

## Detecting attacks on e-mail

Yujia Fang, Gabriela Mogos

Department of Computing, School of Advanced Technology, Xi'an-Jiaotong Liverpool University, Suzhou, China

---

### Article Info

#### Article history:

Received Oct 21, 2023

Revised Nov 20, 2023

Accepted Nov 30, 2023

#### Keywords:

Bayesian filter

E-mail filtering

Machine learning

Spam

Support vector machine

---

### ABSTRACT

E-mail has become a popular communication tool widely used by universities, enterprises and governments. Despite the convenience it brought to people, attacks on e-mail happen very frequently in the range of the world, causing large economic loss and occupying a mass of network bandwidth every year. The hazards from e-mail attacks underline the importance of detecting and resisting spam in an efficient and timely way. Using Python, we built Naïve Bayes (NB) and support vector machine (SVM) filters for emails. The filtering performance of NB and SVM email filters applying different kernel functions was compared and evaluated based on several evaluation indices including accuracy, precision, and total cost ratio (TCR). Also, in order to optimize the filters, the influences of stop words removal, feature numbers and other parameters in the filtering algorithms were monitored.

*This is an open access article under the [CC BY-SA](#) license.*



---

### Corresponding Author:

Gabriela Mogos

Department of Computing, School of Advanced Technology, Xi'an-Jiaotong Liverpool University

Suzhou, China

Email: gabriela.mogos@xjtlu.edu.cn

---

## 1. INTRODUCTION

Since the first e-mail in the world sent by Ray Tomlinson in 1971, e-mail has undergone a steady development over decades and became widely used around the world as the birth of internet browser in 1990s. Nowadays, e-mail becomes a crucial tool in communication with a huge user base, and billions of e-mails are sent and received worldwide every year. From Radicati [1], including both business and consumer e-mail users, there are over 4,2 billion e-mail users with a growth rate of 3% in 2022, which means over half of the world population are using e-mail. Interaction and communication between people have been greatly promoted through e-mail, and the development of society has also been facilitated. However, the flourishing development of e-mail also causes the problem of junk e-mail, or called spam. Spam does not have a worldwide unified definition but according to the Internet Society of China, spams have the following four characteristics:

- Containing advertisements, electronic publications and other promotional e-mails that the recipient did not request or agree to receive in advance.
- Recipient cannot reject.
- Hiding sender identity, address, title, and other information.
- Containing false information sources, senders, and routes.

Most of spams are delivered through simple mail transfer protocol (SMTP) relay service. Relay refers to the process of e-mails being transmitted by third party mail servers in the e-mail path. Because the SMTP protocol do not authenticate users when transmitting e-mails, e-mails can be sent anonymously or under a pseudonym by anyone. This causes the problem of open relay. Spam senders can use automated tools on the internet to set up effective third-party forwarding e-mail servers. By simultaneously using open forwarding

services to transmit e-mails, it can lead to a large amount of spam on the network at once. Many spam senders utilize open relay to bypass the blacklist of e-mail servers.

Dangers of spam: spam causes congestion on the internet. There are large amounts of spams flushing into e-mail servers every day. According to Kaspersky [2], 56.51% of e-mails in mail traffic are spam in 2019. Considering the order of magnitude of sent e-mails, the quantities of spam are surprisingly huge, which leads to the waste of a great number of internet network resources (e.g., bandwidth) and time for recipients to deal with them. For communication institutions, the large amounts of spam require them to significantly improve computer performance to maintain the normal operation of mail servers, which raises expenses. In addition, spam also brings economical loss to individuals and organizations. It is common for spams to contain advertisements. For example, spams may claim to offer consumers free opportunity to watch new films or sports broadcasts. Once e-mail users fall into this bait, their economic information may be exposed by typing in their bank accounts and passwords into a fake payment website, which is usually attached in the spam. As for the organizations, phishing attacks may occur when attackers use social engineering or other means to gain the trust of employees. Attackers may use deceptive or hacked accounts to impersonate trusted sources and send targeted e-mails to employees, business partners or customers, causing loss of sensitive information and funds. This is also called business e-mail compromise (BEC). From FBI [3], the economic loss of US businesses due to BEC between January 2014 and October 2019 is over 2 billion dollars. Also, spam does harm to the image of internet service provider (ISP). Users who are tired of receiving spam may want to exchange the initial internet service providers because they have not established a comprehensive spam filtering system. A more critical result is that some famous anti-spam organizations may add some of the IP addresses of ISPs into blacklist. For example, Spamhause, which is the most famous anti-spam organization in Europe, has banned a lot of problematic IP addresses in China, leading to many enterprise users lose connection with their customers. The ISPs of these banned IP addresses are mainly China Telecom and China Netcom.

Moreover, spam is becoming an important carrier of virus transmission in the internet era. For example, the large-scale outbreak of Melissa in 1999 caused the computer email systems of over 300 companies worldwide, including Microsoft and Intel, to crash, causing losses of over 80 million dollars. In conclusion, considering the great harm of spam on the internet, economy, and society, it is essential to emphasize the importance of finding an effective way to filter the spams out of normal e-mails.

Anti-spam methods: since last century, lots of efforts have been put into the field of anti-spam. Many organizations have been found for the sake of anti-spam, such as the establishment of MAPS (1996), which collects malicious IP addresses to maintain the real-time blackhole list (RBL), and the creation of SpamAssassin by Justin Mason (2001), which is still an extremely popular spam-filtering platform at the moment. There are mainly three aspects in the methods of anti-spam, which are:

- Modify the existing SMTP protocol and fix the vulnerabilities of it. Such as the internet 2,000 protocol raised by Bernstein [4], in which e-mails are saved in the sender terminal and the recipient needs to get the e-mail from the sender terminal. However, it is very difficult and time-consuming to promote it worldwide.
- Legislation can be a powerful tool to regulate and prevent the problems of spam, multiple countries have formulated or are in the process of formulating legal documents related to spam. For example, the United States passed a law called the CAN-SPAM act of 2003 [5], which is the first national standard to regulate commercial e-mails and reduce spam e-mails [6]. Violation of this law would result in high fines. In China, which is still in the early stages of anti-spam legislation, the legal basis people can refer to is very few.
- From technical aspect, there are many technical methods raised in the field of anti-spam over the last decades, and they can mainly be divided into 3 categories [7]. The first type is content-based filtering, such as key word filtering and Bayesian filtering. This technology is mainly used by message user agent (MUAs) to identify and handle spam.

The second category of anti-spam techniques [8] is query and authentication, including domain keys identified mail (DKIM) and patching. This technique is built on the basis that forged sender addresses are used in most spam, and it examine the header of each e-mail without violate privacy [9]. The last type is behavior-based filtering [10]. In this method, the anti-spam engine will establish a reasonable model to decide whether an e-mail is legitimate based on its behavioral characteristics during delivery. For parts that exceed normal values, they are considered abnormal.

E-mail theory and transmission: like the mails having envelop and content in the real world, an e-mail mainly consists of two parts, which are header and body. As for the e-mail body, it usually contains a text

message or a structured message, which follows the multipurpose internet mail extensions (MIME) standard. It is a coding standard for e-mail, which aims at solving information exchange issues in transmitting non-ASCII character text and non-ASCII format (e.g., images, audios, and videos). In the architecture of mail transfer system (MTS), MUA is the module users directly used to edit the e-mail content when sending an e-mail. Typical MUAs are Outlook Express, NetEase email, and Foxmail. After the user clicking on the "send" button, the message submission agent (MSA) would accept the e-mail and submit it to the message transfer agent (MTA). The MTA of sender then uses SMTP to deliver the email to the recipient's MTA, which may require one or more MTA transfers. This process can be ignored if the sender and the recipient share an MTA. Then the message delivery agent (MDA) would deliver the e-mail to the recipients based on POP3 or IMAP. And the recipient can see the e-mail content through his MUA.

## 2. METHOD

The main objective of our research is to efficiently detect attacks on e-mail by building spam filters implementing machine-learning techniques, and to investigate the pros and cons of different machine-learning algorithms in order to improve the performance of the spam filter [11]. Based on the e-mail transfer system, there are many e-mail filtering methods raised over years [12]-[14]. From the perspective of technology [15]-[17], the filtering methods can be based on key words, blacklist, and content. The methodology we used for the email spam filtering is based on Naïve Bayes (NB) algorithms. The NB technique helped us to determine the probabilities spam email. The Figure 1 shows the process for e-mail spam filtering based on the NB algorithm.

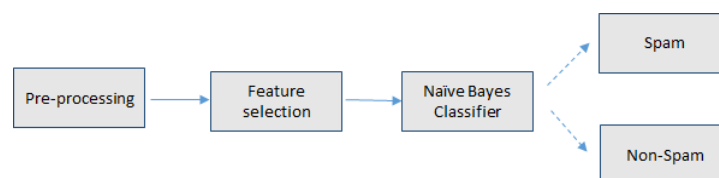


Figure 1. E-mail spam filtering process

### 2.1. Experimental software

In our research, Python is used to implement the e-mail filters. Python is a popular programming language designed by Rossum in the early 1990s and has the advantages of being concise and having rich open-source libraries. Jupyter notebook (previously known as IPython notebook), which is a web application essentially, is selected as the experimental software because it supports segmented code operation, and the results are displayed timely and intuitive.

### 2.2. Dataset

The dataset used in this project is from 2006 text retrieval conference (TREC) Public Spam Corpora [18]. TREC is the most popular and authoritative evaluation conference in the field of text retrieval held by National Institute of Standards and Technology (NIST) and Defence Advanced Research Projects Agency (DARPA). In this spam corpora, there are corpus in English (trec06p) and in Chinese (trec06c). The corpus includes three folders named data, delay and full. In the data folder there are a total of 64620 e-mails with nearly 66% of them are spam, and the content and the header for each e-mail are spaced with one blank row. The indexes for the e-mails are included in the delay and full folders with labels (ham or spam) and mail paths.

### 2.3. E-mail preprocessing

Before applying the filtering algorithms on the dataset, it is essential to preprocess the e-mails because the algorithms cannot directly learn from the e-mail body and header. In this research the content of e-mails will be focused on because normally the content contains more information. In general situation of content-based filtering, the e-mails are expressed as multidimensional vectors where the extracted e-mail features serve as dimensions, which enable algorithms (e.g., NB) to calculate the spam possibilities or to determine the separating hyperplane SVM. A good e-mail preprocessing can effectively decrease the training time (Attributes reduction) and improves the overall performance for the filter.

### 2.3.1. Email content extraction

To connect the e-mail paths saved in the index folder with their corresponding e-mail content, a function to acquire the e-mail path is written first. Then, since the body and content of e-mails are divided by a blank row in the data folder, the “split” function is used to store the body and the content separately in two variables named mailHeader\_list and mailContent\_list. The body and header then can be accessed using the index. The code realization is shown in figure with a printed example of e-mail body see in Figure 2.

```
def getFilePathList(rootDir):
    filePath_list = []
    for w in os.walk(rootDir):
        part_filePath_list = [os.path.join(w[0]+'/', file) for file in w[2]]
        filePath_list.extend(part_filePath_list)
    return filePath_list
filePath_list = getFilePathList(r"C:\Users\FY\Desktop\机器学习数据集\trec06c\data")
print(filePath_list[6666])
print(len(filePath_list))

C:\Users\FY\Desktop\机器学习数据集\trec06c\data\022\066
64620

mailHeader_list = []
mailContent_list = []
for filePath in filePath_list:
    with open(filePath, errors='ignore') as file:
        file_str = file.read()
        mailHeader = file_str.split('\n', 1)[0]
        mailContent = file_str.split('\n', 1)[1]
        mailHeader_list.append(mailHeader)
        mailContent_list.append(mailContent)

mailContent_list[1]
```

```
'讲的是孔子后人的故事。一个老领导回到家乡。跟儿子感情不和。跟贪财的孙子孔为本和睦。
\n老领导的弟弟魏宗万是赶马车的。
\n有个洋妞大概是考察民俗的。在他们家过年。
\n孔为本总想出国。被爷爷教育了。
\n最后。一家人基本和解。
\n顺便问另一类电影。北京青年电影制片厂的。中越战背景。一军人被介绍了一个对象。去相亲。女方是军队医院的护士。犹豫不决。总是在回忆战场上负伤的男友。好像还没死。最后
\n男方表示理解。归队了。
\n'
```

Figure 2. Extracted e-mail body

### 2.3.2. E-mail word segmentation

The quality of word segmentation is a key factor influencing the filter performance because wrong word segmentation would cause misinterpretation of sentence meaning. Since English and Chinese e-mails are to be processed in this project, only English and Chinese word segmentation are covered here.

- English word segmentation. The process of English word segmentation is relatively easy compared with Chinese word segmentation. In English e-mail, the words are separated by spaces so that the content between two spaces is considered to be one word. To improve the English effects of word segmentation, extra process like stemming and lemmatization can be added. Stemming applies reduction to obtain stems from original words. For example, it transforms “boots” to “boot” and “richness” to “rich”. While lemmatization converts the words to their original forms, such as converting “drove” to “drive”. Stemming and lemmatization can effectively reduce the number of features without distorting the meaning of words.
- Chinese word segmentation. Unlike English, there exist no spaces in Chinese sentences to separate words, which make Chinese word segmentation more difficult. There exist many word segmentation methods and techniques, such as algorithms based on neural networks, maximum matching (MM) method and word frequency statistics method [19]. The maximum matching method takes the longest word length in the dictionary as the length of words to be scan each time. And if the dictionary does not contain the scanned words, the length of scanned words would be reduced by 1. This process will repeat again and again until one word in dictionary is recognized. However, there are mainly two problems existing in this method. The first one is the setting up of the longest word length. If set too short, some long words may be skipped when scanning. If set too long, the efficiency of word segmentation would be significantly influenced. The second problem is that a sentence may have multiple ways to be segmented. For example, considering the sentence “一家人生活不错” (The family is doing well). The segmentation result of this sentence is “一家”, “人生”, “活”, “不错” (Family, Life, Live, Not bad). While the expecting result would be “一家人”, “生活”, “不错” (Family, Life, Nice). This kind of segmentation mistake causes the misinterpretation of sentences, degrading the filter performance. Considering the drawbacks of MM method, a third-party library

called *jieba* is used to implement the Chinese word segmentation in this project. *Jieba* firstly constructs a prefix dictionary according to the statistic dictionary. Then, *jieba* uses the prefix dictionary to segment the sentences to get all possible word-forming situations of Chinese characters in sentences, which are lately used to form a directed acyclic graph. Through dynamic programming algorithm, *jieba* calculates the maximum probability path and the final segmentation form [20]-[22]. For those words not in the prefix dictionary, *jieba* uses hidden markov model (HMM) to recognize them. *Jieba* supports 3 modes of word segmentation, which are precise mode, full mode and search engine mode. The return type of *jieba* is list. An example of an e-mail with Chinese segmented content is shown in Figure 3. Note that the words are separated by spaces and stop words have been deleted. The segmentation result by *jieba* is gratifying.

```
In      mailHeader_list = []
      mailContent_list = []
      for filePath in filePath_list:
          with open(filePath, errors='ignore') as file:
              file_str = file.read()
              mailHeader = file_str.split('\n\n', 1)[0]
              mailContent = file_str.split('\n\n', 1)[1]
              mailHeader_list.append(mailHeader)
              mailContent_list.append(mailContent)

In      mailContent_list[1]
Out     '讲的是孔子后人的故事。一个老领导回到家乡，跟儿子感情不和，跟贪财的孙子孔为本和睦。老领导的弟弟魏宗万是赶马车的。有个姐姐大概是考察民俗的，在他们家过年。孔为本总想出国，被爷爷教育了。最后，一家人基本和解。顺便问另一类电影，北京青年电影制片厂的。中越战背景，一军人被介绍了一个对象，去相亲。女方是军队医院的护士，犹豫不决，总是在回忆战场上负伤的男友，好像还没死。最后男方表示理解，归队了。'

In      with open(r"C:\Users\FY\Desktop\机器学习数据集\trec06c\baidu_stopwords.txt", 'r', encoding='utf-8') as file: #打开停用词文件
      stopwords = file.read()
      stopwords_list = stopwords.split('\n') #去除读txt文件里的换行

In      from tqdm.notebook import tqdm

In      cutwords_list = [] #建立一个分词空列表
      for line in tqdm(mailContent_list[0:5000], position = 0): #对这个每行邮件正文迭代，加速度条
          cutwords = [i for i in jieba.lcut(line) if i not in set(stopwords_list) and i!="\n"]
          cutwords_list.append(cutwords) #分好的一组词加入到分词列表
      cutwords_listDeal = [' '.join(text) for text in cutwords_list]

100% ██████████ 5000/5000 [05:59<00:00, 8.37W/s]

Building prefix dict from the default dictionary ...
Loading model from cache C:\Users\FY\AppData\Local\Temp\jieba.cache
Loading model cost 0.941 seconds.
Prefix dict has been built successfully.

'新加坡 联合早报 ( www.zaobao. ) ● 于祥远 ( 北京 ) \u3000\u3000昨天 中国 抗日战争 胜利 60 周年 纪念日，中央政府 大规模 纪念 活动，地方 民间团体 也 开展 活动，纪念 日本 投降 60 周年，高科技 企业 雪兰 产业 报国 \u3000\u3000北京 昨天 举办 一场 召开 生面 晋朝 大会，《 大刀 鬼子 头上 砍去 歌 旋律 背景音乐，100 多家 高科技 产业 企业 领导人 物 聚首 中关村，同声 宣誓 自主创新，产业 报国，宣誓 后，10 0 家 企业 宣布 成立 'V815' 民族 品牌 推广 委员会，\u3000\u3000新浪网 引述 参与 宣誓 华国 资讯 总裁 冯军 说：60 年前，先辈 正义 战争 证明 中国人 军事 上 不屈 不挠；60 年后，中国 拉开 历史 崭新一 幕，行动 证明 中国人 在 经济 上 不屈 不挠！\u3000\u3000来自 中国 大陆、香港、澳门 台湾 豫数 位 教授，昨天 北京 观礼 寺 孔 子 祭 孔 大典 暨 和平 祈拜 大会，肯定 国民党 抗战 \u3000\u3000古城 西安，陕西省 军区 政治 部 单位 举办 纪念 抗战 胜利 60 周年 大型 活动 一 一 百名 抗战 老兵 入城 式 拜日 隆重 举行，60 年前 浴血 奋战 抗日 杀敌 100 位 老战士 代表，胸佩 功勋 奖章，乘坐 十余辆 藏蓝 军用 吉普车 大轿 车 进城，热情 西安 市民 热烈 欢迎，\u3000\u3000\u3000以往 震撼 中 共 领袖 中 领导 地位，昨天 纪念 活动 缅怀 国民党 军队 抗战 作出 牺牲，中央 电视台 正 黄金 时段 详细 介绍 美国 驻华 空军 飞地 队 抗日 事迹，长沙 拉响 防空 警报 \u3000\u3000国民党 军队 日 军 激战 湖南 长沙，官方 上午 11 点 拉响 防空 警报，时间 长达 三分 钟，长沙 全城 司机 防空 警报 鸣响 汽车 喇叭，这是 长沙 首次 为一个 特定 纪念 日 拉响 防空 警报，\u3000\u3000\u3000福建省 会 福州市 昨天 三山 陵园 纪念 国民党 中 山 舰 沉 没 70 周 年 活动，中山 舰 1938 年 10 月 24 日 武汉 长江 流域 巡 航 时 遭 日 军 飞机 轰炸，25 名 国民党 海军 官兵 激战 中 殉 职，福州 籍 舰 长 李 承 恩 殉 职 者 断 肠 下 作 期 待，直 奔 前线，《 人民 日报 》 中共 领导 抗战 胜利 \u3000\u3000\u3000活动 肯定 国民党 军队 抗日，中 共 中央 机关 报 《 人民 日报 》 却 头 版 发表 长篇 特 约 评论 员 文章，是 因为 中共 领导，中国 才 赢得 抗战 胜利，北京 历史 学者 指出，中 共 抗战 中 领导 地位 取代 国民党 政府 合理性，推行 党 执政 合法性，\u3000\u3000\u3000中国社会 科学院 近代 史 研究所 曾 景 忠 说，抗战 期间，国民党 军队 阵 亡 官'
```

Figure 3. Example segmentation result by jieba

– Elimination of stop words. To preserve storage space and increase search effectiveness, stop words are usually filtered out when process text content of emails. Generally speaking, stop words are meaningless or useless words and can be roughly classified into two groups. One is function words which are extremely common. Compared with other words, function words have no classification value for the filter, such as “the”, “an”, and “am” for English words. The other type includes vocabulary words such as ‘want’, which are widely used in all kinds of texts. The removal of stop words can benefit the filter by reducing the training time and decreasing the dimensions of feature. In this research, a stop list from baidu is used, which contains stop words both in English and in Chinese.

## 2.4. Feature selection

Feature selection is a key process when implementing machine learning method. Only those most representative features should be selected because too many features would degrade the efficiency of training for classifier. A good feature selection can reduce the dimensions of e-mail features by deleting miscellaneous and irrelevant features, increasing the precision of classifier. Typical feature selection methods include term frequency-inverse document frequency (TF-IDF), mutual information, information gain (IG) and Chi-square statistic.

- Bag-of-words (BoW) model. BoW model [23] transforms the texts into feature vectors. Its basic idea is to assume that for a text, ignoring its word order, grammar, and syntax, only treating it as a collection of vocabulary. In our research, each e-mail is considered as a bag full of words. The dimensions of the feature vector are the number of total selected features. The frequencies of words in an e-mail are recorded by changing the numbers on the positions of the words. In our research, the function named CountVectorizer from sklearn library is used to form such a word frequency matrix. The main disadvantage of BoW model is that it ignores the relationship between words.
- TF-IDF. TF-IDF [24] is a feature selection tool in the field of data mining to calculate term weights and evaluate the classification value for words in the texts. Its main idea is that if one term occurs very frequently in small quantities of text, high value of classification should be attached to it. While those terms which appear in many texts of corpus contain less classification value. TF-IDF method is an empirical formula, but research has proved that it is a valid tool in the text processing field. Compared with just recording the frequencies of all terms (words) in the documents (e-mails), TF-IDF method effectively handles the problem of high order of dimensions and sparseness of generated matrix by adding the calculation of term weight. However, it still does not study the relationship between words, which sets a limitation on its performance.
- Feature hashing. Hash table is a data structure, which accesses data according to the key value  $k$ . The mapping between key and value is realized by a hash function for each hash table. Feature hashing [25] applies the hash method on feature selection in purpose of reducing the dimensionalities. Considering the sparse matrix with high dimensionalities formed by the term frequency and TF-IDF methods, feature hashing is able to convert all raw data into hash values within the specified range, effectively decreasing the time of feature selection. However, hash tables have errors when the hash collision happens, which refers to different keys mapped to one value. Besides, hash values are unexplainable and irreversible. In our research, a function named hashing vectorizer from the library sklearn is used to implement feature hashing.

## 2.5. Naïve Bayes filters

There are many models of NB e-mail filters [26], but we examined 3 models: Multinomial Naïve Bayes algorithm (MNB), Complement Naïve Bayes algorithm (CNB), and Bernoulli Naïve Bayes algorithm (BNB). MNB models [27] are mainly suitable to deal with discrete features. It considers the relationship between word frequencies in the text and the category of the text. CNB algorithm [28] is an improvement of the standard MNB to deal with corpuses unbalanced in categories and the problems brought by the attribute conditional independence assumption. Simply put, CNB uses the probability of complements from each label category to calculate the weight of each feature.

BNB algorithm [29] assumes that variables obey Bernoulli distribution, which is a random variable distribution with only “yes” or “no” results. For example, the positive or negative of coin flipping is a very typical Bernoulli distribution. In BNB model, the values of features are transformed to either “0” or “1”. If the value of feature is not “0” or “1” initially, a threshold value under which the value of feature is transformed to “0” should be set.

## 3. RESULTS AND DISCUSSION

After building the NB and SVM filters using Python, we extracted and preprocessed the content of the emails from the 2006 TREC. We also evaluated the performance of the models by splitting the corpus into training set and test set in a 7:3 ratio. The last part presents an analysis of the results as well as a comparative analysis based on several evaluation indices, which include precision, accuracy, and total cost ratio (TCR).

### 3.1. Evaluation indexes

To start with, confusion matrix is a machine learning analysis table that lists the outcomes of classification model predictions. For a binary classification problem, the results are divided into four categories through the comparison between the predicted class and the actual class, which are false positive, false negative, true positive and true negative. This table helps readers to learn the filter performance on different types of samples intuitively. Below shows a graph of binary confusion matrix model and an example confusion matrix from a NB filter Figure 4.

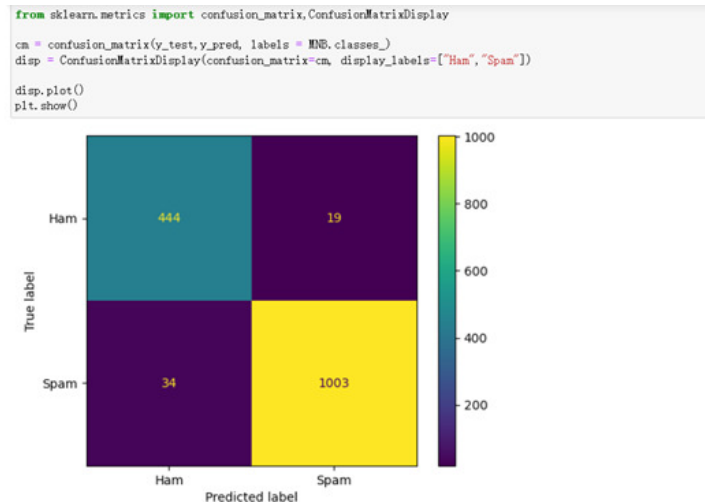


Figure 4. Confusion matrix

To evaluate the performance of e-mail filters and compare different algorithms, unified evaluation indexes should be set. Set  $N_{s \rightarrow h}$  (false positive) as the number of spams misclassified to hams by the e-mail filter,  $N_{h \rightarrow s}$  (false negative) as the number of legitimated e-mails identified as spams by the e-mail filter,  $N_{s \rightarrow s}$  (true negative) as the number of spams correctly filtered and  $N_{h \rightarrow h}$  (true positive) as the number of correctly identified hams. Set  $N_S$  and  $N_H$  as the total numbers of spams and hams in the corpus. Then the accuracy, error, precision and recall rates can be expressed as:

$$ACC = \frac{N_{s \rightarrow s} + N_{h \rightarrow h}}{N_S + N_H} \quad (1)$$

$$Err = \frac{N_{s \rightarrow h} + N_{h \rightarrow s}}{N_S + N_H} \quad (2)$$

$$Pre = \frac{N_{s \rightarrow s}}{N_{s \rightarrow s} + N_{h \rightarrow s}} \quad (3)$$

$$R = \frac{N_{s \rightarrow s}}{N_S} \quad (4)$$

Recall and precision are a pair of contradictory measures. Generously, when the classification confidence of a category increases, precision will rise, while recall is opposite. To consider these two indexes comprehensively, F-measure is raised, which is the harmonic mean of precision and recall:

$$F_1 = \left( \frac{Pre^{-1} + R^{-1}}{2} \right)^{-1} = 2 \times \frac{Pre \times R}{Pre + R} \quad (5)$$

For the above formulas, the mistakes of misclassifying a spam to a ham and misclassifying a ham to a spam are considered to have a same weight. However, it is usually more serious for the e-mail users to miss an important e-mail rejected by the filters than receiving a few spams. Therefore, the concept of weight is introduced by considering missing one ham is as valuable as misclassifying  $\lambda$  spams. The weighted accuracy and weighted error rate can be expressed as:

$$W_{acc} = \frac{N_{s \rightarrow s} + \lambda N_{h \rightarrow h}}{N_S + \lambda N_H} \quad (6)$$

$$W_{err} = \frac{N_{s \rightarrow h} + \lambda N_{h \rightarrow s}}{N_S + \lambda N_H} \quad (7)$$

Considering if there is no filter, all hams are accepted, and all spam are misclassified to hams because they will not be rejected. Then, the weighted accuracy and weighted error rate of reference can be obtained:

$$W_{acc}(R) = \frac{\lambda N_H}{N_S + \lambda N_H} \quad (8)$$

$$W_{err}(R) = \frac{N_S}{N_S + \lambda N_H} \quad (9)$$

And TCR, which is an important index for the filtering performance, is defined as:

$$TCR = \frac{W_{err}(R)}{W_{err}} = \frac{N_S}{N_{s \rightarrow h} + \lambda N_{h \rightarrow s}} \quad (10)$$

### 3.2. Naïve Bayes filter performance

The NB filter usually has a good performance in precision, but further improvement of performance can be hard. Bayesian algorithm is widely applied in e-mail filtering, and many research have been conducted to improve its performance. In this part, four factors influencing the filter performance will be examined, which are e-mail preprocessing (word segmentation and stop words removal), the number of features, different feature extraction methods and different types of NB algorithms. The corpus is divided into a training set and testing set with a ratio of 7:3.

#### 3.2.1. Influence of e-mail preprocessing and feature number

Figure 5 and Tabel 1 show four evaluation indexes for a MNB filter using term frequency to extract features without e-mail preprocessing. From the Table 1, as the feature number increases, both the accuracy and precision increase, while the value of recall fluctuates in the range from 0.73 to 0.79. The F1-score also has a slight rise from 0.79 to 0.84. But all the increases saturated when the feature number is above 10,000. More features would not bring improvement to the performance after this number. Overall, the NB filter without e-mail preprocessing has a disappointed performance with an error rate of about 25%. Considering F1-score as an effective evaluation index, the importance of e-mail preprocessing can be underlined in the graph in Figure 6. As it shown, the F1-score of NB with e-mail preprocessing is about 20% higher than the other, indicating a better filtering performance. In addition, the saturation of feature number is also witnessed, which means a larger feature number may not result in a better filter.

```
#def Multinomial_predict(mail_content=[]):
    predict_list=[]
    for i in mail_content:
        predict_list = MNB.predict(mail_content)
    for k in predict_list:
        if k == 1:
            print('Spam')
        else:
            print('Ham')
Multinomial_predict(Content_tfidf1[0:10])

svc_clf1 = SVC(kernel='rbf',C=2) svc_clf1.fit(X_train, y_train) y_pred1 = svc_clf1.predict(X_test) print("\nclassification_report: \n",
classification_report(y_test,y_pred1)) accuracy_score(y_test,y_pred1).round(4)

svc_clf2 = SVC(kernel='poly', degree=4,C=2)
svc_clf2.fit(X_train, y_train)

y_pred2 = svc_clf2.predict(X_test)
print(confusion_matrix(y_test, y_pred2))
print(classification_report(y_test, y_pred2))

[[463  0]
 [ 94 943]]
      precision    recall  f1-score   support

     0       0.83      1.00      0.91       463
     1       1.00      0.91      0.95      1037

 accuracy          0.94      1500
 macro avg          0.92      0.95      0.93      1500
 weighted avg          0.95      0.94      0.94      1500
```

Figure 5. Python code-the evaluation for MNB filter



Table 1. Evaluation for MNB filter

Feature number	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
2,000	0.6667	0.8649	0.7328	0.7934
4,000	0.7133	0.8928	0.7634	0.823
6,000	0.74	0.8966	0.7939	0.8421
8,000	0.7267	0.9018	0.771	0.8313
10,000	0.76	0.9703	0.7481	0.8448
12,000	0.76	0.9703	0.7481	0.8448
14,000	0.76	0.9798	0.7405	0.8435
16,000	0.76	0.9798	0.7405	0.8435
18,000	0.7533	0.9796	0.7328	0.8384
20,000	0.76	0.9798	0.7405	0.8435

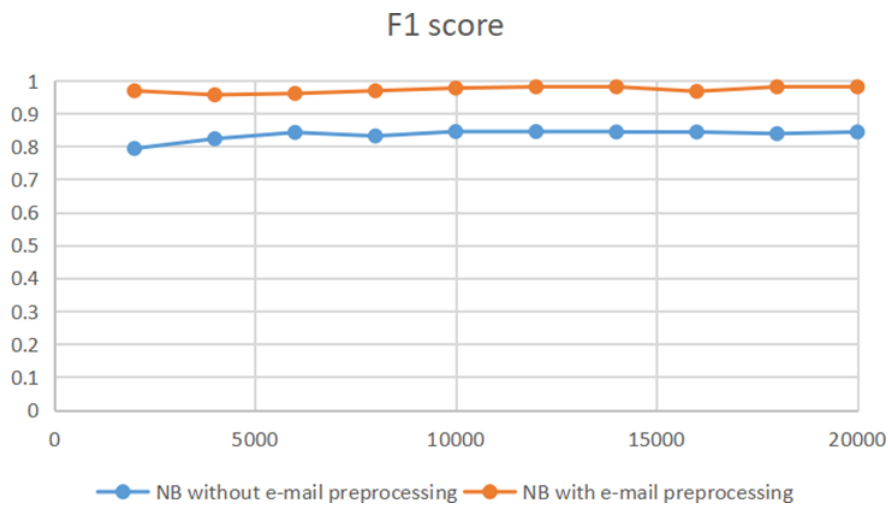


Figure 6. Influence of e-mail preprocessing

### 3.2.2. Comparison in different feature extraction methods

There are three feature extraction methods used in this research, which are term frequency, TF-IDF and feature hashing. The performance of a MNB filter implementing these three methods will be tested in five e-mail sets in terms of F1-score, each containing 500 e-mails. From the Table 2, it can be concluded that TF-IDF method is the best feature extraction method for a MNB filter. The F1-scores of TF-IDF and term frequency methods are similar, and the latter one is generally slightly inferior. While feature hashing has the advantages of low-dimensions and speed, it is not a good method compared with the other two in terms of the filter performance. In addition, TF-IDF method has less features compared with term frequency because of the addition of term weighting, which simplifies the generated sparse matrix.

Table 2. The performance of a MNB filter

E-mail set	Term frequency	TF-IDF	Feature hashing
1	0.9805	0.985	0.9324
2	0.9789	0.9504	0.8989
3	0.9537	0.9683	0.9362
4	0.9507	0.9741	0.9105
5	0.9524	0.9589	0.8724

### 3.2.3. Comparison between three Naïve Bayes filters

In this research, there are three types of NB algorithms implemented on the e-mail filters, which are MNB, BNB, and CNB. Applying TF-IDF method for feature selection, the 3 NB filters are compared on a corpus with 5,000 e-mails. As shown in the Figure 7, MNB has the best performance over the three NB filters. The precision and accuracy for MNB are both high, achieving around 97%. The recall of MNB is 98.6%,

meaning less than 2% spams pass the filter. CNB has a similar filtering performance compared with MNB, and its F1-score is just 0.4% less than MNB. In the last, BNB has the worst performance over three filters, while its precision of 99.2% is high.

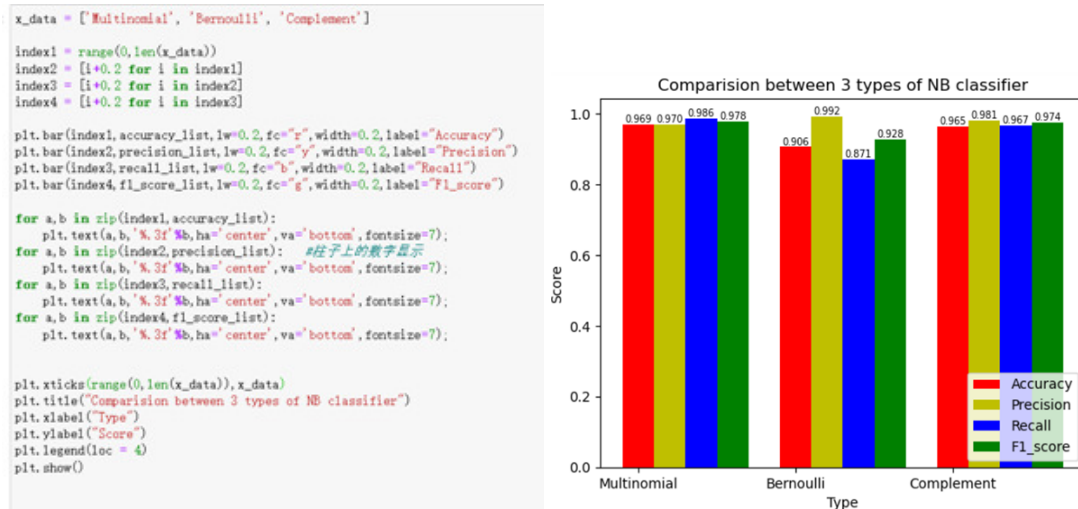


Figure 7. NB filters performance

### 3.3. SVM filter performance

Aforementioned, two key factors for SVM filters are the selection of kernel function and the value of C. In our research, 4 kernel functions of SVM are implemented, which are linear SVM, rbf SVM, polynomial SVM, and sigmoid SVM. Applying e-mail preprocessing and TF-IDF method, performances of these filters are shown in two Tables 3 and 4. Although TCRs for some of the SVM filters are very high, it is more reasonable to set  $\lambda$  as a high value in reality since the loss of misclassifying one legitimate e-mail to a spam is much larger than allowing a spam to pass the filter. Polynomial SVM has the best TCR when  $\lambda$  is set to 10, meaning that there are very few legitimate e-mails being misidentified. However, it is very sensitive to the change of C as its TCR drops to 0.5 when C is changed to 10 see in Figure 8. Normally, a TCR below 1 means the filter performance is so bad that the absence of filter would be considered better than using this filter. Moreover, the F1 score of polynomial SVM is the lowest among the four filters. Therefore, this kernel function is not suitable for the spam filter. Considering the other 3 kernel functions, they all have a high F1-score at around 99%, but the linear SVM has a better performance in terms of TCR. Therefore, linear SVM is considered to be the model with the best performance.

Table 3. SVM filters performance with C=1

Kernels	Precision (%)	Recall (%)	F1-score	Wacc (%) ( $\lambda = 1$ )	TCR ( $\lambda = 1$ )	Wacc (%) ( $\lambda = 10$ )	TCR ( $\lambda = 10$ )
Linear	0.9885	0.9942	0.9913	0.988	57.6	0.9778	8.23
RBF	0.9847	0.9913	0.988	0.9833	41.5	0.817	6.14
Polynomial	1	0.9055	0.9504	0.9347	10.6	0.9827	10.6
Sigmoid	0.9866	0.9952	0.9909	0.9873	54.6	0.9744	7.15

Table 4. SVM filters performance with C=10

Kernels	Precision (%)	Recall (%)	F1-score	Wacc (%) ( $\lambda = 1$ )	TCR ( $\lambda = 1$ )	Wacc (%) ( $\lambda = 10$ )	TCR ( $\lambda = 10$ )
Linear	0.9838	0.9942	0.989	0.9847	51.85	0.979	8.71
RBF	0.9894	0.9913	0.9903	0.9867	45.1	0.9689	5.89
Polynomial	0.8348	0.999	0.9096	0.8627	5.03	0.638	0.5
Sigmoid	0.9838	0.9961	0.9899	0.986	49.38	0.9693	5.96

```

from sklearn.svm import SVC
vectorizer = CountVectorizer(min_df=1, ngram_range=(1, 1), max_features=20000)
Content_bow = vectorizer.fit_transform(mailContent_list[0:5000])
Content_bowl = vectorizer.fit_transform(cutwords_list_deal[0:5000])
tfidf1 = TfidfVectorizer(token_pattern='(?u)\b*\b', max_features=50000, max_df=0.6, min_df=5)
Content_tfidf1 = tfidf1.fit_transform(cutwords_list_deal[0:5000])
Hashing = HashingVectorizer(n_features=20000, alternate_sign=False)
Content_hashing = Hashing.fit_transform(cutwords_list_deal[0:5000])
from sklearn.preprocessing import LabelEncoder
labelEncoder = LabelEncoder()
y_encode = labelEncoder.fit_transform(y)
y_encode1 = labelEncoder.fit_transform(y[0:5000])
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB, BernoulliNB, ComplementNB, GaussianNB
from sklearn.metrics import confusion_matrix, recall_score, precision_score, accuracy_score, f1_score
MNB = MultinomialNB(alpha=1)
BNB = BernoulliNB()
CNB = ComplementNB()
GNB = GaussianNB()
linear = SVC(kernel='linear', degree=4, C=1)
rbf = SVC(kernel='rbf', degree=4, C=10)
poly = SVC(kernel='poly', degree=4, C=10)
sigmoid = SVC(kernel='sigmoid', degree=4, C=10)

start = time.time() # 记录开始时间
accuracy_list, precision_list, recall_list, f1_score_list = [], [], [], []
for X in [Content_tfidf1]:
    X_train, X_test, y_train, y_test = train_test_split(X, y_encode1,
                                                    test_size=0.3,
                                                    random_state=45)
    for model_name, model in zip(['MNB', 'BNB', 'CNB'], [MNB, BNB, CNB]):
        model.fit(X_train, y_train) # 训练过程
        y_pred = model.predict(X_test) # 在测试集上计算召回率
        y_pred1 = model.predict(X_train)
        model_accuracy_score = accuracy_score(y_test, y_pred).round(4) # 在测试集上计算准确率
        model_precision_score = precision_score(y_test, y_pred).round(4)
        model_recall_score = recall_score(y_test, y_pred).round(4)
        model_f1_score = f1_score(y_test, y_pred).round(4)

        accuracy_list.append(model_accuracy_score)
        precision_list.append(model_precision_score)
        recall_list.append(model_recall_score)
        f1_score_list.append(model_f1_score)

        print(model_name+'Accuracy:' +str(model_accuracy_score))
        print(model_name+'Precision:' +str(model_precision_score))
        print(model_name+'Recall:' +str(model_recall_score))
        print(model_name+'F1 score:' +str(model_f1_score))
        print(confusion_matrix(y_test, y_pred))

end = time.time() # 记录结束时间
total = time_end - time_start
print(total)

MNB Accuracy:0.9687
MNB Precision:0.9696
MNB Recall:0.9855
MNB F1 score:0.9775
[[ 431  32]
 [ 15 1022]]
BNB Accuracy:0.906
BNB Precision:0.9923
BNB Recall:0.8708
BNB F1 score:0.9276
[[456  7]
 [124 903]]
CNB Accuracy:0.9647
CNB Precision:0.9814
CNB Recall:0.9672
CNB F1 score:0.9743
[[ 444  19]
 [ 34 1003]]
0.0

```

Figure 8. The filters performance with C=10

### 3.4. Comparison between Naïve Bayes and SVM filters

For the NB filter, MNB is considered to be the best model according to the former comparison. Its performance is compared with the linear SVM on a corpus containing 5,000 e-mails, with TF-IDF as the feature selection method. From the Table 5, although the two algorithms both have a relatively high F1-score, the linear SVM beats MNB in aspects of all the evaluation indexes. Both precision and recall of linear SVM are better than MNB, which results in a higher F1-score of above 99%. The weighted accuracy of linear SVM is about 3% higher than that of MNB, while its TCR is above 2 times higher than that of MNB at  $\lambda = 10$ . Overall, linear SVM has a better performance than MNB when applying them on spam filters. Looking back on the cores of these two machine learning algorithms, it is not surprising that SVM performs better than NB. NB is a linear classifier in the problem of spam classification based on equation 5. However, practical e-mails in the corpus may not be linearly separable.

In comparison, SVM solves this problem by applying kernel functions to map samples in the initial space to a feature space with higher dimensions, and it find a hyperplane to divide these transformed samples. Although SVM has a better filtering performance, NB filters have the advantage of easy implementation, which

results in less training time than SVM. For an e-mail filter, the consumed time for it to process the e-mails is very important because the e-mail filter receives a large number of spams every day.

Table 5. Linear SVM vs MNB

Algorithms	Precision (%)	Recall (%)	F1-score	Wacc (%) ( $\lambda = 10$ )	TCR ( $\lambda = 10$ )
Linear SVM	0.9885	0.9942	0.9913	0.9778	8.23
MNB	0.9696	0.9855	0.9775	0.9408	3.10

#### 4. CONCLUSIONS

As e-mail is commonly used by people all around the world, attacks on e-mail have become a serious problem in society, leading to hazards in many aspects. This thesis summarizes the hazards brought by spam and introduces different aspects of anti-spam topic, which include legislation, protocol improvement and e-mail filtering techniques. And this research focuses on the e-mail filtering techniques. To give an insightful investigation of them, the basic architecture and protocols of e-mail is firstly introduced. Then, different anti-spam filtering techniques are investigated. Can be concluded as, the filter performs 20% better with e-mail preprocessing compared with that without e-mail preprocessing. As for the feature selection methods, TF-IDF is found to be the best methods over 3 methods investigated. Besides, the comparison of MNB, BNB, and CNB shows that MNB is the NB e-mail filter with the best performance. To improve the SVM e-mail filter performance, the influences of the choice of kernel functions and the value of C are evaluated. The SVM e-mail filter with a linear kernel performs best when C is set as 10 in terms of F1-score and TCR. The linear SVM performs better in all the evaluation indexes, but MNB is easier to implement, and its training time is much less. The linear SVM e-mail filter achieves a F1-score of over 99%, a weighted accuracy of over 97% and TCR of 8.23 when  $\lambda$  is set to 10. As a future work, we consider three approaches, both the TF-IDF method and the NB algorithms do not consider the connections between words, which is bad for the filter performance. A new feature selection method called word2vec can be used to generate word vectors in the future, which in nature is a neural network model. The constructed filters need to be tested on different corpuses to test its generalization ability. And the headers of e-mails need to be considered in future work. In this research, only the e-mail body is tested; Many spams contain malicious contents in the form of images and hyperlinks, which the constructed e-mail filters are unable to detect. We will address this issue in the future.

#### ACKNOWLEDGEMENTS

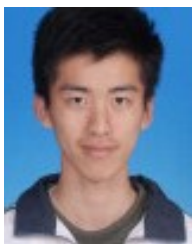
This paper is supported by Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou, China with the Research Development Fund (RDF-21-02-010).




#### REFERENCES

- [1] I. The Radicati Group, "Email statistics report, 2015-2019," *Email Statistics Report*, 2015. <http://www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019-Executive-Summary.pdf>.
- [2] M. Vergelis, T. Shcherbakova, T. Sidorina, and T. Kulikova, "Spam and phishing in 2019," *Securelist*, 2020. <https://securelist.com/spam-report-2019/96527/>.
- [3] FBI, "Cyber criminals conduct business email compromise through exploitation of cloud-based email services, costing US businesses more than 2 billion USD," *ic3*, 2020. <https://www.ic3.gov/Media/Y2020/PSA200406>.
- [4] D. J. Bernstein, "Internet mail 2000," *Internet mail*, 2000. <http://cr.yip.to/im2000.html>.
- [5] Federal Financial Institutions Examination Council, "Social media: consumer compliance risk management guidance," *Federal Register*, 2013. <https://www.fdic.gov/news/financial-institution-letters/2013/fil13056.html>.
- [6] I. Androustopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos, "Experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages," *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, pp. 160–167, 2000, doi: 10.1145/345508.345569.
- [7] J. Isacenkova and D. Balzarotti. "Measurement and evaluation of a real world deployment of a challenge-response spam filter", *Proc. of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, New York, NY, USA, pp. 413–426, 2011. <https://doi.org/10.1145/2068816.2068855>.
- [8] A. Ramachandran, N. Feamster, and S. Vempala, "Filtering spam with behavioral blacklisting," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2007, pp. 342–351, doi: 10.1145/1315245.1315288.
- [9] L. Polkowski, "Rough sets," *Rough Sets*, vol. 11, no. 8, pp. 41–356, 2002, doi: 10.1007/978-3-7908-1776-8.
- [10] W. Q. Zhao, Z. Y. Li, and G. Wei, "Information filtering model based on decision theoretic rough set theory," *Computer Engineering and Applications*, vol. 43, no. 7, 2007.
- [11] X. Carreras and L. Marquez, "Boosting trees for anti-spam email filtering," in *Proceeding of Euro Conference Recent Advances in NLP (RANLP 2011)*, 2001, [Online]. Available: <http://arxiv.org/abs/cs/0109015>.




- [12] W. W. Cohen, "Learning rules that classify e-mail," *Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access*, pp. 18–25, 1996, [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.8517>.
- [13] X. Liping, "Research on Chinese spam filtering technology based on content mining," Master dissertation, Dongbei University of Finance and Economics, China, 2010.
- [14] T. M. Cover, "Hart P E. nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, 1977.
- [15] Z. Junli and Z. Fan, "Application of improved KNN algorithm in spam e-mail filtering," *Data Analysis and Knowledge Discovery*, pp. 75–78, 2007, doi: 0.3969/j.issn.1003-3513.2018.04.018.
- [16] X. Jianfeng, L. Chengqi, and H. Chuanhua, "Research of anti-spam mail filtering technology and rough set-bayesian spam filtering algorithm," *Journal of Nanchang University (Natural Science)*, 2018.
- [17] H. Drucker, W. Donghui, V.N. Vapnik, "Support vector machines for spam categorization", *Neural Networks, IEEE Transactions*, vol. 10, no. 5, pp.1048-1054, 2000.
- [18] Universitas Waterloo "Programming languages group" <https://plg.uwaterloo.ca/~gvcormac/trecspamtrack06/> (accessed Oct. 7, 2023).
- [19] T. Xianqun and Z. Zhongmei, "KNN improved algorithm based on attribute value information entropy," *hangzhou Normal University*, 2010, doi: 10.3778/j.issn.1002-8331.2010.03.034.
- [20] S. Turner, R. Housley, *Implementing email security and tokens*, John Wiley and Sons, pp.296, 2008.
- [21] I. Androustopoulos and J. Koutsias, "An evaluation of Naïve bayesian anti-spam filtering," *Proceeding of the workshop on Machine learning in the New Information Age*, The European conference on Machine Learning Barceloma, Spain, pp. 9-17, 2010.
- [22] F. Z. Ruskanda, "Study on the effect of preprocessing methods for spam email detection," *Indonesian Journal on Computing (Indo-JC)*, vol. 4, no. 1, p. 109, 2019, doi: 10.21108/indojc.2019.4.1.284.
- [23] V. M. Nisha, R. Kumar, and Dr. Ashok, "Implementation on text classification using bag of words model," *Proceedings of the Second International Conference on Emerging Trends in Science and Technologies For Engineering Systems (ICETSE-2019)*, 2019, doi: 10.2139/ssrn.3507923.
- [24] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for IDF," *Journal of Documentation*, vol. 60, no. 5, pp. 503–520, 2004, doi: 10.1108/00220410410560582.
- [25] D. Moon, J. K. Lee, and M. K. Yoon, "Compact feature hashing for machine learning based malware detection," *ICT Express*, vol. 8, no. 1, pp. 124–129, 2022, doi: 10.1016/j.icte.2021.08.005.
- [26] Q. Luo, B. Liu, J. Yan, and Z. He, "Research of a spam filtering algorithm based on naïve bayes and AIS," in *Proceedings - 2010 International Conference on Computational and Information Sciences, ICCIS 2010*, Dec. 2010, pp. 152–155, doi: 10.1109/IC-CIS.2010.43.
- [27] S. Kadam, A. Gala, P. Gehlot, A. Kurup, and K. Ghag, "Word embedding based multinomial naïve bayes algorithm for spam filtering," in *Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018*, Aug. 2018, pp. 1–5, doi: 10.1109/ICCUBEA.2018.8697601.
- [28] N. F. Rusland, N. Wahid, S. Kasim, and H. Hafit, "Analysis of naïve bayes algorithm for email spam filtering across multiple datasets," *IOP Conference Series: Materials Science and Engineering*, vol. 226, no. 1, p. 012091, Aug. 2017, doi: 10.1088/1757-899X/226/1/012091.
- [29] F. Ye et al., "A spam classification method based on naïve bayes," in *IEEE 6th Information Technology and Mechatronics Engineering Conference, ITOEC 2022*, Mar. 2022, pp. 1856–1861, doi: 10.1109/ITOEC53115.2022.9734386.

## BIOGRAPHIES OF AUTHORS



**Yujia Fang**    received his bachelor's degrees in Information and Computer Science, from the University of Liverpool (UK) and Xi'an Jiaotong-Liverpool University, China in July 2023. His research interests include Information Security, Cybersecurity and some of the interdisciplinary fields joint together with techniques of computer science. He can be contacted at email: yujia.fang19@student.xjtlu.edu.cn.



**Gabriela Mogos**    is an Associate Professor of School of Advanced Technology, Department of Computing, Xi'an Jiaotong-Liverpool University, China. She received her Ph.D. in Computer Science from the Alexandru Ioan Cuza University of Iasi, Romania. Her research interests are in the areas of quantum computing, quantum cryptography and cybersecurity. She can be contacted at email: gabriela.mogos@xjtlu.edu.cn.