IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 4, 2021, accepted October 4, 2021, date of publication October 8, 2021, date of current version October 15, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3118901

The Development of the Open Machine-Learning-Based Anti-Spam (Open-MaLBAS)

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

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

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

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

ABSTRACT Spam e-mails are unsolicited e-mails received by users of the e-mail service. Spam e-mails cause serious harm to organizations, for they waste, among other things, their computational and networking resources. To reduce the damage caused by them, organizations use anti-spams. Anti-spams are software systems that classify e-mails in order to separate legitimate from spam e-mails. The best current commercial and open-source anti-spams, and in particular the well-known commercial anti-spam CanIt-PRO, make use of various techniques, such as blacklists and/or SMTP extensions, to classify e-mails. Unfortunately, both blacklists and SMTP extensions have serious drawbacks, such as low scalability and high computational and network costs. This paper introduces the Open Machine-Learning-Based Anti-Spam (Open-MaLBAS). Unlike the best current anti-spams, Open-MaLBAS does not make use of blacklists and SMTP extensions, but only of machine learning models for e-mail classification. Open-MaLBAS was compared to CanIt-PRO in a series of experiments on a database composed of 862,227 real e-mails, collected over three months at the Federal University of Itajubá, Brazil. The e-mails were previously classified by CanIt-PRO. From the experiments, it was observed that Open-MaLBAS was able to correctly classify 81.48% and 98.13% of the e-mails in the database, using, respectively, the two models — Multi-Layer Perceptron and Random Forest — evaluated. In addition, it managed to obtain times of up to 88% shorter than those of CanIt-PRO to classify all e-mails in the database. Open-MaLBAS is implemented in Java language, under free software license, for free use. It is available on GitHub.

INDEX TERMS Electronic mail (e-mail), internet, machine learning, network security, open source software, simple mail transfer protocol (SMTP), software engineering, unsolicited electronic mail (spam).

I. INTRODUCTION

An anti-spam (AS) is a software system that classifies e-mails in order to separate legitimate from spam e-mails. Spam e-mails are unsolicited electronic messages posted blindly to many recipients, usually for commercial advertisement. Spam wastes computational and networking resources and causes large losses to organizations, for the number of spam e-mails circulating on computer networks is high. Indeed, according to Statista Co., over 50% of e-mail traffic circulating in the Internet consists in some kind of spam [1].

The best current commercial and open-source anti-spams (ASes) make use of address lists — blacklists [2], greylists [3], whitelists [2] — on the Internet for e-mail classification. Blacklists, the most important address lists, are lists which contain addresses or domains of suspicious e-mail senders or servers. Blacklists have four serious drawbacks. Firstly, they may not be updated as fast as the spammers[1] change their sender addresses or domains. Secondly, a legitimate e-mail service provider runs always the risk of having any of its addresses (or domains) unduly inserted into one or more blacklists, and that causes it

The associate editor coordinating the review of this manuscript and approving it for publication was Claudio Agostino Ardagna.

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

considerable annoyances [4]–[7]. Thirdly, blacklists are not scalable, i.e., they will grow largely in size with the full adoption of the Internet Protocol version 6 (IPv6) [8], causing a serious decay in performance to access them. At last, the most trusteable blacklists are managed by operators who charge for their use. Thus, organizations have to pay annual fees for using ASes that use them.

The best current commercial and open-source ASes also make use of extensions of the Simple Mail Transfer Protocol (SMTP) [9] for e-mail classification. The most employed extensions are the Sender Policy Framework (SPF) [10], Domain-Keys Identified Mail (DKIM) Signatures [11], and Domain-based Message Authentication, Reporting, and Conformance (DMARC) [12]. SPF policy requires querying the records of the Domain Name System (DNS) servers of the domain to check whether or not the e-mail server that sent the e-mail has permission from the domain to send e-mails. SPF is incompatible with e-mail forwarders as well. DKIM, in turn, requires the use of encrypted signatures to validate the e-mail sender, and DMARC builds on SPF and DKIM. Thus, all three SMTP extensions generate additional expenses, both in terms of computation and time. In addition, they show vulnerabilities when implemented together in a same e-mail server. Indeed, Chen *et al.* [13] discovered various vulnerabilities in e-mail servers of ten popular e-mail providers. All these state-of-the-art servers implement SPF, DKIM, and DMARC.

This paper introduces the Open Machine-Learning-Based Anti-Spam (Open-MaLBAS). Unlike the best current commercial and open-source ASes, and in particular the well-known commercial AS CanIt-PRO [14], Open-MaLBAS does not make use of blacklists on the Internet and of SMTP extensions, but only of machine learning (ML) models for e-mail classification.

Open-MaLBAS was thoroughly evaluated experimentally. It was compared to CanIt-PRO 9.2.4 in a series of experiments on a large database composed of 862,227 real e-mails, collected over three months at the Federal University of Itajubá (UNIFEI), Brazil. The e-mails were previously classified by CanIt-PRO.

From the experiments, it was observed that Open-MaLBAS, using the two ML models — Multi-Layer Perceptron and Random Forest — evaluated, performed very close to CanIt-PRO in terms of e-mail classification. In addition, it achieved a much better performance in terms of the time required for classification.

CanIt-PRO was chosen as the representative of all other existing anti-spams for four main reasons. First, it includes the best techniques (described in Section II-E) currently used for spam detection. Second, it is an anti-spam that has been on the market for many years, which indicates that it is very well regarded by organizations. Third, it receives constant and periodic updates and improvements, and is currently in its 10.2.3 version. Fourth, it was used over several years, until July-2019, at our university — UNIFEI. From August-2019 on, the university network services, including e-mail service, have been providing through the G-Suite platform of Google.

The paper makes three important contributions. Firstly, it introduces the Open-MaLBAS, implemented in Java language. Open-MaLBAS is a free-use AS, under the GNU general public license version 3 [15], available on GitHub [16]. Secondly, it thoroughly assesses Open-MaLBAS on a large database of real e-mails and compares its results with those obtained by CanIt-PRO. Thirdly, it shows that Open-MaLBAS may be both as much efficient in terms of e-mail classification as, and more efficient in terms of the time required for classification than the best current commercial and open-source ASes.

The paper is divided into sections as follows. The second section reviews some existing ASes. The third section provides an overview of Open-MaLBAS. The fourth section details the modules of Open-MaLBAS. The fifth and sixth sections describe, respectively, the data representation and processing, and the metrics employed in the experiments. The seventh section presents the experiments performed as well as evaluates their results. Finally, the eighth section concludes the paper and provides some directions for future work.

II. A REVIEW ON EXISTING ANTI-SPAMS

This section reviews some of the best current and well-known commercial and open-source ASes.

A. SpamAssassin

SpamAssassin [17] is an open-source anti-spam. It can be integrated either with e-mail servers or with e-mail clients. It makes use of a large set of rules to determine whether each e-mail received is ham[2] or spam. Most of the rules are based on regular expressions, which are searched for within the body of the e-mail and/or in its header. SpamAssassin includes several spam detection techniques, such as Bayesian filtering, DNS blacklist (DNSBL), Uniform Resource Identifier blacklist (URIBL), DNS whitelist (DNSWL), SPF, among others.

B. ASSP

Anti-Spam SMTP Proxy (ASSP) [18] is an open-source anti-spam. It is implemented in language Perl and runs as a proxy server. It includes several spam detection techniques, such as Bayesian filtering, HELO (or EHLO) command validation, blacklists (e.g., DNSBL, URIBL), greylist, whitelist and SPF. The ASSP administrator can allow e-mail users to have their own private white and black lists. The destination addresses of e-mails sent by users are automatically included in their whitelists.

C. QPSMTPD

Qpsmtpd [19] is an open-source anti-spam. It consists in a daemon process that executes a SMTP code implemented in language Perl. It also implements a set of plugins that

[2]Legitimate e-mails are also referred to as hams.

allows e-mail service administrators to perform spam filtering in an easier way. The set of plugins includes HELO (or EHLO) command validation, DNSBL, URLBL, greylist, SPF, spam filters (e.g., SpamAssassin) and anti-virus (e.g., Bitdefender [20], ClamAV [21], among others). The qpsmtpd daemon was designed to run in front of some Mail Transfer Agent (MTA) (e.g., qmail [22], postfix [23], exim [24]). The e-mail is received by qpsmtpd and processed through the plugins, to be evaluated and classified as ham or spam.

D. BARRACUDA

Barracuda Email Security Gateway [25] is a commercial anti-spam. It includes several features, such as spam and virus blocking, protection of sensitive data through encryption, protection against e-mail sender forgery, protection against phishing [26], protection against Distributed Denial of Service (DDoS) attacks, among others.

E. CanIt-PRO

CanIt-PRO [14] is a commercial anti-spam. It includes several features, such as protection against spam and viruses, blacklists, whitelists, greylists, SPF, DKIM, DMARC, Bayesian filtering, e-mail archiving, reports and statistics on the e-mails it processes, among others. The CanIt-PRO administrator can allow e-mail users to manage their own private configuration.

F. DISCUSSION

In addition to the three anti-spams — SpamAssassin, ASSP, Qpsmtpd — listed above, there are other open-source anti-spams, such as Rspamd, Scrolloutf1, MailCleaner, and Proxmox. Most open-source anti-spams are, however, just an interface to another known open-source anti-spam (e.g., Postfix, SpamAssassin, Rspamd, ClamAV, among others). All these current open-source anti-spams implement the best spam detection techniques, such as SMTP extensions — SPF, DKIM, DMARC — and permission and blocking lists — whitelist, greylist, blacklist [27].

Similarly, in addition to the two anti-spams — Barracuda, CanIt-PRO — listed above, there are other commercial anti-spams, such as Proofpoint Email Security and Protection, SpamTitan Email Security, SolarWinds Mail Assure, and DuoCircle Spam Filtering. Commercial anti-spams, obviously, do not have open source codes. Thus, it is very difficult to know which spam detection techniques they implement. In fact, G2.com had to resort to reviews gathered from its user community, as well as data aggregated from online sources and social networks to rate the quality and performance of eighty anti-spams, including open-source and commercial [28]. Given the performance of the commercial anti-spams assessed and the satisfaction of the G2.com community with these performances, it is very likely that most, if not all, of current commercial anti-spams also implement, just like current open-source anti-spams, the best spam detection techniques, such as SMTP extensions and address lists.

III. OPEN-MaLBAS OVERVIEW

Open-MaLBAS is an anti-spam for e-mail servers. It does not make use of blacklists on the Internet and of SMTP extensions in order not to suffer the disadvantages of them both. Instead, it makes use of ML models for e-mail classification.

Open-MaLBAS may be run as a foreground process or as a daemon process.[3] It has two operating modes — training mode and running mode. The training mode is used to the periodic training of ML models. The training makes use of a set of e-mails collected during a period of time. These e-mails are previously classified, as spam or ham, by the Open-MaLBAS users. In the running mode — its normal mode of operation —, the Open-MaLBAS classifies, as spam or ham, the e-mails it receives from the Internet. The training mode and running mode are always performed offline and online, respectively.

Open-MaLBAS has a modular architecture. It lets its administrator run each of their modules individually or altogether. Its implementation is based on design patterns [29] in language Java. The implementation took into consideration its computational performance.

The source code of Open-MaLBAS is clear, simple and easily maintainable. All comments and documentation (JavaDocs) are written in English. In addition, all messages issued by Open-MaLBAS are saved in files in order to facilitate their translation to other languages. The source code is licensed under the GNU general public license version 3 [15], for free use. It is available in GitHub [16].

IV. OPEN-MaLBAS MODULES

Figure 1 presents the modular architecture of Open-MaLBAS. Its modules are described next.

A. POSTFIX-ABL

Open-MaLBAS makes use of both the Mail Delivery Agent (MDA) and the Mail Transfer Agent (MTA) of Postfix 2.7.5 [23]. Postfix is an e-mail server. Its MDA has the function of receiving e-mails from the Internet.

The daemon *smtpd* of the Postfix MDA was modified to contain the Active Blacklist (ABL) [30]. ABL is based on a modification of the SMTP. It differs from usual passive blacklists in that it produces three advantageous consequences. Firstly, it promptly rejects, during SMTP negotiation, the spam e-mails thus defined by each e-mail user of an organization, avoiding a waste of the computational and network resources of the organization. Secondly, it returns the spam e-mails to the spammer, penalizing him/her, for his/her server will use more computational and network resources to handle the rejected spam e-mails. Thirdly, owing to the cost of the refusal, the spammer usually removes the user e-mail address from his/her distribution lists.

After receiving each e-mail from the Internet, the Postfix-ABL MTA forwards it to the SMTP Module, using the SMTP protocol [9].

[3]Background processes are also referred to as daemon processes.

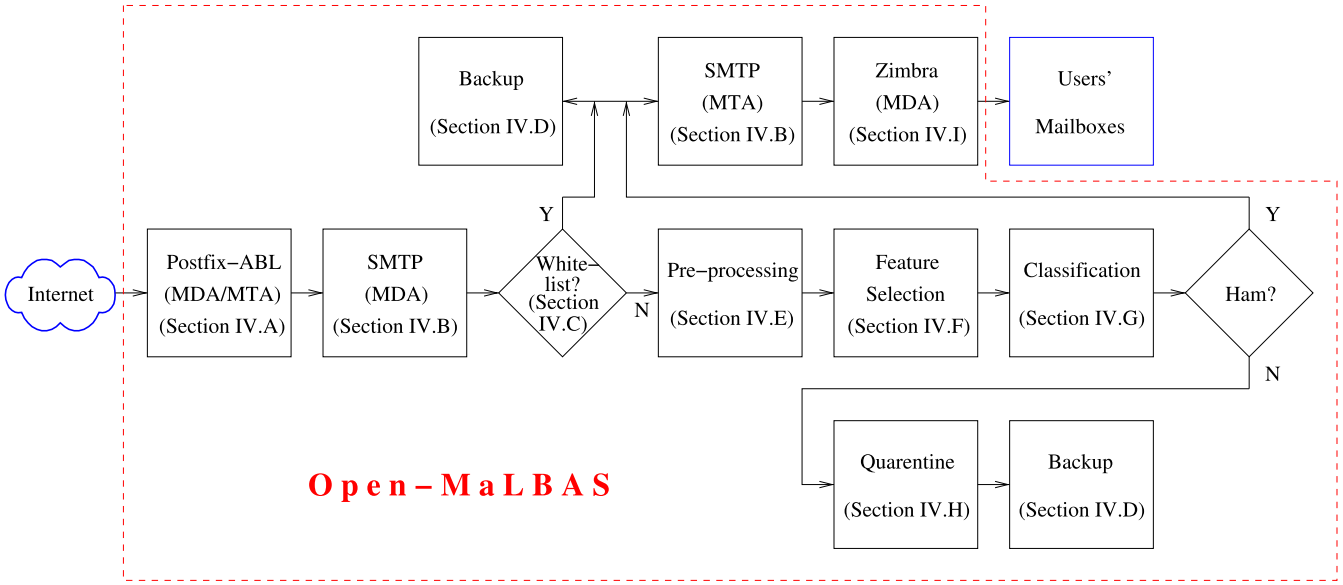

FIGURE 1. Open-MaLBAS modules.

B. SMTP MODULE

The SMTP Module includes both an MDA and an MTA. Through its MDA, the SMTP Module receives e-mails sent by Postfix-ABL MTA. In turn, through its MTA, the SMTP Module sends e-mails to the MDA of the Zimbra server (Section IV-I). The SMTP Module allows Open-MaLBAS to run on a different computer than those running Postfix-ABL and Zimbra servers.

The implementation of the SMTP Module aims to reduce the processing time for sending/receiving e-mails. For example, the implementation makes use of reusable buffers for storing e-mails and threads. The number of threads is configurable and defines the maximum number of e-mails that can be handled simultaneously. The implementation makes use of the open-source library SubEtha-SMTP [31].

C. WHITELIST MODULE

The Whitelist Module implements a whitelist for each e-mail user. E-mails whose sender addresses are in the whitelists are sent directly to their recipients and also to the Backup Module, so that they can be stored. In turn, e-mails whose sender addresses are not in the whitelists pass through the following three modules — Pre-processing, Feature Selection, and Classification Modules — to be classified.

The whitelists are populated indirectly by e-mail users via the Quarentine Module (Section IV-H). The Quarentine Module periodically sends reports containing e-mails classified as spam to each user. Thus, when an user deselects an e-mail classified as spam in the report, the sender address of the e-mail is registered in the user whitelist.

D. BACKUP MODULE

The seventh article of the Brazilian Information Access Law (Law No. 12,527, Nov. 18, 2011 [32]) requires, for auditing purposes, the storage of e-mails received by any server [33]. Thus, to comply with the legislation, the Backup Module stores, in a specific directory (folder), a copy of each e-mail that Open-MaLBAS processes when operating in running mode.

The Backup Module is also responsible for storing, in another specific directory (folder), a copy of each e-mail that Open-MaLBAS has correctly classified as spam. The stored spam e-mails are integrated into the set of e-mails used in the periodic training of ML models.

E. PRE-PROCESSING MODULE

The Pre-processing Module detects, in the body and subject of the e-mail, many of the techniques used by spammers [34], marking them, if necessary, with specific tags, in order to increase the probability of the e-mail being classified as ham or spam. The body of any e-mail can contain plain text and/or text in HTML format. If it contains both, the Pre-processing Module analyzes only its text in HTML format.

The Pre-processing Module considers each e-mail to be composed of text — words, numbers, special characters — and HTML tags. Therefore, it processes each e-mail through two types of filters — text filters and HTML filters. Text filters standardize or convert the text of the e-mail body and subject. For example, letters are converted into lower case, accent marks are removed from the accented letters, and URL addresses, e-mail addresses, currency, and percentage are converted respectively into the specific tags "*!_LINK*", "*!_EMAIL*", "*!_MONEY*", and "*!_PERCENTAGE*".

The HTML filters make use of the Java library Jsoup [35] to process HTML tags of the e-mail body and subject. For this purpose, the HTML tags are divided into three categories, according to the relevance of the information they enclose. They are processed according to the category which they belong to.

HTML tags in the first category are related, in the vast majority, to the description of the document. So, they are totally discarded, that is, the tags, their attributes, and the contents they enclose are completely removed. The HTML tag "*<title>*", for instance, is employed to display the contents it encloses in the navigation bar of browsers. Thus, the block "*<title> contents </title>*" is totally discarded during pre-processing.

HTML tags in the second category present contents parcially significant to the classification of e-mails. Therefore, they only have their attributes removed during pre-processing. Moreover, each one of these tags is replaced by a specific tag. For instance, the block "* contents *" is converted into "*!_IN_P contents*" during pre-processing.

HTML tags in the third category, in its turn, present contents totally significant to the classification of e-mails. So, they are processed in their entirety except for the parameters of their attributes which are removed. More, each one of these tags is replaced by another specific tag. For instance, the block "*<form action = "results.php"> contents </form>*" is converted into "*!_IN_FORM action contents*" during pre-processing.

Each e-mail received by the Pre-processing Module is represented, after the end of its pre-processing, by a set of tokens. Each token is either a word or a specific tag of the e-mail body or subject.

F. FEATURE SELECTION MODULE

All tokens that represent all possible e-mails can be used to represent each e-mail as a multidimensional vector in $\Re^n$. With this, however, e-mails would be represented by very high dimensionality vectors, generating equally high storage and processing costs. Furthermore, as a server anti-spam, Open-MaLBAS would spend a lot of time to classify each e-mail, something that should be avoided.

The Feature Selection Module has two functions. The first, performed only when Open-MaLBAS operates in Training Mode, is to order, in order of relevance, the tokens found in a set of e-mails. Thus, e-mails can be represented by much lower dimensionality vectors, in which each dimension represents a relevant token for their classification in the ham and spam classes. The module makes use of two statistical methods — Frequency Distribution (FD) and Mutual Information (MI) — to sort the tokens, in order of relevance.

Frequency Distribution (FD) [36] assigns relevance to each token by the number of times it appears in the set of e-mails. It is very simple and fast. In turn, Mutual Information (MI) [37] weighs the degree of relevance of each token to the ham and spam classes. For that, it uses probability theory.

The second function of the Feature Selection Module, performed when Open-MaLBAS operates in both Training Mode and Running Mode, is to represent each e-mail as a multidimensional vector in $\Re^n$, in which each dimension represents a relevant token, selected by one of the two statistical methods. The multidimensional vectors are normalized. To this end, the module implements three normalization algorithms. Both the dimensionality n, that is, the number of most relevant tokens, and the normalization algorithm to be used are defined by the Open-MaLBAS administrator.

When Open-MaLBAS operates in Training Mode, each vector generated is saved either in a ham file or in a spam file, depending on the classification of the e-mail it represents. Both files are used in the training of the classifier model (Section IV-G). In turn, when Open-MaLBAS operates in Running Mode, the generated vector is sent directly to the classifier model.

G. CLASSIFICATION MODULE

The Classification Module also has two functions. The first, performed only when Open-MaLBAS operates in Training Mode, is to train the classifier model, in order to enable it to correctly classify e-mails, represented by vectors, in the ham and spam classes. The second function, performed only when Open-MaLBAS operates in Running Mode, is to classify, in the ham and spam classes, new e-mails, represented by vectors, which are sent by Postfix-ABL Module.

The Classification Module uses, as classifier models, ML models provided by the open-source library Weka [38]. In this study, only two ML models were used — Multi-Layer Perceptron (MLP) and Random Forest (RF).

MLP [39] is an ML model inspired by the neural structure of human beings. It consists in interconnected artificial neural units that simulate the behavior of human neurons. For example, in the human brain, each neuron is activated by other neurons through connections, known as synapses. When a neuron receives activation from other neurons and it exceeds its threshold, it transmits a new activation to the following neurons. Similarly, in MLP, each artificial neural unit in a layer l receives activation from the artificial neural units in layer $l-1$ and transmits a new activation to the artificial neural units in layer $l+1$. MLPs have been widely employed in pattern recognition problems.

RF [40] is an ML model that consists in an ensemble of models with decision tree architecture. As an ensemble of models, the RF produces better classification results than those produced by any of its individual models. RFs have been widely employed in pattern recognition problems as well.

H. QUARENTINE MODULE

Open-MaLBAS classifies each e-mail as ham or spam. If classified as ham, the e-mail is delivered to its recipient, but if classified as spam, it is sent to the Quarentine Module. To avoid false positive[4] situations, the Quarentine Module forwards, at intervals defined by the Open-MaLBAS administrator, a report to each user containing the spam e-mails the

[4]A *false positive* is a ham e-mail incorrectly classified as spam by anti-spam. In turn, a *false negative* is a spam e-mail incorrectly classified as ham by anti-spam.

user received since the last report. Thus, the user will be able to deselect and retrieve e-mails incorrectly classified as spam.

E-mails retrieved by users are stored in a ham e-mail file. Those that have not been retrieved are stored in a spam e-mail file. The ham and spam e-mail files are later used in the training of the classifier model (Section IV-G).

I. ZIMBRA

Any anti-spam should focus solely on performing its only task — e-mail classification — so that it can perform well. For this reason, Open-MaLBAS uses the MTA of its SMTP Module to send e-mails, already classified, to the MDA of the e-mail server Zimbra 8.0 [41]. The Zimbra server then takes care of dispatching each e-mail, already classified, to the recipient's mailbox.

V. DATA REPRESENTATION AND PROCESSING

The e-mails used in the experiments are real. They were classified in the ham and spam classes by the anti-spam CanIt-PRO and collected from its database.

The university imposed conditions for collecting e-mails from the CanIt-PRO database, in order to preserve the confidentiality of the information contained therein. Thus, 353,151 ham e-mails and 509,076 spam e-mails were collected in a period of just three months. Likewise, only software programs processed the e-mails. Each e-mail was processed by the Pre-processing Module to transform it into a file containing tokens. At the end of the processing, all the original e-mails were destroyed.

The database of processed e-mails, henceforth called *UNIFEI database*, is therefore composed of real e-mails, but under the representation given by token files. The representation by token files does not allow the reconstruction of the original e-mail, thus preserving the anonymity of the sender and recipients, of the route traveled, as well as the confidentiality of the body and attachments of each original e-mail.

The histogram in Figure 2 shows the sizes, in kiloBytes (KB), of the token files of the UNIFEI database. From the histogram, it is possible to see that there may be empty files. An empty file contains an e-mail with no tokens. This can occur, for example, if the original e-mail contained only attachments or if it contained only invalid characters in its body. Most e-mails are between 0 (zero) and 3 KB in size. They represent about 70% of the UNIFEI database.

Five steps were taken in order to make the e-mails of the UNIFEI database graphically visible. First, the Feature Selection Module was used to select, using the FD statistical method, the 1024 most relevant tokens for the classification of the e-mails from the UNIFEI database. Second, the Feature Selection Module was used again to convert each e-mail (i.e., each token file) into a real vector $\vec{v}$ of dimensionality 1024 ($\vec{v} \in \Re^{1024}$). Third, each group of identical vectors was stored in a single set. Fourth, it was verified, in each set, if CanIt-PRO had classified its vectors, all identical, in a single class. When this did not occur, a new ham/spam class

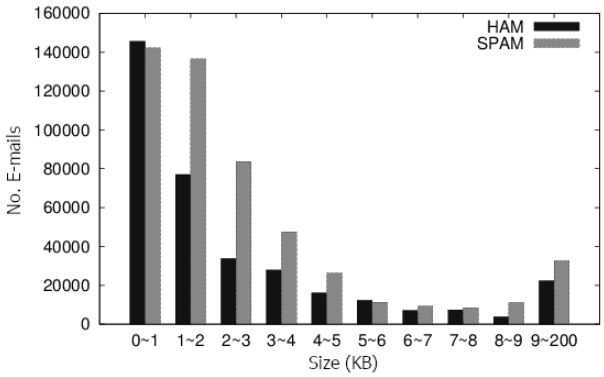

FIGURE 2. Sizes of the token files of the UNIFEI database.

was assigned to all vectors in the set. Finally, the real vectors $\vec{v}$ of dimensionality 1024 were exhibited in two dimensions ($\Re^2$), using the t-SNE technique [42].

Figure 3 graphically exhibits the e-mails of the UNIFEI database in two dimensions. In the figure, ham e-mails appear as blue dots, spam e-mails appear as red dots, and ham/spam e-mails appear as black dots.

Since e-mails were represented by real 1024-dimensional vectors, there is a high probability that equal vectors do, in fact, represent equal e-mails. Thus, by the amount of black dots in the Figure 3, it is clear that the UNIFEI database is highly inconsistent. The inconsistency of the database is due solely to the inconsistent classification of e-mails made by the anti-spam CanIt-PRO.

The inconsistent classification may have occurred for several reasons. Such reasons are difficult to discern, since CanIt-PRO, being a commercial anti-spam, does not have open source code. Probably, however, it can be assumed that inconsistent classifications were made at different points in time, during which the consulted blacklists were updated to include the addresses or domains of spammers.

To correct the inconsistency of the UNIFEI database, a consistency-generating tool was developed. The tool uses two integer constants — $\delta \in \mathbb{N}$ and $n \in \mathbb{N}^*$ — both defined by the Open-MaLBAS administrator. The first constant δ indicates both the degree of dissimilarity between e-mails and between vectors, for the e-mails are represented by tokens (Section IV-E) which, in turn, are represented by vector coordinates. For example, if the administrator defines δ to be zero, this means either that only e-mails that have the same tokens or that only vectors that have the same values in their coordinates are considered identical. If the administrator defines δ to be one or two, this means either that only e-mails that differ at most by one or two tokens or that only vectors that differ at most by one or two values of their coordinates, respectively, are considered identical. The second constant n indicates the dimensionality of the vectors.

The consistency-generating tool performs four steps. In the first, it verifies, through the analysis of their tokens, which are the e-mails identical to each other by the degree of dissimilarity δ (henceforth, δ-dissimilarity e-mails) and puts each group

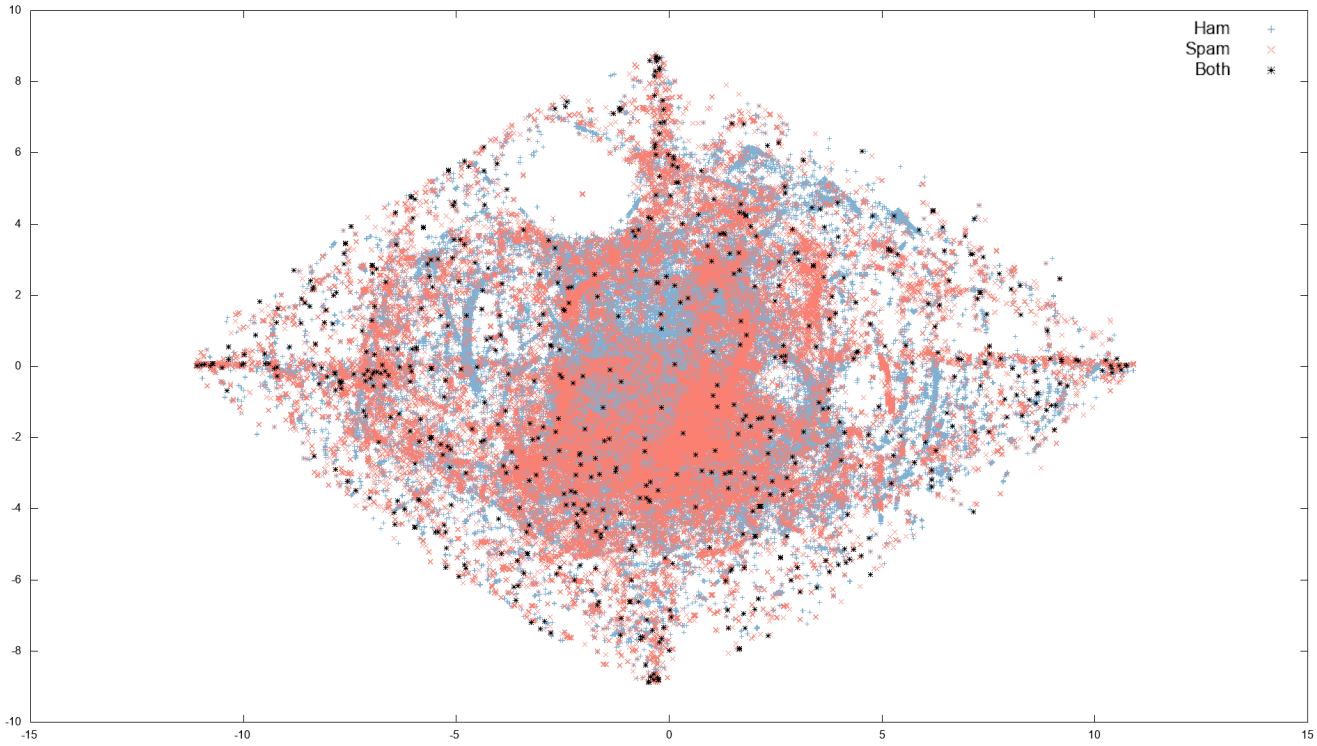

FIGURE 3. E-mails of the UNIFEI database — ham: blue; spam: red; ham/spam: black.

of δ-dissimilarity e-mails in a separate set. In the second, the dominant class (i.e., with the largest number of e-mails) of each set is determined, and then the value of that class is assigned to all e-mails in the set. For example, if a set of δ-dissimilarity e-mails contains 17 ham e-mails and 54 spam e-mails, the spam class is assigned to all 71 e-mails in the set. In the third, the Feature Selection Module is executed, in order to convert the e-mails from the representation by tokens to the representation by n-dimensional vectors. In the fourth and last step, the first and second steps are performed again, but this time, on the vectors obtained in the third step. It is important to note that this last step may change the class of vectors. This means that, indirectly, e-mails may change class again.

A new database was created from the UNIFEI database, using the value of the constant $\delta = 0$ in the consistency-generating tool. From Figure 4, it is possible to verify that the new database, called UNIFEI-δ0, is more consistent than the UNIFEI database. Both databases were used in the experiments. The UNIFEI database has 353,151 ham e-mails and 509,076 spam e-mails. The UNIFEI-δ0 database has 353,910 ham e-mails and 508,317 spam e-mails.

Tables 1 and 2 show the number of vectors generated by the FD and MI methods, respectively, on the UNIFEI and UNIFEI-δ0 databases. The vectors were generated with dimensionalities $n = 8, 16, 32, 64, 128, 256, 512, 1024$.

Based on the two tables, it can be seen that the FD method has the least amount of e-mail losses, that is, of null vectors, in all dimensionalities. This fact was expected, for FD selects the features that are more common in the e-mail sets. With 8-dimension vectors, there are, respectively, 546 (0.15%) and 12,831 (2.52%) of null ham and spam vectors. With 1024-dimension vectors, there are, respectively, 287 (0.08%) and 12,591 (2.47%) of null ham and spam vectors.

TABLE 1. Number of vectors in each set of UNIFEI database.

Dimensionality	Class	FD	MI
8	ham	353,365	301,234
	spam	495,487	431,983
16	ham	353,405	339,642
	spam	495,523	471,830
32	ham	353,416	339,736
	spam	495,523	471,916
64	ham	353,441	353,395
	spam	495,533	495,502
128	ham	353,448	353,427
	spam	495,538	495,502
256	ham	353,501	353,468
	spam	495,538	495,530
512	ham	353,582	353,503
	spam	495,566	495,530
1024	ham	353,624	353,503
	spam	495,727	495,530

In the MI method, with 8-dimension vectors, there are, respectively, 14.88% and 15.02% of null ham and spam vectors. With 1024-dimension vectors, however, there are, respectively, 0.12% and 2.52% of null ham and spam vectors, percentages very close to those of the FD method. In fact, from dimensionality 64 onwards, the MI method starts to generate a number of null vectors similar to that of the FD method.

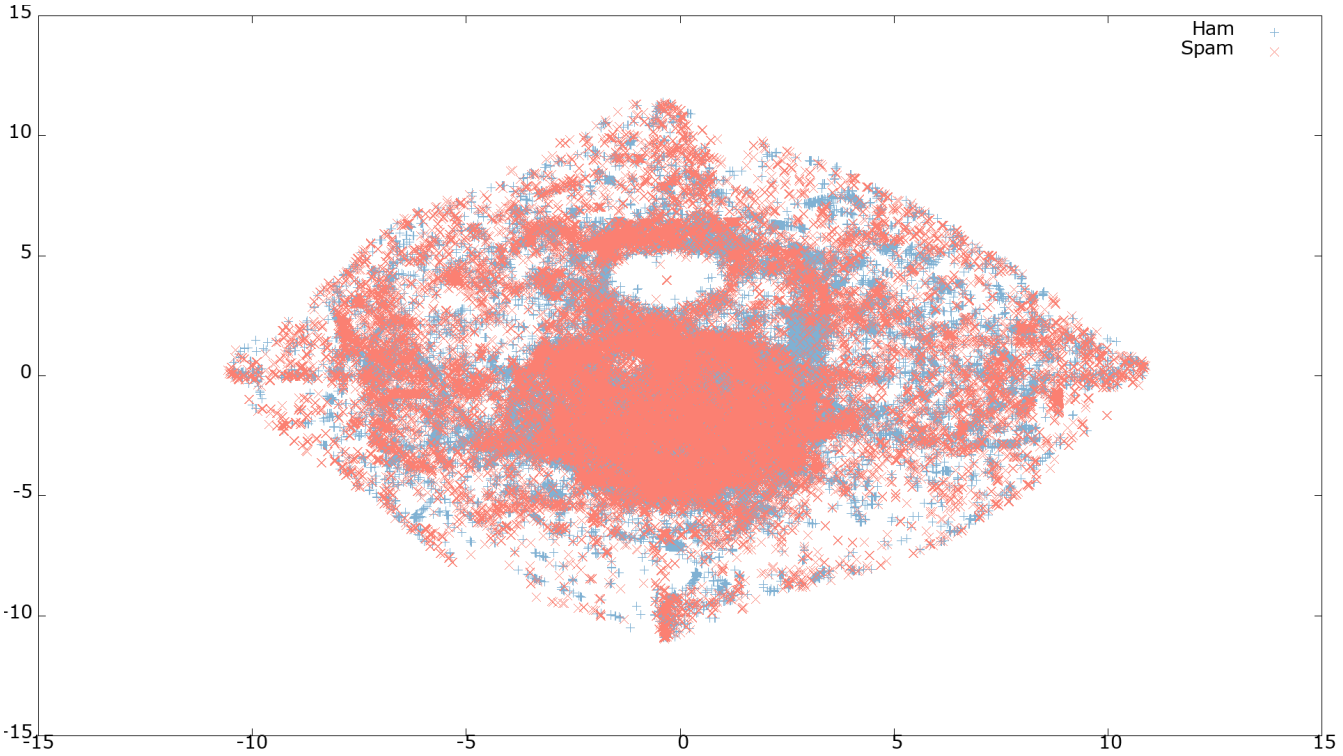

FIGURE 4. E-mails of the UNIFEI-δ0 database — ham: blue; spam: red.

TABLE 2. Number of vectors in each set of UNIFEI-δ0 database.

Dimensionality	Class	FD	MI
8	ham	347,446	102,806
	spam	501,406	630,411
16	ham	349,701	323,961
	spam	499,227	487,511
32	ham	355,024	319,521
	spam	493,915	492,131
64	ham	355,019	347,165
	spam	493,955	501,732
128	ham	355,284	356,422
	spam	493,702	492,507
256	ham	354,859	354,027
	spam	494,180	494,971
512	ham	354,579	354,041
	spam	494,569	494,992
1024	ham	353,868	353,906
	spam	495,483	495,127

Several sets of vectors were created from the UNIFEI and UNIFEI-δ0 databases for carrying out the experiments (Section VII). To create them, a two-step methodology was followed. The first step has already been described above. It consists in creating, through the two statistical methods of token selection — FD and MI —, sets of vectors with eight different dimensionalities — 8, 16, 32, 64, 128, 256, 512 and 1024. The second step consists in reducing, without significant loss of information, the dimensionalities of the vectors in the sets, through the use of a software tool, called dimensionality-reduction tool. This reduction of the dimensionality of the vectors in the sets (for example, from dimensionality 1024 to 208) allows Open-MaLBAS to significantly reduce the training time of its classifier models as well as the time needed to classify each e-mail. The implementation of the dimensionality-reduction tool was based on the Multi-Objective Evolutionary Feature Selection algorithm [43].

VI. METRICS

Precision and recall metrics were used to evaluate performance, in terms of e-mail classification, of the MLP and RF models. To calculate these metrics, the following variables are required:

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

Precision metrics measure the accuracy of the classification, given by the amount of false positives obtained. Three precision metrics — P(HAM), P(SPAM) and P(GEN.[5]) — were used to evaluate the two classifier models. They are calculated by the Equations (1), (2) and (3).

$$P(HAM) = \frac{n_{H\to H}}{n_{H\to H} + n_{S\to H}} \tag{1}$$

$$P(SPAM) = \frac{n_{S\to S}}{n_{S\to S} + n_{H\to S}} \tag{2}$$

$$P(GEN.) = \frac{N_{HAM} * P(HAM) + N_{SPAM} * P(SPAM)}{N_{HAM} + N_{SPAM}} \tag{3}$$

[5]GEN. is the abbreviation for GENERAL.

Recall metrics measure the completeness of the classification, given by the number of false negatives obtained. Three recall metrics — R(HAM), R(SPAM) and R(GEN.) — were used to evaluate the two classifier models. They are calculated by the Equations (4), (5) and (6).

$$R(HAM) = \frac{n_{H\to H}}{n_{H\to H} + n_{H\to S}} \tag{4}$$

$$R(SPAM) = \frac{n_{S\to S}}{n_{S\to S} + n_{S\to H}} \tag{5}$$

$$R(GEN.) = \frac{N_{HAM} * R(HAM) + N_{SPAM} * R(SPAM)}{N_{HAM} + N_{SPAM}} \tag{6}$$

VII. EXPERIMENTS

A. FIRST EXPERIMENT

The first experiment aimed to assess the performance of the two anti-spams — CanIt-PRO 9.2.4 and Open-MaLBAS. Performance was evaluated in terms of the time required to classify all e-mails from the UNIFEI database. Figure 5 shows the architecture used in the experiment.

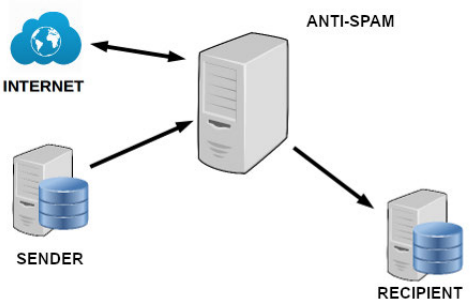

FIGURE 5. Architecture used in the experiment.

The architecture consists of three computers. The first one, on the left, sends e-mails to the second computer using a modified version of the MTA of the SMTP Module (Section IV-B), called MTA-X. MTA-X records the sending time of each e-mail in its *Subject* field, immediately before sending it. This computer has a 1.6 GHz dual-core processor, 2 GB of RAM, and runs the Linux Mint 17.2 operating system.

The second computer, in the center, runs the two anti-spams in turn. It uses either Postfix (if running CanIt-PRO) or SMTP Module (if running Open-MaLBAS) both to receive e-mails from the first computer and to send them to the third computer. The second computer has a processor with eight cores of 2.95 GHz and 4 GB of RAM. CanIt-PRO 9.2.4 is distributed as a package. When installing the package, a customized version of the Debian 6 operating system is also installed. Therefore, CanIt-PRO runs on Debian 6. Open-MaLBAS also runs on Debian 6, installed on another partition of the hard drive of the computer. In this way, a fair comparison between the two anti-spams is guaranteed, since both run, in turn, on the same computing platform.

The third computer, on the right, receives e-mails from the second computer through the MDA of the SMTP module (Section IV-B). It runs a modified version of the Backup Module (Section IV-D). The module receives each e-mail, already classified by one of the anti-spams, and records, in the same line of a file, both the time it was received and the time it was sent, both contained in the field *Subject* of the e-mail. The hardware configuration of the third computer is identical to that of the second one. However, it runs the Ubuntu 16.04 operating system.

The third computer also runs a Network Time Protocol (NTP) server [44]. The other two computers synchronize, through the NTP server, their times with the time of the third computer. Thus, the server keeps the times of the three computers equal, ensuring that both the sending and receiving times of the e-mails be recorded as accurately as possible, within an error range of approximately one millisecond.

The three computers are connected to each other through a router, forming a local network. This network is connected to the Internet, since CanIt-PRO depends on this connection, at least, to validate its license and to consult blacklists.

MTA-X, which runs on the first computer, made use of threads in order to allow it to send e-mails simultaneously. Four sending modes were evaluated. In the first mode, using only one thread, e-mails were sent individually, one at a time. In the second mode, using two threads, e-mails were sent simultaneously every two. In the third mode, using four threads, e-mails were sent simultaneously every four. In the fourth mode, using eight threads, e-mails were sent simultaneously every eight. The four sending modes are called T1, T2, T4 and T8, as they use one, two, four and eight threads, respectively.

The e-mails from the UNIFEI database were grouped into six sets, according to their sizes. E-mails between `1K–2K` went to the first set. E-mails between `2K–3K`, `3K–4K`, `4K–5K`, `10K–20K` and `20K–30K` went to the second, third, fourth, fifth and sixth sets, respectively. In the first four sets, the average processing time of e-mails whose sizes vary in the range of `1K` is evaluated. In the last two, the average processing time of e-mails whose sizes vary in the range of `10K` is evaluated. The other e-mails from the UNIFEI database were not used because, with the e-mails from the six sets, it was already possible to evaluate the average processing times of the two anti-spams.

The experiment produces, as a result, the time spent by each anti-spam to receive, classify and deliver the e-mails to the modified Backup Module. In this way, it is possible to evaluate not only the impact caused by the size of the e-mail, but also how fast each anti-spam processes the e-mails.

Tables 3 and 4 show the total time spent, in hours, minutes and seconds (HH:MM:SS), by CanIt-PRO and Open-MaLBAS, respectively, to receive, classify and deliver all e-mails, from each set, to the modified Backup Module.

Open-MaLBAS total times are up to 88% and 86.7% shorter than CanIt-PRO total times, using one and eight threads, respectively, to process the e-mails from the set `1K–2K`. Owing to the fact that CanIt-PRO does not have open source code, it is not possible to determine what reasons lead it to spend more time to process the e-mails.

TABLE 3. Total time (HH: MM: SS) spent by CanIt-PRO to process all e-mails in each set.

	1K-2K	2K-3K	3K-4K	4K-5K	10K-20K	20K-30K
T1	25:37:03	14:25:37	09:41:07	09:33:33	07:57:52	02:32:30
T2	13:47:42	08:01:48	05:31:54	05:31:01	04:02:32	01:17:20
T4	10:00:52	05:46:24	03:47:49	03:47:26	02:24:45	00:40:21
T8	08:04:35	04:26:46	02:53:47	02:54:59	01:47:08	00:24:43

TABLE 4. Total time (HH: MM: SS) spent by Open-MaLBAS to process all e-mails in each set.

	1K-2K	2K-3K	3K-4K	4K-5K	10K-20K	20K-30K
T1	03:02:47	02:01:25	01:31:36	01:32:28	00:56:26	00:15:05
T2	01:34:45	01:05:58	00:53:18	00:52:34	00:30:35	00:07:57
T4	01:09:04	00:54:15	00:44:55	00:44:51	00:25:00	00:05:22
T8	01:04:29	00:53:12	00:45:25	00:43:59	00:25:11	00:05:09

From the results, it can be seen that the total time to process all e-mails in each set decreases as the number of threads increases. This is due to the fact that when one e-mail is being processed by either of the two anti-spams, another may be being received and yet another may be being sent to the modified Backup Module. The reduction in the total processing time for each set of e-mails, however, is not linear, since, with the increase in the number of threads, there is also an increase in the simultaneous demand for non-shareable resources.

B. SECOND EXPERIMENT

The second experiment aimed to evaluate the two classifier models — MLP and RF — on the e-mails of the UNIFEI database. The experiment was performed on a computer with a 3.20 GHz four-core processor, 32 GB of memory, running the Linux Mint 18.3 operating system.

The methodology used to carry out the experiment consists of two steps. First, the set to be tested is chosen. For example, the set of 8-dimensional vectors, obtained by executing the FD method. This set has, according to Table 1, 353,365 ham e-mails and 495,487 spam e-mails. Second, the vectors from the set that will be used in the training and testing of the model are selected.

For the MLP classifier model, 40% and 20% of the vectors in the set, not including null vectors, are used in training and validation, respectively. The remaining 40%, including all null vectors, are used in the test. For the RF classifier model, 50% of the vectors in the set, not including null vectors, are used in the training and the remaining 50%, including all null vectors, are used in the test.[6]

For the MLP classifier model, the values 0.3, 0.2 and 5,000 were set to the parameters learning rate, momentum and maximum number of epochs, respectively. The number of artificial neural units in the input, first hidden, second hidden and output layers were $\mathtt{NF'}$, $(\mathtt{NF'}+2)/2$, $(\mathtt{NF'}+2)/4$ and 2, respectively. $\mathtt{NF'}$ is the dimensionality of the vectors in the sets after the execution of the dimensionality-reduction tool (Section V). For the RF classifier model, the number of trees built and of features considered were 100 and $\mathtt{NF'}$, respectively.

In the experiment, each result, as well as its confidence interval [45], was calculated from the average of ten runs of the classifier model. The T-test statistical significant test was used to calculate the confidence intervals. The confidence intervals were calculated with a 5% confidence level, that is, they are valid with 95% certainty.

Tables 5, 6, 7 and 8 show, respectively, the accuracy, in terms of precision and recall metrics (Section VI), in the classification of e-mails, training time and classification time of the MLP model. Similarly, Tables 9, 10, 11 and 12 present the values obtained by the RF model. In these eight tables, NF and $\mathtt{NF'}$ indicate, respectively, the dimensionality of the vectors in the sets before and after the execution of the dimensionality-reduction tool (Section V).

TABLE 5. Precision of the MLP model in e-mail classification.

Method	NF	NF´	P(HAM)	P(SPAM)	P(GEN.)
FD	8	5	46.88 ± 17.68	64.96 ± 5.69	55.93 ± 11.64
FD	16	3	58.68 ± 0.15	75.80 ± 0.17	67.25 ± 0.05
FD	32	6	60.41 ± 2.78	76.06 ± 4.38	68.24 ± 1.00
FD	64	22	71.30 ± 1.09	81.99 ± 2.14	76.65 ± 1.15
FD	128	65	75.48 ± 1.54	86.12 ± 2.00	80.81 ± 0.38
FD	256	51	75.00 ± 1.22	81.52 ± 0.84	78.26 ± 0.38
FD	512	101	75.39 ± 1.01	83.98 ± 1.21	79.69 ± 0.44
FD	1024	105	71.79 ± 5.48	80.53 ± 4.80	76.16 ± 0.74
MI	8	4	56.09 ± 1.28	76.23 ± 2.22	66.44 ± 0.53
MI	16	7	57.68 ± 0.13	77.99 ± 0.66	68.05 ± 0.39
MI	32	5	58.05 ± 0.12	77.10 ± 0.55	67.78 ± 0.30
MI	64	9	63.30 ± 0.73	81.42 ± 2.22	72.36 ± 0.77
MI	128	18	67.98 ± 5.83	81.58 ± 6.34	74.78 ± 0.41
MI	256	31	71.30 ± 0.94	84.95 ± 1.85	78.13 ± 0.50
MI	512	125	74.69 ± 1.50	85.59 ± 2.04	80.15 ± 0.37
MI	1024	208	75.43 ± 2.03	88.10 ± 2.45	81.78 ± 0.48

From the Tables 5 and 6, it can be seen that the best result — 81.78% under the precision metric and 80.26% under the recall metric — of the MLP model was obtained with the set created by the MI method, with 1024-dimensional vectors reduced to 208-dimensional vectors through the dimensionality-reduction tool. Its second best result — 80.81% under the precision metric and 79.73% under the recall metric — was obtained with the set created by the FD method, with 128-dimensional vectors reduced to 65-dimensional vectors. In turn, from the Tables 9 and 10, it can be seen that the best result — 94.60% under the precision metric and 94.42% under the recall metric — of the RF model was obtained with the set created by the FD method, with 128-dimensional vectors reduced to 65-dimensional vectors through the dimensionality-reduction tool. Its second best result — 93.89% under the precision metric and 93.71% under the recall metric — was obtained with the set created by the FD method, with 64-dimensional vectors reduced to 22-dimensional vectors. Therefore, in terms of accuracy in the classification of e-mails, the RF model produced better results than those produced by the MLP model.

From the Table 11, it can be seen that the worst training time of the RF model — 11 minutes and 46 seconds — was obtained with the set created by the MI method, with 1024-dimensional vectors reduced to 208-dimensional vectors. This time, however, is considerably short, compared to the worst training time of the MLP model (Table 7) — 2 hours, 11 minutes and 27 seconds — also obtained on the same set. The significant difference between the

[6]Unlike the MLP, the RF does not require a validation set for its training.

TABLE 6. Recall of the MLP model in e-mail classification.

Method	NF	NF´	R(HAM)	R(SPAM)	R(GEN.)
FD	8	5	64.00 ± 24.19	54.86 ± 17.15	59.42 ± 3.55
FD	16	3	87.72 ± 0.30	38.37 ± 0.60	63.01 ± 0.15
FD	32	6	84.41 ± 9.71	43.22 ± 10.05	63.79 ± 0.69
FD	64	22	85.36 ± 2.45	65.67 ± 2.29	75.50 ± 0.99
FD	128	65	88.27 ± 2.38	71.22 ± 3.17	79.73 ± 0.57
FD	256	51	83.53 ± 1.43	72.14 ± 2.27	77.83 ± 0.56
FD	512	101	86.17 ± 1.54	71.89 ± 1.92	79.02 ± 0.47
FD	1024	105	82.23 ± 10.12	65.31 ± 8.42	73.76 ± 0.97
MI	8	4	88.15 ± 4.39	34.32 ± 5.83	60.49 ± 0.86
MI	16	7	88.93 ± 0.44	37.52 ± 0.29	62.67 ± 0.21
MI	32	5	87.80 ± 0.51	39.27 ± 0.44	63.00 ± 0.16
MI	64	9	88.65 ± 2.62	48.53 ± 3.03	68.58 ± 0.29
MI	128	18	83.74 ± 14.64	56.84 ± 10.41	70.28 ± 2.26
MI	256	31	88.38 ± 1.95	64.43 ± 2.45	76.39 ± 0.37
MI	512	125	87.93 ± 2.50	70.11 ± 3.13	79.00 ± 0.36
MI	1024	208	90.14 ± 3.00	70.42 ± 3.95	80.26 ± 0.76

TABLE 7. Training time of the MLP model.

Method	NF	NF´	Training time (MLP)
FD	8	5	00:00:57.230 ± 00:00:19.496
FD	16	3	00:07:36.711 ± 00:07:08.330
FD	32	6	00:01:39.661 ± 00:01:20.227
FD	64	22	00:05:53.577 ± 00:01:48.272
FD	128	65	00:19:09.338 ± 00:03:33.137
FD	256	51	00:15:13.454 ± 00:03:37.718
FD	512	101	00:59:25.592 ± 00:21:14.650
FD	1024	105	00:38:45.791 ± 00:02:27.502
MI	8	4	00:12:45.004 ± 00:07:36.121
MI	16	7	00:02:00.906 ± 00:01:08.805
MI	32	5	00:01:59.880 ± 00:01:03.761
MI	64	9	00:03:39.381 ± 00:01:42.241
MI	128	18	00:05:11.680 ± 00:01:27.164
MI	256	31	00:11:24.748 ± 00:02:38.636
MI	512	125	00:50:57.477 ± 00:02:52.817
MI	1024	208	02:11:27.315 ± 00:08:38.977

TABLE 8. Classification time of the MLP model.

Method	NF	NF´	Classification time (MLP)
FD	8	5	00:00:00.438 ± 00:00:00.014
FD	16	3	00:00:00.383 ± 00:00:00.022
FD	32	6	00:00:00.539 ± 00:00:00.017
FD	64	22	00:00:01.631 ± 00:00:00.033
FD	128	65	00:00:08.263 ± 00:00:00.076
FD	256	51	00:00:05.505 ± 00:00:00.070
FD	512	101	00:00:18.106 ± 00:00:00.108
FD	1024	105	00:00:19.570 ± 00:00:00.071
MI	8	4	00:00:00.444 ± 00:00:00.029
MI	16	7	00:00:00.565 ± 00:00:00.018
MI	32	5	00:00:00.454 ± 00:00:00.016
MI	64	9	00:00:00.625 ± 00:00:00.023
MI	128	18	00:00:01.292 ± 00:00:00.028
MI	256	31	00:00:02.579 ± 00:00:00.030
MI	512	125	00:00:27.160 ± 00:00:00.017
MI	1024	208	00:01:11.920 ± 00:00:00.065

training times of the two models is due to the fact that, on WEKA, the implementation of the RF model is multi-threaded, whereas that of the MLP model is not. Therefore, in terms of the time required for training the models, the RF model, once again, produced better results than those produced by the MLP model.

Finally, from Tables 8 and 12, it can be seen that the worst classification time for MLP and RF models was, respectively, 1 minute and 11 seconds and 2 minutes and 20 seconds. Therefore, in terms of the time required to classify e-mails, the MLP model produced better results than those produced by the RF model.

TABLE 9. Precision of the RF model in e-mail classification.

Method	NF	NF´	P(HAM)	P(SPAM)	P(GEN.)
FD	8	5	90.94 ± 0.06	88.91 ± 0.05	89.92 ± 0.03
FD	16	3	63.39 ± 0.02	92.63 ± 0.08	78.03 ± 0.04
FD	32	6	80.13 ± 0.05	90.26 ± 0.11	85.20 ± 0.04
FD	64	22	91.03 ± 0.04	96.75 ± 0.03	93.89 ± 0.02
FD	128	65	91.78 ± 0.03	97.41 ± 0.02	94.60 ± 0.01
FD	256	51	88.99 ± 0.06	96.69 ± 0.07	92.85 ± 0.02
FD	512	101	90.66 ± 0.03	97.12 ± 0.02	93.89 ± 0.02
FD	1024	105	85.81 ± 0.04	97.03 ± 0.04	91.43 ± 0.02
MI	8	4	62.93 ± 0.43	71.81 ± 0.24	67.49 ± 0.33
MI	16	7	60.46 ± 0.02	94.21 ± 0.10	77.70 ± 0.06
MI	32	5	60.58 ± 0.03	94.90 ± 0.07	78.12 ± 0.04
MI	64	9	74.98 ± 0.07	90.05 ± 0.12	82.52 ± 0.04
MI	128	18	70.40 ± 0.04	95.38 ± 0.03	82.90 ± 0.02
MI	256	31	82.03 ± 0.07	94.71 ± 0.06	88.38 ± 0.03
MI	512	125	88.73 ± 0.02	97.12 ± 0.03	92.93 ± 0.02
MI	1024	208	90.44 ± 0.04	96.94 ± 0.04	93.70 ± 0.02

TABLE 10. Recall of the RF model in e-mail classification.

Method	NF	NF´	R(HAM)	R(SPAM)	R(GEN.)
FD	8	5	88.60 ± 0.06	91.20 ± 0.07	89.90 ± 0.03
FD	16	3	96.45 ± 0.05	44.43 ± 0.06	70.41 ± 0.02
FD	32	6	91.63 ± 0.11	77.33 ± 0.09	84.47 ± 0.02
FD	64	22	96.95 ± 0.03	90.47 ± 0.04	93.71 ± 0.02
FD	128	65	97.57 ± 0.02	91.29 ± 0.02	94.42 ± 0.01
FD	256	51	96.98 ± 0.07	88.04 ± 0.08	92.50 ± 0.02
FD	512	101	97.32 ± 0.02	90.00 ± 0.03	93.66 ± 0.02
FD	1024	105	97.42 ± 0.03	83.94 ± 0.05	90.67 ± 0.03
MI	8	4	76.12 ± 0.04	57.57 ± 0.76	66.59 ± 0.38
MI	16	7	97.50 ± 0.05	38.94 ± 0.05	67.58 ± 0.03
MI	32	5	97.80 ± 0.03	39.11 ± 0.05	67.81 ± 0.02
MI	64	9	92.34 ± 0.11	69.21 ± 0.13	80.77 ± 0.03
MI	128	18	97.13 ± 0.02	59.21 ± 0.05	78.16 ± 0.03
MI	256	31	95.57 ± 0.06	79.13 ± 0.08	87.34 ± 0.03
MI	512	125	97.40 ± 0.03	87.67 ± 0.04	92.52 ± 0.02
MI	1024	208	97.16 ± 0.04	89.77 ± 0.06	93.46 ± 0.02

TABLE 11. Training time of the RF model.

Method	NF	NF´	Training time (RF)
FD	8	5	00:00:30.808 ± 00:00:00.333
FD	16	3	00:00:29.647 ± 00:00:00.528
FD	32	6	00:00:36.723 ± 00:00:00.455
FD	64	22	00:01:09.291 ± 00:00:00.677
FD	128	65	00:02:39.321 ± 00:00:00.865
FD	256	51	00:03:12.799 ± 00:00:01.835
FD	512	101	00:04:29.471 ± 00:00:01.929
FD	1024	105	00:07:26.325 ± 00:00:02.502
MI	8	4	00:00:17.138 ± 00:00:00.728
MI	16	7	00:00:34.198 ± 00:00:00.842
MI	32	5	00:00:18.094 ± 00:00:00.845
MI	64	9	00:00:52.094 ± 00:00:00.871
MI	128	18	00:01:14.909 ± 00:00:01.037
MI	256	31	00:02:35.688 ± 00:00:01.113
MI	512	125	00:08:00.978 ± 00:00:02.845
MI	1024	208	00:11:46.847 ± 00:00:03.511

C. THIRD EXPERIMENT

The third experiment aimed to evaluate the two classifier models — MLP and RF — on the e-mails of the UNIFEI-δ0 database. The experiment was carried out on the same computer used in the second experiment (Section VII-B). The parameters values of the MLP and RF models and the methodology used for the execution of the experiment (i.e., creation of the vector sets of the UNIFEI-δ0 database, selection of the vectors of the training and test sets, number of runs performed, and calculation of confidence intervals) are also the same used in the second experiment.

TABLE 12. Classification time of the RF model.

Method	NF	NF′	Classification time (RF)
FD	8	5	00:00:42.935 ± 00:00:03.355
FD	16	3	00:00:27.233 ± 00:00:01.045
FD	32	6	00:00:46.636 ± 00:00:03.520
FD	64	22	00:00:50.898 ± 00:00:01.471
FD	128	65	00:00:58.227 ± 00:00:02.559
FD	256	51	00:01:25.623 ± 00:00:03.702
FD	512	101	00:01:26.540 ± 00:00:01.037
FD	1024	105	00:01:57.638 ± 00:00:02.656
MI	8	4	00:00:19.674 ± 00:00:01.366
MI	16	7	00:00:29.684 ± 00:00:02.517
MI	32	5	00:00:19.125 ± 00:00:00.904
MI	64	9	00:00:44.231 ± 00:00:04.190
MI	128	18	00:00:42.425 ± 00:00:01.413
MI	256	31	00:01:05.949 ± 00:00:02.598
MI	512	125	00:01:49.900 ± 00:00:02.148
MI	1024	208	00:02:20.178 ± 00:00:02.078

TABLE 13. Precision of the MLP model in e-mail classification.

Method	NF	NF′	P(HAM)	P(SPAM)	P(GEN.)
FD	8	5	60.36 ± 0.34	73.94 ± 2.20	67.21 ± 0.97
FD	16	3	51.97 ± 13.06	76.30 ± 6.58	64.25 ± 9.76
FD	32	8	64.24 ± 4.88	73.70 ± 5.04	69.03 ± 1.37
FD	64	11	65.15 ± 1.06	78.28 ± 1.05	71.80 ± 0.82
FD	128	24	75.31 ± 1.41	81.39 ± 1.11	78.39 ± 0.64
FD	256	60	73.74 ± 0.66	85.35 ± 1.38	79.62 ± 0.38
FD	512	79	75.24 ± 2.69	83.16 ± 1.78	79.25 ± 0.84
FD	1024	77	71.64 ± 1.51	83.78 ± 1.71	77.78 ± 0.35
MI	8	3	80.57 ± 3.37	77.63 ± 3.80	79.12 ± 0.21
MI	16	3	69.87 ± 0.05	79.75 ± 0.07	74.93 ± 0.04
MI	32	7	76.78 ± 2.84	77.94 ± 0.23	77.37 ± 1.35
MI	64	10	66.97 ± 6.19	78.34 ± 6.00	72.70 ± 0.35
MI	128	30	70.28 ± 0.63	84.25 ± 2.44	77.36 ± 1.09
MI	256	49	74.34 ± 3.88	81.48 ± 3.76	77.96 ± 0.62
MI	512	118	73.06 ± 1.19	85.89 ± 1.57	79.56 ± 0.34
MI	1024	155	77.22 ± 2.90	85.63 ± 1.70	81.48 ± 0.71

TABLE 14. Recall of the MLP model in e-mail classification.

Method	NF	NF′	R(HAM)	R(SPAM)	R(GEN.)
FD	8	5	83.12 ± 2.93	46.28 ± 2.61	64.55 ± 0.26
FD	16	3	81.52 ± 20.51	41.48 ± 14.72	61.31 ± 2.72
FD	32	8	76.92 ± 11.79	54.74 ± 15.86	65.69 ± 3.55
FD	64	11	83.96 ± 1.23	56.19 ± 2.39	69.89 ± 0.99
FD	128	24	82.67 ± 1.75	73.48 ± 2.36	78.02 ± 0.72
FD	256	60	87.64 ± 1.63	69.56 ± 1.64	78.48 ± 0.11
FD	512	79	84.67 ± 2.69	72.43 ± 4.92	78.47 ± 1.36
FD	1024	77	86.63 ± 2.59	66.33 ± 3.15	76.36 ± 0.45
MI	8	3	76.47 ± 6.60	80.13 ± 5.75	78.28 ± 0.52
MI	16	3	82.41 ± 0.07	66.11 ± 0.06	74.06 ± 0.03
MI	32	7	77.18 ± 1.24	77.29 ± 3.59	77.24 ± 1.25
MI	64	10	80.45 ± 16.01	56.73 ± 10.53	68.49 ± 2.65
MI	128	30	87.47 ± 2.96	63.97 ± 2.01	75.55 ± 0.68
MI	256	49	82.35 ± 7.36	71.07 ± 6.28	76.64 ± 1.09
MI	512	118	88.38 ± 1.84	68.13 ± 2.58	78.12 ± 0.49
MI	1024	155	86.92 ± 2.42	74.49 ± 5.31	80.63 ± 1.58

Tables 13, 14, 15 and 16 show, respectively, the accuracy, in terms of precision and recall metrics (Section VI), in the classification of e-mails, training time and classification time of the MLP model. Similarly, Tables 17, 18, 19 and 20 present the values obtained by the RF model. In these eight tables, NF and NF′ indicate, respectively, the dimensionality of the vectors in the sets before and after the execution of the dimensionality-reduction tool (Section V).

From the Tables 13 and 14, it can be seen that the best result — 81.48% under the precision metric and 80.63% under the recall metric — of the MLP model was obtained with the set created by the MI method, with 1024-dimensional vectors reduced to 155-dimensional vectors through the dimensionality-reduction tool. Its second best result — 79.62% under the precision metric and 78.48% under the recall metric — was obtained with the set created by the FD method, with 256-dimensional vectors reduced to 60-dimensional vectors. In turn, from the Tables 17 and 18, it can be seen that the best result — 98.13% under both the precision and recall metrics — of the RF model was obtained with the set created by the FD method, with 8-dimensional vectors reduced to 5-dimensional vectors through the dimensionality-reduction tool. Its second best result — 94.64% under the precision metric and 94.30% under the recall metric — was obtained with the set created by the FD method, with 512-dimensional vectors reduced to

TABLE 15. Training time of the MLP model.

Method	NF	NF′	Training time (MLP)
FD	8	5	00:00:52.779 ± 00:00:37.226
FD	16	3	00:01:06.080 ± 00:00:30.760
FD	32	8	00:01:29.640 ± 00:00:45.851
FD	64	11	00:02:52.848 ± 00:01:11.867
FD	128	24	00:08:22.381 ± 00:02:44.154
FD	256	60	00:21:29.634 ± 00:05:45.896
FD	512	79	00:27:28.849 ± 00:04:23.578
FD	1024	77	00:26:09.283 ± 00:03:28.547
MI	8	3	00:00:58.559 ± 00:00:33.555
MI	16	3	00:00:43.803 ± 00:00:21.798
MI	32	7	00:01:06.504 ± 00:00:17.764
MI	64	10	00:02:05.908 ± 00:00:50.601
MI	128	30	00:13:21.549 ± 00:03:25.748
MI	256	49	00:17:22.840 ± 00:02:22.171
MI	512	118	00:54:48.178 ± 00:14:06.396
MI	1024	155	01:43:32.567 ± 00:09:52.741

TABLE 16. Classification time of the MLP model.

Method	NF	NF′	Classification time (MLP)
FD	8	5	00:00:00.453 ± 00:00:00.031
FD	16	3	00:00:00.379 ± 00:00:00.001
FD	32	8	00:00:00.598 ± 00:00:00.019
FD	64	11	00:00:00.742 ± 00:00:00.018
FD	128	24	00:00:01.782 ± 00:00:00.022
FD	256	60	00:00:07.205 ± 00:00:00.096
FD	512	79	00:00:11.703 ± 00:00:00.146
FD	1024	77	00:00:11.120 ± 00:00:00.189
MI	8	3	00:00:00.572 ± 00:00:00.045
MI	16	3	00:00:00.383 ± 00:00:00.035
MI	32	7	00:00:00.545 ± 00:00:00.027
MI	64	10	00:00:00.762 ± 00:00:00.037
MI	128	30	00:00:02.500 ± 00:00:00.039
MI	256	49	00:00:05.198 ± 00:00:00.190
MI	512	118	00:00:24.309 ± 00:00:00.070
MI	1024	155	00:00:40.818 ± 00:00:00.661

TABLE 17. Precision of the RF model in e-mail classification.

Method	NF	NF′	P(HAM)	P(SPAM)	P(GEN.)
FD	8	5	97.79 ± 0.01	98.47 ± 0.02	98.13 ± 0.01
FD	16	3	70.30 ± 0.07	89.24 ± 0.17	79.86 ± 0.07
FD	32	8	88.07 ± 0.07	94.22 ± 0.10	91.18 ± 0.02
FD	64	11	75.85 ± 0.03	94.17 ± 0.10	85.13 ± 0.04
FD	128	24	86.40 ± 0.02	97.53 ± 0.04	92.04 ± 0.02
FD	256	60	90.66 ± 0.03	98.36 ± 0.03	94.56 ± 0.01
FD	512	79	90.68 ± 0.03	98.48 ± 0.02	94.64 ± 0.02
FD	1024	77	89.14 ± 0.05	97.20 ± 0.07	93.22 ± 0.02
MI	8	3	96.77 ± 0.02	72.16 ± 0.03	84.63 ± 0.02
MI	16	3	94.42 ± 0.05	89.18 ± 0.03	91.74 ± 0.02
MI	32	7	97.01 ± 0.02	90.89 ± 0.02	93.89 ± 0.01
MI	64	10	68.00 ± 0.05	96.78 ± 0.04	82.51 ± 0.02
MI	128	30	85.74 ± 0.07	94.75 ± 0.12	90.31 ± 0.04
MI	256	49	86.10 ± 0.04	97.23 ± 0.04	91.73 ± 0.01
MI	512	118	90.38 ± 0.03	98.09 ± 0.02	94.28 ± 0.01
MI	1024	155	89.90 ± 0.03	98.50 ± 0.03	94.25 ± 0.01

TABLE 18. Recall of the RF model in e-mail classification.

Method	NF	NF´	R(HAM)	R(SPAM)	R(GEN.)
FD	8	5	98.45 ± 0.02	97.81 ± 0.01	98.13 ± 0.01
FD	16	3	92.41 ± 0.15	61.70 ± 0.15	76.91 ± 0.04
FD	32	8	94.49 ± 0.11	87.52 ± 0.09	90.96 ± 0.01
FD	64	11	95.52 ± 0.08	70.37 ± 0.06	82.78 ± 0.03
FD	128	24	97.79 ± 0.03	85.00 ± 0.04	91.31 ± 0.02
FD	256	60	98.46 ± 0.03	90.12 ± 0.03	94.23 ± 0.01
FD	512	79	98.57 ± 0.02	90.14 ± 0.03	94.30 ± 0.02
FD	1024	77	97.39 ± 0.06	88.42 ± 0.06	92.85 ± 0.01
MI	8	3	63.25 ± 0.05	97.83 ± 0.01	80.31 ± 0.03
MI	16	3	87.91 ± 0.04	95.05 ± 0.05	91.56 ± 0.02
MI	32	7	89.83 ± 0.03	97.34 ± 0.02	93.66 ± 0.01
MI	64	10	98.15 ± 0.03	54.60 ± 0.14	76.19 ± 0.05
MI	128	30	95.18 ± 0.12	84.60 ± 0.11	89.82 ± 0.03
MI	256	49	97.53 ± 0.03	84.65 ± 0.04	91.01 ± 0.01
MI	512	118	98.21 ± 0.02	89.81 ± 0.03	93.95 ± 0.01
MI	1024	155	98.60 ± 0.03	89.20 ± 0.03	93.84 ± 0.01

TABLE 19. Training time of the RF model.

Method	NF	NF´	Training time (RF)
FD	8	5	00:00:34.150 ± 00:00:00.283
FD	16	3	00:00:29.915 ± 00:00:00.372
FD	32	8	00:00:49.848 ± 00:00:00.486
FD	64	11	00:01:24.405 ± 00:00:02.222
FD	128	24	00:01:58.231 ± 00:00:01.367
FD	256	60	00:04:14.807 ± 00:00:02.097
FD	512	79	00:05:30.383 ± 00:00:01.690
FD	1024	77	00:05:19.756 ± 00:00:01.987
MI	8	3	00:00:18.330 ± 00:00:00.179
MI	16	3	00:00:26.374 ± 00:00:00.191
MI	32	7	00:00:45.323 ± 00:00:00.277
MI	64	10	00:01:23.869 ± 00:00:02.673
MI	128	30	00:02:52.458 ± 00:00:00.980
MI	256	49	00:04:03.581 ± 00:00:02.118
MI	512	118	00:07:41.898 ± 00:00:02.527
MI	1024	155	00:13:30.913 ± 00:00:10.112

TABLE 20. Classification time of the RF model.

Method	NF	NF´	Classification time (RF)
FD	8	5	00:00:38.227 ± 00:00:03.353
FD	16	3	00:00:30.882 ± 00:00:03.135
FD	32	8	00:00:46.834 ± 00:00:06.070
FD	64	11	00:01:01.692 ± 00:00:07.017
FD	128	24	00:01:00.205 ± 00:00:05.707
FD	256	60	00:01:27.565 ± 00:00:05.863
FD	512	79	00:01:28.888 ± 00:00:02.975
FD	1024	77	00:01:29.501 ± 00:00:02.371
MI	8	3	00:00:10.758 ± 00:00:00.829
MI	16	3	00:00:27.742 ± 00:00:03.525
MI	32	7	00:00:41.540 ± 00:00:03.053
MI	64	10	00:00:50.739 ± 00:00:02.594
MI	128	30	00:01:22.403 ± 00:00:04.564
MI	256	49	00:01:31.318 ± 00:00:08.911
MI	512	118	00:01:40.479 ± 00:00:01.903
MI	1024	155	00:02:19.988 ± 00:00:09.063

79-dimensional vectors. Therefore, in terms of accuracy in the classification of e-mails, the RF model produced better results than those produced by the MLP model.

From the Table 19, it can be seen that the worst training time of the RF model — 13 minutes and 30 seconds — was obtained with the set created by the MI method, with 1024-dimensional vectors reduced to 155-dimensional vectors. This time, however, is considerably short, compared to the worst training time of the MLP model (Table 15) — 1 hour, 43 minutes and 32 seconds — also obtained on the same set. As in the second experiment (Section VII-B), the significant difference between the training times of the two models is due to the fact that, on WEKA, the implementation of the RF model is multithreaded, while that of the MLP model is not. Therefore, in terms of the time required for training the models, the RF model, once again, produced better results than those produced by the MLP model.

From Tables 16 and 20, it can be seen that the worst classification time for MLP and RF models was, respectively, 40 seconds and 2 minutes and 20 seconds. Therefore, in terms of the time required to classify e-mails, the MLP model produced better results than those produced by the RF model.

Finally, by comparing the results obtained in this third experiment with those obtained in the second experiment (Section VII-B), it is possible to conclude the importance of data treatment. With the reduction of the inconsistency of the UNIFEI database, the MLP model maintained the accuracy in the classification of e-mails. However, the RF model produced more accurate results. With the MLP model, the accuracy practically remained from 81.78% to 81.48% under the precision metric and from 80.26% to 80.63% under the recall metric. With the RF model, the accuracy increased from 94.60% to 98.13% under the precision metric and from 94.42% to 98.13% under the recall metric.

VIII. CONCLUSION

This paper introduces a novel anti-spam — Open-MaLBAS. Unlike commercial and open-source anti-spams, the Open-MaLBAS does not make use of blacklists on the Internet and of SMTP extensions in order not to suffer the disadvantages of them both. Instead, it makes use of machine learning models for e-mail classification.

Open-MaLBAS is an anti-spam for e-mail servers. It has a modular architecture. Its implementation is based on design patterns in language Java. The implementation took into consideration its computational performance.

The source code of Open-MaLBAS is clear, simple and easily maintainable. All comments and documentation (JavaDocs) are written in English. In addition, all messages issued by Open-MaLBAS are saved in files in order to facilitate their translation to other languages. The source code is licensed under the GNU general public license version 3 [15], for free use. It is available in GitHub [16].

Open-MaLBAS was compared to commercial anti-spam CanIt-PRO 9.2.4. Through the results of the experiments, it was observed that the average processing time of each e-mail spent by Open-MaLBAS is almost ten times less than that of CanIt-PRO. It was also observed that the machine learning model Random Forest, from the Classification Module of Open-MaLBAS, was able to learn how to correctly classify the e-mail database previously classified by CanIt-PRO. Even though this database is inconsistent, owing to the inconsistent classifications of its e-mails carried out by CanIt-PRO, the Random Forest model managed to classify the e-mail database with accuracy of 94.60% under the precision metric and 94.4% under the recall metric. With the reduction of inconsistency of the database, the Random Forest model produced even more accurate results — 98.13% under both the precision and recall metrics.

Three directions for future work may be proposed. First, the implementation of a graphical user interface for the configuration of Open-MaLBAS, since its configuration is currently done through configuration files. Second, the inclusion of the image anti-spam, developed by Carpinteiro *et al.* [46], as a new Open-MaLBAS module, since Open-MaLBAS is currently only a text anti-spam. Finally, the test of other machine learning models, made available by the open-source library Weka [38], on the e-mail database classified by CanIt-PRO and on public e-mail databases. This test has already started and its results will be reported, shortly, in a new paper.

REFERENCES

[1] *Global Spam Volume*, Statista Company. Accessed: Aug. 2021. [Online]. Available: https://www.statista.com/statistics/420391/spam-email-traffic-share/

[2] J. Levine, *DNS Blacklists and Whitelists*, Internet Engineering Task Force, document RFC 5782, Feb. 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5782

[3] M. Kucherawy and D. Crocker, *Email Greylisting: An Applicability Statement for SMTP*, Internet Engineering Task Force, document RFC 6647, Jun. 2012. [Online]. Available: https://www.rfc-editor.org/info/rfc6647

[4] I. Harbarczyk. (Feb. 2020). *Email Blacklist: What is it and How to Avoid it in 2020?*. [Online]. Available: https://www.usebouncer.com/email-blacklist-what-is-it-and-how-to-avoid-it-in-2020/

[5] G. Croce and A. Killeen. *How to Avoid Email Domain Blacklist Remove Your IP From One*. Accessed: Aug. 2021. [Online]. Available: https://hubsell.com/insights/how-to-avoid-email-domain-blacklist-and-remove-ip/

[6] J. Huckaby. (Jan. 2019). *The 8 Email Blacklists You Should Actually Care About*. [Online]. Available: https://www.rackaid.com/blog/email-blacklists/

[7] B. Gardiner. (Oct. 2020). *Why do IP Addresses Get Blacklisted?*. [Online]. Available: https://www.fasthosts.co.uk/blog/why-do-ip-addresses-get-blacklisted-2/

[8] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification, STD 86*, Internet Engineering Task Force, document RFC 8200, Jul. 2017. [Online]. Available: https://www.rfc-editor.org/info/rfc8200

[9] J. Klensin, *SMTP 521 and 556 Reply Codes*, Internet Engineering Task Force, document RFC 7504, Jun. 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7504

[10] S. Kitterman, *Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1*, Internet Engineering Task Force, document RFC 7208, Apr. 2014. [Online]. Available: https://www.rfc-editor.org/info/rfc7208

[11] D. Crocker, T. Hansen, and M. Kucherawy, *Domain Keys Identified Mail (DKIM) Signatures*, Internet Engineering Task Force, document RFC 6376, Sep. 2011. [Online]. Available: https://www.rfc-editor.org/info/rfc6376

[12] M. Kucherawy and E. Zwicky, *Domain-Based Message Authentication, Reporting, Conformance (DMARC)*, Internet Engineering Task Force, document RFC 7489, Mar. 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7489

[13] J. Chen, V. Paxson, and J. Jiang, "Composition kills: A case study of email sender authentication," in *Proc. USENIX Secur. Symp.*, 2020, pp. 2183–2199. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/chen-jianjun

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

[15] *GNU Licenses*. Free Software Foundation. Accessed: Aug. 2021. [Online]. Available: http://www.gnu.org/licenses/licenses.en.html

[16] *Open Machine-Learning-Based Anti-Spam*. GitHub. Accessed: Aug. 2021. [Online]. Available: https://github.com/Isaac44/Open-MaLBAS

[17] *The Apache Spam Assassin Project*. Apache Software Foundation. Accessed: Aug. 2021. [Online]. Available: https://spamassassin.apache.org/

[18] *Anti Spam SMTP Proxy*. Sourceforge Platform. Accessed: Aug. 2021. [Online]. Available: http://www.thockar.com/assp-home/

[19] *Qpsmtpd*. GitHub Platform. Accessed: Aug. 2021. [Online]. Available: http://smtpd.github.io/qpsmtpd/

[20] *Bitdefender Antivirus*. Bitdefender. Accessed: Aug. 2021. [Online]. Available: https://www.bitdefender.com/

[21] *ClamAV Antivirus*. ClamAV. Accessed: Aug. 2021. [Online]. Available: https://www.clamav.net/

[22] D. J. Bernstein. *Qmail*. Accessed: Aug. 2021. [Online]. Available: https://cr.yp.to/qmail.html

[23] W. Z. Venema. *Postfix*. Accessed: Aug. 2021. [Online]. Available: http://www.postfix.org/

[24] *Exim Internet Mailer*. University of Cambridge, U.K. Accessed: Aug. 2021. [Online]. Available: https://www.exim.org/index.html

[25] *Barracuda Email Security Gateway*. Barracuda Networks. Accessed: Aug. 2021. [Online]. Available: https://www.barracuda.com/products/emailsecuritygateway

[26] A. Bhardwaj, V. Sapra, A. Kumar, N. Kumar, and S. Arthi, "Why is phishing still successful?" *Comput. Fraud Secur.*, vol. 2020, no. 9, pp. 15–19, Sep. 2020.

[27] *Comparativo Soluções Mail Gateway/Antispam OpenSource (Gratuitas)*. Accessed: Aug. 2021. [Online]. Available: https://nvlan.com.br/comunidade/comparativo-solucoes-antispam-opensource-gratuitas/

[28] *Best Email Anti-Spam Software*. Accessed: Aug. 2021. [Online]. Available: https://www.g2.com/categories/email-anti-spam

[29] V. Sarcar, *Java Design Patterns: A Hands-on Experience With Real-World Examples*, 2nd ed. New York, NY, USA: Apress, 2018.

[30] P. M. Oliveira, M. B. Vieira, I. C. Ferreira, E. M. Oliveira, E. M. Moreira, and O. A. S. Carpinteiro, "ABL: An original active blacklist based on a modification of the SMTP," *PlOs ONE*, to be published.

[31] *SubEtha SMTP*. GitHub Platform. Accessed: Aug. 2021. [Online]. Available: https://github.com/voodoodyne/subethasmtp

[32] *Lei no. 12.527, de 18 de Novembro de 2011*, Presidência da República—Casa Civil. Accessed: Aug. 2021. [Online]. Available: http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2011/lei/l12527.htm

[33] *Teor de Emails Oficiais Pode ser Requisitado Por Qualquer Pessoa, Confirma CGU*. Accessed: Aug. 2021. [Online]. Available: http://livre.jor.br/teor-de-emails-oficiais-pode-ser-requisitado-por-qualquer-pessoa-confirma-cgu/

[34] A. Cournane and R. Hunt, "An analysis of the tools used for the generation and prevention of spam," *Comput. Secur.*, vol. 23, no. 2, pp. 154–166, Mar. 2004.

[35] *jsoup: Java HTML Parser*. Accessed: Aug. 2021. [Online]. Available: https://jsoup.org/

[36] T. Yamane, *Statistics: An Introductory Analysis*. Manhattan, NY, USA: Harper & Row, 1973.

[37] P. E. Latham and Y. Roudi. (2009). *Mutual Information*. Scholarpedia. [Online]. Available: http://www.scholarpedia.org/article/Mutual_information

[38] *Weka: The workbench for machine learning*. Machine Learning Group, University of Waikato. Accessed: Aug. 2021. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/index.html

[39] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.

[40] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[41] *Zimbra*. Synacor. Accessed: Aug. 2021. [Online]. Available: https://www.zimbra.com/open-source-email-overview/

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

[43] 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. https://www.sciencedirect.com/science/article/abs/pii/S0925231216315612

[44] D. Mills, J. Martin, J. Burbank, and W. Kasch, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, Internet Engineering Task Force, document RFC 5905, Jun. 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5905

[45] M. Gardner and D. Altman, "Confidence intervals rather than P values: Estimation rather than hypothesis testing," *Brit. Med. J. (Clin. Res. Ed.)*, vol. 292, no. 6522, pp. 746–750, 1986.

[46] O. A. S. Carpinteiro, B. C. Sanches, and E. M. Moreira, "Detecting image spam with an artificial neural model," *Int. J. Comput. Sci. Inf. Secur.*, vol. 15, no. 1, pp. 296–314, Jan. 2017. [Online]. Available: https://www.academia.edu/36003643/Journal_of_Computer_Science_IJCSIS_January_2017_Full_Volume_pdf

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.

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.

EDVARD M. OLIVEIRA received the B.Sc. degree in computer science from the Pontifical Catholic University (PUC) of Minas Gerais, in 2010, and the M.Sc. and Ph.D. degrees in computer science and computational mathematics from the University of São Paulo (USP), 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. degree in computer science and computational mathematics and the Ph.D. degree in computer science from the University of São Paulo, 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: 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, the B.Sc. degree in music, and the M.Sc. degree in systems and computer engineering from the Federal University of Rio de Janeiro, 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 presently 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.