IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 26, 2021, accepted November 15, 2021, date of publication November 18, 2021,
date of current version December 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3129203

A Study and Evaluation of Classifiers for Anti-Spam Systems

MARCELO V. C. ARAGÃO[1], ISAAC C. FERREIRA[2], EDVARD M. OLIVEIRA[3], BRUNO T. KUEHNE[3], EDMILSON M. MOREIRA[3], AND OTÁVIO A. S. CARPINTEIRO[3]

[1]National Institute of Telecommunications, Santa Rita do Sapucaí, Minas Gerais 37540-000, Brazil
[2]TRICOD Equipamentos Eletrônicos Indústria e Comércio LTDA, Itajubá, Minas Gerais 37500-005, Brazil
[3]Research Group on Systems and Computer Engineering, Federal University of Itajubá, Itajubá, Minas Gerais 37500-903, Brazil

Corresponding author: Otávio A. S. Carpinteiro (otavio@unifei.edu.br)

This work was supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil.

ABSTRACT The volume of e-mails has been increasing in recent years. However, since 2005, at least half of these e-mails have been made up of spam. This massive traffic of unwanted messages causes losses to users, such as the excessive and unnecessary use of the bandwidth of their networks, loss of productivity, exposure of inappropriate content to inappropriate audiences etc. This paper proposes the study and the application of machine learning models to the classification of e-mails in existing anti-spam systems and, in particular, in the new anti-spam system Open-MaLBAS. After carrying out many experiments on different data sets, it was possible both to prove the feasibility of the proposal and to develop a powerful combination of techniques, methods, and models that can be successfully applied to the classification of e-mails in anti-spam systems.

INDEX TERMS Unsolicited electronic mail, machine learning, internet, network security.

I. INTRODUCTION

Nowadays, with the advent of mobile data networks, an increasing number of people have access to the internet from anywhere. This causes an increase in the number of e-mail users (from 2.5 million in 2014 to 3.8 million in 2018) and, consequently, in the number of e-mail messages (from 196 billion in 2014 to 281 billion in 2018) [1]. However, at least half of the messages since 2005 is composed of spam [2]–[4], that is, illegal and/or unsolicited messages whose recipient is not a known individual, customer or member voluntarily subscribed to a distribution list.

Spams are usually sent to a large number of people to promote products or services and often include advertisements, pyramid schemes, false donation proposals, message chains, advisory proposals, among others. This massive traffic of unwanted messages causes several negative consequences, such as excessive use of network bandwidth, wasted electricity [5], decreased user productivity, exposure of users to inappropriate or offensive content, financial/legal losses caused by fraud, among others.

Thus, it is important to classify the e-mails received in order to filter out those that are spam. The reduction of spam traffic circulating on the internet would result in the reduction of the load on telecommunications systems and in a better user experience in the use of his/her e-mail.

The e-mail classification process can be done through blacklists [6], [7]. Blacklists provide an opinion related to the legitimacy of the e-mail sender's address and/or domain.

Anti-spam systems that use blacklists to classify e-mails have several disadvantages, such as cost to consult them, cost to remove addresses and domains improperly registered, slowness to consult them, among others.

Another approach is to use the data contained in the e-mail itself (i.e., sender, recipient, header, subject, body, attachments etc) to build an automatic and intelligent classification system. To this end, machine learning (ML) models, capable of continuously learning to identify new patterns, are employed to classify e-mails in the ham[1] and spam classes, with ever greater precision.

This paper presents eight ML models for classifying e-mails. The models are evaluated in terms of the F_1 score,

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Wei Tsai.

[1] Just as illegitimate e-mails are referred to as *spam*, legitimate e-mails are referred to as *ham*.

AUC-ROC [8], training time, classification time, influence of the space dimensionality value, feature selection methods, dimensionality reduction, and computer suitability.

The eight models can be easily incorporated into the open-source anti-spam system Open-MaLBAS [9], developed by the same authors of this paper. Open-MaLBAS is available in GitHub [10].

The most relevant contributions of the paper are:

- the study and comparison of feature selection methods and the use of a dimensionality reduction method;
- the test and validation of the eight ML models on different e-mail databases;
- the comparison of the experimental results with those obtained by a current and well-known commercial anti-spam;
- the possibility of including the eight models in the classification module of the Open-MaLBAS anti-spam system.

The paper is structured as follows. Section II reviews papers on e-mail classification, covering models and methods used and results achieved. Section III discusses the pre-processing of e-mails. The feature selection methods and the dimensionality reduction method used in this paper are described, respectively, in Sections IV and V. In the Sections VI, VII and VIII, the ML models, e-mail databases and evaluation metrics used are detailed, respectively. In the Sections IX and X, the proposed methodology and the experiments performed are presented, as well as analysis and conclusions about the results. Finally, Section XI summarizes the e-mail classification problem and the proposed approach to solving it. In addition, it makes a qualitative analysis of the results and presents proposals for future work.

II. RELATED WORK

The study and development of anti-spam systems are recurrent subjects in papers in the area of computer engineering. Researchers have been approaching the topic in many different ways, proposing solutions that use from statistical techniques [11] and artificial neural networks [12] to the analysis of the reputation of senders [7] and the detection of recurrent patterns and writing styles in spams [13]. In this section, research papers involving e-mail classification are presented.

Drucker *et al.* [14] compared a Support Vector Machine (SVM) with three other models — boosted decision trees, Repeated Incremental Pruning to Produce Error Reduction (RIPPER)[2] and Rocchio[3] — for e-mail classification. The experiments were conducted on two e-mail databases from the American telecommunications company (AT&T). The e-mails were processed by several feature selection methods and empty words[4] were removed. In addition to the good results (precision around 98%), the authors highlighted other important results obtained by SVM, such as better training times when using binary representation to represent the features of e-mails as well as its ability to handle well a large number of e-mail features.

Using the Naïve Bayes (NB) probabilistic model proposed by Sahami *et al.* [15] combined with linguistic lemmatisation techniques[5] and removal of empty words, Androutsopoulos *et al.* [11] obtained classification accuracy near to 99% in most cases on the e-mails of the Ling Spam database [16], with a relatively low computational cost. In the paper, the authors mention that the use of linguistic techniques significantly improved the accuracy in classification, as well as introduced new possibilities in the analysis of e-mails.

Meyer and Whateley [17] proposed a statistical model called chi-squared (χ^2) combining for e-mail classification. The model performs two χ^2 tests to determine the probability that a given e-mail be spam and ham, respectively. These probabilities are combined and scaled to provide, for each e-mail, an overall spam score in the range 0 and 1. Five e-mail databases — four SpamBayes [18] and Spam Assassin [19] — were processed with *n*-gram and tiling-based feature selection methods, and used to train the model. The number of e-mails requiring manual classification after training was just over 1%, proving the efficacy of the model.

Carpinteiro *et al.* [12] proposed a pre-processing of e-mails to simplify them. Different feature selection methods were applied to the subject and body of the e-mails. A Multilayer Perceptron (MLP) neural network was used as a classifier model. The experiments showed classification accuracy over 99% on the Spam Assassin database [19].

Zhang *et al.* [7] proposed a reputation-based e-mail classification system — IPGroupRep. The reputation is provided by a server that stores Internet Protocol (IP) addresses and their respective scores. Scores are made up of sending histories of e-mail users, data from other anti-spam systems and from recipients. The authors used a database composed of almost three million e-mails from a university e-mail server. The experiments showed that the proposed system performed as well as the existing technique Distributed Checksum Clearinghouses (DCC) [20], and outperformed others such as Gossip Optimization for Selective Spam Prevention (GOSSiP) [21] and RepuScore [22], reaching rates above 95% of accuracy, precision and recall.

Pérez-Díaz *et al.* [23] used the rough set model [24] for e-mail classification. They also used the MFD (Most Frequent Decision), LNO (Largest Number of Objects), and LTS (Largest Total Strength) heuristics to make decisions about indeterminate e-mail classifications. The authors used binary or frequency representations to represent the features of the

[2] Classification model based on rules induced from a training set. It is known for handling unbalanced and noisy data sets well.

[3] Classification model used in information retrieval systems. It makes use of the relevance (or not), assigned by users of the system, to documents.

[4] Empty words are words that are filtered before or after natural language processing. They usually refer to the most common words in a language.

[5] Lemmatisation is the representation of words in their basic form, which disregards their inflected forms.

e-mails from Spam Assassin database [19]. They compared the classification results obtained with those of the AdaBoost, Flexible Bayes (FB), Naïve Bayes and SVM models. The rough set model obtained F_1 score rates of 98%, surpassing those of the other models. The authors highlighted the importance of periodically regenerating the rule set of the model. In addition, they highlighted the long training time of the model and suggested the adoption of methods to reduce the e-mail feature space.

Barigou *et al.* [25] proposed a Cellular Automaton combined with K-Nearest Neighbors algorithm (CA-KNN) in order to reduce the amount of similar e-mails selected during the e-mail classification process. The use of CA-KNN produced a reduction in memory usage and an increase in performance of the model when compared to traditional KNN. The model showed an accuracy of more than 98% on the e-mails from the Ling Spam database [16], outperforming existing models such as NB [11], [26], stacked classifiers [27], open-source filters [28], and Topic-based Vector Space Model (eTVSM) [29].

Kaya and Ertuğrul [30] proposed a new feature selection method — shifted one-dimensional local binary pattern (shifted-1D-LBP). In experiments, the method was applied to the e-mails of the Ling Spam [16], Spam Assassin [19] and TREC 2006 [31] databases. Then, six ML models — Fisher Linear Discriminant Analysis (FLDA), NB, BayesNet (BN), Functional Tree (FT), Random Tree (RT), and Random Forest (RF) — were trained on the e-mails of the databases. The performance of the models in e-mail classification was evaluated under several metrics, such as precision, recall and F_1 score. The results were promising, reaching approximately 92%, 93% and 95% on the Ling Spam, Spam Assassin and TREC databases, respectively.

Shams and Mercer [13] proposed a new method that consists in using stylometry attributes[6] to train e-mail classifying models. Examples of these attributes include the number of spelling and grammatical errors, indicators of ease of reading — Gunning fog index, Simple Measure of Gobbledygook (SMOG), Flesch Reading Ease Score (FRES), Forcast, Flesch-Kincaid readability —, quantities of simple words (with up to two syllables) and complex (with three or more syllables), and average size of e-mail and words. The method was tested with the NB, RF, SVM, Bagging, and Adaboost.M1 classifying models on the CSDMC2010 [32], Spam Assassin [19], Ling Spam [16], and Enron-Spam [33] e-mail databases. The Bagging and Adaboost.M1 models achieved the best results. The average classification accuracies ranged from approximately 92% to 95%. In addition, the authors concluded that the method is relevant in detecting spam on personalized e-mail databases (i.e., in those in which the collection of e-mails is not random), but limited on non-personalized e-mail databases, owing to the multiplicity of e-mail writing patterns on these databases.

Yang *et al.* [34] proposed the Anti-Spam Filter algorithm based on One-Class Information Bottleneck (SFOC-IB) model. SFOC-IB is suitable for training with small training sets. According to the authors, the frequent change of content in e-mails reduces the availability of large training sets with up-to-date content. SFOC-IB extracts highly significant samples from training sets in order to build clusters. The clusters are used to classify the e-mails, in the ham and spam classes, through a similarity function — Jensen-Shannon divergence [35]. The SFOC-IB, SVM, NB, and AdaBoost models were evaluated on the e-mails from the Ling Spam [16], Spambase [36], PU3 [37] and TREC 2007 [38] databases. SFOC-IB presented accuracy, recall and F_1 score results comparable to those presented by the AdaBoost, NB, and SVM models, when trained with large training sets. With small training sets, however, SFOC-IB had less deterioration in its performance.

Tyagi [39] proposed the Stacked Denoising Autoencoder (SDAE) model, based on a deep neural network [40], for e-mail classification. She used the Term Frequency-Inverse Document Frequency (TF-IDF) feature selection method to select the most relevant features of the e-mails from the PU1, PU2, PU3, PUA [37] and Enron-Spam [33] databases. SDAE was compared to three other models — Deep Belief Network (DBN), Dense Multi-Layer Perceptron (Dense-MLP), and SVM. It outperformed the other three models, achieving accuracy, precision, recall and F_1 score around 95%.

Kumaresan *et al.* [41] proposed a Hybrid-Kernel Support Vector Machine (HKSVM) model[7] for e-mail classification. They used two combined databases — Ling Spam [16] and Spam Archive [42], [43] — to evaluate the model. The combined databases are composed of e-mails containing text and images. The textual characteristics of the e-mails were selected by the Term Frequency (TF)[8] method, and the visual ones, by the correlogram[9] and wavelet moment[10] methods. After selection, the feature space was reduced by a modified version of the Cuckoo search algorithm [44], with heuristics given by Lévy flights[11] [45]. Finally, the proposed model was experimentally compared to SVM models. It achieved classification accuracy rate above 97%, surpassing the 94% obtained by other SVM models.

Douzi *et al.* [46] proposed a new e-mail representation, based on the Paragraph Vector-Distributed Memory (PV-DM) and TF-IDF methods. In this representation, both contextual characteristics (i.e., present in several other e-mails) and specific characteristics (i.e., present only in a particular e-mail) of each e-mail are considered. The authors used a

[6] Stylometry is the study of language styles that aims to attribute authorship to anonymous or contested documents.

[7] Model that uses a combination of two or more kernels, such as linear, polynomial and quadratic kernels.

[8] Denotes the number of occurrences of a term (e.g., a word) in a particular document (e.g., an e-mail).

[9] Graph that presents autocorrelations in a time series.

[10] Technique used to measure the local regularity of a signal.

[11] Random step succession whose lengths follow a probability distribution of heavy tail (e.g., Pareto, Lévy, Cauchy, Burr, and Student's t distributions).

double representation — the one they proposed and the traditional Bag-of-Words (BoW) [47] — for each e-mail from the Ling Spam [16] and Enron-Spam [33] databases. The Logistic Regression, KNN, and SVM models were trained and evaluated on the e-mails with double representation. They presented F_1 scores ranging from approximately 92% to 98% in the classification of e-mails from the two databases.

As summarized in Table 1, the reviewed papers proposed several approaches and models for e-mail classification. The models were evaluated on several e-mail databases and obtained an accuracy greater than 90% in the classification. Some papers made use of feature selection methods, such as those presented in Section IV. However, none of the papers addressed the dimensionality reduction of the feature space of e-mails, as presented in Section V. Thus, the proposal presented in this paper differs from those presented in the reviewed papers.

III. PRE-PROCESSING

The pre-processing of the body and subject of e-mails aims to increase the accuracy of the ML models in their classifications. In this paper, two types of filters — plain text and HTML (HyperText Markup Language) — were used.

A. PLAIN TEXT FILTER

The plain text filter has two functions. First, it makes the body text and subject text of e-mails uniform. For example, it converts uppercase characters into lowercase and removes accents from words. Second, it replaces parts of the text with special tags. For example, numbers are replaced by the `!_NUMBER` tag, values with currency symbols are replaced by the `!_MONETARY` tag, and words less than 4 or more than 19 characters are replaced by the `!_SMALL_WORD` and `!_BIG_WORD` tags, respectively.

B. HTML FILTER

The HTML filter, as expected, processes the HTML tags of the body and subject of e-mails. The processing performed by the filter is done at three levels, according to the relevance of the information contained in the tag and/or its attributes. The three levels of processing are described below:

- Tags with information typically focused on document description are entirely removed. For example, the tag "`<title>Lipsum</title>`" is entirely removed;
- Tags partially relevant to the classification of e-mails have their attributes removed and are replaced by a corresponding special tag. For example, the tag "`<p id="par">text</p>`" has its attribute "`id`" removed and is replaced by the special tag "`!_IN_P text`";
- Tags totally relevant to the classification of e-mails have only the parameters of their attributes removed and are replaced by a corresponding special tag. For example, the tag "`<form action ="script.php">contents</form>`" has the parameter "`script.php`" of its attribute "`action`" removed and is replaced by the special tag "`!_IN_FORM action contents`".

After pre-processing, each e-mail is therefore represented by a set of tokens. Each token is either a word or a special tag of the e-mail body or subject.

IV. FEATURE SELECTION

It is necessary to represent the pre-processed e-mails so that they can be classified by the ML models. This representation, which consists in a set of features (i.e., tokens — words and special tags) of the e-mail, can be simplified by using feature selection methods [48].

Three feature selection methods — chi-square statistics, frequency distribution, and mutual information — were used in this paper. They were chosen for four main reasons. First, because they are very well known. The literature contains many papers that report their use. Practically, if not everyone who works in the ML field knows them. Second, they are easy to implement. Third, they have a low computational cost, which is advantageous when using them in anti-spams. Finally, in the experiments carried out, they presented very good results. The three methods are described next.

A. CHI-SQUARE STATISTICS (CHI2)

The χ^2 statistics (CHI2) measures the dependency between a feature t and a class c, in particular. It is defined by the Equation (1),

$$CHI2(t, c) = \frac{n \cdot F(t)^2 \cdot (p_c(t) - P_c)^2}{F(t) \cdot (1 - F(t)) \cdot P_c \cdot (1 - P_c)} \tag{1}$$

in which:

- n is the total number of e-mails in the set;
- $p_c(t)$ is the conditional probability c for e-mails that contain the feature t;
- P_c is the global fraction of e-mails that contain the class c;
- $F(t)$ is the global fraction of e-mails that contain the feature t.

B. FREQUENCY DISTRIBUTION (FD)

Frequency Distribution (FD) is a statistical method that measures the frequency with which a feature t occurs in a class c, in particular. It is defined by the Equation (2),

$$FD(t, c) = \frac{C(t, c)}{\sum_{t_i \in T(c)} C(t_i, c)} \tag{2}$$

in which:

- the numerator is the number of occurrences of t in e-mails of the class c;
- the denominator is the sum of the number of occurrences of all features in e-mails of the class c.

C. MUTUAL INFORMATION (MI)

Mutual Information (MI) is derived from the information theory [49]. It consists in the amount of information that a

TABLE 1. **Summary of the reviewed papers.**

Paper	Proposed Model(s)	Compared Model(s)	Used Database(s)	Feature Selection Method(s) / Representation	Accuracy
Drucker et al. [14]	SVM	Boosted C4.5 Trees, RIPPER, and Rocchio	Two e-mail databases from the American telecommunications company (AT&T)	TF, TD-IDF, and binary representation (with or without stop words removal and lemmatisation)	≈98%
Androutsopoulos et al. [11]	NB	NB	Ling Spam	MI (with or without stop words removal and lemmatisation)	≈99%
Meyer and Whateley [17]	χ^2 combining	NB	SpamBayes and SpamAssassin	Unigrams, bigrams, unigrams/bigrams and tiling scores	≈99%
Carpinteiro et al. [12]	MLP	N/A	SpamAssassin	FD and CHI2	≈99%
Zhang et al. [7]	IPGroupRep	GOSSiP, DCC, and RepuScore	Database collected from an university e-mail server	Sender IP address	≈96%
Pérez-Díaz et al. [23]	Rough sets	AdaBoost, FB, NB, and SVM	SpamAssassin	TF and binary represen-tatiton	≈98%
Barigou et al. [25]	CA-KNN	NB, stacked classifiers, open-source filters, and eTVSM	Ling Spam	Binary representation (with stemming and stop words removal)	≈98%
Kaya and Ertuğrul [30]	FLDA, NB, BN, FT, RT, and RF	N/A	Ling Spam, Spam Assassin, and TREC 2006	Shifted-1D-LBP	≈95%
Shams and Mercer [13]	NB, RF, SVM, Bagging, and Adaboost	N/A	CSDMC2010, Spam Assassin, Ling Spam, and Enron-Spam	Stylometry attributes (SMOG, FRES, Forcast etc.)	≈95%
Yang et al. [34]	SFOC-IB	AdaBoost, NB, and SVM	Ling Spam, Spambase, PU3, and TREC 2007	Jensen-Shannon divergence to centroids that represent spam and ham, respectively	≈95%
Tyagi [39]	SDAE	DBN, Dense-MLP, and SVM	PU1, PU2, PU3, PUA and Enron-Spam	TF-IDF	≈96%
Kumaresan et al. [41]	HKSVM	Linear SVM, quadratic SVM and polynomial SVM	Ling Spam and Spam Archive	TF optimized with S-Cuckoo search (text) and correlogram (images)	≈97%
Douzi et al. [46]	Logistic regression, KNN and SVM	Logistic regression, KNN and SVM	Ling Spam and Enron-Spam	PV-DM, TF-IDF and BoW	≈98%

feature aggregates in relation to a given class. The mutual information $M(t, c)$, between the feature t and the class c, is based on the level of co-occurrence between the class c and the feature t. It is defined by the Equation (3),

$$MI(t, c) = \log\left(\frac{F(t) \cdot p_c(t)}{F(t) \cdot P_c}\right) = \log\left(\frac{p_c(t)}{P_c}\right) \tag{3}$$

in which:

- $F(t) \cdot P_c$ is the expected co-occurrence between the class c and the feature t, based on mutual independence;
- $F(t) \cdot p_c(t)$ is the actual co-occurrence (in practice, this value can be much higher or lower than expected, depending on the level of correlation between the class c and the feature t).

Clearly, the feature t is positively correlated to the class c when $MI(t, c) > 0$, and negatively correlated when $MI(t, c) < 0$.

V. DIMENSIONALITY REDUCTION

The e-mail classification problem usually presents feature spaces with high dimensionality, even after the most relevant features have been selected through feature selection methods. The high dimensionality of the feature spaces can make it difficult to train certain ML models.

Thus, it is necessary to reduce the dimensionality of the feature space, so that the training algorithms of the models do not spend impractical amounts of time and computational resources. In addition, it is desired that the reduction of the dimensionality of the original feature space to another space preserves the greatest possible amount of relevant information, so as not to impair the generalization and classification capabilities of the models.

Several methods have been proposed to explore the feature space and find a relevant subset of features according to some evaluation metric. In this paper, the method employed — Multi-Objective Evolutionary Feature Selection (MOEFS) [50] — consists in a multi-objective evolutionary search that explores the feature space in order to generate the candidate subsets while two objectives are optimized simultaneously:

- Maximization of the ''merit'' metric, given in terms of the correlation between features and classes of the problem and the intercorrelation between the features of the candidate subsets [51];

- Minimization of the cardinality (i.e., number of features) of the candidate subsets.

Class balancing is carried out after the dimensionality reduction. Its purpose is to ensure that there is the same amount of samples of ham and spam e-mails in the set, as the imbalance can cause the ML model to emphasize one class more than another, obtaining biased results.

VI. MACHINE LEARNING MODELS

Eight ML models — Naïve Bayes (NB), Averaged One-Dependence Estimators (AODE), Single Layer Perceptron (SLP), Radial Basis Function Network (RBF), Reduced-Error Pruning Tree (REPT), Boosted Reduced-Error Pruning Tree (AB-M1), Linear Support Vector Machine (L-SVM) and Non-Linear Support Vector Machine (NL-SVM) — were used in the experiments. The eight models are made available by the open-source library Weka [52].

NB and AODE are probabilistic models. Probabilistic models are models capable of predicting, given an input pattern, a probability distribution across a set of classes, rather than just predicting the most likely class to which the pattern belongs.

SLP and RBF are artificial neural models. Artificial neural models are models inspired by the biological neural system.

REPT and AB-M1 are models of decision trees commonly used in operational research to identify strategies most likely to achieve an objective.

Finally, L-SVM and NL-SVM are support vector machines. Support vector machines are models that define a mathematically optimal separability boundary between classes.

A. Naïve BAYES

Naïve Bayes (NB) is a model known both for its theoretical simplicity and ease of implementation, and for its effectiveness. The model learns the probability that any object (e.g., e-mail) with certain features belongs to a certain class or category. In addition, it is called *Bayesian* because it is based on Bayes' theorem and *naïve* because it supposes that the occurrence of a particular feature is independent of the occurrence and/or influence of the other features. The classifier model consists in the function that, among the existing classes $\{C_1, \ldots, C_f, \ldots, C_K\}$, assigns to the object (e.g., e-mail) the class $\hat{C} = C_f$, for some f, as described in Equation (4).

$$\hat{C} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k). \tag{4}$$

The model in Figure 1 illustrates the NB model. In it, it is noted that the probabilities of class $P(c)$ depend directly on the features $x_1, x_2, \ldots, x_n$, and no probability between them is considered.

B. AVERAGED ONE-DEPENDENCE ESTIMATORS

Averaged One-Dependence Estimators (AODE) is an extension of NB that introduces the notion of x-dependencies estimators, whereby the probability of the value of each feature is conditioned by the class and a predefined amount of other features. In this paper, the AODE with the value of $x = 1$ was used, which makes it a classifier model "less naïve" than the NB [54].

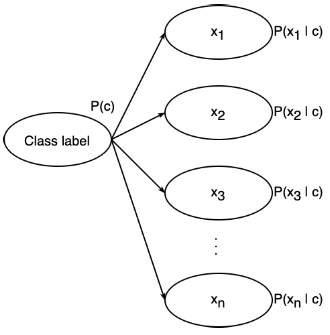


FIGURE 1. NB model (based on Murakami and Mizuguchi [53]).

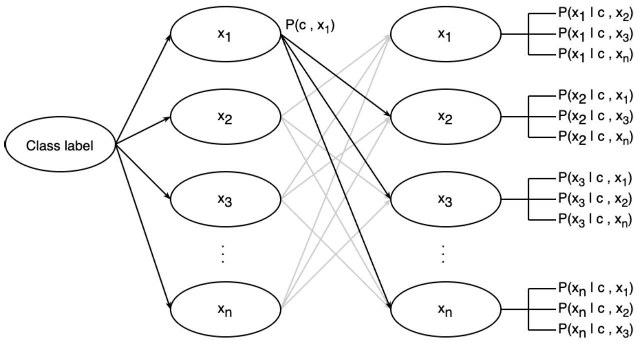


FIGURE 2. AODE model (based on Murakami and Mizuguchi [53]).

Figure 2 illustrates the AODE model. In it, $P(c)$ is conditioned to the features that, in turn, take into account the joint probabilities in relation to a single other feature, thus characterizing a 1-dependency estimator.

C. SINGLE LAYER PERCEPTRON

Single Layer Perceptron (SLP) is a model that consists in a single artificial neuron with all its inputs connected directly to its outputs (Figure 3) [55]. If the linear combination of its inputs exceeds a predetermined threshold, it will produce an excitatory potential at its output. Thus, if an output is produced, the SLP classifies the e-mail, for example, as being spam. Otherwise, it is classified as ham.

D. RADIAL BASIS FUNCTION NETWORK

Radial Basis Function Network (RBF) is a model that consists in a network of artificial neurons. The activations of the RBE artificial neurons start at the neurons of the input layer, then run through the neurons with Gaussian activation functions $g_1^{(1)}, g_2^{(1)}, \ldots, g_{n_1}^{(1)}$ of the single hidden layer and, finally, reach the neurons with linear activation functions $g_1^{(2)}, \ldots, g_{n_2}^{(2)}$ of the output layer [57]. Figure 4 illustrates the architecture of the RBF.

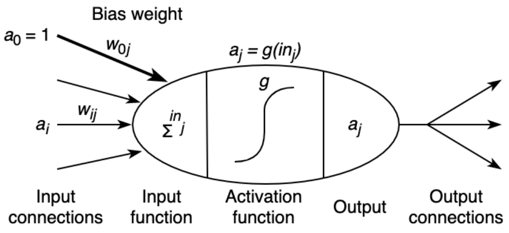


FIGURE 3. Simple mathematical model of a neuron (based on Russell and Norvig [56]).

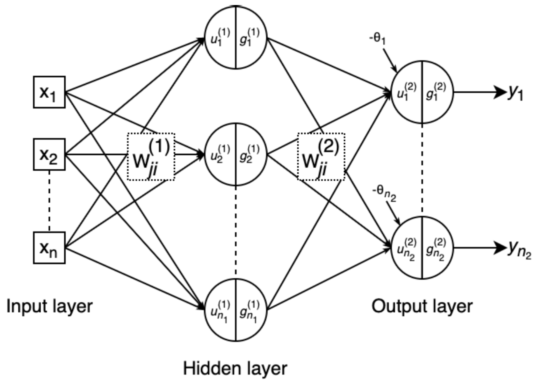


FIGURE 4. Radial Basis Function Network (based on Silva *et al.* [58]).

E. REDUCED-ERROR PRUNING TREE

Reduced-Error Pruning Tree (REPT) is a model that consists in a decision tree whose simplification process (i.e., the process of removing subtrees that make its structure unnecessarily complex and that reduce its ability to generalize) is based on the Reduced-Error Pruning method, proposed by Quinlan [59].

F. BOOSTED REDUCED-ERROR PRUNING TREE

Boosted Reduced-Error Pruning Tree (AB-M1) is a model that consists in a decision tree that makes use of a boosting method to improve its performance. Generically, this method repeatedly runs a "weak" learning algorithm (i.e., which produces slightly better responses than random guesses) over different distributions of training data [60]. It then combines the classifiers $h_1, \ldots, h_n$ produced by the algorithm with their respective weights $\alpha_1, \ldots, \alpha_n$ to form a single composite classifier H.

AdaBoost [61] is a boosting method that has two versions [62] — AdaBoost.M1, focused on binary classification problems, and AdaBoost.M2, focused on multiclass classification. The first version, illustrated in Figure 5, was used in this paper. The result of executing this method is a binary value, built from the combination of the outputs of the models taken into account. This binary value indicates the class resulting from the classification. The REPT model, described in the previous subsection, was used as a "weak" classifier.

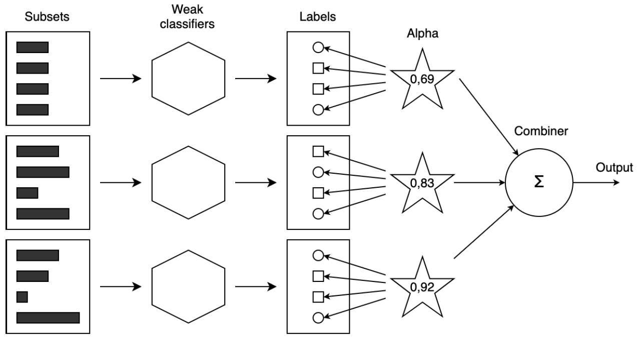


FIGURE 5. AdaBoost.M1 operating process (based on Harrington [63]).

TABLE 2. Types of kernel functions.

Type	Function
Linear	$K(\vec{x}, \vec{x}'_i) = \vec{x}^T \vec{x}_i$
Polynomial	$K(\vec{x}, \vec{x}'_i) = (\vec{x}^T \vec{x}_i)^d$
Gaussian	$K(\vec{x}, \vec{x}'_i) = \exp\left(-\frac{1}{2\sigma^2}\|\|\vec{x} - \vec{x}_i\|\|^2\right)$
Sigmoid	$K(\vec{x}, \vec{x}'_i) = \tanh(\eta \vec{x} \vec{x}_i + \theta)$

Thus, by comparing the results of the REPT model — REPT tree, without boosting — with those of the AB-M1 model — REPT tree, with AdaBoost.M1 boosting — it was possible to assess whether or not the method AdaBoost.M1 improved the classification accuracy of e-mails.

G. LINEAR SUPPORT VECTOR MACHINE

Linear Support Vector Machine (L-SVM) is a linear model that searches, in the original feature space, for a maximum margin hyperplane (i.e., a hyperplane capable of separating samples of different classes with the greatest possible distance) [64].

H. NON-LINEAR SUPPORT VECTOR MACHINE

Non-Linear Support Vector Machine (NL-SVM) is a non-linear model proposed by Boser *et al.* [65]. Its approach consists in applying a kernel function [66] to the maximum margin hyperplanes, transforming the input space into a linearly separable feature space of equal or greater dimensionality. Table 2 presents the most used types of kernel functions. In the experiments of this paper, the NL-SVM used a polynomial kernel function (P-SVM).

VII. E-MAIL DATABASES

The experiments used three public databases — Ling Spam, Spam Assassin, TREC — and two private ones — UNIFEI, UNIFEI-$\delta 0$ —, all composed by real e-mails. The databases are described below.

A. LING SPAM

The Ling Spam database [16] was compiled and made available in the public domain by Androutsopoulos *et al.* [11]. The database e-mails were obtained from several sources and pre-processed, in order to remove some information (e.g., attachments and HTML tags) deemed irrelevant or private.

The database has four versions — bare, lemm, lemm_stop, stop. Bare is the most original version. In the lemm version, the words in the e-mails are lemmatized. In the lemm_stop version, in addition to lemmatizing the words, the empty words were also removed. In the stop version, only the empty words have been removed. Each version has 2,893 e-mails, containing 2,412 (83.4%) hams and 481 (16.6%) spams. In the experiments, the four versions were joined, forming a single database composed of 11,572 e-mails.

The database was used in several papers, such as, by Barigou *et al.* [25], Kaya and Ertuğrul [30], Shams and Mercer [13], Yang *et al.* [34] and Kumaresan *et al.* [41].

B. SPAM ASSASSIN

The Spam Assassin database [19] is comprised of e-mails from various sources. It is divided into five parts — spam, spam_2, easy_ham, easy_ham_2, hard_ham. The spam part contains 500 spam e-mails. The spam_2 part consists of a new addition of 1,397 spam e-mails to the database. The easy_ham and easy_ham_2 parts contain e-mails that are ham and are easily identified as ham. The easy_ham and easy_ham_2 parts contain 2,500 and 1,400 e-mails, respectively. The hard_ham part contains 250 e-mails that are ham, but which are hardly identified as ham. In the experiments, the five parts were joined, forming a single database composed of 6,047 e-mails, of which 4,150 (68.6%) are ham and 1,897 (31.4%) are spam.

The database was used in the development of the Apache Spam Assassin anti-spam [67]. It was also used in several papers, such as those by Meyer and Whateley [17], Carpinteiro *et al.* [12], Sirisanyalak and Sornil [68], Pérez-Díaz *et al.* [23], Barigou *et al.* [25], Kaya and Ertuğrul [30] and Shams and Mercer [13].

C. TREC

The TREC Spam Track database [69], hereafter referred to as TREC, is composed of e-mails from various sources. It is divided into three parts — TREC 2005, TREC 2006 and TREC 2007.

The TREC 2005 part [70] is composed of e-mails related to 150 executives of the Enron company. The e-mails were collected and made publicly available as a result of the US federal investigation into the company's collapse. Other public domain spam e-mails were later added to TREC 2005.

The TREC 2006 part [31] includes two private and two public parts — TREC06p, TREC06c. The TREC06p and TREC06c parts contain e-mails with content in English and Chinese, respectively, collected from the Internet.

The TREC 2007 part [38] includes a private part and a public part — TREC07p. The TREC07p part consists of e-mails received, over a period of three months, by a private e-mail server.

In the experiments, the parts TREC 2005, TREC06p and TREC07p were joined, forming a single database composed of 205,430 e-mails, of which 77,529 (37.7%) are ham and 127,901 (62.3%) are spam.

D. UNIFEI

The UNIFEI database is composed of e-mails collected, during the second semester of 2016, by the Research Group in Systems and Computer Engineering (GPESC) of the Federal University of Itajubá (UNIFEI). The database represents the reality of the university, containing e-mails received by professors, technical and administrative personnel, and students. The university imposed conditions for the collection of e-mails in order to preserve the confidentiality of the information contained therein.

The database contains 862,229 e-mails, of which 353,151 (41%) are ham and 509,076 (59%) are spam. The classification, in the ham and spam classes, of the database e-mails was performed by the commercial anti-spam CanIt-PRO 9.2.4 [71].

CanIt-PRO remained in use at the Federal University of Itajubá (UNIFEI), Brazil, until July-2019. From August-2019 on, the university network services, including e-mail service, have been providing through the Google G-Suite platform.

The classification carried out by CanIt-PRO was the subject of suspicion. There was a strong suspicion that identical e-mails could be classified into different classes. Thus, to check whether or not the classification of e-mails was consistent, five steps were taken:

1) Each e-mail was represented, through the pre-processing described in Section III, by a set of tokens;
2) The 1,024 most relevant tokens for the classification of the e-mails were selected using the FD feature selection method (Section IV). Then, each e-mail was represented as a multidimensional vector in $\Re^{1024}$, in which each dimension represents a token selected by the FD method;
3) Each group of identical vectors was placed in a separate set;
4) It was verified if CanIt-PRO had assigned the same class to all vectors in each set, for they were all identical. Whenever this was not the case, a brand-new ham/spam class was assigned to all vectors in the set;
5) The original 1024-dimensional ($\Re^{1024}$) vectors were exhibited in two dimensions ($\Re^2$), by means of the t-SNE method [72].

Figure 6 displays the e-mails of UNIFEI database in the bidimensional space. In the figure, ham e-mails are shown as blue "$+$" marks, spam e-mails as red "$\times$" marks, and ham/spam e-mails as black "$\star$" marks. It should be noted that the axes x and y of the figure have no meaning, as the t-SNE method takes into account only the distances between the points $x \in \Re^{1024}$ of the database and the probability distributions between these distances [73].

Because e-mails were represented by vectors in $\Re^{1024}$, it is very likely that equal vectors represent equal e-mails. Hence, since the amount of black marks in Figure 6 is substantial, it follows that the UNIFEI database is highly inconsistent.

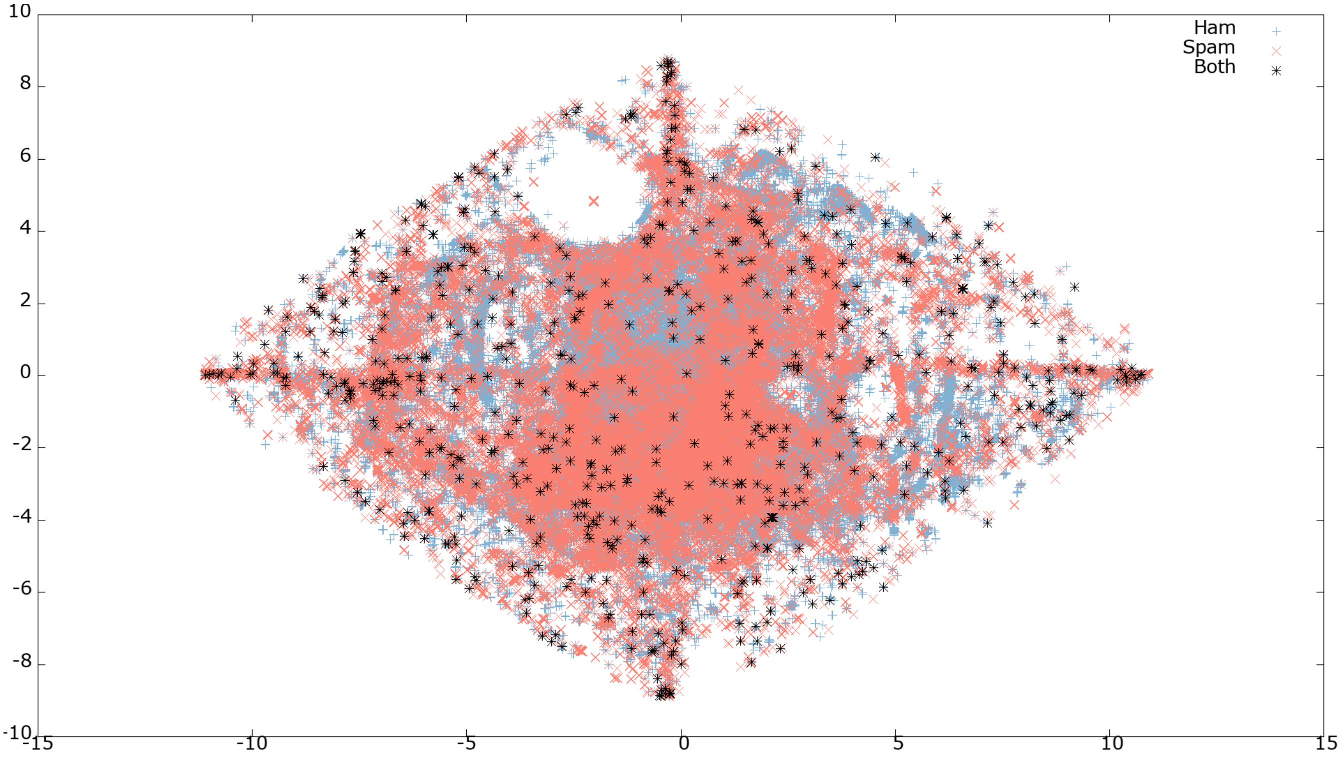


FIGURE 6. E-mails of the UNIFEI database — ham: blue; spam: red; ham/spam: black.

E. UNIFEI-δ0

The consistency-generating tool of the Open-MaLBAS anti-spam [9] was employed to correct the inconsistency of the UNIFEI database. The tool utilizes two integer parameters — $\delta \in \mathbb{N}$ and $n \in \mathbb{N}^*$ — that are defined beforehand. The first parameter δ denotes both the level of divergence among e-mails and among vectors, as the e-mails are represented by tokens (Section III) which, in turn, are represented by vector coordinates. For example, if δ is set to be zero, this indicates either that only e-mails with identical tokens or that only vectors with identical values in their coordinates are deemed to be identical. If δ is set to be one or two, this indicates either that only e-mails that vary at most by one or two tokens or that only vectors that vary at most by one or two values of their coordinates, respectively, are deemed to be identical. The second parameter n denotes the number of dimensions of the vectors.

The consistency-generating tool performs the four steps below:

1) It checks, by examining their tokens, which are the e-mails identical to each other by the level of divergence δ (from now on, δ-divergence e-mails) and places each group of δ-divergence e-mails in a separate set;
2) It identifies the predominant class (i.e., with the highest amount of e-mails) of each set, and then attributes the label of that class to every e-mail in the set. For example, if a set of δ-divergence e-mails comprises 35 ham e-mails and 65 spam e-mails, the spam label is attributed to all 100 e-mails in the set;
3) It changes the representation of the e-mails. It changes their representation from tokens to n-dimensional vectors;
4) It executes again the first and second steps, but now, on the vectors produced in the third step. It is worth noticing that this fourth step may change the class of the vectors, i.e., e-mails may change indirectly their classes again.

The UNIFEI-δ0 database was derived from the UNIFEI database, by running the consistency-generating tool with the value of the parameter $\delta = 0$. From Figure 7, it is possible to verify that the consistency of UNIFEI-δ0 database is higher than that of UNIFEI database. The UNIFEI-δ0 database contains 862,227 e-mails, of which 353,910 (41%) are ham and 508,317 (59%) are spam.

VIII. PERFORMANCE METRICS

Classifying models of anti-spam systems should avoid false positives and false negatives. A false positive is a ham e-mail incorrectly classified as spam. In turn, a false negative is a spam e-mail incorrectly classified as ham.

The precision and recall metrics measure the percentage of absence of false positives and false negatives, respectively. The F_1 score combines the metrics precision and recall, in order to assess the accuracy of a classifying model in terms of the amount of false positives and false negatives it

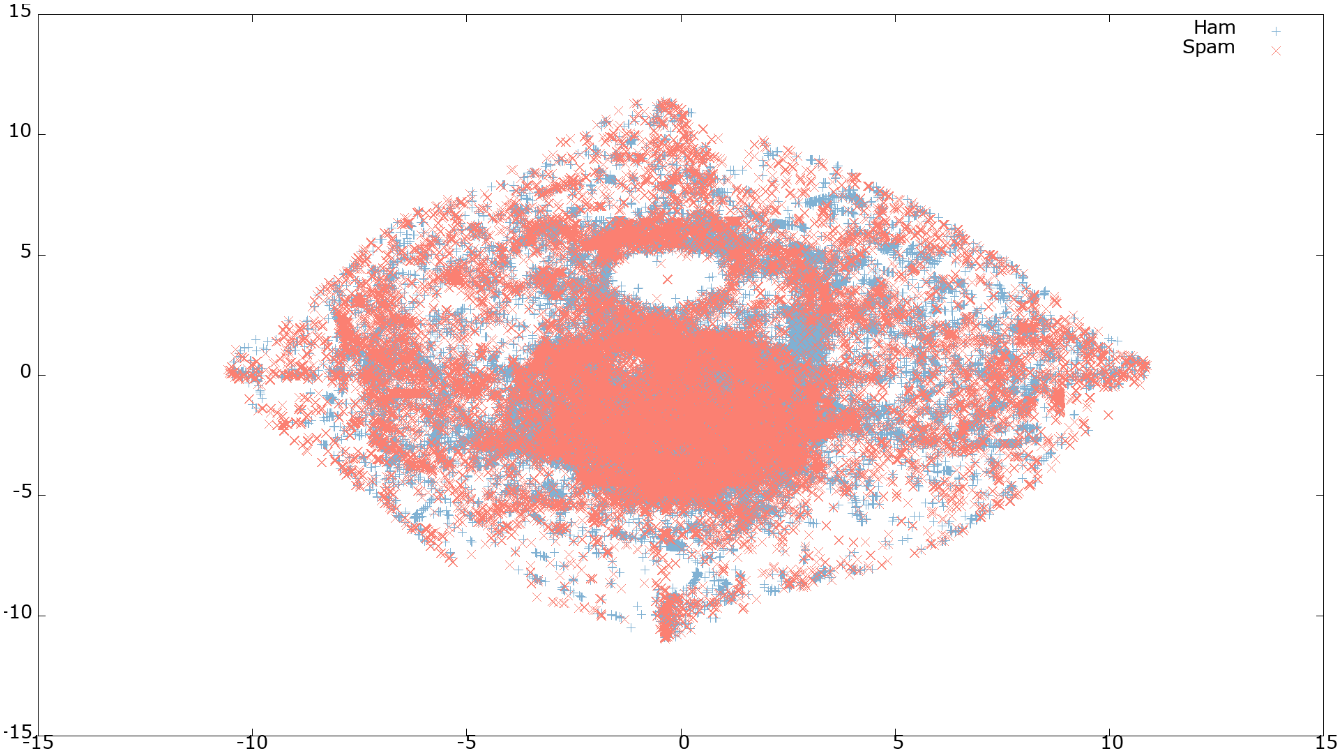


FIGURE 7. E-mails of the UNIFEI-δ0 database — ham: blue; spam: red.

produces. The area under Receiver Operating Characteristic curve (AUC-ROC) is also a concise way to assess the classification ability of binary classifiers.

Two other metrics, also used to evaluate the ML models, are metrics related to training and classification times. The first metric measures the time required to train the model on the training set. The second, measures the time the model spends to classify all e-mails in the test set.

To describe the metrics, in terms of their equations, the following variables are needed:

- N_{HAM}: total number of ham e-mails in the test set;
- N_{SPAM}: total number of spam e-mails in the test set;
- $n_{H \to H}$: number of ham e-mails correctly classified as ham;
- $n_{H \to S}$: number of ham e-mails incorrectly classified as spam;
- $n_{S \to S}$: number of spam e-mails correctly classified as spam;
- $n_{S \to H}$: number of spam e-mails incorrectly classified as ham.

A. PRECISION

The precision metric is calculated both to indicate the accuracy in the classification of ham e-mails (Equation (5)) and spam e-mails (Equation (6)). The general[12] precision is given by Equation (7).

$$P_{HAM} = \frac{n_{H \to H}}{n_{H \to H} + n_{S \to H}} \tag{5}$$

$$P_{SPAM} = \frac{n_{S \to S}}{n_{S \to S} + n_{H \to S}} \tag{6}$$

$$P_{GEN} = \frac{N_{HAM} * P_{HAM} + N_{SPAM} * P_{SPAM}}{N_{HAM} + N_{SPAM}} \tag{7}$$

B. RECALL

The recall metric is also calculated both to indicate the recall in the classification of ham e-mails (Equation (8)) and spam e-mails (Equation (9)). The general recall is given by Equation (10).

$$R_{HAM} = \frac{n_{H \to H}}{n_{H \to H} + n_{H \to S}} \tag{8}$$

$$R_{SPAM} = \frac{n_{S \to S}}{n_{S \to S} + n_{S \to H}} \tag{9}$$

$$R_{GEN} = \frac{N_{HAM} * R_{HAM} + N_{SPAM} * R_{SPAM}}{N_{HAM} + N_{SPAM}} \tag{10}$$

C. F_1 SCORE

The F_1 score is given by the harmonic mean between the values obtained by the metrics precision and recall. Thus, its best and worst values are 1 and 0, respectively. The F_1 score is calculated both to indicate the score in the classification of ham e-mails (Equation (11)) and spam e-mails

[12] General is abbreviated as *GEN* in the Equations 7, 10, 13, and 16.

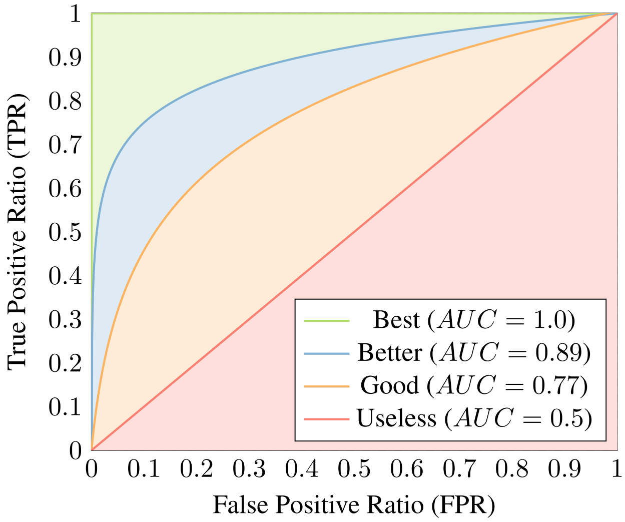


FIGURE 8. Area under different ROC curves, from useless to ideal classifiers (based on Tourassi [74]).

(Equation (12)). The general score is given by Equation (13).

$$F_{1,HAM} = 2 \cdot \frac{1}{\frac{1}{P_{HAM}} + \frac{1}{R_{HAM}}} = 2 \cdot \frac{P_{HAM} \cdot R_{HAM}}{P_{HAM} + R_{HAM}} \tag{11}$$

$$F_{1,SPAM} = 2 \cdot \frac{1}{\frac{1}{P_{SPAM}} + \frac{1}{R_{SPAM}}} = 2 \cdot \frac{P_{SPAM} \cdot R_{SPAM}}{P_{SPAM} + R_{SPAM}} \tag{12}$$

$$F_{1,GEN} = \frac{N_{HAM} * F_{1,HAM} + N_{SPAM} * F_{1,SPAM}}{N_{HAM} + N_{SPAM}} \tag{13}$$

D. AREA UNDER RECEIVER OPERATING CHARACTERISTIC CURVE (AUC-ROC)

A Receiver Operator Characteristic (ROC) curve is a graphical plot used to show the classification ability of binary classifiers. It is constructed by plotting the true positive rate (TPR) — also known as sensitivity or recall, given by Equation (10) — against the false positive rate (FPR) — also known as fall-out or false alarm ratio, given by Equation (16). In Equation (16), FPR_{HAM} and FPR_{SPAM} are given by Equations (14) and (15), respectively.

$$FPR_{HAM} = \frac{n_{S \to H}}{n_{S \to H} + n_{S \to S}} \tag{14}$$

$$FPR_{SPAM} = \frac{n_{H \to S}}{n_{H \to S} + n_{H \to H}} \tag{15}$$

$$FPR_{GEN} = \frac{N_{HAM} * FPR_{HAM} + N_{SPAM} * FPR_{SPAM}}{N_{HAM} + N_{SPAM}} \tag{16}$$

As shown in Figure 8, classifiers that produce curves closer to the top-left corner have better performances. As a baseline, a random classifier is expected to produce points lying along the diagonal (i.e., $FPR = TPR$).

A succinct way to evaluate classifiers using the ROC metric is by calculating the area under the curve, which can be done by trapezoidal approximation. As can be observed from Figure 8, the AUC value lies between 0.5 to 1 (i.e., 50% to 100%) in which the lower value denotes a bad classifier and the higher value denotes an excellent classifier.

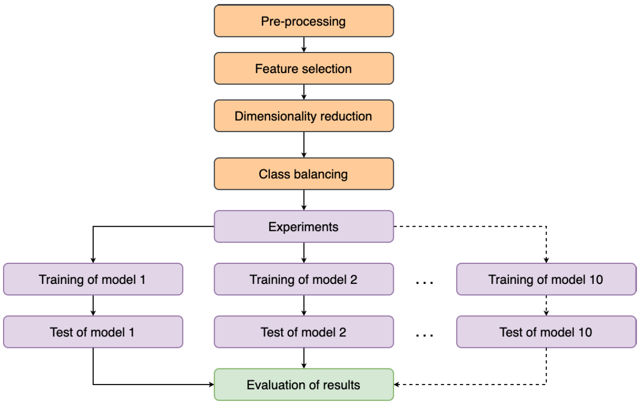


FIGURE 9. Methodology.

E. TRAINING TIME

The training time metric measures the time needed to train the model on the training set. The metric is given by Equation (17), in which t_i^T and t_f^T mark, respectively, the start and end times (HH:MM:SS) of the training.

$$t^T = t_f^T - t_i^T \tag{17}$$

F. CLASSIFICATION TIME

The classification time metric measures the time the model spends to classify all e-mails in the test set. The metric is given by the Equation (18), in which t_i^C and t_f^C mark, respectively, the start and end times (HH:MM:SS) of the classification.

$$t^C = t_f^C - t_i^C \tag{18}$$

IX. METHODOLOGY

In order to evaluate the performance of the different ML models (Section VI) as well as the feature selection methods (Section IV) and dimensionality reduction of the feature space (Section V), a methodology was developed to carry out the experiments. The steps of the methodology are specified by the diagram blocks in Figure 9. The steps are described below.

A. PRE-PROCESSING

In the pre-processing step, described in detail in Section III, the subject text and body text of each e-mail are standardized (e.g., letters are converted to lowercase, accents to words are removed) and each information relevant to its classification is converted into a specific tag (e.g., attachments and figures are replaced by their corresponding specific tags). The information relevant to the classification of e-mails is based on techniques used by spammers,[13] described by Cournane and Hunt [75]. At the end of pre-processing, each e-mail is represented by a set of tokens. Each token is either a word or specific tag in the subject or body of the e-mail.

[13] Spammers are individuals who send spam e-mails.

B. FEATURE SELECTION

In the feature selection step, described in detail in Section IV, the most relevant tokens for the classification of e-mails are selected by any of the three feature selection methods. Then, each e-mail is represented as a multidimensional vector in $\Re^n$, in which each dimension represents a selected token. The multidimensional vectors are normalized.

C. DIMENSIONALITY REDUCTION

In the dimensionality reduction step, described in detail in Section V, the MOEFS method is used to reduce the dimensionality of the e-mail feature space, that is, to reduce the dimensionality of the vectors that represent the e-mails.

D. CLASS BALANCING

In this step, the classes of e-mails are balanced in order to balance the contribution of each class in the training of the model. To this end, e-mails belonging to the class with the least amount of e-mails were randomly replicated. At the end of the step, both classes — ham and spam — have the same number of e-mails.

E. EXPERIMENTS

In this step, the e-mail database is shuffled and divided in half, preserving the balance of classes. The first half is used as training set and the second half as test set for the ML models. The null vectors (i.e., $\vec{x} = [x_1, x_2, \ldots, x_n] = [0, 0, \ldots, 0]$) from the training set are removed. On the other hand, the null vectors of the test set are maintained.

Each ML model is trained and tested ten times. Thus, the results obtained by each model are described in terms of the average and the confidence interval $C = 0.95$ [76] of the ten training sessions and the ten tests. In addition, the classification results are presented in the form of the metrics described in Section VIII.

F. EVALUATION OF RESULTS

In this step, the results obtained by the ML models are evaluated in terms of the F$_1$ score, AUC-ROC, training time, classification time, influence of the space dimensionality value, feature selection methods, dimensionality reduction, and computer suitability.

X. EXPERIMENTS AND RESULTS

All experiments were carried out on a single computer. The computer had an Intel Core(TM) i5-4570, 3.2-3.6 GHz, 6 MB cache processor, 32 GB DDR3-1333 RAM and ran the Linux Mint 18.3 Sylvia operating system.

The experiments followed the steps of the methodology described in Section IX. First, the pre-processing step was performed on the e-mails of each database. Then, the remaining steps were performed. The results of the execution of these steps are presented below.

TABLE 3. Percentages of null vectors generated in each e-mail database — *NF*: number of features; H_0: percentage of ham null vectors; S_0: percentage of spam null vectors.

Method	NF	Ling Spam		Spam Assassin		TREC		UNIFEI		UNIFEI-δ0	
		H_0	S_0	H_0	S_0	H_0	S_0	H_0	S_0	H_0	S_0
CHI2	8	97.8	99.3	4.2	11.7	99.6	98.9	99.7	99.9	98.8	98.6
	16	95.8	99.3	3.9	9.9	99.1	97.5	98.6	98.9	98.0	97.6
	32	91.9	99.3	3.5	9.1	98.9	97.2	98.4	98.8	96.0	96.2
	64	85.2	99.3	2.6	8.5	98.2	96.6	98.3	98.8	93.6	94.6
	128	74.2	99.3	2.0	6.5	97.2	94.6	97.8	98.7	89.9	88.7
	256	59.7	99.3	0.6	0.6	94.1	91.9	95.5	97.9	85.9	86.9
	512	34.2	61.9	0.6	0.6	82.2	78.4	86.3	88.2	69.5	60.2
	1024	34.2	49.1	0.5	0.5	57.5	45.1	42.2	47.3	41.2	36.2
FD	8	0.0	0.0	0.0	0.7	4.4	2.4	0.8	2.1	0.2	2.5
	16	0.0	0.0	0.0	0.2	4.3	2.4	0.8	2.1	0.1	2.5
	32	0.0	0.0	0.0	0.2	4.2	2.4	0.8	2.1	0.1	2.5
	64	0.0	0.0	0.0	0.2	4.1	2.4	0.8	2.1	0.1	2.5
	128	0.0	0.0	0.0	0.2	4.0	2.4	0.8	2.0	0.1	2.5
	256	0.0	0.0	0.0	0.2	4.0	2.4	0.8	2.0	0.1	2.5
	512	0.0	0.0	0.0	0.2	3.9	2.3	0.8	2.0	0.1	2.5
	1024	0.0	0.0	0.0	0.2	3.9	2.3	0.7	2.0	0.1	2.5
MI	8	0.0	0.0	0.0	1.0	7.8	3.0	15.1	14.9	14.9	15.0
	16	0.0	0.0	0.0	0.9	5.9	3.0	4.7	6.7	4.0	7.2
	32	0.0	0.0	0.0	0.2	4.4	2.8	4.7	6.7	4.0	7.2
	64	0.0	0.0	0.0	0.2	4.3	2.4	0.8	2.1	0.1	2.5
	128	0.0	0.0	0.0	0.2	4.3	2.4	0.8	2.1	0.1	2.5
	256	0.0	0.0	0.0	0.2	4.2	2.4	0.8	2.0	0.1	2.5
	512	0.0	0.0	0.0	0.2	4.2	2.4	0.8	2.0	0.1	2.5
	1024	0.0	0.0	0.0	0.2	3.9	2.3	0.8	2.0	0.1	2.5

TABLE 4. Reduction of the dimensionality of the vectors of the Ling Spam database.

NF	CHI2		FD		MI	
	NF'	Reduction (%)	NF'	Reduction (%)	NF'	Reduction (%)
8	1	87.5%	4	50.0%	6	25.0%
16	1	93.8%	4	75.0%	12	25.0%
32	1	96.9%	10	68.8%	19	40.6%
64	3	95.3%	19	70.3%	34	46.9%
128	4	96.9%	38	70.3%	52	59.4%
256	14	94.5%	58	77.3%	68	73.4%
512	14	97.3%	103	79.9%	139	72.9%
1024	242	76.4%	121	88.2%	157	84.7%

A. STEP: FEATURE SELECTION

The set of tokens that represent all e-mails in a database is much larger than the set of most relevant tokens, selected by the feature selection methods. Thus, the execution of the feature selection step generates null vectors. The Table 3 shows, for each feature selection method, the percentages of null vectors generated in each e-mail database.

From the results presented in the table, it can be seen that the CHI2 and FD methods are the ones that have, respectively, the highest and lowest amount of null vectors in all e-mail databases. Likewise, it can be verified that the number of null vectors generated by each method is inversely proportional to the number of features that represent the e-mails.

B. STEP: DIMENSIONALITY REDUCTION

In this step, the MOEFS method is executed to reduce, from *NF* to *NF*′, the dimensionality of the e-mail feature space. In other words, the method is executed to reduce the dimensionality of the vectors that represent the e-mails.

The Tables 4, 5, 6, 7 and 8 present, for each feature selection method, the reduction of the dimensionality of the vectors of the databases Ling Spam, Spam Assassin, TREC, UNIFEI and UNIFEI-δ0, respectively. In the tables, *NF* is the original dimension of the vectors, *NF*′ is the reduced dimension of the vectors and *Reduction (%)* is the percentage of reduction.

TABLE 5. Reduction of the dimensionality of the vectors of the Spam Assassin database.

NF	CHI2		FD		MI	
	NF'	Reduction (%)	NF'	Reduction (%)	NF'	Reduction (%)
8	4	50.0%	5	37.5%	3	62.5%
16	7	56.3%	6	62.5%	6	62.5%
32	14	56.3%	9	71.9%	18	43.8%
64	20	68.8%	16	75.0%	33	48.4%
128	25	80.5%	26	79.7%	58	54.7%
256	29	88.7%	53	79.3%	85	66.8%
512	7	98.6%	67	86.9%	96	81.3%
1024	20	98.0%	58	94.3%	84	91.8%

TABLE 6. Reduction of the dimensionality of the vectors of the TREC database.

NF	CHI2		FD		MI	
	NF'	Reduction (%)	NF'	Reduction (%)	NF'	Reduction (%)
8	4	50.0%	6	25.0%	5	37.5%
16	11	31.3%	7	56.3%	6	62.5%
32	10	68.8%	14	56.3%	9	71.9%
64	12	81.3%	7	89.1%	16	75.0%
128	45	64.8%	22	82.8%	26	79.7%
256	68	73.4%	24	90.6%	37	85.5%
512	133	74.0%	51	90.0%	66	87.1%
1024	47	95.4%	49	95.2%	204	80.1%

TABLE 7. Reduction of the dimensionality of the vectors of the UNIFEI database.

NF	CHI2		FD		MI	
	NF'	Reduction (%)	NF'	Reduction (%)	NF'	Reduction (%)
8	8	0.0%	5	37.5%	4	50.0%
16	10	37.5%	3	81.3%	7	56.3%
32	20	37.5%	6	81.3%	5	84.4%
64	48	25.0%	22	65.6%	9	85.9%
128	86	32.8%	65	49.2%	18	85.9%
256	34	86.7%	51	80.1%	31	87.9%
512	45	91.2%	101	80.3%	125	75.6%
1024	17	98.3%	105	89.7%	208	79.7%

TABLE 8. Reduction of the dimensionality of the vectors of the UNIFEI-δ0 database.

NF	CHI2		FD		MI	
	NF'	Reduction (%)	NF'	Reduction (%)	NF'	Reduction (%)
8	4	50.0%	5	37.5%	3	62.5%
16	16	0.0%	3	81.3%	3	81.3%
32	28	12.5%	8	75.0%	7	78.1%
64	42	34.4%	11	82.8%	10	84.4%
128	69	46.1%	24	81.3%	30	76.6%
256	116	54.7%	60	76.6%	49	80.9%
512	121	76.4%	79	84.6%	118	77.0%
1024	152	85.2%	77	92.5%	155	84.9%

From the results presented in the tables, it can be seen that:

- The CHI2 method was the one that provided the highest percentage of dimensionality reduction on the Ling Spam and Spam Assassin databases. In turn, the FD method provided the highest percentage of reduction on the TREC, UNIFEI and UNIFEI-δ0 databases;
- The highest percentage of dimensionality reduction was 98.6% and occurred with the CHI2 method on the Spam Assassin database. In turn, the lowest percentages of reduction occurred on the UNIFEI and UNIFEI-δ0 databases. In these two databases, there were occurrences of reduction percentages of 0% (i.e., there was no reduction in the number of features);
- The highest and lowest percentages of dimensionality reduction provided by the FD method were 95.2% and 25%, respectively, both on the TREC database;
- The highest percentage of dimensionality reduction provided by the MI method was 91.8% on the Spam Assassin database. The lowest percentage of reduction provided by the method was 25% on the Ling Spam database.

C. STEP: EXPERIMENTS

As described in Section IX-E, each ML model is trained and evaluated ten times on each e-mail database. Then, the mean and confidence interval of the F_1 score, AUC-ROC, training time and classification time metrics (Section VIII) are calculated.

Tables 9, 10, 11, 12 and 13 present the best results, ordered by F_1 score, obtained by each of the eight ML models on each e-mail database. In the tables, times are given in hours, minutes, seconds and milliseconds (HH:MM:SS.mmm), *FS* is the feature selection method, and *NF* and *NF′* are, respectively, the quantity of features before and after the dimensionality reduction.

Table 14 shows the number of occurrences of each feature selection method in the results of the Tables 9 to 13.

Similarly, Table 15 shows the number of occurrences of each original quantity of features *NF* in the results of the Tables 9 to 13.

D. STEP: EVALUATION OF RESULTS

The results are evaluated according to the four metrics — F_1 score, AUC-ROC, training time, classification time — as well as according to the quality of the feature selection method, dimensionality reduction, and computer suitability. In the evaluation, results whose confidence intervals overlap are considered equivalent results.

1) F_1 SCORE AND AUC-ROC

- The AB-M1 model, with tree architecture and boosting method, achieved the best performance on all e-mail databases. Thus, it surpassed all other models with other architectures as well as, in particular, the REPT model, also with tree architecture, but without the boosting method;
- The probabilistic model NB had the worst performance on all e-mail databases. Thus, in particular, it produced worse results than those produced by the other probabilistic model AODE;
- The SLP model, with neural network architecture, surpassed the RBF model, also with neural network architecture, on all e-mail databases with the exception of the UNIFEI-δ0 database. In turn, the RBF model obtained the seventh best performance on all databases except the UNIFEI-δ0, on which it obtained the sixth best performance;
- The nonlinear model P-SVM, a support vector machine with polynomial kernel, surpassed the linear model

TABLE 9. Best results obtained by the ML models on the Ling Spam database.

FS	NF	NF'	Model	F_1 score	AUC-ROC	Training time	Classification time
FD	1024	121	AB-M1	99.23 ± 0.07	99.97 ± 0.01	00:00:05.321 ± 00:00:00.249	00:00:00.018 ± 00:00:00.001
FD	512	103	AODE	98.55 ± 0.05	99.93 ± 0.01	00:00:00.236 ± 00:00:00.000	00:00:00.418 ± 00:00:00.000
MI	512	139	SLP	98.42 ± 0.21	98.17 ± 0.09	00:00:06.196 ± 00:00:00.017	00:00:00.011 ± 00:00:00.000
FD	1024	121	REPT	98.26 ± 0.14	99.45 ± 0.07	00:00:00.308 ± 00:00:00.010	00:00:00.004 ± 00:00:00.000
FD	1024	121	P-SVM	98.20 ± 0.15	98.20 ± 0.14	00:00:03.431 ± 00:00:00.352	00:00:00.388 ± 00:00:00.002
FD	1024	121	L-SVM	98.16 ± 0.08	97.62 ± 0.45	00:00:00.250 ± 00:00:00.005	00:00:00.145 ± 00:00:00.002
MI	512	139	RBF	97.01 ± 0.32	99.00 ± 0.15	00:00:18.229 ± 00:00:00.918	00:00:05.242 ± 00:00:00.021
MI	64	34	NB	95.20 ± 0.50	98.59 ± 0.07	00:00:00.022 ± 00:00:00.000	00:00:00.098 ± 00:00:00.001

TABLE 10. Best results obtained by the ML models on the Spam Assassin database.

FS	NF	NF'	Model	F_1 score	AUC-ROC	Training time	Classification time
MI	128	58	AB-M1	96.97 ± 0.29	99.36 ± 0.08	00:00:00.955 ± 00:00:00.034	00:00:00.007 ± 00:00:00.000
MI	256	85	P-SVM	95.48 ± 0.27	95.51 ± 0.28	00:00:01.835 ± 00:00:00.091	00:00:00.135 ± 00:00:00.001
MI	128	58	SLP	95.22 ± 0.20	95.21 ± 0.19	00:00:01.104 ± 00:00:00.029	00:00:00.002 ± 00:00:00.000
MI	128	58	L-SVM	95.04 ± 0.28	95.07 ± 0.28	00:00:00.087 ± 00:00:00.001	00:00:00.071 ± 00:00:00.003
MI	128	58	AODE	94.70 ± 0.22	99.18 ± 0.07	00:00:00.037 ± 00:00:00.000	00:00:00.066 ± 00:00:00.000
MI	128	58	REPT	94.04 ± 0.35	97.04 ± 0.24	00:00:00.064 ± 00:00:00.003	00:00:00.001 ± 00:00:00.000
FD	256	53	RBF	93.38 ± 0.47	96.80 ± 0.37	00:00:02.782 ± 00:00:00.221	00:00:00.729 ± 00:00:00.002
FD	512	67	NB	89.74 ± 0.37	95.54 ± 0.24	00:00:00.017 ± 00:00:00.000	00:00:00.078 ± 00:00:00.002

TABLE 11. Best results obtained by the ML models on the TREC database.

FS	NF	NF'	Model	F_1 score	AUC-ROC	Training time	Classification time
MI	1024	204	AB-M1	90.63 ± 0.07	96.55 ± 0.02	00:19:21.268 ± 00:00:29.870	00:00:01.610 ± 00:00:00.104
MI	1024	204	REPT	89.05 ± 0.06	95.05 ± 0.04	00:01:30.832 ± 00:00:01.972	00:00:00.136 ± 00:00:00.003
MI	1024	204	AODE	87.67 ± 0.07	95.48 ± 0.05	00:00:13.285 ± 00:00:00.050	00:00:30.387 ± 00:00:00.568
MI	1024	204	P-SVM	86.80 ± 0.11	86.53 ± 0.11	00:08:49.209 ± 00:01:02.402	00:00:09.555 ± 00:00:00.008
MI	1024	204	L-SVM	82.75 ± 2.73	81.98 ± 2.31	00:00:03.994 ± 00:00:00.113	00:00:01.943 ± 00:00:00.013
MI	1024	204	SLP	81.67 ± 3.73	81.77 ± 3.57	00:01:48.535 ± 00:00:00.208	00:00:00.201 ± 00:00:00.000
MI	128	26	RBF	77.20 ± 1.42	84.17 ± 0.57	00:01:54.845 ± 00:00:24.283	00:00:08.976 ± 00:00:00.026
FD	256	24	NB	72.08 ± 0.70	83.12 ± 0.25	00:00:00.312 ± 00:00:00.000	00:00:00.813 ± 00:00:00.008

TABLE 12. Best results obtained by the ML models on the UNIFEI database.

FS	NF	NF'	Model	F_1 score	AUC-ROC	Training time	Classification time
FD	128	65	AB-M1	93.96 ± 0.02	98.46 ± 0.03	00:17:11.884 ± 00:00:38.035	00:00:03.276 ± 00:00:00.121
FD	128	65	REPT	93.27 ± 0.02	97.65 ± 0.03	00:01:18.676 ± 00:00:01.901	00:00:00.356 ± 00:00:00.002
FD	128	65	AODE	92.37 ± 0.06	97.67 ± 0.01	00:00:29.522 ± 00:00:00.336	00:00:17.162 ± 00:00:00.255
MI	1024	208	P-SVM	83.00 ± 0.09	82.78 ± 0.09	01:32:39.551 ± 00:28:47.995	00:00:40.832 ± 00:00:00.019
MI	1024	208	L-SVM	80.91 ± 0.41	80.65 ± 0.57	00:00:22.785 ± 00:00:01.653	00:00:08.090 ± 00:00:00.028
MI	1024	208	SLP	78.78 ± 1.32	78.72 ± 1.17	00:07:43.886 ± 00:00:00.809	00:00:00.839 ± 00:00:00.011
MI	512	125	RBF	77.52 ± 0.23	87.03 ± 0.34	00:24:12.225 ± 00:03:07.822	00:04:07.769 ± 00:00:00.561
MI	512	125	NB	75.44 ± 0.17	84.27 ± 0.08	00:00:07.338 ± 00:00:00.064	00:00:16.914 ± 00:00:00.034

TABLE 13. Best results obtained by the ML models on the UNIFEI-δ0 database.

FS	NF	NF'	Model	F_1 score	AUC-ROC	Training time	Classification time
FD	8	5	AB-M1	97.80 ± 0.03	99.42 ± 0.01	00:00:39.308 ± 00:00:01.229	00:00:02.491 ± 00:00:00.066
FD	8	5	REPT	96.66 ± 0.02	98.71 ± 0.02	00:00:04.153 ± 00:00:00.014	00:00:00.293 ± 00:00:00.003
FD	8	5	AODE	95.07 ± 0.04	98.64 ± 0.01	00:00:02.406 ± 00:00:00.028	00:00:00.747 ± 00:00:00.040
MI	1024	155	P-SVM	82.27 ± 0.17	82.08 ± 0.17	00:58:43.293 ± 00:12:58.240	00:00:25.196 ± 00:00:00.013
MI	1024	155	L-SVM	81.23 ± 0.30	81.00 ± 0.50	00:00:14.037 ± 00:00:00.911	00:00:06.054 ± 00:00:00.018
MI	32	7	RBF	79.24 ± 1.04	86.01 ± 0.58	00:01:46.418 ± 00:00:03.602	00:00:07.948 ± 00:00:00.026
MI	512	118	SLP	79.17 ± 4.57	75.21 ± 0.94	00:04:52.090 ± 00:00:00.531	00:00:00.540 ± 00:00:00.012
FD	512	79	NB	71.96 ± 1.17	81.94 ± 0.09	00:00:04.986 ± 00:00:00.061	00:00:10.410 ± 00:00:00.082

TABLE 14. Number and percentage of occurrences of each feature selection method in the results of the Tables 9 to 13.

Feature selection method	CHI2	FD	MI
Number of occurrences	0	15	25
Percentage of occurrences	0%	37.5%	62.5%

TABLE 15. Number and percentage of occurrences of each original quantity of features *NF* in the results of the Tables 9 to 13.

Number of features	8	16	32	64	128	256	512	1024
Number of occurrences	3	0	1	1	9	3	8	15
Percentage of occurrences	7.5%	0%	2.5%	2.5%	22.5%	7.5%	20.0%	37.5%

L-SVM, also a support vector machine, on all e-mail databases.

2) TRAINING TIME

- The AODE model presented the best training time on the UNIFEI-δ0 database. In turn, the NB model presented the best training time on the remaining databases;
- The RBF model had the worst training time on the smallest bases — Ling Spam, Spam Assassin;
- The model AB-M1 had the worst training time on the medium-sized database — TREC;
- The P-SVM model had the worst training time on the largest databases — UNIFEI, UNIFEI-δ0.

3) CLASSIFICATION TIME

- The REPT model presented the best classification time on all e-mail databases;
- The RBF model presented the worst classification time on the Ling Spam, Spam Assassin and UNIFEI databases. In turn, the AODE and P-SVM models had the worst classification times on the TREC and UNIFEI-δ0 databases, respectively.

4) FEATURE SELECTION AND DIMENSIONALITY

- The MI feature selection method obtained the highest number of occurrences (62.5%) in the results of the Tables 9 to 13. Therefore, it is the method that allows the models to obtain their best results;
- The FD method obtained the second highest number of occurrences (37.5%) in the results of the Tables 9 to 13;
- The CHI2 method has no occurrences (0%) in the results of the Tables 9 to 13. Therefore, it is the method that prevents the models from obtaining their best results;
- The original dimensions of 1024, 128 and 512 features present, respectively, the largest (37.5%), the second largest (22.5%) and the third largest (20%) number of occurrences in the results of the Tables 9 to 13. The other dimensionalities add up to the remaining 20% of occurrences;
- The use of the MOEFS dimensionality reduction method (Section V) allowed a significant reduction in training and classification times of the ML models.

5) COMPUTER SUITABILITY

The worst training time was obtained by the P-SVM model on the UNIFEI database. This time, however, is not significant, since the training process for ML models is always carried out offline. In turn, the worst classification time (approximately four minutes) was obtained by the RBF model on the UNIFEI database. This time is not acceptable. However, all other ML models studied, whose classification times are in the order of seconds, can be used as classification models for anti-spam systems. Thus, the resources — CPU, RAM — of the computer used in the experiments were adequate.

XI. CONCLUSION

The large amount of spam e-mails circulating on the internet requires the development of anti-spam systems with a high degree of accuracy in the classification of e-mails. The main objective of this paper consisted in evaluating machine learning models in the classification of e-mails. Such models can be incorporated into existing anti-spam systems and, in particular, into the open-source anti-spam Open-MaLBAS [9], developed by the same authors of this paper. The Open-MaLBAS is available in GitHub [10].

The models were trained and tested on three public databases — Ling Spam, Spam Assassin, TREC — and two private ones — UNIFEI, UNIFEI-δ0 —, all made up of real e-mails. The experimental results indicate that machine learning models, combined with feature selection methods and the MOEFS dimensionality reduction method can be successfully applied to the classification of e-mails in the ham and spam classes.

Three directions for future work may be proposed. First, the test of other feature selection methods (e.g., Information Gain (IG) and Term Strength (TS) [48]) and dimensionality reduction methods (e.g., Principal Component Analysis (PCA) [77] and Autoencoders [78]). Second, the test of deep learning models [79], [80]. Finally, the incorporation of the models, evaluated in this paper, into the Classification Module of the Open-MaLBAS anti-spam.

REFERENCES

[1] The Radicati Group. (2018). *Email Statistics Report 2018–2022*. [Online]. Available: https://www.radicati.com/?p=15185

[2] Symantec Corporation. (2005). *Internet Security Threat Report 8*. [Online]. Available: https://docs.broadcom.com/doc/istr-05-Sep.-en

[3] Symantec Corporation. (2012). *Internet Security Threat Report 17*. [Online]. Available: https://docs.broadcom.com/doc/istr-12-april-volume-17-en

[4] Symantec Corporation. (2018). *Internet Security Threat Report 23*. [Online]. Available: https://docs.broadcom.com/doc/istr-22-2017-en

[5] *The Carbon Footprint of Email Spam Report*, ICF International, Fairfax, VA, USA, 2009.

[6] J. Jung and E. Sit, "An empirical study of spam traffic and the use of DNS black lists," in *Proc. 4th ACM SIGCOMM Conf. Internet Meas.*, Taormina, Italy, 2004, pp. 370–375.

[7] H. Zhang, H. Duan, W. Liu, and J. Wu, "IPGroupRep: A novel reputation based system for anti-spam," in *Proc. Symp. Workshops Ubiquitous, Autonomic Trusted Comput.*, 2009, pp. 513–518.

[8] D. M. W. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation," *J. Mach. Learn. Technol.*, vol. 2, no. 1, pp. 37–63, 2011.

[9] I. C. Ferreira, M. V. C. Aragão, E. M. Oliveira, B. T. Kuehne, E. M. Moreira, and O. A. S. Carpinteiro, "The development of the open machine-learning-based anti-spam (Open-MaLBAS)," *IEEE Access*, vol. 9, pp. 138618–138632, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9565223

[10] I. C. Ferreira, *Open Machine-Learning-Based Anti-Spam*. San Francisco, CA, USA: GitHub, 2020. [Online]. Available: https://github.com/Isaac44/Open-MaLBAS

[11] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos, "An evaluation of naive Bayesian anti-spam filtering," 2000, *arXiv:cs/0006013*.

[12] O. A. S. Carpinteiro, I. Lima, J. M. C. Assis, A. C. Z. de Souza, E. M. Moreira, and C. A. M. Pinheiro, "A neural model in anti-spam systems," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2006, pp. 847–855.

[13] R. Shams and R. E. Mercer, "Supervised classification of spam emails with natural language stylometry," *Neural Comput. Appl.*, vol. 27, no. 8, pp. 2315–2331, Nov. 2016.

[14] H. Drucker, D. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1048–1054, Sep. 1999.

[15] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian approach to filtering junk E-mail," in *Proc. Learn. Text Categorization, Papers Workshop*, Madison, Wisconsin, vol. 62, 1998, pp. 98–105.

[16] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos. (2000). *Ling-Spam Corpus*. [Online]. Available: http://www2.aueb.gr/users/ion/data/lingspam_public.tar.gz

[17] T. A. Meyer and B. Whateley, "SpamBayes: Effective open-source, Bayesian based, email classification system," in *Proc. Conf. Email Anti-Spam (CEAS)*, Mountain View, CA, USA, 2004, pp. 1–8.

[18] T. A. Meyer. (Apr. 2004). *Corpora Used for CEAS-04 Testing*. [Online]. Available: http://www.massey.ac.nz/~tameyer/research/spambayes/ceas.html

[19] Apache Software Foundation. (2005). *Spam Assassin Public Mail Corpus*. [Online]. Available: https://spamassassin.apache.org/publiccorpus/

[20] Rhyolite Software. (Jul. 2008). *Distributed Checksum Clearinghouses*. [Online]. Available: https://www.dcc-servers.net/dcc/

[21] S. Pollei. (Sep. 2004). *The GOSSiP (Gossip Optimization for Selective Spam Prevention) Project*. [Online]. Available: http://gossip-project.sourceforge.net/

[22] G. Singaraju and B. B. Kang, "RepuScore: Collaborative reputation management framework for email infrastructure," in *Proc. LISA*, vol. 7, 2007, pp. 243–251.

[23] N. Pérez-Díaz, D. Ruano-Ordás, J. R. Méndez, J. F. Gálvez, and F. Fdez-Riverola, "Rough sets for spam filtering: Selecting appropriate decision rules for boundary E-mail classification," *Appl. Soft Comput.*, vol. 12, no. 11, pp. 3671–3682, Nov. 2012.

[24] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning About Data*, vol. 9. Dordrecht, The Netherlands: Springer, 1991.

[25] F. Barigou, B. Beldjilali, and B. Atmani, "Using cellular automata for improving KNN based spam filtering," *Int. Arab J. Inf. Technol.*, vol. 11, no. 4, pp. 345–353, 2014.

[26] K.-M. Schneider, "A comparison of event models for naive Bayes anti-spam E-mail filtering," in *Proc. 10th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2003, pp. 1–8.

[27] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, and P. Stamatopoulos, "Stacking classifiers for anti-spam filtering of E-mail," 2001, *arXiv:cs/0106040*.

[28] G. V. Cormack and T. R. Lynam, "Online supervised spam filter evaluation," *ACM Trans. Inf. Syst.*, vol. 25, no. 3, p. 11, Jul. 2007.

[29] I. Santos, C. Laorden, B. Sanz, and P. G. Bringas, "Enhanced topic-based vector space model for semantics-aware spam filtering," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 437–444, Jan. 2012.

[30] Y. Kaya and Ö. F. Ertuärul, "A novel approach for spam email detection based on shifted binary patterns," *Secur. Commun. Netw.*, vol. 9, no. 10, pp. 1216–1225, Jul. 2016.

[31] G. V. Cormack and T. R. Lynam. (2006). *2006 TREC Public Spam Corpora*. [Online]. Available: https://plg.uwaterloo.ca/~gvcormac/treccorpus06/

[32] B. Tao. (2010). *CSDMC2010 SPAM Corpus*. [Online]. Available: http://csmining.org/index.php/spam-email-datasets-.html

[33] W. W. Cohen. (2008). *Enron Email Dataset*. [Online]. Available: https://www.cs.cmu.edu/~./enron/

[34] C. Yang, S. Zhao, D. Zhang, and J. Ma, "An anti-spam filter based on one-class IB method in small training sets," *Int. Arab J. Inf. Technol. (IAJIT)*, vol. 13, no. 6, pp. 677–685, 2016.

[35] J. Lin, "Divergence measures based on the Shannon entropy," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 145–151, Jul. 1991.

[36] M. Hopkins, E. Reeber, G. Forman, and J. Suermondt. (1999). *Spam Base Dataset*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/spambase

[37] I. Androutsopoulos, G. Paliouras, and E. Michelakis. (2003). *PU1, PU2, PU3, and PUA Corpora*. [Online]. Available: https://labs-repos.iit.demokritos.gr/skel/i-config/downloads/

[38] G. V. Cormack and T. R. Lynam. (2007) *2007 TREC Public Spam Corpus*. [Online]. Available: http://plg.uwaterloo.ca/~gvcormac/treccorpus07/

[39] A. Tyagi. (2016). *Content Based Spam Classification—A Deep Learning Approach*. [Online]. Available: https://prism.ucalgary.ca/handle/11023/3478

[40] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[41] T. Kumaresan, S. Saravanakumar, and R. Balamurugan, "Visual and textual features based email spam classification using S-Cuckoo search and hybrid kernel support vector machine," *Cluster Comput.*, vol. 22, no. 1, pp. 33–46, Jan. 2017.

[42] M. Dredze, R. Gevaryahu, and A. Elias-Bachrach. (2007). *Image Spam Dataset*. [Online]. Available: http://www.cs.jhu.edu/~mdredze/datasets/image_spam/

[43] M. Dredze, R. Gevaryahu, and A. Elias-Bachrach, "Learning fast classifiers for image spam," in *Proc. Conf. Email Anti-Spam (CEAS)*, Berlin, Germany, 2007, pp. 1–9.

[44] T. Kumaresan and C. Palanisamy, "E-mail spam classification using S-Cuckoo search and support vector machine," *Int. J. Bio-Inspired Comput.*, vol. 9, no. 3, pp. 142–156, 2017.

[45] X.-S. Yang and S. Deb, "Cuckoo search via Levy flights," in *Proc. World Congr. Nature Biologically Inspired Comput. (NaBIC)*, Coimbatore, India, 2010, pp. 210–214.

[46] S. Douzi, F. A. AlShahwan, M. Lemoudden, and B. E. Ouahidi, "Hybrid email spam detection model using artificial intelligence," *Int. J. Mach. Learn. Comput.*, vol. 10, no. 2, pp. 316–322, Feb. 2020.

[47] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[48] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proc. ICML*, vol. 97, Jul. 1997, pp. 412–420.

[49] T. M. Cover and J. A. Thomas, *Elements Information Theory*, 2nd ed. New York, NY, USA: Wiley, 1991.

[50] F. Jiménez, G. Sánchez, J. M. García, G. Sciavicco, and L. Miralles, "Multi-objective evolutionary feature selection for online sales forecasting," *Neurocomputing*, vol. 234, pp. 75–92, Apr. 2017.

[51] M. A. Hall, "Correlation-based feature selection of discrete and numeric class machine learning," in *Proc. 17th Int. Conf. Mach. Learn.* Burlington, MA, USA: Morgan Kaufmann, 2000, pp. 359–366.

[52] E. Frank, M. A. Hall, and I. H. Witten, "The WEKA workbench," in *Data Mining: Practical Machine Learning Tools and Techniques*, 2016. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

[53] Y. Murakami and K. Mizuguchi, "Homology-based prediction of interactions between proteins using averaged one-dependence estimators," *BMC Bioinf.*, vol. 15, no. 1, pp. 1–11, Dec. 2014.

[54] M. Sahami, "Learning limited dependence Bayesian classifiers," in *2nd Int. Conf. Knowl. Discovery Data Mining*, vol. 96, Portland, OR, USA, 1996, pp. 91–94.

[55] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, 1943.

[56] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.

[57] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Syst.*, no. 2, pp. 321–355, 1988.

[58] I. N. Silva, D. H. Spatti, and R. A. Flauzino, *Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas-Curso Prático*. São Paulo, CA, USA: Artliber, 2010, pp. 31–55.

[59] J. R. Quinlan, "Simplifying decision trees," *Int. J. Man-Mach. Stud.*, vol. 27, no. 3, pp. 221–234, Sep. 1987.

[60] R. E. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, 1990.

[61] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[62] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, vol. 96, Bari, Italy, 1996, pp. 148–156.

[63] P. Harrington, *Machine Learning Action*. Shelter Island, NY, USA: Manning, 2012.

[64] V. Vapnik, "Pattern recognition using generalized portrait method," *Automat. Remote Control*, vol. 24, no. 6, pp. 774–780, Jan. 1963.

[65] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory*, Pittsburgh, PA, USA, 1992, pp. 144–152.

[66] M. A. Aizerman, E. M. Braverman, and L. I. Rozoner, "Theoretical foundations of the potential function method in pattern recognition learning," *Autom. Remote Control*, vol. 25, pp. 821–837, Jun. 1964.

[67] Apache Software Foundation. (2015). *Apache Spam Assassin Project*. [Online]. Available: http://spamassassin.apache.org/

[68] B. Sirisanyalak and O. Sornil, "An artificial immunity-based spam detection system," in *Proc. Congr. Evol. Comput. (CEC)*. Singapore, 2007, pp. 3392–3398.

[69] G. V. Cormack and T. R. Lynam. (2007). *Text Retrieval Conference (TREC) Spam Track*. [Online]. Available: https://trec.nist.gov/data/spam.html

[70] G. V. Cormack and T. R. Lynam. (2005). *2005 TREC Public Spam Corpus*. [Online]. Available: https://plg.uwaterloo.ca/ gvcormac/treccorpus/

[71] Roaring Penguin Software. *CanIt-PRO*. Accessed: Aug. 2021. [Online]. Available: https://www.roaringpenguin.com/products/canit-pro

[72] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[73] Data Science Stack Exchange. *Are T-SNE Dimensions Meaningful*. Accessed: Aug. 2021. [Online]. Available: https://datascience.stackexchange.com/questions/17314/are-t-sne-dimensions-meaningful

[74] G. Tourassi, *Receiver Operating Characteristic Analysis: Basic Concepts Practical Applications*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2018, pp. 227–244.

[75] A. Cournane and R. Hunt, "An analysis of the tools used for the generation and prevention of spam," *Comput. Secur.*, vol. 23, pp. 154–166, 2004.

[76] J. Neyman, "Outline of a theory of statistical estimation based on the classical theory of probability," *Phil. Trans. Roy. SoC. London Ser. A, Math. Phys. Sci.*, vol. 236, no. 767, pp. 333–380, 1937.

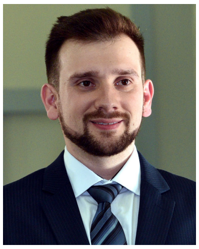
[77] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philos. Mag.*, vol. 2, no. 6, pp. 559–572, 1901.

[78] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation* (Explorations in the Micro-Structure of Cognition). La Jolla, CA, USA: MIT Press, 1985, no. 2, ch. 8, pp. 318–362.

[79] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 445–458, Feb. 2019.

[80] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "FS-Net: A flow sequence network for encrypted traffic classification," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2019, pp. 1171–1179.

MARCELO V. C. ARAGÃO received the B.Sc. degree in computer engineering from the National Institute of Telecommunications (INATEL), Brazil, in 2014, and the M.Sc. degree in computer science and technology from the Federal University of Itajubá, Brazil, in 2018. He is currently pursuing the Ph.D. degree in telecommunication engineering with INATEL. From 2011 to 2018, he was as a System Specialist with the INATEL Competence Center, where he developed business support systems (BSS) solutions in continuous integration environment. He also teaches undergraduate courses, coordinates the graduate course in application development for mobile devices and cloud computing with INATEL. His research interests include machine learning, data science, and software engineering.



ISAAC C. FERREIRA received the B.Sc. degree in computer engineering and the M.Sc. degree in computer science and technology from the Federal University of Itajubá, Brazil, in 2014 and 2018, respectively. He is currently working as a Research and Development Engineer with the TRICOD Equipamentos Eletrônicos Indústria e Comércio LTDA (TRICOD Electronic Equipments Industry and Commerce Ltd.), Brazil.

EDVARD M. OLIVEIRA received the B.Sc. degree in computer science from the Pontifical Catholic University (PUC) of Minas Gerais, Brazil, in 2010, and the M.Sc. and Ph.D. degrees in computer science and computational mathematics from the University of São Paulo (USP), Brazil, in 2013 and 2018, respectively. He is currently an Assistant Professor with the Federal University of Itajubá, Brazil. He also works with distributed systems, with emphasis in e-science, scientific gateways, service oriented architectures (SOA), scientific workflows, and the Internet of Things, and is involved in research and development projects within the Research Group on Systems and Computer Engineering, Federal University of Itajubá.

BRUNO T. KUEHNE received the B.Sc. degree in computer science from the Pontifical Catholic University of Minas Gerais, Brazil, in 2006, and the M.Sc. and Ph.D. degrees in computer science and computational mathematics from the University of São Paulo (USP), Brazil, in 2009 and 2015, respectively. Since 2014, he has been an Assistant Professor and a member of the Research Group on Systems and Computer Engineering, Federal University of Itajubá, Brazil. He has experience in computer science, with emphasis on computer systems, working mainly on the following topics, such as QoS, web services, cloud computing, fog computing, the IoT, and performance evaluation.

EDMILSON M. MOREIRA received the B.Sc. degree in computer science from the University of Alfenas, Brazil, in 1994, the B.Sc. degree in mathematics from the Faculty of Philosophy, Sciences and Linguistics, Varginha, Brazil, in 1996, and the M.Sc. and Ph.D. degrees in computer science and computational mathematics from the University of São Paulo (USP), Brazil, in 2000 and 2005, respectively. He is currently an Associate Professor with the Federal University of Itajubá, Brazil. He works with graph theory, distributed systems, and discrete events simulation. He is a member of the Research Group on Systems and Computer Engineering, Federal University of Itajubá.

OTÁVIO A. S. CARPINTEIRO was born in Rio de Janeiro, Brazil. He received the B.Sc. degree in mathematics and the B.Sc. degree in music and the M.Sc. degree in systems and computer engineering from the Federal University of Rio de Janeiro (UFRJ), and the D.Phil. degree in cognitive and computer science from the University of Sussex, U.K. He worked as a System Analyst for many years, and ended his professional career as a Full Professor with the Federal University of Itajubá, Brazil, in which, he did research, supervised graduate students, and taught undergraduate and graduate courses in computer engineering. He is currently retired, but is still working in research and development projects as a member of the Research Group on Systems and Computer Engineering, Federal University of Itajubá.