proceedings of European Conference on Artificial Intelligence, Stockholm. pp. 147–149.

Clark, P., Niblett, T., 1989. The CN2 induction algorithm. Mach. Learn. 3 (4), 261–283.

Cohen, P.R., 1995. Empirical Methods for Artificial Intelligence. MIT Press, Cambridge, MA.

Cohen, W., 1996. Learning rules that classify e-mail. In: Hearst, M.A., Hirsh, H. (Eds.), Proceedings of AAAI Spring Symposium on Machine Learning in Information Access, Stanford, CA. pp. 18–25.

Dagan, I., Karov, Y., Roth, D., 1997. Mistake-driven learning in text categorization. In: Cardie, C., Weischedel, R. (Eds.), Proceedings of Conference on Empirical Methods in Natural Language Processing, Rhode Island. pp. 55–63.

Domingos, P., Pazzani, M., 1997. On the optimality of the simple Bayesian classifier under zero–one loss. Mach. Learn. 29 (2–3), 103–130.

Gama, J., 2000. A linear-Bayes classifier. In: Monard, M.C., Sichman, J.S. (Eds.), Lecture Notes in Artificial Intelligence, vol. 1952. Springer-Verlag, Heidelberg, pp. 269–279.

Güngör, T., 1995. Computer processing of Turkish: morphological and lexical investigation. Ph.D. Thesis. Boğaziçi University, İstanbul.

John, G., Langley, P., 1995. Estimating continuous distributions in Bayesian classifiers. In: Besnard, P., Hanks, S. (Eds.), Proceedings of Conference on Uncertainty in Artificial Intelligence, Montreal. pp. 338–345.

Jones, K.S., Walker, S., Robertson, S.E., 2000a. A probabilistic model of information retrieval: development and comparative experiments Part I. Inf. Process. Manage. 36, 779–808.

Jones, K.S., Walker, S., Robertson, S.E., 2000b. A probabilistic model of information retrieval: development and comparative experiments Part 2. Inf. Process. Manage. 36, 809–840.

Kononenko, I., 1990. Comparison of inductive and Naive Bayesian learning approaches to automatic knowledge acquisition. In: Wielinga, B. (Ed.), Current Trends in Knowledge Acquisition. IOS Press, Amsterdam, pp. 190–197.

Kornfilt, J., 1997. Turkish. Routledge Press, London.

Langley, P., Iba, W., Thompson, K., 1992. An analysis of Bayesian classifiers. In: Proceedings of National Conference on Artificial Intelligence, San Mateo, CA. pp. 223–228.

Lee, P.Y., Hui, S.C., Fong, A.C.M., 2002. Neural networks for web content filtering. IEEE Intell. Syst. 17 (5), 48–57.

Lewis, D., 1992. Feature selection and feature extraction for text categorization. In: Proceedings of Workshop on Speech and Natural Language, Harriman, New York. pp. 212–217.

Lewis, D., Croft, W.B., 1990. Term clustering of syntactic phrases. In: Vidick, J.L. (Ed.), Proceedings of International Conference on Research and Development in Information Retrieval, Brussels. pp. 385–404.

Lewis, G.L., 2002. Turkish Grammar. Oxford University, Oxford.

McCallum, A., Nigam, K., 1998. A comparison of event models for Naive Bayes text classification. In: Sahami, M. (Ed.), Proceedings of AAAI Workshop on Learning for Text Categorization, Madison, WI. pp. 41–48.

Mitchell, T.M., 1997. Machine Learning. McGraw-Hill, New York.

Sahami, M., 1996. Learning limited dependence Bayesian classifiers. In: Proceedings of International Conference on Knowledge Discovery and Data Mining, Portland. pp. 335–338.

Sanchez, S.N., Triantaphyllou, E., Kraft, D., 2002. A feature mining based approach for the classification of text documents into disjoint classes. Inf. Process. Manage. 38 (4), 583–604.