Spam Filtering Using Integrated Distribution-Based Balancing Approach and Regularized Deep Neural Networks

Aliaksandr Barushka[1], Petr Hajek[1*]
[1]Institute of System Engineering and Informatics, Faculty of Economics and Administration,
University of Pardubice, Studentska 95, Pardubice, 532 10,
Czech Republic
aliaksandr.barushka@upce.cz
petr.hajek@upce.cz
tel.: +420 466 036 147, fax: +420 466 036 010
* corresponding author

Abstract. Rapid growth in the volume of unsolicited and unwanted messages has inspired the development of many anti-spam methods. Supervised anti-spam filters using machine-learning methods have been particularly effective in categorizing spam and non-spam messages. These automatically integrate spam corpora pre-processing, appropriate word lists selection, and the calculation of word weights, usually in a bag-of-words fashion. To develop an accurate spam filter is challenging because spammers attempt to decrease the probability of spam detection by using legitimate words. Complex models are therefore needed to solve such a problem. However, existing spam filtering methods usually converge to a poor local minimum, cannot effectively handle high-dimensional data and suffer from overfitting issues. To overcome these problems, we propose a novel spam filter integrating an *N*-gram *tf.idf* feature selection, modified distribution-based balancing algorithm and a regularized deep multi-layer perceptron NN model with rectified linear units (DBB-RDNN-ReL). As demonstrated on four benchmark spam datasets (Enron, SpamAssassin, SMS spam collection and Social networking), the proposed approach enables capturing more complex features from high-dimensional data by additional layers of neurons. Another advantage of this approach is that no additional dimensionality reduction is necessary and spam dataset imbalance is addressed using a modified distribution-based algorithm. We compare the performance of the approach with that of state-of-the-art spam filters

(Minimum Description Length, Factorial Design using SVM and NB, Incremental Learning C4.5, and Random Forest, Voting and Convolutional Neural Network) and several machine learning algorithms commonly used to classify text. We show that the proposed model outperforms these other methods in terms of classification accuracy, with fewer false negatives and false positives. Notably, the proposed spam filter classifies both major (legitimate) and minor (spam) classes well on personalized / non-personalized and balanced / imbalanced spam datasets. In addition, we show that the proposed model performs better than the results reported by previous studies in terms of accuracy.

1 Introduction

Spam can be defined as an unsolicited and unwanted message sent electronically by a sender that has no current relationship with the recipient [20]. Email spam, a subset of electronic spam, consumes users' time, as users must identify and remove undesired messages; it also takes up limited mailbox space and buries important personal emails [83]. Meanwhile, SMS spam is typically transmitted over a mobile network [21]. Recently, social network spam has received increased attention from both researchers and practitioners due to both the considerable amount of spammers and the potential negative effects of social network spam on convenience and understanding of all the followers [85].

It would be impossible to identify the first person who sent spam. The idea of spam is very simple: to send a message to millions of people and profit from the one person who replies. The availability of unlimited pre-pay SMS packages has enabled the same approach for SMS spam. Increasing the cost of sending spam and reducing the burden spam places on users require highly accurate spam filters [64].

Serious negative effects on the worldwide economy have been observed as a result of high rates of spam [37,46,56], including reduced productivity, the costs associated with delivering spam, and the costs due to viruses or phishing attacks. Therefore, an effective spam filter may also improve user productivity and reduce the consumption of information technology resources such as the help desk. For individuals, more accurate spam filters may increase their trust in email communication [77].

Various spam filters have been developed with machine learning methods being particularly effective, including methods such as Naïve Bayes (NB) classifiers [7,51], decision trees [16,65], support vector machines (SVMs) [8,23], *k*-nearest neighbor algorithm (*k*-NN) [39], K-means clustering [53], artificial immune systems (AIS) [81], multilayer perceptron neural network (MLP) [19,78], and meta-learning methods [42,77]. Machine learning approaches aim to automatically construct word lists and their weights by classifying messages into two classes; the incoming message is either spam or not spam. Misclassifying a legitimate message as spam (a false positive) and misclassifying spam as non-spam (a false negative) carries costs [84]. This is a challenging task because spammers usually attempt to decrease the probability their messages are detected as spam by using legitimate words [64]. Alternative approaches to machine learning methods have also been developed, such as black (white) lists of spammers (trusted senders) and hand-crafted rules [77]. Recent surveys [17,21,31] suggest that Bayesian approaches remain highly popular with researchers, while neural networks (NNs) are significantly under-researched. By contrast to Bayesian approaches, NNs (and SVMs) are more computationally expensive, limiting their maximum potential application to online spam filtering [17]. However, NNs have recently shown promising potential for classifying text, especially when equipped with advanced techniques, such as rectified linear units and dropout regularization [55]. Such techniques can address the main limitations of existing spam filters, namely their optimization convergence to a poor local minimum, problems with handling high-dimensional data, and problems with overfitting. To overcome these drawbacks is a challenging task to solve such a complex problem. Therefore, we propose a novel DBB-RDNN-ReL spam filter that integrates a high-dimensional *N*-gram *tf.idf* feature selection, a modified distribution-based balancing algorithm (DBB) [11], and a regularized deep multi-layer perceptron NN model with rectified linear units (RDNN-ReL) [35] to capture complex features from the high-dimensional data. Compared to existing spam filters [4,48,71,82], an important advantage of the proposed approach is that no additional dimensionality reduction is necessary. In fact, here we show that dimensionality reduction methods specifically designed for high-dimensional datasets deteriorate the performance of DBB-RDNN-ReL.

We initially investigated regularized NNs for spam filtering and presented the results in a conference paper [9]; we present significantly extended results here. Unlike the previous version, limited to shallow NNs (with one hidden layer), here we use a deep NN with two and three hidden layers. Additional hidden layers enable feature hierarchies to increase complexity and abstraction. More complex features can be captured by additional layers of neurons, which recombine features from previous layers to handle high-dimensional data. In the case of spam filtering, this allows larger bags of words to be utilized as NN inputs, which is desirable. In addition, we propose a modification of DBB algorithm to address the issue of imbalanced spam datasets. Unlike under-sampling and total replacement in the original version of the DBB algorithm [11], artificially generated data are used to over-sample the minority class in training data. This approach can be justified by the fact that, unlike the NB classifiers used in [11], RDNN-ReL can effectively handle large datasets [35]. Further, this paper examines the effects of using feature space size and *N*-gram lengths (unigrams, bigrams, and trigrams) as NN inputs. Finally, the results are compared with several state-of-the-art methods for spam filtering, showing that the deep learning NN architecture not only significantly increases the accuracy of spam filtering compared with shallow NNs but also outperforms the state-of-the-art methods for spam filtering on three benchmark datasets.

Specifically, we compare the proposed approach with several state-of-the-art spam filters and machine learning algorithms commonly used to classify text [41] in terms of their accuracy, including rates of false positives and false negatives. We demonstrate that the DBB-RDNN-ReL outperforms other methods on three benchmark spam datasets: Enron, SpamAssassin, and SMS spam collection. We also show that the proposed spam filter performs better than methods previously tested using these datasets. The main contribution of this paper is a novel spam filter methodology that has all four of the following properties: (1) deep learning architecture enables learning complex features from high-dimensional *N*-gram spam data; (2) it is effective as no dimensionality reduction is necessary; (3) the problem of imbalanced spam datasets is handled using a modified DBB algorithm (4) it is effective for all kind of spam datasets, including personalized / non-personalized e-mail spam, SMS spam and social network spam. The results obtained from extensive comparative analyses confirm the effectiveness of our approach compared with the state-of-the-art spam filtering methods, providing more accurate classification on personalized / non-personalized and balanced / imbalanced spam datasets.

The remainder of this paper is organized as follows. Section 2 briefly reviews related literature. Section 3 presents the research methodology, including spam datasets, their pre-processing, and the RDNN-ReL model, along with methods used for comparative analysis. The experiments are performed in Section 5, and Section 6 discusses the obtained results and concludes.

2 Spam Filtering Using Machine Learning: A Literature Review

Spam filtering techniques can be categorized into non-machine learning and machine learning approaches. The former include legislative approaches [15,68], changes to protocols and models of operation [34], rule-, signature-, and hash-based filtering, whitelists and blacklists, and traffic analysis [17]. Kaya and Ertugrul [40] proposed an effective approach based on the probability of using characters in similar orders with respect to their UTF-8 values.

Machine learning spam filters automatically identify whether or not a message is spam based on its content [26]. Following [61] and [83], automated spam filtering can be defined as follows. Let $D = \{d_1, d_2, \dots, d_i, \dots, d_N\}$ be a message set and C = {spam, legitimate} be a class set. The task of a spam filter is to build a model to classify each message $d_i \in D$ as spam or legitimate. With machine learning approaches, spam filtering starts with text pre-processing [32], with tokenization performed first to extract the words (multi-words) in each message. Next, typically, the initial set of words is reduced by stemming, lemmatization, and stop-words removal. Bag-of-words (BoW; also known as the vector-space model) is a common approach to represent the weights of the pre-processed words. Term frequency–inverse document frequency (*tf.idf*) is a popular specific weighting scheme. Feature selection algorithms, such as filters or wrappers [4,48,71,82], may then be applied to reduces the size of the feature space, which is useful mainly because not all classification methods can handle high-dimensional data. Finally, machine learning methods are applied to classify the pre-processed dataset.

The first spam classifiers employed NB algorithms due primarily to their simplicity and computational efficiency [7,51,59]. Concerning SVM, another popular spam-classification algorithm, it was showed that SVMs are robust to both different datasets and pre-processing techniques [23]. Its superiority to NB, *k*-NN, decision trees, and MLP approaches was demonstrated is recent comparative studies [45,75,82]. AISs [76] represent another promising method for spam filtering. Zitar and Hamdan [86] used a genetic algorithm to train AISs to improve the filter performance. Meta-learning algorithms [29] have also recently attracted increasing attention [69]. The combination of boosting and SVM outperformed single classifiers on several benchmark datasets in [70]. Similarly, boosting and bagging were reported to perform significantly better than NB and SVM in a stylometric spam filter [63]. Laorden et al. [46] proposed an anomaly-based spam-filtering system that uses a data reduction algorithm on the labelled dataset, reducing processing time while maintaining high detection rates. Incremental training also reduces processing time [60]. The above-mentioned classification methods usually require sufficient labelled data for the training process, data which are not always available in real-world applications. Semi-supervised approaches have therefore been employed to overcome this problem [2]. In addition, Bosma et al. [13] introduced a framework for unsupervised spam detection in social networking sites, based on user spam reports.

3 Research Methodology

The research methodology employed in this study is introduced in Fig. 1. We used three well-known benchmark datasets as spam corpora so that the performance of the proposed spam filter can be easily compared with the results of previous studies. The corpora were pre-processed with traditional methods, and the bags of words were selected according to their *tf.idf* ranking. The RDNN-ReL model can be effectively handle the high dimensionality of these features, as will be shown by comparative analysis with the benchmark models.

Fig. 1: Research methodology

3.1 Datasets

To evaluate the performance of different spam filters, several benchmark datasets are usually employed. In this paper, to examine the performance of the RDNN-ReL model compared to its competitors, we used the following publicly available spam datasets: (1) Enron[1], (2) SpamAssassin[2], (3) SMS[3] and (4) Social networking[4].

The popular Enron dataset [51] has both spam and ham email messages and has been used in many studies, as overviewed in [31]. The dataset, also called Enron 1, contains a total of 5,172

[1] http://csmining.org/index.php/enron-spam-datasets.html

[2] http://csmining.org/index.php/spam-assassin-datasets.html

[3] https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

[4] http://ilps.science.uva.nl/framework-unsupervised-spam-detection-social-networking-sites/

emails, with 3,672 legitimate and 1,500 spam emails. The messages are in their original forms of non-Latin encodings, with several slight modifications (legitimate emails owners of the mail-boxes sent to themselves and a handful of virus-infected emails were removed).

The SpamAssassin dataset [34], another popular corpus used as a benchmark in many studies, contains 3,252 emails, of which 2,751 are legitimate and 501 are spam emails. Compared to the Enron spam dataset, this dataset is notably more imbalanced, with 84.6% legitimate emails. Comprised of emails randomly collected over a given time period, the dataset is suitable for testing non-personalized spam filters [63]. Almost all headers were reproduced as received.

A SMS spam dataset [4] was chosen here to diversify the spam corpora. Unlike the Enron and SpamAssassin datasets, the SMS spam dataset includes 4,827 legitimate and 747 spam SMS messages, for a total of 5,574 messages. The sources used in this corpus were the Grumbletext Web site (425 SMS spam messages), the NUS SMS Corpus (3,375 legitimate SMS messages), 450 legitimate SMS messages collected from Caroline Tag’s PhD thesis, and the SMS Spam Corpus v.0.1 Big (1,002 legitimate SMS ham messages and 322 spam messages). The average legitimate SMS had 13.18 tokens while the average spam SMS had 23.48 tokens.

The Social networking dataset consists of messages and spam reports from Hyves, the Dutch social networking cite [13]. The original dataset was collected in the year 2010. Unsolicited and promotional messages were labelled as spam. Most of these messages were non-commercial spam messages, such as friend and group invitations or requests to follow a user on Twitter. The dataset contains 355 legitimate and 466 spam messages. The messages are represented as an array of JSON objects with the following fields: the bag of words representation of the message (each word was assigned an anonymized id) and the annotation of the object (spam / ham). Similarly to SMS spam, messages in social networks are generally short, corresponding to sparser datasets. The average legitimate message had 33.15 tokens while the average spam message had 34.70 tokens.

To study the performance of the DBB-RDNN-ReL in comparison with those of the other spam filtering method, we first investigated the complexity of the spam datasets. To control the datasets' complexity, we applied several of the measures proposed by [36] in the Keel software. We included F1 Fisher's discriminant ratio to measure overlaps of individual feature values, N1 to measure the separability of classes' distributions and L1 to measure linear separability of classes. The high value of F1 suggests that the SpamAssassin dataset is a more linear problem with less overlaps compared with the other three datasets, whereas the SMS dataset seems to represent a strongly nonlinear problem, probably attributed to the greater variety of corpus sources included in this dataset. The values of the N1 and L1 suggest that the SpamAssassin dataset has larger margins between classes and is more linearly separable, respectively. Generally, the linear separability of classes increased with the dimensionality of the datasets.

Fig. 2: Complexity measures of spam datasets

3.2 Data Pre-processing

Before attempting to classify legitimate and spam messages, we performed data pre-processing. First, all words were converted to lower-case letters, and tokenization was performed. Unigrams, bigrams, and trigrams were used as tokens, with the following delimiters: `. , ; : ' " ( ) ? !`. Furthermore, we removed stop-words from represented messages using the Rainbow stop-word handler. Stop-words usually provide no semantic information, adding noise to the model [46]. The Snowball stemmer was used as a stemming algorithm. To represent the weights of the pre-processed words, we used *tf.idf*, the most common BoW approach. In this scheme, weights w_{ij} are calculated as follows:

$$w_{ij} = (1 + \log(tf_{ij})) \times \log(N/df_i), \tag{1}$$

where N denotes the total number of messages, tf_{ij} is the frequency of the i^{th} word in the j^{th} message, and df_i denotes the number of messages with at least one occurrence of the i^{th} term. Unlike raw term frequency, *tf.idf* considers both term rareness and document length. To select the most relevant words, we ranked them according to their *tf.idf* weights. For our experiments, we used the Top 200, 1,000, and 2,000 words, in a BoW fashion. Related studies have reported that the most relevant 2,000 words are enough to classify documents [22]. It is also important to include bigrams and trigrams, as previous studies have indicated their potential value [32]. By contrast, using too many features in a spam filter may not only extend calculation time but also deteriorate classification performance due to the higher complexity. Therefore, the use of various numbers of top *N*-grams may also be considered a feature selection method in spam filtering.

To handle the problem of imbalanced datasets, a modified version of DBB algorithm is proposed here. The original version of the DBB combines under- and over-sampling with total replacement of the training set. In this algorithm, new artificial data are created based on learning probability distributions from the training set. Specifically, probability distribution $P(x_i \mid c_k)$ is learnt for each feature x_i, i=1,…,n, and for each class c_k. By using these distributions, a smaller dataset can be artificially generated to enhance the effectiveness of NB algorithms [11]. As RDNN-ReL can effectively handle large datasets [35], we modified the DBB algorithm in order to over-sample the minority class. In other words, the modified DBB algorithm works without replacement, adding b new samples from minority class to the training set. Thus, overlapping among classes is reduced in Algorithm 1. The Gaussian probability distribution was selected because it performed best for spam datasets in [11]:

$$f\left(x_i = w_{ij}\right) = \frac{1}{\sigma\sqrt{2\pi}} exp\left[-\frac{(w_{ij}-\mu)^2}{2\sigma^2}\right], \tag{2}$$

where μ and σ respectively represent the mean and standard deviation of weights for feature x_i restricted to class c_k.

Algorithm 1.

```
Inputs: training set: $D_h$, minority class: $c_k$, number of features: $n$, number
of new instances to sample for minority class: $b$
Output: balanced training set: $D_h^{balanced}$
      For minority class $c_k \in C$ {
            For each feature $x_i$, $i$=1,…,$n$ {
                  learn Gaussian probability distribution $P_{ik}$ from $D_h^{x_i,C}$};
            $D_h^{balanced} \leftarrow D_h$;
            For $p$=1 to $b$ {
                  $newD$ = new double[$n$+1];
                  For each feature $x_i$ {
                        $newD$[$i$] = sample value from $P_{ik}$;
                        $newD$[$n$+1]=$c_k$ //add class label;
                        $D_h^{balanced} = D_h^{balanced} \cup newD$};
            }
      }
      return $D_h^{balanced}$;
```

3.3 Model of a Deep Neural Network

This section introduces the RDNN-ReL model. Complex tasks require many hidden units to model them accurately. Deep NNs with many parameters are extremely powerful machine learning systems that contain multiple hidden layers to process complicated relationships between inputs and outputs. However, the large number of these relationships leads to sampling noise. As a result, complex adaptation to training data may lead to overfitting, preventing high accuracy on testing data. Overfitting can be effectively addressed through dropout regularization. In dropout, the units (hidden and visible) in a NN are temporarily removed from the network, including all their incoming and outgoing connections. In the fully connected layers of a feed-forward NN, dropout regularization randomly sets a given proportion (usually half) of activations to zero during training, thus omitting hidden units that activate the same output.

Commonly used sigmoidal units reportedly suffer from the vanishing gradient problem, often accompanied by slow convergence of optimization to a poor local minimum [49]. Rectified linear (ReL) units tackle this problem. When activated above 0, their partial derivative is 1. Moreover, ReL units saturate upon reaching 0, a characteristic that might be helpful in scenarios in which hidden activations are used as input features for the classifier. The ReL function can be defined as follows:

$$h_i = \max(w_i^T x, 0) = \begin{cases} w_i^T x & \text{if } w_i^T x > 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where w_i is the weight vector of the i^{th} hidden unit and x is the input vector. The ReL function is therefore one-sided and does not enforce a sign symmetry or anti-symmetry. The main disadvantage of ReL is the fact that an NN using this function can easily obtain sparse representation. On the other hand, such an NN has less intensive computation, exploiting the sparsity by avoiding the need to compute the exponential function in activations. The combination of dropout regularization and ReL units has shown promising synergistic effects [38].

We examined different numbers of hidden layers (from one to three) and units in the hidden layers (from 10 to 200; see Fig. 3). Training of the RDNN-ReL was performed with the mini-batch gradient descent algorithm, which updates the synapse weights for every mini-batch *b* of *m* training examples as follows:

$$\theta = \theta - \eta \nabla_{\theta} J(\theta; w^{(j:j+m)}; c^{(j:j+m)}), \tag{4}$$

where every mini-batch includes m training examples, θ are the parameters of the RDNN-ReL, $J(\theta)$ is an objective function to be minimized w.r.t. to the parameters θ, and η denotes learning rate.

On the one hand, this algorithm reduces the updates' variance, thus achieving a more stable convergence. On the other hand, calculating the gradient w.r.t. a mini-batch makes this algorithm highly effective because it utilizes highly optimized matrix optimizations present in deep learning. The structure and parameters of the RDNN-ReL learning were found using a grid search procedure.

Fig. 3: The structure of regularized deep neural network with rectified linear units for spam filtering (crossed neurons are dropped)

3.4 Performance Evaluation

We used common measures to evaluate the performance of the spam filter, namely Accuracy, *FP* (false positive) rates, and *FN* (false negative) rates. *FP* are legitimate messages that are mistakenly regarded as spam, whereas *FN* are spam messages that are not detected. See the confusion matrix in Table 1.

Table 1

Accuracy can be defined as the percentage of correctly classified messages

$$\text{Accuracy} = (TP + TN) / (TP + FP + FN + TN), \tag{5}$$

where *TP* is the number of spam messages classified as spam and *TN* is the number of legitimate messages classified as legitimate.

FP rate (type I error) is calculated as a percentage, the number of legitimate messages incorrectly classified as spam divided by the number of all legitimate messages

$$FP \text{ rate} = FP / (FP + TN). \tag{6}$$

Lastly, *FN* rate (type II error) is also a percentage, the number of spam messages incorrectly classified as legitimate divided by the number of all spam messages

$$FN \text{ rate} = FN / (FN + TP). \tag{7}$$

4 Comparative Spam Filters

To demonstrate the effectiveness of the DBB-RDNN-ReL, we compared the results with recent approaches proposed for spam classification, namely (1) Minimum Description Length [6], (2) Factorial Design using SVM and NB [8], (3) Incremental Learning C4.5 [66], (4) Random Forest [42], (5) Voting [54], and (6) CNN. These comparative methods were used as they represent the state-of-the-art machine learning approaches to spam filtering with supervised learning. We briefly describe these methods below. In addition, we used several traditional machine learning methods, such as *k*-NN, AIS, MLP and AdaBoost to include all types of machine learning methods presented in previous review studies [31].

Minimum Description Length

The Minimum Description Length (MDL) for spam filtering, as introduced in [6,67], is based on the idea the model that fits better the data results in a more compact description for the data. In other words, the model is selected that provides the shortest description length, represented by the sum of the description length of model M, $L(M)$, and the description length of data X when encoded by model M, $L(X \mid M)$. In this study, we use the MDL variant also known as the compression-based spam filter [6]. This algorithm first searches for a term in the training database, and then it either updates the number of messages on the class (spam or non-spam) the term appeared or inserts the term in the database. Thus, the model is incrementally built by the term frequencies.

Factorial Design Analysis using SVM and NB

In the spam filter proposed by [8], factorial design analysis (FDA) is used to obtain the optimal filter setup. Specifically, FDA finds the best combination of three text pre-processing parameters for SVM and NB classifiers. The parameters are represented by stop-words removal (yes/no), lemmatization (yes/no), and the number of features (128/1024), leading to 2^3 factorial design matrix. In [8], SVM-based spam filter performed better without stop-words removal and lemmatization, whereas these linguistic techniques were effective for the NB classifier. For both spam filters, performance increased with a high level of features.

Incremental Learning with C4.5 Decision Tree

The J48 training algorithm is a popular version of the well-known C4.5 decision tree [58]. J48 generates a decision tree model with varying classification rates based on cross-validation. Using fewer features to create the model may benefit performance efficiency by minimizing the number of branches on the tree which must be calculated. In this study, we use an incremental learning mechanism using C4.5 (IL_C4.5) proposed to better adapt to the dynamic environment [66]. In this algorithm, a critical attribute is selected based on the maximum value of Gain Ratio, and the base of association rules is formed using the paths from root nodes to leaf nodes.

Voting

Voting is an ensemble method, combining the decisions of several base learners. Here we use the combination of NB, SVM, and Stochastic Gradient Descent (SGD) classifiers proposed for

SMS spam filtering in [54]. This approach employs majority voting, and it was reported to be more effective in spam filtering than the above-mentioned classifiers trained individually. This was attributed to computational effectivity, fast convergence, and resiliency to overfitting [54].

Random Forests

Recently, it was showed that Random Forests are effective classifiers in spam filtering owing to its non-differentiable decision boundary [42]. Random Forests [14] combine tree predictors in such a way that each single tree depends on the values of a random vector sampled independently from the others, and all trees in the forest have the same distribution. Once the number of trees in the forest grows large enough, the generalization error for the forest converges to a limit. The generalization error depends on two factors: the strengths of individual trees and the correlations between them. Using a random selection of features to split each node yields error rates that compare favorably to AdaBoost, but that are more robust with respect to noise, thus improving the performance of a spam filter [44].

Convolutional Neural Network

A CNN is a variant of MLP, utilizing layers with convolving filters that are applied to the local features of adjacent layers [47]. The filters in any given layer form a feature map and share the same parametrization. Each hidden layer comprises multiple feature maps, obtaining a complex data representation. To capture the most important feature for each feature map, a max-pooling operation is applied over that map. Although originally developed for the computer vision domain, CNNs have recently shown effectiveness in text-categorization tasks [43]. Despite this interest in their use in general text categorization, to the best of our knowledge CNNs have not yet been applied to spam filtering.

5 Experimental Results

The most important parameter of the modified DBB algorithm to set is the number *b* of new instances to sample for minority class. The imbalance ratios ranged from 1:1.3 (Social networking) to 1:6.5 (SMS) in the spam datasets. First, we tested the datasets without over-sampling (*b*=0) as the baseline. In further experiments, we set *b* to achieve the imbalance ratio of 1:2 and 1:1, respectively. In this study, the RDNN-ReL was trained using a mini-batch gradient descent algorithm with the following parameters: number of hidden layers = {1, 2, 3}; number of units in the hidden layer = {10, 20, 50, 100, 200}; learning rate = {0.05, 0.10}; size of each mini-batch used in computing gradients *b* = 100; input layer dropout rate = 0.2; hidden layer dropout rate = 0.5; and number of iterations = 1000. The structure and parameters of the RDNN-ReL were found using a grid search procedure, and we did the same for each comparative method. To estimate the generalization performance of the classifiers, we used 10-fold cross-validation on the three spam datasets, with the overall performance estimate represented by the average and standard deviation over the 10 classifiers.

In the first run of experiments, we investigated the effects of (1) the number of features and (2) the number of *N*-grams on the accuracy of the DBB-RDNN-ReL model. Table 2 shows that DBB-RDNN-ReL performed best for spam datasets with high dimensionality of 2,000 features. However, increasing the feature complexity by using bigrams or trigrams was effective only for the SpamAssassin dataset, which suggests that SpamAssassin presents the most complex spam-filtering problem of the three datasets. Unigrams provided sufficient complexity for spam filtering the remaining two datasets. Note that it was not possible to extract bigrams and trigrams from the Social networking datasets because the order of words in the bag of words representation in the JSON objects was already reorganized. Student's paired *t*-test at *p* = 0.05 was used to test differences in performance. The average differences over all *k*=10 pairs of validation folds were tested with *k*-1 degrees of freedom. Specifically, this test checks whether the average difference in the performance of two compared classifiers is significantly different from zero.

Table 2

Table 3 presents more insight into DBB-RDNN-ReL behavior in terms of how the performance of DBB-RDNN-ReL is affected by the numbers of hidden layers and features. The results indicate that additional hidden layers are beneficial only when using high-dimensional data. In other words, for *n* = 200, DBB-RDNN-ReL performed best with one hidden layer (except for the Social networking dataset). The benefits of additional hidden layers, that is, the development of feature hierarchies and recombinations to increase complexity and abstraction were achieved when using *n* = 2000 features, regardless of the number of *N*-grams.

Table 3

In the further set of experiments, we demonstrated the dominance of the modified DBB algorithm over its original version. Fig. 4 shows that the modified version of the DBB outperformed its original counterpart for all the spam datasets, regardless of the number of features. The greatest improvement was achieved for the SMS and Social networking dataset, suggesting that the oversampling modification is particularly suitable for sparse spam datasets.

Fig. 4 Classification accuracy of DBB-RDNN-Rel with original vs. modified DBB algorithm

Several previous studies on spam filtering have utilized feature selection to decrease data dimensionality, because not all classifiers can gracefully handle high dimensions [82,83]. In addition, due to a lower complexity, reductions in dimensionality may even improve the performance of machine learning methods. The selection of relevant features requires an objective function and a search strategy. Here, we examined the effect of feature selection on DBB-RDNN-ReL performance using two different feature selection algorithms specifically designed for high-dimensional datasets [30]. Generally, filter feature selection methods are preferred in high-dimensional problems due to their computational efficiency. We applied a fast correlation-based filter (CBF) as a subset evaluator and particle swarm optimization (PSO) as a search method. CBF evaluates a feature subset by considering the individual predictive ability of each feature along with the degree of redundancy between them [79]. The following settings were used for the PSO algorithm: number of particles in the swarm = 20; mutation probability = 0.01; individual weight = 0.34; inertia weight = 0.33; and social weight = 0.33. To achieve superior learning performance, the advantages of filters and wrappers have recently been combined in hybrid feature selection methods. The best possible classification performance of a particular machine learning algorithm can thus be achieved with similar time complexity as filter algorithms. Here, we used incremental wrapper feature subset selection (IWSS) with an NB classifier proposed specifically to handle high-dimensional datasets [12]. This wrapper with an embedded NB classifier presents the advantages of a wrapper search alongside the time complexity of a filter algorithm, which is achieved through feature ranking. To alleviate feature selection bias, the selection methods were applied separately to the 10 training datasets. The CBF_PSO method performed better on the SMS dataset, whereas the IWSS_NB dominated for the Enron dataset, regardless of the number of *N*-grams (Table 4). The highest accuracy was achieved with unigrams for both datasets. For the SpamAssassin and Social networking datasets, the feature selection methods performed similarly to each other. However, using feature selection led to comparatively ineffective performance of DBB-RDNN-ReL (see Table 3), indicating that reducing dimensionality deteriorates the accuracy of DBB-RDNN-ReL. Decreases in performance were significant for the Enron and SMS datasets.

Table 4

In the further set of experiments, we compared the results of the DBB-RDNN-ReL to those obtained by other state-of-the-art spam filters, namely MDL [6], FDA [8], IL_C4.5 [66], Voting [54], and Random Forest [42]. All experiments were performed in Weka 3.8 environment. Namely, MDL discretization filter[5] was used for the MDL, StringToWordVector for the factorial design in the FDA, the modification of J4.8 without concept drift judgment for the IL_C4.5, and wekaDeeplearning4jCore 1.0.6 was used to train the CNN.

The incrementally updateable MDL filter was used to select terms. For the FDA, the parameters were represented by stop-words removal (yes/no), lemmatization (yes/no), and the number of features (200/1000). In agreement with [8], SVM and NB were used as classifiers in the FDA framework. SVMs were trained by the SMO algorithm. In the experiments, we examined SVMs with a polynomial kernel function and complexity parameter $C = \{2^0, 2^1, 2^2, \dots , 2^8\}$. To train the IL_C4.5 spam filter, we used the J48 implementation of the C4.5 algorithm with confidence factor = 0.25 and minimum number of instances per leaf = 2. Following the selection of base learners used in [54], NB, SVM and SGD algorithms were used in Voting. The setting of the SVM was the same as for the FDA, while Hinge loss function was used in the SGD. Finally, Random Forest worked with 100 random trees.

As with the DBB-RDNN-ReL, the CNN was trained using a mini-batch gradient descent algorithm with patch size 5×5 and max pool size 2×2, each with number of feature maps = {10, 20,

[5] https://sourceforge.net/projects/weka-mdl-df/

50, 100, 200}; learning rate = {0.05, 0.10}; size of each mini-batch used in computing gradients *b* = 100; input layer dropout rate = 0.2; hidden layer dropout rate = 0.5; and number of iterations = 1000.

Furthermore, we compared the results of the DBB-RDNN-ReL to those obtained by other machine learning methods used in previous spam filters, namely *k*-NN algorithm, logistic regression, MLP, AIS, and AdaBoost. We used the *k*-NN classifier with the Euclidean distance function and number of neighbors set to *k* = 3. The MLP was trained using a backpropagation algorithm with the following parameters: number of neurons in the hidden layer = {10, 20, 50, 100, 200}; learning rate = {0.05, 0.10, 0.30}; momentum = 0.2; and number of iterations = 1000. As a representative of AISs, parallel AIRS version 2 was trained with the following parameters: affinity threshold = 0.2; clonal rate = 10; hyper-mutation rate = 2; *k*-NN = 3; stimulation threshold = 0.9; and number of allocable resources = 150. The AdaBoost M1 version was trained with Decision Stump as base learners and number of iterations = 10.

Classification results are summarized in Table 5. Note that we examined nine configurations (200, 1,000, and 2,000 words using unigrams, bigrams, and trigrams). Here we report only the results for the best configurations in terms of accuracy. In order to evaluate the performance of different algorithms with different parameters set and the accuracy with different word-class sizes, *FN* and *FP* rates were chosen as performance criteria. As before, we employed Student's paired *t*-test at *p* = 0.05 to test average differences in performance.

The results show that DBB-RDNN-ReL achieved higher classification accuracy on the Enron, SMS and Social networking datasets than the other algorithms, while FDA+SVM slightly outperformed DBB-RDNN-ReL on the SpamAssassin dataset. Besides DBB-RDNN-ReL, the CNN, FDA+SVM, Voting and Random Forests algorithms show quite good results for the Enron, SMS and Social networking datasets. For the SpamAssassin dataset, only AIRS2Parallel, MDL and FDA+NB were significantly outperformed by DBB-RDNN-ReL. This may be attributed to a high linear separability of the SpamAssassin dataset.

Table 5

The DBB-RDNN-ReL algorithm also performed significantly better than most of the other algorithms in terms of *FN* rate (Table 6), with nearly one-tenth as many false negatives than the third-best Random Forest algorithm on the Enron dataset. For the SMS and SpamAssassin datasets, the difference in *FN* rates were also significant except Voting on the SpamAssassin dataset. For the Social networking dataset, FDA+NB performed best, suggesting that it is effective in detecting spam messages in social networking. However, this is achieved at the expense of a high rate of legitimate messages mistakenly classified as spam (Table 7).

Table 6

Regarding *FP* rate, DBB-RDNN-Rel also showed good results, but it only achieved the best score for the most complex SMS dataset (Table 7). FDA+NB performed significantly better than DBB-RDNN-ReL for *FP* on the Enron dataset. For the SpamAssassin dataset, the DBB-RDNN-Rel was not significantly outperformed on *FP* rates by any compared method, but AdaBoost, Voting and IL_C4.5 had fewer errors. In addition, Adaboost and MDL performed better than the DBB-RDNN-Rel on the Social networking dataset in terms of *FP* rate. However, these methods should not be preferred due to a relatively poor performance on detecting spam messages (*FN* rate performance).

Since the original SpamAssassin and SMS datasets are strongly imbalanced in favor of legitimate messages, the methods' classification performance in terms of *FN* and *FP* rates is particularly important. It is therefore notable that DBB-RDNN-ReL performed reasonably well on both measures.

Table 7

To detect statistical differences in the performance of the used spam filters across the four datasets (and all the twelve configurations of the datasets), we performed nonparametric Friedman test because the reliability of parametric tests could not be guaranteed. In this test, Friedman statistic is used to rank the methods. Average ranks were calculated in case of ties. The null hypothesis was tested which states that all the spam filters perform similarly. The Friedman *p*-values obtained in Table 8 indicate the existence of significant differences between the evaluated spam filtering methods except *FP* rate. To determine which spam filters performed significantly worse, we next performed the Holm post-hoc procedure (DBB-RDNN-ReL was used as a control algorithm). This procedure adjusts the level of significance in a step-down manner [28]. The results show that only FDA+NB and AIRS2Parallel were significantly outperformed for all the three datasets, while four existing spam filters (FDA+SVM, Voting, CNN, and Random Forest) performed statistically similar at *p*=0.05. Notably, Voting performed well in terms of *FN* rate, indicating a high accuracy on minor (spam) classes. This is in agreement with previous studies on spam filtering [54].

Table 8

To further demonstrate the effectiveness of the proposed spam-filtering model, we compared the average accuracy obtained with that of previous studies that examined the same datasets.

To ensure fair comparability of the results, Table 9, Table 10 and Table 11 only report accuracies obtained with 10-fold cross-validation. Similarly, Table 12 presents the area under ROC curve obtained with 10-fold cross-validation.

Regarding the Enron dataset (Table 9), the best performance thus far reported was achieved by Bagged Random Forest [60] and Deep Belief Networks [70]. The results for Random Forest obtained here agree with those from [60]. Therefore, we believe that these results suggest that DBB-RDNN-ReL performs better other methods in terms of accuracy. Even larger increases in accuracy were achieved in the case of the SMS dataset (Table 10). The SVM proposed in [4] has performed the best so far on this dataset, and the SVM used here reproduced similar results, suggesting that our approach is also more effective for SMS spam filtering. For the SpamAssassin dataset, several methods have performed similarly to ours in previous studies, including SVM, AIS, NB, and Boosting, and our comparative results corroborate these findings. However, DBB-RDNN-ReL achieved slightly higher accuracy than prior studies have reported, as presented in Table 11.

Table 9

Table 10

Table 11

6 Conclusion

This study demonstrates that the DBB-RDNN-ReL outperforms existing spam filtering methods on two of three datasets in terms of classification accuracy. More importantly, it classified both major (legitimate) and minor (spam) classes well. The comparative analysis with the state-of-the-art spam filters showed that DBB-RDNN-Rel ranked first, but FDA+SVM, Voting and

Random Forest spam filters achieved statistically similar performance on the benchmark spam datasets.

By contrast, the remaining algorithms performed relatively poorly in terms of accuracy, *FN* or *FP* rates. The proposed spam filter also outperformed previous approaches on all datasets under investigation, implying that deep NNs represent a promising technique for constructing spam filters. The results also suggest that the performance of DBB-RDNN-ReL improves with higher data dimensionality and that feature selection deteriorates performance. Furthermore, the use of unigrams and bigrams with two hidden layers seems sufficient for DBB-RDNN-ReL to be effective. The relatively large number of units in the hidden layers were examined mainly due to the high number of input features. However, the results showed that adding too many hidden layers and units would model noise in the training data, eventually causing poor generalization performance.

The main limitation of the proposed model is that it is significantly more computationally intensive than the other algorithms (Table 12), with average elapsed training time about ten times higher than that of FDA+SVM for SMS dataset, about thirty times higher than that of FDA+SVM for the Enron dataset, and about hundred times higher for the SpamAssassin dataset. On one hand, this finding limits the application of DBB-RDNN-ReL as an online spam filter [64]. On the other hand, the results suggest that DBB-RDNN-ReL can be effectively used for static datasets. Returning to challenges specific to spam filters, we conclude that the DBB-RDNN-ReL classifier may effectively address imbalanced class distributions and uncertain misclassification costs, as well as complex text patterns. However, its high computational expenses make it difficult to tackle the problem of concept drift. Further investigation and experimentation regarding concept drift is therefore strongly recommended. It would also be interesting to assess the effects of additional text components, such as the syntactic structure and semantic features, on classification accuracy. A further study could also assess the performance of DBB-RDNN-ReL in terms of multi-objective optimization [10]. Finally, we believe that the proposed method can be effectively applied to related high-dimensional imbalanced text categorization problems such as news classification or social network profiling.

Table 12

References

[1] Abi-Haidar A, Rocha LM (2008) Adaptive spam detection inspired by the immune system. In Artificial Life XI, Proc of the 11th Int Conf on the Simul and Synthesis of Living Syst, pp 1-8. doi: 10.1007/978-3-540-85072-4

[2] Ahmed I, Ali R, Guan D, Lee YK, Lee S, Chung T (2015) Semi-supervised learning using frequent itemset and ensemble learning for SMS classification. Expert Syst with Appl 42(3): 1065-1073. doi: 10.1016/j.eswa.2014.08.054

[3] Almeida TA, Almeida J, Yamakami A (2011a) Spam filtering: how the dimensionality reduction affects the accuracy of Naive Bayes classifiers. J of Internet Serv and Appl 1(3): 183-200. doi: 10.1007/s13174-010-0014-7

[4] Almeida TA, Hidalgo JMG, Yamakami A (2011b) Contributions to the study of SMS spam filtering: New collection and results. In Proc of the 11th ACM Symposium on Document Engineering, pp 259-262. doi: 10.1145/2034691.2034742

[5] Almeida TA, Yamakami A (2012) Occam’s razor-based spam filter. J of Internet Serv and Appl 3(3): 245-253. doi: 10.1007/s13174-012-0067-x

[6] Almeida TA, Yamakami A (2016) Compression-based spam filter. Secur and Commun Netw 9(4): 327-335. doi: 10.1002/sec.639

[7] Androutsopoulos I, Koutsias J, Chandrinos KV, Spyropoulos CD (2000) An experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with personal e-

mail messages. In Proc of the 23rd Annual Int ACM SIGIR Conf on Res and Development in Inf Retr, pp 160-167. doi: 10.1145/345508.345569

[8] Aragão MV, Frigieri EP, Ynoguti CA, Paiva AP (2016) Factorial design analysis applied to the performance of SMS anti-spam filtering systems. Expert Syst with Appl 64: 589-604. doi: 10.1016/j.eswa.2016.08.038

[9] Barushka A, Hajek P (2016) Spam filtering using regularized neural networks with rectified linear units. In AI*IA 2016 Adv in Artif Intell, Springer, pp 65-75. doi: 10.1007/978-3-319-49130-1_6

[10] Basto-Fernandes V, Yevseyeva I, Méndez JR, Zhao J, Fdez-Riverola F, Emmerich MT (2016) A spam filtering multi-objective optimization study covering parsimony maximization and three-way classification. Appl Soft Comput 48: 111-123. doi: 10.1016/j.asoc.2016.06.043

[11] Bermejo P, Gámez JA, Puerta JM (2011) Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets. Expert Syst with Appl 38(3): 2072-2080. doi: 10.1016/j.eswa.2010.07.146

[12] Bermejo P, Gámez JA, Puerta JM (2014) Speeding up incremental wrapper feature subset selection with Naive Bayes classifier. Knowl-Based Syst 55: 140-147. doi: 10.1016/j.knosys.2013.10.016

[13] Bosma M, Meij E, Weerkamp W (2012) A framework for unsupervised spam detection in social networking sites. In Eur Conf on Info Retriev, Springer, pp 364-375. Springer, Berlin, Heidelberg. doi: 10.1007/978-3-642-28997-2_31

[14] Breiman L (2001) Random forests. Mach Learn 45(1): 5-32. doi: 10.1023/A:1010933404324

[15] Carpinter J, Hunt R (2006) Tightening the net: A review of current and next generation spam filtering tools. Comput & Secur 25(8): 566-578. doi: 10.1016/ j.cose.2006.06.001

[16] Carreras X, Marquez L (2001) Boosting trees for anti-spam email filtering. In Proc of RANLP 2001, Bulgaria, pp 58-64.

[17] Caruana G, Li M (2012) A survey of emerging approaches to spam filtering. ACM Comput Surv 44(2): 1-27. doi: 10.1145/2089125.2089129

[18] Chhogyal K, Nayak A (2016) An empirical study of a simple Naive Bayes classifier based on ranking functions. In Australasian Jt Conf on Artif Intell, Springer, 324-331. doi: 10.1007/978-3-319-50127-7_27

[19] Clark J, Koprinska I, Poon J (2003) A neural network based approach to automated e-mail classification. In Proc of the IEEE/WIC Int Conf on Web Intell (WI'03), IEEE, pp 702-705. doi: 10.1109/WI.2003.1241300

[20] Cormack GV (2006) Email spam filtering: A systematic review. Found and Trends in Inf Retr 1(4): 335-455. doi: 10.1561/1500000006

[21] Delany SJ, Buckley M, Greene D (2012) SMS spam filtering: Methods and data. Expert Syst with Appl 39(10): 9899-9908. doi: 10.1016/j.eswa.2012.02.053

[22] Dhillon IS, Mallela S, Kumar R (2003) A divisive information-theoretic feature clustering algorithm for text classification. J of Mach Learn Res 3: 1265-1287. doi: 10.1162/153244303322753661

[23] Drucker H, Wu D, Vapnik V (1999) Support vector machines for spam categorization. IEEE Trans on Neural Netw 10(5): 1048-1054. doi: 10.1109/72.788645

[24] El Boujnouni M (2017) SMS spam filtering using N-gram method, information gain metric and an improved version of SVDD classifier. J of Eng Sci & Technol Rev 10(1): 131-137.

[25] Fang A (2016) Applications of the maximum entropy principle in spam email classification. J of Residuals Sci & Technol 13(6): 1-4. doi: 10.12783/issn.1544-8053/13/6/1

[26] Fawcett T (2003) In vivo spam filtering: A challenge problem for KDD. ACM SIGKDD Explor Newsl 5(2): 140-148. doi: 10.1145/980972.980990

[27] Fdez-Riverola F, Iglesias EL, Diaz F, Méndez JR, Corchado JM (2007) SpamHunting: An instance-based reasoning system for spam labelling and filtering. Dec Supp Syst 43(3): 722-736. doi: 10.1016/j.dss.2006.11.012

[28] Freund Y, Schapire R, Abe N (1999) A short introduction to boosting. Journal-Japanese Soc For Artif Intell 14(5): 771-780.

[29] Garcia S, Fernandez A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Inf Sci 180(10): 2044-2064. doi: 10.1016/j.ins.2009.12.010

[30] Gheyas IA, Smith LS (2010) Feature subset selection in large dimensionality domains. Pattern Recognit 43(1): 5-13. doi: 10.1016/j.patcog.2009.06.009

[31] Guzella T, Caminhas W (2009) A review of machine learning approaches to spam filtering. Expert Syst with Appl 36(7): 10206-10222. doi: 10.1016/j.eswa.2009.02.037

[32] Hagenau M, Liebmann M, Neumann D (2013) Automated news reading: Stock price prediction based on financial news using context-capturing features. Dec Supp Syst 55(3): 685-697. doi: 10.1016/j.dss.2013.02.006

[33] Hassan D (2016) Investigating the effect of combining text clustering with classification on improving spam email detection. In Madureira A, Abraham A, Gamboa D, Novais P (Eds), Int Conf on Intell Syst Des and Appl, Springer, Cham, pp 99-107. doi: 10.1007/978-3-319-53480-0_10

[34] Henning JL (2006) SPEC CPU2006 benchmark descriptions. ACM SIGARCH Comput Archit News 34(4): 1-17. doi: 10.1145/1186736.1186737

[35] Hinton G, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov R (2012) Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580

[36] Ho TK, Basu M (2002) Complexity measures of supervised classification problems. IEEE T Pattern Anal 24(3): 289-300. doi: 10.1109/34.990132

[37] Hoanca B (2006) How good are our weapons in the spam wars?. IEEE Technol and Soc Mag 25(1): 22-30. doi: 10.1109/MTAS.2006.1607720

[38] Jaitly N, Hinton G (2011) Learning a better representation of speech soundwaves using restricted Boltzmann machines. In IEEE Int Conf on Acoustics, Speech and Signal Processing (ICASSP), pp 5884-5887. doi: 10.1109/ICASSP.2011.5947700

[39] Jiang S, Pang G, Wu M, Kuang L (2012) An improved k-nearest-neighbor algorithm for text categorization. Expert Syst with Appl 39(1): 1503-1509. doi 10.1016/ j.eswa.2011.08.040

[40] Kaya Y, Ertuğrul ÖF (2016) A novel approach for spam email detection based on shifted binary patterns. Secur and Commun Netw 9(10): 1216-1225. doi: 10.1002/sec.1412

[41] Khan A, Baharudin B, Lee L (2010) A review of machine learning algorithms for text-documents classification. J of Adv in Inf Technol 1(1): 4-20. doi: 10.1016/j.eswa.2011.08.040

[42] Khorshidpour Z, Hashemi S, Hamzeh A (2017) Evaluation of random forest classifier in security domain. Appl Intell. doi:10.1007/s10489-017-0907-2

[43] Kim Y (2014) Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[44] Koprinska I, Poon J, Clark J, Chan J (2007) Learning to classify e-mail. Inf Sci 177(10): 2167–2187. doi: 10.1016/j.ins.2006.12.005

[45] Lai C (2007) An empirical study of three machine learning methods for spam filtering. Knowl-Based Syst 20(3): 249-254. doi: 10.1016/j.knosys.2006.05.016

[46] Laorden C, Ugarte-Pedrero X, Santos I, Sanz B, Nieves J, Bringas PG (2014) Study on the effectiveness of anomaly detection for spam filtering. Inf Sci 277: 421-444. doi: 10.1016/j.ins.2014.02.114

[47] LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc of the IEEE 86(11): 2278-2324. doi: 10.1109/5.726791

[48] Liu Y, Wang Y, Feng L, Zhu X (2016). Term frequency combined hybrid feature selection method for spam filtering. Pattern Anal and Appl 19(2): 369-383. doi: 10.1016/j.asoc.2016.06.043

[49] Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In Proc of the 30th Int Conf on Mach Learn, Vol. 30, pp 1-6.

[50] Méndez J, Corzo B, Glez-Peña D, Fdez-Riverola F, Díaz F (2007) Analyzing the performance of spam filtering methods when dimensionality of input vector changes. In Perner P (Ed) Mach Learn and Data Min in Pattern Recognit, Springer, Berlin, Heidelberg, pp 364-378. doi: 10.1007/978-3-540-73499-4_28

[51] Metsis V, Androutsopoulos I, Paliouras G (2006) Spam filtering with Naive Bayes - Which Naive Bayes?. In Third Conf on Email and AntiSpam (CEAS), pp 27-28. doi: 10.1.1.61.5542

[52] Mishra R, Thakur RS (2013) Analysis of random forest and Naive Bayes for spam mail using feature selection catagorization. Int J of Comput Appl 80(3): 42-47.

[53] Nagwani NK, Sharaff A (2017) SMS spam filtering and thread identification using bi-level text classification and clustering techniques. J of Inf Sci 43(1): 75-87. doi: 10.1177/0165551515616310

[54] Najadat H, Abdulla N, Abooraig R, Nawasrah S (2016) Spam detection for mobile Short messaging service using data mining classifiers. Int J of Comput Sci and Inf Secur 14(8): 511-517.

[55] Nam J, Kim J, Mencía EL, Gurevych I, Fürnkranz J (2014) Large-scale multi-label text classification - Revisiting neural networks. In Calders T, Esposito F, Hüllermeier E, Melo R (Eds), Mach Learn and Knowl Discovery in Databases, Springer, Berlin, Heidelberg, pp 437-452. (2014). doi: 10.1007/978-3-662-44851-9_28

[56] Obied A, Alhajj R (2009) Fraudulent and malicious sites on the web. Appl Intell 30(2): 112-120. doi: 10.1007/s10489-007-0102-y

[57] Rozza A, Lombardi G, Casiraghi E (2009) Novel IPCA-based classifiers and their application to spam filtering. In Ninth Int Conf on Intell Syst Des and Appl, ISDA'09, pp 797-802. IEEE. doi: 10.1109/ISDA.2009.21

[58] Quinlan JR (1996) Improved use of continuous attributes in C4. 5. J of Artificial Intell Res 4: 77-90. doi: 10.1613/jair.279

[59] Sahami M, Dumais S, Heckerman D, Horvitz E (1998) A Bayesian approach to filtering junk e-mail. In Learn for Text Categorization, Papers from the 1998 Workshop, Vol. 62, pp 98-105. doi: 10.1.1.48.1254

[60] Sanghani G, Kotecha K (2016) Personalized spam filtering using incremental training of support vector machine. In Int Conf on Comput, Anal and Secur Trends (CAST), IEEE, pp 323-328. doi: 10.1109/CAST.2016.7914988

[61] Sebastiani F (2002) Machine learning in automated text categorization. ACM Comput Surv (CSUR) 34(1): 1-47. doi: 10.1145/505282.505283

[62] Shams R, Mercer RE (2013) Personalized spam filtering with natural language attributes. In 12th Int Conf on Mach Learn and Appl (ICMLA), Vol. 2, IEEE, pp 127-132. doi: 10.1109/ICMLA.2013.117

[63] Shams R, Mercer RE (2016) Supervised classification of spam emails with natural language stylometry. Neural Comput and Appl 27(8): 2315-2331. doi: 10.1007/s00521-015-2069-7

[64] Shen H, Li Z (2014) Leveraging social networks for effective spam filtering. IEEE Trans on Comput 63(11): 2743-2759. doi: 10.1109/TC.2013.152

[65] Sheu JJ, Chen YK, Chu KT, Tang JH, Yang WP (2016) An intelligent three-phase spam filtering method based on decision tree data mining. Secur and Commun Netw 9(17): 4013-4026. doi: 10.1002/sec.1584

[66] Sheu JJ, Chu KT, Li NF, Lee CC (2017) An efficient incremental learning mechanism for tracking concept drift in spam filtering. PloS One 12(2): e0171518. doi: 10.1371/journal.pone.0171518

[67] Silva RM, Alberto TC, Almeida TA, Yamakami A (2017) Towards filtering undesired short text messages using an online learning approach with semantic indexing. Expert Syst with Appl 83: 314-325. doi: 10.1016/j.eswa.2017.04.055

[68] Talbot D (2008) Where spam is born. MIT Technol Rev.

[69] Trivedi SK, Dey S (2013) An enhanced genetic programming approach for detecting unsolicited emails. In IEEE 16th Int Conf on Comput Sci and Eng (CSE), pp 1153-1160. doi: 10.1109/CSE.2013.171

[70] Trivedi SK, Dey S (2016a) A combining classifiers approach for detecting email spams. In 30th Int Conf on Adv Infn Netw and Appl Workshops (WAINA), IEEE, pp 355-360. doi: 10.1109/WAINA.2016.127

[71] Trivedi SK, Dey S (2016b) A comparative study of various supervised feature selection methods for spam classification. In Proc of the 2nd Int Conf on Inf and Commun Technol for Competitive Strateg, ACM, p. 64. doi: 10.1145/2905055.2905122

[72] Tzortzis G, Likas A (2007) Deep belief networks for spam filtering. In 19th IEEE Int Conf on Tools with Artif Intell, ICTAI 2007, Vol. 2, IEEE, pp 306-309. doi: 10.1109/ICTAI.2007.65

[73] Uysal AK, Gunal S (2012) A novel probabilistic feature selection method for text classification. Knowl-Based Syst 36: 226-235. doi: 10.1016/j.knosys.2012.06.005

[74] Uysal AK, Gunal S, Ergin S, Gunal ES (2012) A novel framework for SMS spam filtering. In 2012 Int Symp on Innov in Intell Syst and Appl (INISTA), IEEE, pp 1-4. doi: 10.1109/INISTA.2012.6246947

[75] Vyas T, Prajapati P, Gadhwal S (2015) A survey and evaluation of supervised machine learning techniques for spam e-mail filtering. In IEEE Int Conf on Electr, Comput and Communn Technol (ICECCT), IEEE, pp 1-7. doi: 10.1109/ICECCT. 2015.7226077

[76] Watkins A, Timmis J (2004) Artificial immune recognition system (AIRS): An immune-inspired supervised learning algorithm. Genetic Program and Evolvable Mach 5(3): 291-317. doi: 10.1023/B:GENP.0000030197.83685.94

[77] Wei CP, Chen HC, Cheng TH (2008) Effective spam filtering: A single-class learning and ensemble approach. Decis Supp Syst 45(3): 491-503. doi: 10.1016/j.dss.2007.06.010

[78] Wu CH, Tsai CH (2009) Robust classification for spam filtering by back-propagation neural networks using behavior-based features. Appl Intell 31: 107-121. doi: 10.1007/s10489-008-0116-0

[79] Yu L, Liu H (2003) Feature selection for high-dimensional data: A fast correlation-based filter solution. In Int Conf on Mach Learn, Vol. 3, pp 856-863.

[80] Yu B, Xu, ZB (2008) A comparative study for content-based dynamic spam classification using four machine learning algorithms. Knowl-Based Syst 21(4): 355-362. doi: 10.1016/j.knosys.2008.01.001

[81] Yue X, Abraham A, Chi ZX, Hao YY, Mo H (2007) Artificial immune system inspired behavior-based anti-spam filter. Soft Comput - A Fusion of Found, Methodol and Appl 11(8): 729-740. doi: 10.1007/s00500-006-0116-0

[82] Zhang Y, Wang S, Phillips P, Ji G (2014) Binary PSO with mutation operator for feature selection using decision tree applied to spam detection. Knowl-Based Syst 64: 22-31. doi: 10.1016/j.knosys.2014.03.015

[83] Zhang L, Zhu J, Yao T (2004) An evaluation of statistical spam filtering techniques. ACM Trans on Asian Lang Inf Process 3(4): 243-269. doi: 10.1.1.109.7685

[84] Zheng X, Zeng Z, Chen Z, Yu Y, Rong C (2015) Detecting spammers on social networks. Neurocomput 159: 27-34. doi: 10.1016/j.neucom.2015.02.047

[85] Zhou B, Yao Y, Luo J (2014) Cost-sensitive three-way email spam filtering. J of Intell Inf Syst 42(1): 19-45. doi: 10.1007/s10844-013-0254-7

[86] Zitar RA, Hamdan A (2013) Genetic optimized artificial immune system in spam detection: a review and a model. Artif Intell Rev 40(3): 305-377. doi: 10.1007/s10462-011-9285-z

Table 1: Confusion matrix for spam filtering

		Actual	
		Spam	Legitimate
Predicted	Spam	*TP*	*FP*
	Legitimate	*FN*	*TN*

Table 2: Effect of number of features and *N*-grams on DBB-RDNN-ReL accuracy

Features / *N*-grams	Dataset			
	Enron	SMS	SpamAssassin	Social networking
200 / 1	96.23±0.84*	95.86±0.76*	99.45±0.45	90.14±3.87
200 / 2	91.79±1.07*	88.96±0.80*	99.84±0.20	-
200 / 3	82.36±1.55*	86.86±0.45*	99.84±0.22	-
1000 / 1	98.37±0.59	98.23±0.57	99.75±0.28	92.27±3.04
1000 / 2	95.83±0.87*	93.98±1.18*	99.82±0.22	-
1000 / 3	87.79±1.47*	91.71±0.98*	99.70±0.32	-
2000 / 1	**98.76±0.57**	**98.51±0.51**	99.79±0.25	**92.81±2.84**
2000 / 2	96.69±0.81	95.50±0.93*	**99.89±0.19**	-
2000 / 3	91.39±1.18*	92.98±0.84*	99.85±0.30	-

* significantly lower (*p*=0.05).

Table 3: Effect of number of hidden layers and features on DBB-RDNN-ReL accuracy

Hidden layers / Features	Dataset			
	Enron	SMS	SpamAssassin	Social networking
1 / 200	96.23±0.84*	95.86±0.76*	99.84±0.22	89.69±3.75*
1 / 1000	98.37±0.59	98.23±0.57	99.80±0.23	92.27±3.04
1 / 2000	96.69±0.81	**98.51±0.51**	99.84±0.22	92.17±3.17
2 / 200	95.05±1.41*	95.78±0.50*	99.45±0.45	90.14±3.87
2 / 1000	97.93±0.84	98.17±0.48	99.82±0.22	92.20±3.00
2 / 2000	**98.76±0.57**	98.47±0.39	**99.89±0.19**	91.96±3.21
3 / 200	92.24±7.59*	90.25±4.77*	99.38±0.49	89.77±4.15*
3 / 1000	84.80±11.93*	87.75±3.61*	99.75±0.19	**92.81±2.84**
3 / 2000	87.49±14.18*	87.70±3.50*	**99.89±0.19**	**92.81±3.07**

* significantly lower at *p*=0.05.

Table 4: Effect of feature selection on DBB-RDNN-ReL accuracy

Feature selection	*N*-grams	Dataset			
		Enron	SMS	SpamAssassin	Social networking
CBF_PSO	1	93.49±2.55*	96.44±0.72*	99.50±0.36	90.99±3.64
CBF_PSO	2	88.89±1.35*	95.60±0.74*	99.47±0.37	-
CBF_PSO	3	82.64±1.22*	92.32±0.92*	99.72±0.28	-
IWSS_NB	1	94.39±0.99*	95.63±0.81*	99.54±0.40	91.84±2.44
IWSS_NB	2	89.12±1.19*	92.81±0.92*	99.58±0.39	-
IWSS_NB	3	84.52±1.34*	86.68±0.56*	99.47±0.46	-

* significantly lower (*p*=0.05) compared to the best result without feature selection.

Table 5: Accuracy of compared methods

Method	Dataset			
	Enron	SMS	SpamAssassin	Social networking
MDL [6]	95.67±1.13*	97.99±0.55	94.43±1.19*	89.28±3.57*
FDA+NB [8]	91.29±1.18*	94.95±0.88*	92.36±1.56*	82.98±7.08*
FDA+SVM [8]	96.66±0.84	97.52±0.72	**99.90±0.16**	90.45±3.25
IL_C4.5 [66]	93.35±1.29*	95.67±0.90*	99.72±0.31	88.65±3.84*
Voting [54]	97.20±1.06	98.20±0.32	99.88±0.16	92.08±3.06
Random Forest [42]	98.05±0.57	97.89±0.63	99.76±0.28	91.94±3.09
AdaBoost	78.76±1.15*	88.65±0.60*	99.59±0.43	89.10±3.73*
Logistic Regr.	94.54±1.04*	96.80±0.75	99.62±0.69	87.59±3.60*
AIRS2Parallel	71.36±7.65*	87.31±2.83*	94.79±2.80*	80.42±7.78*
MLP	96.29±2.84*	95.50±1.10*	99.61±0.36	88.97±3.86*
k-NN	91.36±1.34*	93.57±0.78*	99.35±0.41	88.37±3.70*
CNN	97.47±0.87	98.01±1.30	99.72±0.27	91.11±4.52
DBB-RDNN-Rel	**98.76±0.57**	**98.51±0.51**	99.89±0.19	**92.81±2.84**

* significantly lower (*p*=0.05).

Table 6: *FN* rates of comparative methods

Method	Dataset			
	Enron	SMS	SpamAssassin	Social networking
MDL [6]	0.0107±0.0105*	0.1341±0.0330*	0.0300±0.0330*	0.1673±0.0558*
FDA+NB [8]	0.1209±0.0163*	0.1318±0.0388*	0.0694±0.0370*	**0.0698±0.0509**
FDA+SVM [8]	0.0459±0.0178*	0.1192±0.0383*	0.0140±0.0184*	0.1126±0.0459*
IL_C4.5 [66]	0.0608±0.0207*	0.2719±0.0604*	0.0167±0.0187*	0.1397±0.0545*
Voting [54]	0.0247±0.0118*	0.1072±0.0259*	0.0040±0.0084	0.0900±0.0481*
Random Forest [42]	0.0178±0.0121*	0.1403±0.0458*	0.0080±0.0136*	0.1000±0.0468*
AdaBoost	0.6838±0.0340*	0.8190±0.0426*	0.0265±0.0272*	0.1706±0.0595*
Logistic Regr.	0.0668±0.0200*	0.1210±0.0382*	0.0082±0.0355*	0.1502±0.0559*
AIRS2Parallel	0.1220±0.1679*	0.8801±0.0885*	0.1888±0.1069*	0.2540±0.1450*
MLP	0.0501±0.0823*	0.2294±0.0608*	0.0144±0.0211*	0.1424±0.0557*
k-NN	0.0378±0.0158*	0.4534±0.0562*	0.0209±0.0187*	0.1609±0.0595*
CNN	0.0347±0.0193*	0.1232±0.0947*	0.0100±0.0141*	0.0916±0.0457*
DBB-RDNN-Rel	**0.0017±0.0018**	**0.0953±0.0348**	**0.0010±0.0032**	0.0900±0.0449*

* significantly higher (*p*=0.05).

Table 7: *FP* rates of compared methods

Methods	Dataset			
	Enron	SMS	SpamAssassin	Social networking
MDL [6]	0.0566±0.0165*	0.0025±0.0038	0.0603±0.0124*	0.0282±0.0131
FDA+NB [8]	**0.0045±0.0057**	0.0380±0.0083*	0.0776±0.0166*	0.3021±0.1946*
FDA+SVM [8]	0.0283±0.0092*	0.0102±0.0056*	0.0036±0.0036*	0.0729±0.0460*
IL_C4.5 [66]	0.0688±0.0159*	0.0079±0.0045*	0.0002±0.0011	0.0791±0.0497*
Voting [54]	0.0294±0.0130*	0.0041±0.0032*	0.0007±0.0015	0.0648±0.0326*
Random Forest [42]	0.0201±0.0073*	0.0027±0.0026	0.0014±0.0024	0.0552±0.0363*
AdaBoost	0.0200±0.0800*	0.0044±0.0026*	**0.0001±0.0004**	**0.0281±0.0249**
Logistic Regr.	0.0496±0.0125*	0.0182±0.0065*	0.0031±0.0038*	0.0899±0.0478*
AIRS2Parallel	0.3535±0.1519*	0.0103±0.0371*	0.0272±0.0331*	0.1190±0.1853*
MLP	0.0318±0.0361*	0.0165±0.0118*	0.0020±0.0031	0.0681±0.0610*
k-NN	0.1062±0.0177*	0.0041±0.0030	0.0039±0.0034*	0.0577±0.0397*
CNN	0.0215±0.0079*	0.0039±0.0021	0.0015±0.0025	0.0855±0.1007*
DBB-RDNN-Rel	0.0212±0.0101*	**0.0024±0.0024**	0.0011±0.0018	0.0480±0.0194*

* significantly higher (*p*=0.05).

Table 8: Results of Friedman and Holm nonparametric tests

	Accuracy		*FN* rate		*FP* rate	
Methods	Aver. Ranking	Holm *p*-value	Aver. Ranking	Holm *p*-value	Aver. Ranking	Holm *p*-value
MDL [6]	7.3	0.033*	7.8	0.026*	6.5	0.146
FDA+NB [8]	10.5	0.001*	9.3	0.006*	8.5	0.029*
FDA+SVM [8]	5.5	0.134	6.5	0.077	7.3	0.085
IL_C4.5 [66]	7.9	0.018*	8.5	0.013*	7.5	0.069
Voting [54]	5.1	0.173	2.0	0.892	7.1	0.093
Random Forest [42]	3.3	0.496	4.1	0.364	4.5	0.468
AdaBoost	9.5	0.003*	10.3	0.002*	4.8	0.414
Logistic Regr.	6.3	0.077	5.8	0.134	8.3	0.037*
AIRS2Parallel	11.0	0.000*	12.5	0.000*	8.8	0.023*
MLP	8.5	0.010*	8.3	0.016*	9.3	0.014*
k-NN	11.0	0.000*	9.8	0.003*	9.9	0.008*
CNN	3.9	0.364	4.8	0.256	6.3	0.173
DBB-RDNN-Rel	1.4	-	1.6	-	2.5	-
Friedman *p*-value	0.002*		0.001*		0.304	

* significantly worse than DBB-RDNN-Rel (*p*=0.05).

Table 9: Comparison of DBB-RDNN-ReL accuracy with the results of previous studies on the Enron 1 dataset

Study	Method	Accuracy
[70]	Deep Belief Networks	97.43
[1]	AIS	90.00
[3]	Multivariate Bernoulli NB	94.79
[71]	Distinguishing Feature Selector	94.35
[5]	Minimum description length	95.56
[60]	Bagged RF	97.75
[67]	Enhanced genetic programming	94.10
[50]	RF	96.39
[69]	Relief + NB	96.30
[32]	*k*-means + SVM	97.35
[17]	Natural language toolkit NB	94.70
[58]	Incremental SVM	96.86
[68]	Boosted NB + SVM	95.60
This study	DBB-RDNN-ReL	**98.76**

Table 10: Comparison of DBB-RDNN-ReL accuracy with the results of previous studies on the SMS dataset

Study	Method	Accuracy
[4]	SVM	97.64
[71]	Distinguishing Feature Selector	97.44
[72]	χ^2 filter + probabilistic classifier	90.17
[2]	Apriori + ensemble learning	96.21
[52]	Discriminative multinomial NB	96.46
[23]	Support Vector Domain Description	89.32
This study	DBB-RDNN-ReL	**98.51**

Table 11: Comparison of DBB-RDNN-ReL accuracy with the results of previous studies on the SpamAssassin dataset

Study	Method	Accuracy
[14]	Heuristic filter + NB	97.67
[48]	SVM	98.53
[26]	Case-based Reasoning	93.58
[70]	Deep Belief Networks	97.50
[78]	SVM	97.00
[55]	Isotropic PCA	98.89
[83]	Genetic optimized AIS	98.92
[67]	Enhanced genetic programming	98.60
[69]	OneR + NB	96.40
[24]	Maximum entropy + incremental learning	97.87
[61]	Natural language stylometry + Adaboost	95.70
[68]	Boosted NB + SVM	98.60
This study	DBB-RDNN-ReL	**99.89**

Table 12: Comparison of DBB-RDNN-ReL accuracy with the results of previous studies on the Social network dataset

Study	Method	Area under ROC
[13]	NB baseline	0.528
[13]	Report baseline	0.548
[13]	HITS unsupervised	0.767
[13]	HITS semi-supervised	0.801
This study	DBB-RDNN-ReL	**0.961**

Table 13: Average elapsed training time in seconds

Methods	Dataset			
	Enron	SMS	SpamAssassin	Social networking
MDL [6]	10.1864±1.4689	5.6712±1.0959	8.8657±0.6047	1.7175±0.0557
FDA+NB [8]	0.8592±0.0374	1.5929±0.0773	1.0731±0.0643	0.0182±0.0052
FDA+SVM [8]	5.4755±1.1081	6.8151±1.3511	1.7081±0.2148	0.9279±0.4131
IL_C4.5 [64]	40.0021±3.6369	57.0352±2.2090	17.7206±1.8892	1.8255±0.1087
Voting [52]	34.3022±2.0173	24.0878±1.0569	23.1259±0.6988	0.3704±0.0130
Random Forest [40]	28.0200±0.3813	137.9478±2.7606	8.8578±0.6931	5.2441±0.1221
AdaBoost	2.7871±0.0530	1.3228±0.0433	30.0971±5.5630	0.2033±0.0261
Logistic Regr.	4.7599±0.8967	33.0418±4.9804	81.4898±10.8791	0.1585±0.0292
AIRS2Parallel	21.2569±0.5605	22.7472±1.3016	12.8325±1.5599	0.3152±0.0252
MLP	347.9439±9.1491	43.1749±0.3345	212.1962±5.3466	5.9765±0.0859
k-NN	0.0018±0.0039	0.0015±0.0036	0.0033±0.0049	0.0009±0.0004
CNN	170.7381±28.4784	225.1482±50.8137	16.4079±2.7690	68.5354±9.9256
DBB-RDNN-Rel	183.5865±28.3987	62.7598±8.6043	181.8043±38.7057	25.8860±4.9062