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

ISAAC C. FERREIRA¹, MARCELO V. C. ARAGÃO², EDVARD M. OLIVEIRA³, BRUNO T. KUEHNE³, EDMILSON M. MOREIRA³, OTÁVIO A. S. CARPINTEIRO³

¹TRICOD Equipamentos Eletrônicos Indústria e Comércio LTDA, R. Cel. Francisco Braz 185, Apto. 21, Itajubá, MG, 37500-005, Brazil
²National Institute of Telecommunications, Av. João de Camargo 510, Santa Rita do Sapucaí, MG, 37540-000, Brazil
³Research Group on Systems and Computer Engineering, Federal University of Itajubá, Av. BPS 1303, Itajubá, MG, 37500-903, Brazil

E-mails: caldas.isaac@gmail.com; marcelovca90@inatel.br; {edvard,kuehne,edmarmo,otavio}@unifei.edu.br

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

This research was supported by 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 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

¹Spammers are individuals who send spam e-mails.
blacklists, and that causes it 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 trustable 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 ham2 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 allows e-mail service administrators to perform spam filtering in an

2Legitimate e-mails are also referred to as hams.
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 its 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 traslation 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.
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 partially 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 “<p align=left> contents </p>” 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 ℜ^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 ℜ^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 a 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 situations, the Quarentine Module forwards, at intervals defined by the Open-MaLBAS administrator, a report to each user containing the spam e-mails the
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, 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 \( \delta \) 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 \( \delta \) 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 \( \delta \) 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 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 $\delta$ (henceforth, $\delta$-dissimilarity e-mails) and puts each group of $\delta$-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 $\delta$-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-$\delta 0$, 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-$\delta 0$ 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-$\delta 0$ 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, however, there are, respectively, 287 (0.08%) and 12,591 (2.47%) of null ham and spam vectors.

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

![FIGURE 3. E-mails of the UNIFEI database — ham: blue; spam: red; ham/spam: black](image_url)
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:

- $\hat{N}_{HAM}$: total number of ham e-mails in the vector set;
- $\hat{N}_{SPAM}$: total number of spam e-mails in the vector set;
- $n_{H\rightarrow H}$: number of ham e-mails correctly classified;
- $n_{H\rightarrow S}$: number of ham e-mails classified as spam;
- $n_{S\rightarrow S}$: number of spam e-mails correctly classified;
- $n_{S\rightarrow H}$: 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 $P(GEN)$ — were used to evaluate the two classifier models. They are calculated by the Equations (1), (2) and (3).

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

$P(SPAM)$ and $P(GEN)$ follow the same formula.

$\text{GEN.}$ 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).

$$P(SPAM) = \frac{n_{S \rightarrow S}}{n_{S \rightarrow S} + n_{H \rightarrow S}}$$

(2)

$$P(GEN.) = \frac{N_{HAM} \times P(HAM) + N_{SPAM} \times P(SPAM)}{N_{HAM} + N_{SPAM}}$$

(3)

$$R(HAM) = \frac{n_{H \rightarrow H}}{n_{H \rightarrow H} + n_{H \rightarrow S}}$$

(4)

$$R(SPAM) = \frac{n_{S \rightarrow S}}{n_{S \rightarrow S} + n_{S \rightarrow H}}$$

(5)

$$R(GEN.) = \frac{N_{HAM} \times R(HAM) + N_{SPAM} \times R(SPAM)}{N_{HAM} + N_{SPAM}}$$

(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.

![Architecture used in the experiment](link)

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 spent time, 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.

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

| Method | NF | NF' | P(HAM) | P(SPAM) | P(GEN) |
|--------|----|-----|--------|---------|--------|
| FD     | 8  | 5   | 64.86 ± 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| 51  | 75.98 ± 1.54  | 86.12 ± 2.00  | 80.81 ± 0.38  |
| FD     | 256| 101 | 75.39 ± 1.01  | 83.98 ± 2.51  | 79.69 ± 0.44  |
| FD     | 1024| 105 | 71.59 ± 0.84  | 80.53 ± 1.40  | 76.16 ± 0.27  |
| 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 ± 0.53  | 67.78 ± 0.30  |
| MI     | 64 | 9   | 63.30 ± 0.73  | 81.42 ± 1.22  | 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  |

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.

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.

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 NF’, (NF’ + 2)/2, (NF’ + 2)/4 and 2, respectively. 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 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 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

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 FD 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 MI method, with 128-dimensional vectors reduced to 65-dimensional vectors. In turn, from the Tables 9 and 10, it can be seen that the model 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 MI method, with 1024-dimensional vectors reduced to 208-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 created by the MI method, with 1024-dimensional vectors reduced to 208-dimensional vectors.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/ACCESS.2021.3118901, IEEE Access
of the two models is due to the fact that, on WEKA, the implementation of the RF model is multithreaded, 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.

### 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.

Tables 13, 14, and 15 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 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 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 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 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 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 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 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 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 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 256-dimensional vectors reduced to 60-dimensional vectors.
under the recall metric — was obtained with the set created by the FD method, with 512-dimensional vectors reduced to 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 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 classification time of the MLP model (Table 15) — 1 hour, 43 minutes and 32 seconds — also obtained on the same set. In the second experiment (Section VII-B), the significant difference between the classification 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 classifying the models, the RF model, once again, produced better results than those produced by the MLP model.
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, available at https://www.statista.com/statistics/420391/spam-email-traffic-share/, Accessed in August, 2021.
[2] J. Levine, DNS blacklists and whitelists, Internet Engineering Task Force, RFC 5782, available at https://www.rfc-editor.org/info/rfc5782, February 2010.
[3] M. Kucherawy and D. Crocker, Email greylisting: An applicability statement for SMTP, Internet Engineering Task Force, RFC 6647, available at https://www.rfc-editor.org/info/rfc6647, June 2012.
[4] I. Harbarczyk, Email blacklist: What is it and how to avoid it in 2020?, available at https://www.usebouncer.com/email-blacklist-what-is-it-and-how-to-avoid-it-in-2020/, February 2020.
[5] G. Croce and A. Killeen, How to avoid an email domain blacklist and remove your IP from one, available at https://hubbell.com/insights/how-to-avoid-email-domain-blacklist-and-remove-ip/, Accessed in August, 2021.
[6] J. Huckaby, The 8 email blacklists you should actually care about, available at https://www.rackaid.com/blog/email-blacklist/January 2019.
[7] B. Gardiner, Why do IP addresses get blacklisted?, available at https://www.fasthosts.co.uk/blog/why-do-ip-addresses-get-blacklisted-2/, October 2020.
[8] E. Peering and R. Hinden, Internet Protocol, Version 6 (IPv6) Specification, STD 86, Internet Engineering Task Force, RFC 8200, available at https://www.rfc-editor.org/info/rfc8200, July 2017.
[9] J. Klensin, SMTP 521 and 556 reply codes, Internet Engineering Task Force, RFC 7504, available at https://www.rfc-editor.org/info/rfc7504, June 2015.
[10] S. Kitterman, Sender policy framework (SPF) for authorizing use of domains in email, version 1, Internet Engineering Task Force, RFC 7208, available at https://www.rfc-editor.org/info/rfc7208, April 2014.
[11] D. Crocker, T. Hansen, and M. Kucherawy, Domain keys identified mail (DKIM) signatures, Internet Engineering Task Force, RFC 6376, available at https://www.rfc-editor.org/info/rfc6376, September 2011.
[12] M. Kucherawy and E. Zwicky, Domain-based message authentication, reporting, and conformance (DMARC), Internet Engineering Task Force, RFC 7489, available at https://www.rfc-editor.org/info/rfc7489, March 2015.
[13] J. Chen, V. Paxson, and J. Jiang, “Composition kills: A case study of email sender authentication,” in Proceedings of the USENIX Security Symposium, available at https://www.usenix.org/conference/usenixsecurity20/presentation/chen-jiangjun, 2020, pp. 2183–2199.
[14] CanIt-PRO, Roaring Penguin Software Inc., available at https://www.roaringpenguin.com/products/canit-pro, Accessed in August, 2021.
[15] GNU Licenses, Free Software Foundation, available at http://www.gnu.org/licenses/licenses.en.html, Accessed in August, 2021.
[16] Open Machine-Learning-Based Anti-Spam, GitHub, available at https://github.com/Isaac44/Open-MaLBAS, Accessed in August, 2021.
[17] The Apache Spam Assassin Project, Apache Software Foundation, available at https://spamassassin.apache.org/, Accessed in August, 2021.
[18] Anti Spam SMTP Proxy, Sourceforge Platform, available at http://www.iflockat.com/asph-home/, Accessed in August, 2021.
[19] qsp-mdp, GitHub Platform, available at http://smtpd.github.io/qsp-mdp/, Accessed in August, 2021.
[20] Bit defender Antivirus, Bitdefender, available at https://www.bitdefender.com/, Accessed in August, 2021.
[21] ClamAV Antivirus, ClamAV, available at https://www.clamav.net/, Accessed in August, 2021.
[22] D. J. Bernstein, Qmail, available at https://cr.yp.to/qmail.html, Accessed in August, 2021.
[23] W. Z. Venema, Postfix, available at http://www.postfix.org/, Accessed in August, 2021.
[24] Exim Internet Mailer, University of Cambridge, UK, available at https://www.exim.org/index.html, Accessed in August, 2021.
[25] Barracuda Email Security Gateway, Barracuda Networks, Inc., available at https://www.barracuda.com/products/emailsecuritygateway, Accessed in August, 2021.
[26] A. Bhdawaj, V. Sapra, A. Kumar, N. Kumar, and S. Arthi, “Why is phishing still successful?” Computer Fraud & Security, vol. 2020, no. 9, pp. 15–19, September 2020.
[27] Comparativo Soluções Mail Gateway/Anitispam OpenSource (gratuitas), NVLan.com.br, available at https://nvlan.com.br/comunidade/comparativo-solucoes-antispam-opensource-gratuitas/, Accessed in August, 2021.
[28] Best Email Anti-spam Software, G2.com, Inc., available at https://www.g2.com/categories/email-anti-spam, Accessed in August, 2021.
[29] V. Sarcar, Java Design Patterns: A Hands-On Experience with Real-World Examples, 2nd ed. Apress, 2018.
[30] P. M. Oliveira, M. B. Vieira, I. C. Ferreira, E. M. Oliveira, E. M. Moreira, and O. A. S. Carpinheiro, “ABL: An original active blacklist based on a modification of the SMTP?,” 2021. Submitted to Plos One.

[31] SubEtha SMTP, GitHub Platform, available at https://github.com/vooodoodynet/subethasmtpl, Accessed in August, 2021.

[32] Lei no. 12.327, 18 de novembro de 2011, Presidência da República — Casa Civil, available at http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2011/lei/l12327.htm, Accessed in August, 2021.

[33] Teor de emails oficiais pode ser requisitado por qualquer pessoa, confirma CGU, Livre.jor, available at http://livre.jor.br/teor-de-emails-oficiais-pode-ser-requisitado-por-qualquer-pessoa-confirma-cgu/, Accessed in August, 2021.

[34] A. Courmane and R. Hunt, “An analysis of the tools used for the generation and prevention of spam,” Computer Security, vol. 23, pp. 154–166, 2004.

[35] jsoup: Java HTML Parser, Jsoup Org, available at https://jsoup.org/, Accessed in August, 2021.

[36] T. Yamane, Statistics: An introductory analysis, Harper & Row, 1973.

[37] P. E. Latham and Y. Roudi, “Mutual information,” Scholarpedia, 2009, available at http://www.scholarpedia.org/article/Mutual_information.

[38] Weka: The workbench for machine learning, Machine Learning Group, University of Waikato, available at https://www.cs.waikato.ac.nz/ml/weka/index.html, Accessed in August, 2021.

[39] S. Haykin, Neural Networks and Learning Machines, 3rd ed. Prentice-Hall, Inc., 2009.

[40] L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[41] Zimbra, Synacor, Inc., available at https://www.zimbra.com/open-source-email-overview/, Accessed in August, 2021.

[42] L. Maaten and G. Hinton, “Visualizing data using t-SNE,” Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.

[43] F. Jiménez, G. Sánchez, J. García, G. Sciavicco, and L. Miralles, “Multi-objective evolutionary feature selection for online sales forecasting,” Neurocomputing, vol. 234, pp. 75–92, 2017, available at 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, RFC 5905, available at https://www.rfc-editor.org/info/rfc5905, June 2010.

[45] M. Gardner and D. Altman, “Confidence intervals rather than P values: Estimation rather than hypothesis testing,” British Medical Journal (Clinical Research Ed.), vol. 292, no. 6522, pp. 746–750, 1986.

[46] O. A. S. Carpinheiro, B. C. Sanches, and E. M. Moreira, “Detecting image spam with an artificial neural model,” International Journal of Computer Science and Information Security, vol. 15, no. 1, pp. 296–314, January 2017, available at https://www.academia.edu/36003643/Journal_of_Computer_Science_IJCSIS_January_2017_Full_Volume_pdf.

EDWARD M. OLIVEIRA is an Assistant Professor at the Federal University of Itajubá, MG, Brazil. He has a B.Sc. degree in Computer Science from the Catholic University (PUC) of Minas Gerais (2010). He also has a M.Sc. (2013) and a Ph.D. (2018) degree in Computer Science and Computational Mathematics from the University of São Paulo (USP). He works with Distributed Systems, with emphasis in e-Science, Scientific Gateways, Service Oriented Architectures (SOA), Scientific Workflows, and Internet of Things, and is involved in research and development projects within the Research Group on Systems and Computer Engineering at the Federal University of Itajubá.

BRUNO T. KUEHNE has a B.Sc. degree (2006) in Computer Science from the Catholic University of Minas Gerais, MG, Brazil, a M.Sc. degree (2009) in Computer Science and Computational Mathematics from the University of São Paulo, SP, Brazil and a PhD degree (2015) in Computer Science from the University of São Paulo, SP, Brazil. Since 2014 he is an Assistant Professor and a member of the Research Group on Systems and Computer Engineering at the Federal University of Itajubá, MG, 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, IoT and Performance Evaluation.

EDMILSON M. MOREIRA is an Associate Professor at the Federal University of Itajubá, Brazil. He has a B.Sc. (1994) degree in Computer Science from the University of Alfenas, Brazil, and a B.Sc. (1996) degree in Mathematics from Faculty of Philosophy, Sciences and Linguistics of Varginha, Brazil. He also has a M.Sc. (2000) and a Ph.D. (2005) degrees in Computer Science and Computational Mathematics from the University of São Paulo (USP), Brazil. He works with Graph Theory, Distributed Systems, and Discrete Events Simulation, and is a member of the Research Group on Systems and Computer Engineering at the Federal University of Itajubá.

ISACC C. FERREIRA received the B.Sc. degree in Computer Engineering, in 2014, and the M.Sc. degree in Computer Science and Technology, in 2018, both from the Federal University of Itajubá, MG, Brazil. He is currently working as a R&D engineer at the TRICOD Equipamentos Eletrônicos e Comércio LTDA (TRICOD Electronic Equipments Industry and Commerce Limited), MG, Brazil.

MARCELO V. C. ARAGÃO has a B.Sc. degree (2014) in Computer Engineering from the National Institute of Telecommunications (INATEL), MG, Brazil, and a M.Sc. degree (2018) in Computer Science and Technology from the Federal University of Itajubá, MG, Brazil. From 2011 to 2018, as a system specialist, he developed Business Support Systems (BSS) solutions in continuous integration environment at the INATEL Competence Center. He currently teaches undergraduate courses, coordinates the graduate course in Application Development for Mobile Devices and Cloud Computing, and is a Ph.D. student in Telecommunication Engineering, all at INATEL. He has interest in the areas of machine learning, data science, and software engineering.

ISAAC C. FERREIRA, received the B.Sc. degree in Computer Engineering, in 2014, and the M.Sc. degree in Computer Science and Technology, in 2018, both from the Federal University of Itajubá, MG, Brazil. He is currently working as a R&D engineer at the TRICOD Equipamentos Eletrônicos e Comércio LTDA (TRICOD Electronic Equipments Industry and Commerce Limited), MG, Brazil.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
OTÁVIO A. S. CARPINTEIRO was born in Rio de Janeiro, RJ, Brazil. He has a B.Sc. degree in Mathematics, a B.Sc. in Music, and a M.Sc. in Systems and Computer Engineering, all from the Federal University of Rio de Janeiro, RJ, Brazil. He has a D.Phil. degree in Cognitive and Computer Science from the University of Sussex, UK. He has worked as a system analyst for many years, and ended his professional career as a full professor at the Federal University of Itajubá, MG, 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 R&D projects as a member of the Research Group on Systems and Computer Engineering at the Federal University of Itajubá.