applied sciences

MDPI

Article

Enhancing Phishing Email Detection through Ensemble Learning and Undersampling

Qinglin Qi [1], Zhan Wang [1,*], Yijia Xu [1], Yong Fang [1] and Changhui Wang [2]

[1] College of Cybersecurity, Sichuan University, Chengdu 610065, China; qiqinglin@stu.scu.edu.cn (Q.Q.); xuyijia@stu.scu.edu.cn (Y.X.); yfang@scu.edu.cn (Y.F.)
[2] Department of Fundamental Courses, Chengdu Textile College, Chengdu 611731, China; 15397645041@163.com
* Correspondence: wangzscu@163.com

Abstract: In real-world scenarios, the number of phishing and benign emails is usually imbalanced, leading to traditional machine learning or deep learning algorithms being biased towards benign emails and misclassifying phishing emails. Few studies take measures to address the imbalance between them, which significantly threatens people's financial and information security. To mitigate the impact of imbalance on the model and enhance the detection performance of phishing emails, this paper proposes two new algorithms with undersampling: the Fisher–Markov-based phishing ensemble detection (FMPED) method and the Fisher–Markov–Markov-based phishing ensemble detection (FMMPED) method. The algorithms first remove benign emails in overlapping areas, then undersample the remaining benign emails, and finally, combine the retained benign emails with phishing emails into a new training set, using ensemble learning algorithms for training and classification. Experimental results have demonstrated that the proposed algorithms outperform other machine learning and deep learning algorithms, achieving an F1-score of 0.9945, an accuracy of 0.9945, an AUC of 0.9828, and a G-mean of 0.9827.

Keywords: phishing emails detection; imbalance; undersampling

Citation: Qi, Q.; Wang, Z.; Xu, Y.; Fang, Y.; Wang, C. Enhancing Phishing Email Detection through Ensemble Learning and Undersampling. *Appl. Sci.* **2023**, *13*, 8756. https://doi.org/10.3390/app13158756

Academic Editor: Mohamed Benbouzid

Received: 13 June 2023
Revised: 24 July 2023
Accepted: 25 July 2023
Published: 28 July 2023

1. Introduction

Humans are often perceived as the weakest link in cybersecurity defense, particularly due to social engineering attacks, such as phishing emails. Phishing emails often involve impersonating a trustworthy entity and utilizing urgency or emotional manipulation tactics to make the message appear authentic. The goal is to trick victims into providing sensitive information or clicking on attachments or links that can lead to further attacks. Phishing emails tend to target specific groups of people or exploit critical moments in time in reality, which only account for a small percentage. For example, since the outbreak of the novel coronavirus in late 2019, attackers have been exploiting people's fears by sending phishing emails that are closely related to COVID-19. As the COVID-19 pandemic has spread, people have become less sensitive to it, and phishing emails related to COVID-19 are no longer as common [1]. Before holidays, such as the traditional Mid-Autumn Festival in China, attackers take advantage of people's greed by posing as organizations and sending phishing emails claiming to offer free mooncakes, which are notoriously rare throughout the year [2]. Therefore, the proportion of benign emails to phishing emails is unbalanced in reality.

Phishing emails are also a significant method used by advanced persistent threats (APTs). Currently, over 80% of reported APT attacks involve phishing emails. According to the 2022 China Corporate Email Security Study, corporate email users in China received 42.59 billion phishing emails in 2022, an increase of 24.5% from 34.22 billion phishing emails in 2021 [3]. According to the Global Email Threat Report for 2022, the average number of phishing email attacks per 1000 email addresses worldwide was 299.27 per month, which represents a 12.36% increase from the previous year [4]. Recently, the cybersecurity firm

Cofense released the 2023 Email Security Report, which revealed a 569 percent surge in malicious email attacks in 2022 [5]. Therefore, there is an urgent need for an effective method to detect phishing emails.

With the emergence of machine learning (ML) and deep learning (DL) in recent years, numerous researchers have utilized them for detecting phishing emails. The steps can be summarized as follows: (1) selecting and extracting features; (2) choosing a machine learning classifier model; (3) training and testing the model. Email features generally fall into one of two categories: the email's contents and the email's body text. The former contains the structural properties of the email, while the latter contains the semantic features of the email body. The common ML and DL algorithms used to detect phishing emails include support vector machine (SVM) [6–11], naïve Bayes (NB) [6,12–14], decision tree (DT) [10,11,15–19], logistic regression (LR) [6,7,10], etc.

Although these algorithms have achieved good results in the field of detecting phishing emails, there are many limitations in the experimental design process [20–23], as follows: (1) Relying only on text-based features or content-based features may not provide comprehensive information; (2) The dataset used is outdated; (3) Using balanced datasets that do not match real-world scenarios; (4) Evaluation metrics are not comprehensive enough. Bountakas et al. propose HELPHED, which addresses the aforementioned issues and obtains satisfactory results [23]. Although they used imbalanced datasets and the experimental design was closer to real scenarios, they did not take any measures to address the imbalance between benign and phishing emails, which may have hindered the model performance and generalization ability. To investigate the impact of the imbalance between benign and phishing emails on the classifier performance and improve the detection rate of phishing emails, this paper presents the following contributions:

- To mitigate the impact of unbalanced datasets on classifier performance, two novel algorithms based on undersampling are proposed: FMPED and FMMPED. These algorithms differ in their approach to undersampling benign emails;
- To improve classification performance, we use an ensemble learning approach for handling the hybrid features of emails, which combines decision tree (DT) and support vector machine (SVM) as basic classifiers to train content-based features and text-based features respectively;
- To simulate real-life scenarios, the ratio of benign emails to phishing emails of the dataset used in this article is almost 10:1. In order to conduct a more comprehensive evaluation of the classifier performance, we utilize a variety of evaluation metrics that apply to an unbalanced dataset, including the F1-score, accuracy, AUC, G-mean, and Matthews correlation coefficient (MCC).

The structure of the remainder of this paper is as follows. In Section 2, we introduce research related to detecting phishing emails. In Section 3, we provide a detailed explanation of the entire process for detecting phishing emails using our method. In Section 4, detailed descriptions of the proposed algorithms FMPED and FMMPED are provided. In Section 5, we present the experiment results and discussion. In Section 6, we will summarize the main points of this paper and provide insights into future work.

2. Related Work

There are various methods for detecting phishing emails, including blacklist-based, machine learning, deep learning, natural language processing, and combinations of these techniques. We compile a list of recent studies on machine learning-based phishing email detection methods that have used both balanced and unbalanced datasets. Due to the fact that our subsequent work is based on [23], we also provide a detailed introduction to it.

2.1. Works Based on Balanced Datasets

Dutta et al., focused on designing a model for phishing email detection and classification using biogeography-based optimization with deep learning [24]. The dataset used in this article is sourced from the CLAIR dataset [25], which comprises 3685 phishing emails

and 4894 legitimate emails. The ratio of legitimate emails to phishing emails is 1.3:1. However, it is important to note that the dataset is quite old, dating back to 2008, and therefore may have limited relevance to current trends in email phishing. Magdy et al. proposed a three-classifier system based on deep learning to classify emails into legitimate, spam, and phishing categories based on content characteristics in the dataset [26]. The dataset used in the experiment consisted of 2758 legitimate emails and 2432 phishing emails, which is a ratio of almost 1:1.

Alhogail et al., proposed a model for classifying phishing emails that combines graph convolutional network (GCN) and natural language processing techniques [27]. The dataset used in the experiment is the fraud dataset [25], which includes 3685 phishing emails and 4894 legitimate emails. Two-thirds of the dataset was used for training. The remaining data are for testing purposes. The accuracy of detecting phishing emails in this dataset is 98.2%, with a false-positive rate of only 0.015.

Somesha et al., proposed a method for classifying phishing emails using a combination of word embedding and machine learning algorithms [28]. They used three datasets for the experiment, one of which was a balanced dataset containing 6295 benign emails and 9135 phishing emails, resulting in an accuracy of 99.5%. Valecha et al. [29] proposed a model for detecting phishing emails that utilizes Word2vec [30] and four machine learning classifiers, which takes into account gain and loss clues present in the emails. The dataset included 19,153 legitimate emails and 17,902 phishing emails. It achieved the highest accuracy for gain (96.52%), loss (96.16%), and a combined accuracy of 95.97%. It is evident that the datasets used in the mentioned work are close to balance or outdated, which is far from the actual scenario.

2.2. Work Based on Unbalanced Datasets

Bountakas et al., combined text-based and content-based email features, utilizing ensemble learning methods to classify emails [23]. One of the highlights of this paper is that it considers the actual situation and the development trend of phishing emails, constructing a more practical and closer to real-world scenario imbalanced dataset containing 32,051 benign emails and 3460 phishing emails. Fatima et al. collected 10,500 benign emails and 10,500 phishing emails as an initial dataset [31]. They then adjusted the ratio of benign emails to phishing emails from 1:1 to 1:10 to create an imbalanced test set and achieved an F1-score of 99.6% on an extremely unbalanced (1:10) ratio. Mehdi Gholampour et al. [32] developed an adversarial dataset using various legitimate text attack techniques. The dataset comprised 5092 legitimate emails and 568 phishing emails. It retrained phishing email detection models using this adversarial dataset. Results showed that the accuracy and F1-score of the models improved with subsequent attacks. It can be observed that although these experimental designs used imbalanced datasets that are closer to real-world scenarios, no measures were taken to reduce the impact of the imbalance on the classification performance.

2.3. Bountakas' HELPHED

From the input of the original email to the categorization of the final output email, HELPHED, a phishing email detection model proposed by Bountakas et al. [23], consists of six stages: the email parsing stage (S1), the content-based feature extraction stage (S2), the preprocessing stage (S3), the text-based feature extraction stage (S4), the feature selection stage (S5), and the ensemble classification stage (S6). The frame diagram of HELPHED is depicted in Figure 1.

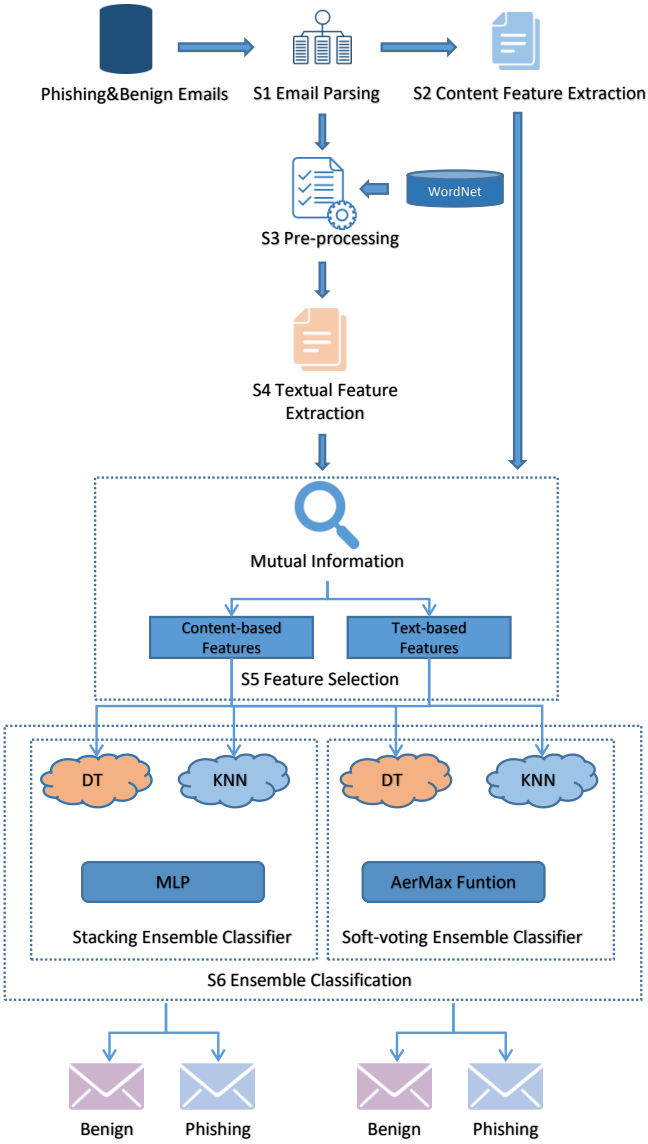

Figure 1. HELPHED architecture.

In Stage S1, the header and the body fields of an email are split and parsed to a vector structure, and they are associated with the respective email class according to the collection from which it originated (e.g., phishing or benign). Then, the content-based features of the message are extracted in the S2 phase, which includes several parts of the emails, such as body, syntax, header, and URL features. Subsequently, the texts are transformed into lowercase; the special characters, stopwords, and punctuation marks are deleted; the hyperlinks are converted to a consistent string; and the lemmatization process is employed in Stage S3. Later, Stage S4 utilizes the Word2vec method [30] to automatically extract the textual features. The innovation in this phase is that both content-based and text-based features are extracted and merged into a robust hybrid feature set for the detection of the newer sophisticated phishing emails. Then, the redundant features that are identified by the mutual Information method are excluded from the hybrid feature set. In this way, only the most informative features will be used in the classification stage. Finally, Stage S6 deploys the selected hybrid features, which offer a comprehensive representation of emails to feed the ensemble learning algorithms for the detection of phishing emails by predicting the email class (namely phishing or benign). The classification algorithms involved in Stage 6 include stacking ensemble learning (SEL) and soft-voting ensemble learning (SVEL). SEL comprises two tiers, where Tier 1 employs the DT and KNN algorithms to perform the

initial classification, and Tier 2 employs the MLP algorithm to detect whether an email is phishing or benign based on the classification results of Tier 1. SVEL employs DT and KNN algorithms for the initial classification in Tier 1, and a voting function is used to judge the class in Tier 2. In our work, we have improved SVEL with an undersampling strategy and provide a detailed description in Section 3.

3. Framework

In this section, we will provide a detailed introduction to the proposed algorithms for classifying phishing emails. The proposed algorithms utilize a combination of undersampling strategy and ensemble learning to enhance performance. Similar to HELPPED, the proposed algorithms have six phases, which include parsing the email (Step 1), extracting content features (Step 2), preprocessing and extracting textual features (Step 3), selecting features (Step 4), undersampling the dataset (Step 5), and ensemble learning classification (Step 6). It follows the framework illustrated in Figure 2.

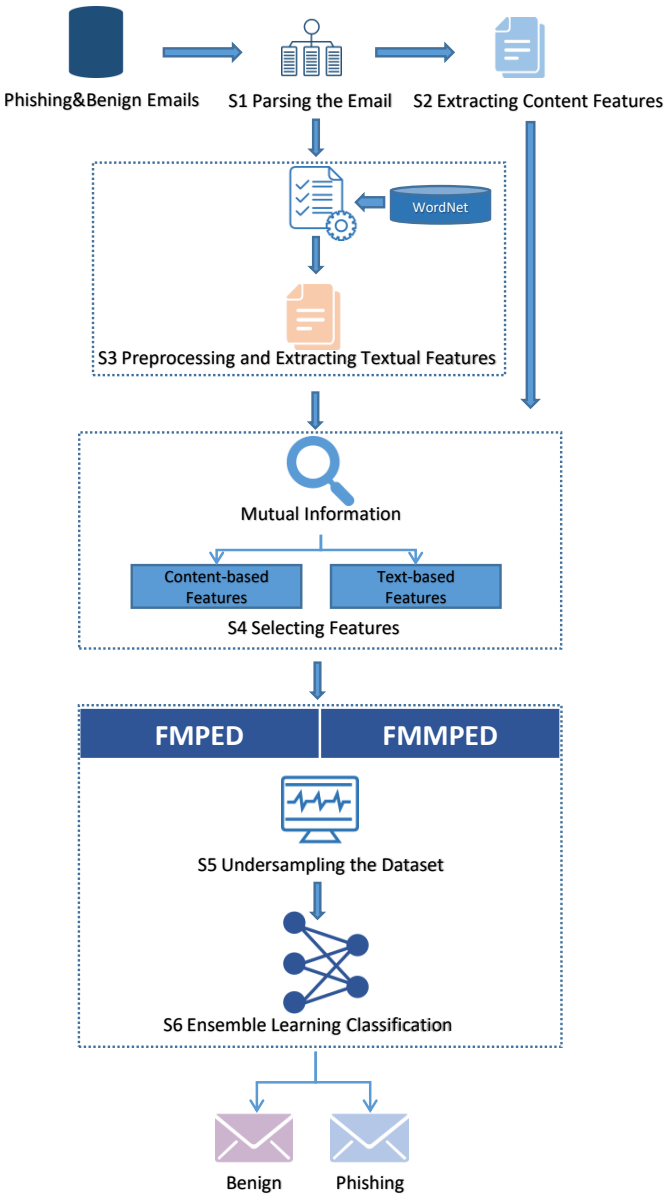

Figure 2. Proposed method framework.

It can be found that FMPED and FMMPED are primarily used for sampling the Bountakas HELPHED dataset and training the classifier. Therefore, the preprocessing, feature selection, and feature extraction of the dataset follow the same rules as HELPHED. In other words, our first four steps align with the first five steps outlined in the HELPHED framework mentioned in Section 2.3. The remainder of this section outlines the six steps mentioned above to illustrate the process of classifying phishing emails.

3.1. Parsing the Email

In Step 1, the header and body of each email are separated and transformed into a vector structure. These vector structures are stored in two distinct arrays, which are later combined with an array that represents the email's category (such as benign or phishing) to create a joint array.

3.2. Extracting Content Features

The main purpose of Step 2 is to extract content features from the header and body information in the joint array. Before the advent of NLP technology, the existing phishing email detection technology relied on email content-based features [33]. Bountakas et al., divided email content-based features into four categories, namely, body features, syntactic functions, headers features, and URL features. Body features mainly refers to the behavioral features that attackers try to deceive the victim, such as inserting malicious web pages, malicious files, etc., which include HTML code, HTML forms, scripts, attachments and image link, etc. Syntactic features mainly extract semantic information that phishing messages differ from benign messages, such as bad words in the email's body, the absence of "RE" in the subject, and the number of characters in the email body. Header features mainly contain information such as the sender, recipient, and encoding method of the message. URL features are obtained from the hyperlinks and domain names of the messages and are used to identify whether they contain malicious hyperlinks and special domain names, such as IP, number of hyperlinks, number of different href, and so on.

Bountakas et al. created four sets to store labels for the above four features in each email and then combined these sets into a matrix. The matrix contains labels corresponding to the above features of each email, as well as the label indicating whether the email is phishing or benign.

3.3. Preprocessing and Extracting Textual Features

Before extracting the text-based features from the email body, Bountakas et al., preprocessed the email body, mainly including the following steps: first separate the words in the email body; convert the words to lowercase; remove all stop words, special characters, punctuation, and HTML elements; replace the hyperlinks in the body with fixed strings (such as "URL" or "LINK"); tokenize the separated words [27]; split the words using delimiters; convert the text to a list; and, finally, to reduce the dimensionality of the email corpus, perform lemmatization and stemming using the WordNet database [34]. After preprocessing the email body text, Bountakas et al. utilized Word2vec's skip-gram method [30] to extract a total of 300 text-based features.

3.4. Selecting Features

The purpose of this step is to extract the most informative content-based and text-based features from the original feature set in order to enhance the accuracy of the classifier and prevent overfitting. Bountakas et al. proposed a filter-based method called mutual information (MI) and used it to extract 18 content-based features and 253 text-based features. The final hybrid feature set consists of a total of 271 features.

3.5. Undersampling the Dataset

The ratio of phishing emails to benign emails in the dataset utilized for this paper is almost 1:10, with 3465 phishing emails and 32,046 benign emails. This dataset is imbalanced;

therefore, it is in line with reality. In order to alleviate the impact of imbalanced factors on classification performance, this paper utilized two proposed undersampling methods to reduce the number of benign emails and obtain a new training set. The two algorithms proposed in this paper will be described in detail in Section 4.

3.6. Ensemble Learning Classification

Ensemble learning classifiers can integrate multiple basic machine learning classifiers and effectively process hybrid features. Specifically, the prediction results from each basic learning classifier are further fused to make a final classification prediction. This paper employs a widely recognized ensemble learning technique known as soft-voting ensemble learning [35]. Soft voting can eliminate the structural sensitivity generated by the base classifier and reduce the variance of the ensemble. There are several machine learning and deep learning algorithms that are suitable for the task of classifying phishing emails [20]. In this paper, we utilize the decision tree (DT) algorithm to process content-based features, while the support vector machine (SVM) is employed to process text-based features. It is worth noting that Bountakas et al. utilized the k-nearest neighbor (KNN) algorithm to analyze text-based features. This article uses the SVM algorithm instead of the KNN algorithm. The specific algorithm process will be detailed in Section 4.

4. Hybrid Ensemble Learning with Data Undersampling

The related works show that there have been many studies on phishing email classification. Some studies do not consider real-world scenarios and train their models on balanced datasets. Other studies, however, use imbalanced datasets that are closer to reality. They do not apply any sampling techniques to address class imbalance, leading to bias towards benign emails and the misclassification of phishing emails, resulting in significant losses. To investigate the impact of the imbalance between benign and phishing emails on the classifier performance and improve the detection rate of phishing emails, two novel ensemble learning with data undersampling algorithms, FMPED and FMMPED, are proposed, which are shown in Figures 3 and 4.

4.1. Soft-Voting Ensemble Learning

We choose soft-voting ensemble learning [23] as our ensemble learning method because of its superior performance. Soft-voting ensemble learning calculates the average of the output probabilities from two base learners, including decision tree (DT) and k-nearest neighbor (KNN), to obtain the final classification result, where the DT algorithm is responsible for handling content-based features, while the KNN algorithm is responsible for handling text-based features.

KNN predicts the class of a sample by calculating the distance between the sample and each training sample, then selects the class based on the votes of the k-nearest neighbors, which can be easily affected by outliers. Therefore, we consider using the SVM algorithm instead of the KNN algorithm to train text-based features, which is suitable for high-dimensional feature space problems and has a certain degree of robustness to noise and outliers.

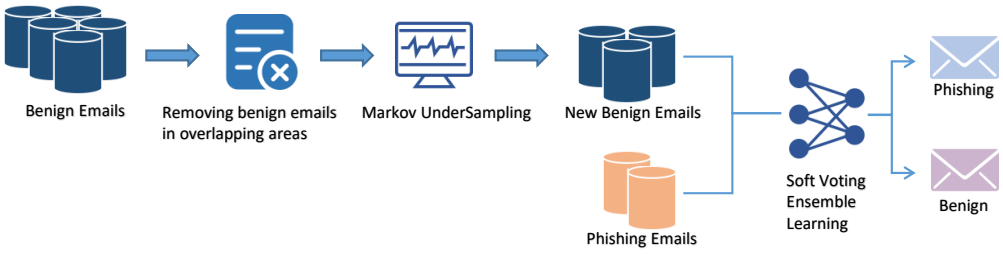

Figure 3. FMPED diagram.

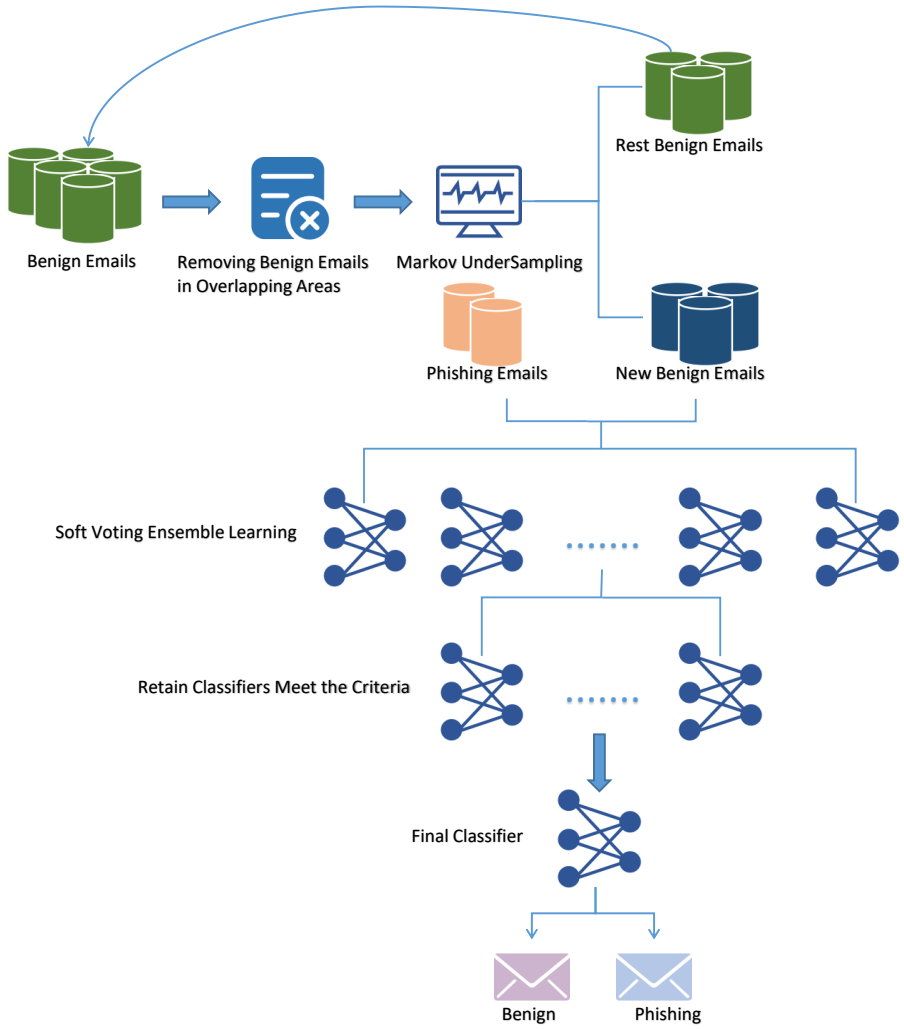

Figure 4. FMMPED diagram.

4.2. FMPED Algorithm

Suppose that $\mathcal{D}_{maj}$ represents the majority class training set, $\mathcal{D}_{min}$ represents the minority class training set, and $\mathcal{D}$ represents the whole training set. The FMPED algorithm includes three stages.

$$D = \mathcal{D}_{maj} \cup \mathcal{D}_{min}. \tag{1}$$

- Stage 1: Remove the overlapping majority class samples.

Liang et al. [36] point out that in imbalanced datasets, the overlapping regions containing majority and minority samples will cause the obtained classifier to deviate from the optimal classifier, resulting in a bias toward the majority class samples, thus leading to misclassification of minority class samples. It is necessary to remove those majority class samples located in overlapping regions before training. Figure 5 displays the overlapping area of the unbalanced dataset. Liang et al. [36] use the squared loss function to compute which majority class samples should be removed and adjust the position of classification boundary by Fisher linear discrimination (FLD) because it requires no hyperparameters. This process is repeated until the G-mean of the retained samples becomes smaller.

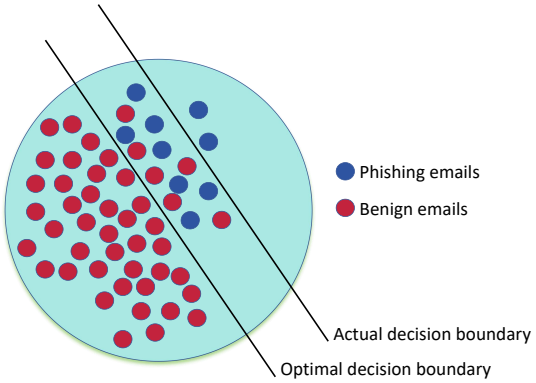

Figure 5. Overlapping area of unbalanced dataset.

Generally speaking, machine learning methods such as SVM and FLD obtain classifier hyperplanes that separate the two classes of samples. However, for unbalanced datasets, the classification hyperplanes obtained by these machine learning methods are likely to lie in the overlapping regions of the two types of samples. Therefore, unlike [36], the RMCSPV (removing majority class samples based on predicted value) algorithm removes samples based on the predicted values. The larger predicted value means the corresponding majority class sample may be closer to minority class samples. Moreover, samples are removed from the original majority class sample set, rather than from the set after the previous removal of each iteration. After removing selected samples, RMCSPV trains a classifier and calculates the G-mean value. This process ends when the maximum number of iterations is reached or the G-mean value decreases. This step can be seen as a preprocessing step before undersampling, so it is not necessary to set the maximum number of iterations to a large value. In this paper, we set it to 5. The specific process of RMCSPV is stated as Algorithm 1. All experimental results of RMCSPV are based on $\theta = 0.005$ in this article. Section 5 will discuss the choice of θ.

Algorithm 1: RMCSPV

Input: $\mathcal{D}_{maj}, \mathcal{D}_{min}, \mathcal{D}, \theta$

Output: $\hat{\mathcal{D}}, \hat{\mathcal{D}}_{maj}$

$\hat{\mathcal{D}} \leftarrow \mathcal{D}, k \leftarrow 1, G_0 \leftarrow 0$;

while $k \leq 5$ **do**

  Get a learning model f by training FLD with $\hat{\mathcal{D}}$, and compute G-mean value G_k of the set $\mathcal{D}$;

  if $G_k < G_{k-1}$ **then**

    break;

  Remove θ percent of samples from D_{maj} according to the ascending order of the value $e^{-f(x)}$ and denote it as $\hat{D}_{maj}$. $\hat{\mathcal{D}} \leftarrow \hat{D}_{maj} \cup \mathcal{D}_{min}$;

- Stage 2: Markov Undersampling

Stage 1 only removes a small fraction of the majority samples in the overlapping region, and the factor of imbalance still exists. So, we attempt to undersample the majority samples. Statistical learning theory introduced by Vapnik [37] indicates that the most "important" samples for classification are samples that are close to the classification hyperplane. So, we use Markov sampling to undersample the majority samples, such as in [36].

Different from Markov sampling in [36,38], the number of samples to be drawn in the Markov undersampling (MUS) algorithm changes dynamically, which eliminates the need for parameter selection. The specific process of the MUS algorithm can be stated as Algorithm 2. All experimental results of MUS are based on $q = 1.3, \hat{n} = 3, \bar{n} = 5$ in this article. Section 5 will discuss the choices of $q, \hat{n}, \bar{n}$.

Algorithm 2: MUS

Input: $\mathcal{D}_{maj}, \mathcal{D}_{min}, \mathcal{D}, q, \bar{n}, \hat{n}$

Output: $\hat{\mathcal{D}}, \hat{\mathcal{D}}_{maj}$

Let $N \leftarrow |D_{maj}|$, $j \leftarrow 1, \tilde{n} \leftarrow 0$. Take a sample z_j from $\mathcal{D}_{maj}$ randomly and denote its index as i;

while $j \leq N$ **do**

  $i \leftarrow i + 1$;

  if $i > N$ **then**

    $i \leftarrow i\%N$;

  Draw the sample at index i from $\mathcal{D}_{maj}$ as z_*;

  $P \leftarrow e^{-V(f,z_*)}/e^{-V(f,z_j)}$;

  if $\tilde{n} > \hat{n}$ **then**

    $P \leftarrow \min\{1, qP\}$;

  if $P \equiv 1$ *and* $y_j y_* = 1$ **then**

    $P \leftarrow e^{-y_* f}/e^{-y_j f}$;

  if $P > rand(1)$ *or* $\tilde{n} > \bar{n}$ **then**

    $z_j \leftarrow z_*, \hat{\mathcal{D}} \leftarrow z_j, j \leftarrow j + 1, N^{'} \leftarrow j, \tilde{n} \leftarrow 0$;

  if z_* *is not accepted* **then**

    $\tilde{n} \leftarrow \tilde{n} + 1$;

Obtain $\hat{\mathcal{D}} = \{z_j\}_{j=1}^{N^{'}} \cup D_{min}$, $\hat{\mathcal{D}}_{maj} = \mathcal{D}_{maj} - \{z_j\}_{j=1}^{N^{'}}$;

- Stage 3: Training stage

Similar to [23], features are divided into content-based features and text-based features. Content-based features are trained by DT, and text-based features are trained by linear SVM. The final classification model is obtained based on two base classifiers, as described in Section 4.1. All parameters of DT used in this paper are the same as [23] and the penalty parameter of the SVM is set to 10.

4.3. FMMPED Algorithm

Like the FMPED algorithm, the FMMPED algorithm also contains three stages as follows.

- Stage 1: Remove the overlapping majority class samples

Different from the RMCSPV algorithm, the RMCSBM (removing majority class samples based on Markov) method combines FLD and Markov sampling to sample majority class samples near a classification hyperplane. The main reason is [38] mentions that Markov sampling can obtain samples near the hyperplane. The specific process of Stage 1 can be stated as Algorithm 3.

- Stage 2: Markov Undersampling

This process is exactly the same as the MUS algorithm and will not be repeated here.

- Stage 3: Ensemble classifier

The FMMPED algorithm controls the number of undersampling rounds based on the imbalance ratio. In each round, the sampled majority class samples are combined with the minority class samples to train a classifier and compute the corresponding weight. If the sum of all metrics of the newly obtained classifier is higher than that of the previous classifier, it is accepted. Otherwise, it is rejected. Finally, the accepted classifiers corresponding to weights above the mean are integrated to obtain the final classifier. The specific process of the FMMPED algorithm can be stated as Algorithm 4. All experimental results of MUS are based on $r_1 = 0.15$ in this article. Section 5 will discuss the choice of r_1.

Algorithm 3: RMCSBM

Input: $\mathcal{D}_{maj}, \mathcal{D}_{min}, \mathcal{D}, \theta, q, \bar{n}, \hat{n}$
Output: $\hat{\mathcal{D}}_{maj}$

$N \leftarrow \theta * |D_{max}|$;
Get a learning model f by training FLD with $\mathcal{D}$;
Let $j \leftarrow 1, \tilde{n} \leftarrow 0$. Take a sample z_j from $\mathcal{D}_{maj}$ randomly;
while $j \leq N$ **do**
  Draw randomly a sample z_* from $\mathcal{D}_{maj}$;
  $P \leftarrow e^{-V(f,z_*)}/e^{-V(f,z_j)}$;
  if $\tilde{n} > \hat{n}$ **then**
    $P \leftarrow \min\{1, qP\}$;
  if $P \equiv 1$ *and* $y_j y_* = 1$ **then**
    $P \leftarrow e^{-y_* f}/e^{-y_j f}$;
  if $P > rand(1)$ *or* $\tilde{n} > \bar{n}$ **then**
    $z_j \leftarrow z_*, \hat{\mathcal{D}} \leftarrow z_j, j \leftarrow j+1, N' \leftarrow j, \tilde{n} \leftarrow 0$;
  if z_* *is not accepted* **then**
    $\tilde{n} \leftarrow \tilde{n} + 1$;
Let $\hat{\mathcal{D}}_{maj} = \mathcal{D}_{maj} - \{z_j\}_{j=1}^{N}$;

Algorithm 4: FMMPED Algorithm

Input: $\mathcal{D}_{maj}, \mathcal{D}_{min}, \mathcal{D}, r_1$
Output: f

$r \leftarrow \frac{|D_{maj}|}{|D_{min}|}, s \leftarrow r, o \leftarrow 0, flag \leftarrow 1, n \leftarrow 0$;
while $|\mathcal{D}_{max}| \neq 0$ **do**
  if $flag == 1$ **then**
    Put $\mathcal{D}$,$\mathcal{D}_{maj}$,$\mathcal{D}_{min}$ in Algorithm 3, get new $\mathcal{D}_{maj} \leftarrow \hat{\mathcal{D}}_{maj}$;
    $r \leftarrow \frac{|D_{maj}|}{|D_{min}|}, flag \leftarrow 0$;
  if $r \geq r_1 * s$ **then**
    Put $\mathcal{D}$,$\mathcal{D}_{maj}$,$\mathcal{D}_{min}$ in Algorithm 2, get $\hat{\mathcal{D}}$ and new $\mathcal{D}_{maj} \leftarrow \hat{\mathcal{D}}_{maj}$;
    Get a learning model f by training $\hat{\mathcal{D}}$ with Soft-voting ensemble learning, and calculate sum of F, A, R, G, M with $\mathcal{D}$;
    if $sum \geq o$ **then**
      $n \leftarrow n+1, w_n \leftarrow -0.5 * \log((5-sum)/(sum)), f_n = f$;
      o=sum;
    $r \leftarrow \frac{|D_{maj}|}{|D_{min}|}$;
Retain n_1 models with weights higher than $\frac{\sum_1^n w_n}{n}$;
$f = sign(\sum_1^{n_1} w_{n_1} * f_{n_1})$;

5. Experimental Section

In order to verify the effectiveness of the proposed methods, we provide comparative experiments in this section. Due to insufficient information provided by some studies such as [39] regarding datasets, experimental settings, features, and machine learning algorithms, ref. [23] mentions that it is difficult or impossible for others to reproduce their methods and results. Therefore, this paper chooses to compare with the two new phishing email classification algorithms: stacking ensemble learning (SEL) and soft-voting ensemble learning (SVEL) provided in [23]. Additionally, in order to demonstrate the superiority of the proposed algorithms, this paper also compares them with six excellent machine

learning algorithms or deep learning algorithms: decision tree (DT), random forest (RF), logistic regression (LR), Gaussian naïve Bayes (GNB), multilayer perceptron (MLP), and k-nearest neighbor (KNN) as presented in [23].

5.1. Dataset

Numerous previous studies use datasets that have several limitations, such as not taking into account the evolution of phishing email attacks but relying on outdated phishing emails (before 2015) rather than current ones; not considering that, in real life, the number of benign emails received by institutions is much higher than that of phishing emails, and the evaluation should be conducted on the unbalanced rather than on the balanced or nearly balanced dataset; not using different data sources; and lacking generalizability.

To avoid the drawbacks of these studies, ref. [23] created a dataset named HELPHED [40]. This dataset contains samples from the Enron email corpus [41], the SpamAssassin public corpus [42], the Nazario phishing corpus [43], and the [23] authors mailboxes. HELPHED merges 28,000 randomly selected emails from the Enron corpus and all benign emails from the SpamAssassin corpus. After removing duplicate emails, a total of 32,046 benign email samples are obtained. HELPHED uses all phishing emails in the Nazario phishing corpus from 2015 to 2020 (i.e., 1472 emails), 1992 emails that appeared before 1995, and 76 phishing emails in the authors' mailboxes as phishing emails. The imbalance ratio of HELPHED is close to 1:10, which is more similar to real-world scenarios than the datasets used in previous studies. To ensure the fairness of experimental results and the closeness to real-world scenarios, we use the HELPHED dataset in this paper.

5.2. Evaluation Metrics

For traditional balanced datasets, accuracy is the most commonly used evaluation metric. However, for unbalanced datasets, it would be biased to use accuracy alone to evaluate. This paper uses multiple evaluation metrics to comprehensively evaluate various algorithms. Let true positive (TP) represent the number of correctly identified phishing emails, true negative (TN) represent the number of correctly identified benign emails, false positive (FP) indicate the number of benign emails misclassified as phishing emails, and false negative (FN) indicate the number of phishing emails misclassified as benign emails.

- F1-Score: A more robust evaluation metric as it considers both precision and recall and is particularly useful when dealing with imbalanced datasets.

$$\text{F1-Score} = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{2}$$

  where $Recall = \frac{TP}{TP+FN}$ is used to measure the proportion of phishing emails that are correctly identified among all the phishing emails, and $Precision = \frac{TP}{TP+FP}$ indicates the proportion of emails correctly classified as phishing among all emails classified as phishing emails.

- Accuracy: The proportion of correctly classified emails with respect to the total number of emails.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3}$$

- G-mean: An important metric based on the geometric mean of the majority class accuracy and minority class accuracy. A higher G-mean value indicates a better decision boundary.

$$\text{G-mean} = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}}. \tag{4}$$

- Matthews Correlation Coefficient (*MCC*): A metric for evaluating binary classification performance. It combines the information from *TP*, *TN*, *FP*, and *FN*, making it suitable

for evaluating the ability of classifiers to handle imbalanced datasets. The *MCC* score ranges from -1 to 1, where a score of 1 represents a perfect classifier, 0 represents a random classifier, and -1 represents a classifier whose predictions are completely opposite to the true classes.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \quad (5)$$

- Area Under the ROC Curve (AUC): This term refers to the region of the ROC curve that is under the curve and is frequently used to assess the effectiveness and quality of binary classification models.

5.3. Experimental Setup

The experiments were conducted on an Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz, 3101 Mhz using Python 3.6.

We divided the dataset HELPHED into training set $\mathcal{D}_{train}$ and testing set D_{test} with a ratio of 7:3 and 8:2, respectively, and conducted 50 repeated experiments. Specifically, the training process can be described as follows: (1) For $\mathcal{D}_{train}$, we performed the FMPED algorithm, the FMMPED algorithm, and 8 other algorithms to obtain 10 classifiers. We then tested them on the same D_{test}. (2) We combined $\mathcal{D}_{train}$ and D_{test} and randomly split the combined dataset into a new training set $\mathcal{D}'_{train}$ and testing set D'_{test} according to a specified proportion. (3) We performed the above procedures 50 times and calculated the average and standard deviation of each metric.

5.4. Results for Proposed Algorithms

Based on the experimental results presented in Tables 1 and 2, it is evident that both FMPED and FMMPED algorithms outperform the other eight previous algorithms in all evaluation metrics.

Table 1. Experimental results when the ratio of training set to test set is 4:1.

Classifier	F1-Score	Accuracy	AUC	G-Mean	MCC
DT	98.50 ± 0.39	98.50 ± 0.39	95.36 ± 1.24	95.26 ± 1.28	91.40 ± 2.24
RF	98.75 ± 0.13	98.75 ± 0.13	93.88 ± 0.62	93.68 ± 0.66	92.71 ± 0.76
LR	90.26 ± **0.07**	90.26 ± **0.07**	50.51 ± 1.02	9.09 ± 5.35	5.82 ± 3.84
GNB	90.25 ± 0.20	90.25 ± 0.20	49.90 ± **0.10**	0.44 ± 1.35	1.11 ± 0.70
MLP	98.29 ± 0.80	98.29 ± 0.80	94.83 ± 3.80	96.63 ± 4.18	90.31 ± 4.71
KNN	94.62 ± 0.19	94.62 ± 0.19	77.27 ± 0.80	74.20 ± 1.06	65.69 ± 1.37
SEL	99.09 ± 0.11	99.10 ± 0.11	97.11 ± 0.67	97.07 ± 0.69	94.81 ± 0.70
SVEL	99.33 ± 0.10	99.33 ± 0.10	96.78 ± 0.46	96.72 ± 0.48	96.17 ± 0.58
FMPED	99.42 ± 0.11	99.42 ± 0.11	**98.28** ± 0.43	**98.27** ± 0.43	96.69 ± 0.65
FMMPED	**99.45** ± 0.10	**99.45** ± 0.10	98.11 ± 0.36	98.10 ± **0.37**	**96.87** ± **0.57**

Table 2. Experimental results when the ratio of training set to test set is 7:3.

Classifier	F1-Score	Accuracy	AUC	G-Mean	MCC
DT	98.44 ± 0.39	98.44 ± 0.39	95.26 ± 1.25	95.17 ± 1.30	91.11 ± 2.23
RF	98.68 ± 0.12	98.68 ± 0.12	93.55 ± 0.58	93.33 ± 0.62	92.32 ± 0.70
LR	90.26 ± 0.11	90.26 ± 0.11	50.70 ± 1.72	8.56 ± 8.88	5.31 ± 6.07
GNB	87.74 ± 6.21	87.74 ± 6.21	50.78 ± 2.25	7.56 ± 17.56	0.36 ± 3.58
MLP	98.42 ± 1.20	98.42 ± 1.20	94.76 ± 6.00	94.33 ± 7.68	90.83 ± 8.06
KNN	94.54 ± 0.18	94.54 ± 0.18	76.80 ± 0.78	73.57 ± 1.05	65.02 ± 1.34
SEL	99.09 ± 0.14	99.09 ± 0.14	97.02 ± 0.56	96.99 ± 0.58	94.80 ± 0.75
SVEL	99.32 ± 0.09	99.32 ± 0.09	97.02 ± 0.56	96.99 ± 0.58	96.11 ± 0.52
FMPED	99.36 ± 0.11	99.36 ± 0.11	**98.20** ± 0.31	**98.19** ± 0.32	96.35 ± 0.61
FMMPED	**99.41** ± **0.08**	**99.41** ± **0.08**	97.97 ± **0.30**	97.95 ± **0.31**	**96.62** ± **0.44**

Specifically, when the ratio of the training set to testing set is 4:1, the FMPED algorithm achieves an improvement of the AUC between 48.38% (GNB) and 1.17% (SEL), an improvement of the G-mean value between 97.83% (GNB) and 1.20% (SEL), and an improvement of the MCC between 95.58% (GNB) and 0.52% (SVEL). The FMMPED algorithm achieves an improvement of the AUC between 48.21% (GNB) and 1.00% (SEL), an improvement of the G-mean value between 97.66% (GNB) and 1.03% (SEL), and an improvement of the MCC between 95.76% (GNB) and 0.70% (SVEL). When the ratio of training set to testing set is 7:3, the FMPED algorithm achieves an improvement of the AUC between 47.50% (LR) and 1.18% (SEL), an improvement of the G-mean value between 90.63% (GNB) and 1.20% (SEL), and an improvement of the MCC between 95.99% (GNB) and 0.24% (SVEL). The FMMPED algorithm achieves an improvement of the AUC between 47.27% (GNB) and 0.95% (SEL), an improvement of the G-mean value between 90.39% (GNB) and 0.96%(SEL), and an improvement of the MCC between 96.26% (GNB) and 0.51% (SVEL).

5.5. Parameters Experiments

Table 3 presents the experimental results of FMPED for various q with $\bar{n} = 5$, $\hat{n} = 3$, $\theta = 0.005, r_1 = 0.15$, where q is chosen from $(1.1, 1.3, 1.5, 1.7)$. Table 4 provides the experimental results of FMPED for various $\bar{n}$ with $q = 1.3$, $\hat{n} = 3$, $\theta = 0.005$, $r_1 = 0.15$, where $\bar{n} = 5$ is chosen from $(5, 10, 15)$. Table 5 provides the experimental results of FMPED for various $\hat{n} = 3$ with $\bar{n} = 5$, $q = 1.3, \theta = 0.005$, $r_1 = 0.15$, where $\hat{n}$ is chosen from $(1, 2, 3, 4)$. Table 6 presents the experimental results of FMPED for various θ with $\bar{n} = 5$, $\hat{n} = 3, q = 1.3, r_1 = 0.15$, where θ is chosen from $(0.003, 0.005, 0.007, 0.009)$. Table 7 presents the experimental results of FMMPED for various r_1 with $\bar{n} = 5$, $\hat{n} = 3$, $q = 1.3$, $\theta = 0.005$, where r_1 is chosen from $(1.5, 2.0, 2.5, 3.0)$. Table 3 suggests that considering all metrics, the algorithm performs best when $q = 1.3$. Table 4 suggests that considering all metrics, the algorithm performs best when $\bar{n} = 5$. Table 5 suggests that considering all metrics, the algorithm performs best when $\hat{n} = 3$. For θ, Table 6 shows that there is no value that always achieves the best performance. By considering all the metric values in the two scenarios, we set $\theta = 0.005$. For r_1, by considering all the metric values in the two scenarios, it can be observed that the best performance is achieved when r_1 equals 0.15.

5.6. Discussion of Results

Since the FMPED and FMMPED algorithms are based on the idea of ref. [23], similar to the SEL and SVEL algorithms, they also use ensemble learning and hybrid features. Therefore, the FMPED and FMMPED algorithms perform better than the DT, RF, LG, GNB, MLP, and KNN algorithms that directly train all features. The SEL and SVEL algorithms proposed in ref. [23] do not take into account the impact of data imbalance on the model performance and do not perform any processing on the data, which may lead to the classification model being skewed towards benign emails and thus misclassifying malicious emails. However, the FMPED and FMMPED algorithms first remove the benign emails in the overlapping region that mislead for classification and then mitigate the degree of imbalance by using the undersampling technique to avoid the negative impact of data imbalance. In addition, when learning text-based features, we use SVM instead of the KNN algorithm, which has better robustness. Therefore, the FMPED and FMMPED algorithms perform better than SEL and SVEL. Since the idea of the FMPED algorithm is the same as that of FMMPED, the overall performances of the two are close to each other, but they are not exactly the same due to the different measures taken in removing benign emails from overlapping areas.

Table 3. Experimental results of FMPED for different q.

q / Data	4:1.				7:3.			
	1.1	**1.3**	**1.5**	**1.7**	**1.1**	**1.3**	**1.5**	**1.7**
F1-Score	**0.9940 ± 0.0009**	**0.9940 ± 0.0009**	0.9939 ± 0.0010	0.9937 ± 0.0010	0.9939 ± 0.0010	**0.9941 ± 0.0008**	0.9940 ± **0.0008**	0.9939 ± 0.0010
Accuracy	**0.9940 ± 0.0009**	**0.9940 ± 0.0009**	0.9939 ± 0.0010	0.9937 ± 0.0010	0.9939 ± 0.0010	**0.9941 ± 0.0008**	0.9940 ± **0.0008**	0.9939 ± 0.0010
AUC	0.9832 ± 0.0032	**0.9836 ± 0.0030**	0.9832 ± 0.0031	0.9821 ± 0.0032	0.9824 ± 0.0033	**0.9930** ± 0.0032	0.9823 ± **0.0030**	0.9825 ± 0.0036
G-mean	0.9831 ± 0.0033	**0.9835 ± 0.0030**	0.9831 ± 0.0031	0.9820 ± 0.0033	0.9823 ± 0.0034	**0.9929** ± 0.0032	0.9821 ± **0.0031**	0.9825 ± 0.0037
MCC	0.9660 ± **0.0051**	**0.9662 ± 0.0051**	0.9657 ± 0.0057	0.9645 ± 0.0057	0.9656 ± 0.0055	**0.9664 ± 0.0048**	0.9657 ± **0.0048**	0.9655 ± 0.0059

Table 4. Experimental results of FMPED for different $\bar{n}$.

$\bar{n}$ / Data	4:1.			7:3.		
	5	**10**	**15**	**5**	**10**	**15**
F1-Score	**0.9939 ± 0.0010**	0.9934 ± 0.0011	0.9932 ± 0.0012	**0.9940** ± 0.0012	0.9936 ± **0.0010**	0.9937 ± 0.0011
Accuracy	**0.9939 ± 0.0010**	0.9934 ± 0.0011	0.9932 ± 0.0012	**0.9940** ± 0.0012	0.9936 ± **0.0010**	0.9937 ± 0.0011
AUC	0.9827 ± **0.0027**	**0.9833** ± 0.0030	0.9830 ± 0.0034	0.9831 ± 0.0041	**0.9839 ± 0.0037**	0.9837 ± **0.0037**
G-mean	0.9826 ± **0.0028**	**0.9833** ± 0.0031	0.9829 ± 0.0034	0.9830 ± 0.0042	**0.9838 ± 0.0038**	0.9836 ± **0.0038**
MCC	**0.9655 ± 0.0056**	0.9628 ± 0.0060	0.9618 ± 0.0066	**0.9659** ± 0.0069	0.9638 ± **0.0058**	0.9642 ± 0.0064

Table 5. Experimental results of FMPED for different $\hat{n}$.

$\hat{n}$ / Data	4:1.				7:3.			
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
F1-Score	0.9938 ± 0.0012	0.9936 ± 0.0010	**0.9939 ± 0.0008**	**0.9939** ± 0.0011	**0.9941** ± 0.0010	0.9939 ± 0.0011	**0.9941** ± 0.0013	**0.9941 ± 0.0009**
Accuracy	0.9938 ± 0.0012	0.9936 ± 0.0010	**0.9939 ± 0.0008**	**0.9939** ± 0.0011	**0.9941** ± 0.0010	0.9939 ± 0.0011	**0.9941** ± 0.0013	**0.9941 ± 0.0009**
AUC	0.9823 ± 0.0034	0.9817 ± 0.0032	**0.9825 ± 0.0030**	**0.9825** ± 0.0035	**0.9834** ± 0.0036	0.9828 ± 0.0037	**0.9834** ± 0.0041	0.9830 ± **0.0032**
G-mean	0.9822 ± 0.0034	0.9816 ± 0.0033	**0.9824 ± 0.0030**	**0.9824** ± 0.0036	**0.9833** ± 0.0036	0.9827 ± 0.0037	**0.9833** ± 0.0041	0.9829 ± **0.0033**
MCC	0.9649 ± 0.0065	0.9637 ± 0.0055	**0.9654 ± 0.0048**	**0.9654** ± 0.0061	**0.9667** ± 0.0058	0.9652 ± 0.0063	**0.9667** ± 0.0071	0.9664 ± **0.0052**

Table 6. Experimental results of FMPED for different θ.

θ / Data	4:1.				7:3.			
	0.009	**0.007**	**0.005**	**0.003**	**0.009**	**0.007**	**0.005**	**0.003**
F1-Score	0.9937 ± 0.0011	0.9934 ± 0.0013	**0.9940 ± 0.0008**	0.9935 ± **0.0008**	0.9939 ± **0.0007**	**0.9942** ± 0.0010	**0.9942 ± 0.0007**	0.9941 ± 0.0010
Accuracy	0.9937 ± 0.0011	0.9934 ± 0.0013	**0.9940 ± 0.0008**	0.9935 ± **0.0008**	0.9939 ± **0.0007**	**0.9942** ± 0.0010	**0.9942 ± 0.0007**	0.9941 ± 0.0010
AUC	**0.9833** ± 0.0037	0.9822 ± 0.0036	0.9830 ± 0.0039	0.9816 ± **0.0032**	0.9812 ± 0.0025	0.9836 ± **0.0021**	0.9826 ± 0.0025	**0.9837** ± 0.0028
G-mean	**0.9833** ± 0.0037	0.9821 ± 0.0037	0.9829 ± 0.0040	0.9815 ± **0.0032**	0.9810 ± 0.0025	0.9835 ± **0.0021**	0.9825 ± 0.0025	**0.9836** ± 0.0028
MCC	0.9646 ± 0.0062	0.9624 ± 0.0073	**0.9658** ± 0.0046	0.9632 ± **0.0044**	0.9655 ± **0.0039**	0.9672 ± 0.0058	**0.9673** ± 0.0041	0.9665 ± 0.0054

Table 7. Experimental results of FMMPED for different r_1

r_1 / Data	4:1.				7:3.			
	1.5	**2.0**	**2.5**	**3.0**	**1.5**	**2.0**	**2.5**	**3.0**
F1-Score	0.9938 ± 0.0012	**0.9942** ± 0.0009	0.9941 ± 0.0009	0.9941 ± **0.0008**	**0.9941** ± 0.0008	0.9939 ± 0.0010	**0.9941 ± 0.0007**	**0.9941** ± 0.0008
Accuracy	**0.9943 ± 0.0008**	0.9942 ± 0.0009	0.9941 ± 0.0009	0.9941 ± **0.0008**	**0.9941** ± 0.0008	0.9939 ± 0.0010	**0.9941 ± 0.0007**	**0.9941** ± 0.0008
AUC	**0.9805 ± 0.0035**	0.9803 ± 0.0036	0.9794 ± **0.0035**	0.9793 ± 0.0037	0.9791 ± 0.0035	0.9785 ± 0.0042	0.9792 ± **0.0032**	**0.9793** ± 0.0034
G-mean	**0.9803** ± 0.0036	0.9802 ± 0.0037	0.9793 ± **0.0035**	0.9791 ± 0.0038	0.9789 ± 0.0035	0.9783 ± 0.0043	0.9790 ± **0.0032**	**0.9791** ± 0.0035
MCC	**0.9675** ± 0.0048	0.9670 ± 0.0050	0.9660 ± **0.0051**	0.9660 ± **0.0046**	**0.9663** ± 0.0049	0.9650 ± 0.0059	0.9662 ± **0.0042**	**0.9663** ± 0.0046

6. Conclusions

To reduce the impact of data imbalance on classification performance, we introduce the idea of undersampling benign emails for the soft-voting ensemble learning algorithm in this paper. We first propose the FMPED algorithm, which removes benign emails in the overlapping region based on the change of the G-mean value, and the decision boundary is subsequently adjusted using the FLD algorithm. Then, Markov sampling is used to selectively receive the retained benign emails. Only the important benign emails that are close to the hyperplane are sampled. The retained benign emails are merged with the phishing emails and trained by the soft-voting ensemble learning algorithm to obtain the final classifier. Based on the inspiration of the FMPED algorithm, we also propose the FMMPED algorithm, which uses Markov sampling for both removing and undersampling benign emails. Unlike FMPED, FMMPED determines the number of undersampled rounds according to the imbalance ratio. The undersampled benign emails are combined with phishing emails to obtain a classifier, and the weight is computed. If the weight is bigger than the last received classifier, it is retained. The final prediction for the testing sample is obtained by integrating the predictions of the retained classifiers with weights above the mean. Experimental results show that FMPED and FMMPED can obtain satisfactory results in F1-score, accuracy, AUC, G-mean, and MCC evaluation metrics compared with other eight phishing email classification algorithms.

There are still some issues to be investigated, such as exploring an undersampling strategy that applies to all phishing email algorithms and exploring a more effective way of locating the overlapping regions. All these questions are currently being researched and investigated.

Author Contributions: Conceptualization, Q.Q. and Z.W.; methodology, Q.Q. and Z.W.; software, Z.W.; validation, Q.Q., Z.W., and Y.X.; formal analysis, Q.Q.; investigation, Z.W.; resources, Y.F.; data curation, C.W.; writing—original draft preparation, Q.Q. and Z.W.; writing—review and editing, Q.Q. and Z.W.; visualization, Z.W.; supervision, Y.F.; project administration, Y.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the National Natural Science Foundation of China (U20B2045).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. INTERPOL Report Shows Alarming Rate of Cyberattacks during COVID-19. Available online: https://www.interpol.int/News-and-Events/News/2020/INTERPOL-report-shows-alarming-rate-of-cyberattacks-during-COVID-19 (accessed on 4 September 2020).
2. The University of Science and Technology of China Sent 40,000 "Free Mooncake Giveaway" Phishing Emails. Available online: https://www.thepaper.cn/newsDetail_forward_19819224 (accessed on 8 August 2022).
3. 2022 China Corporate Email Security Study. Available online: https://www.qianxin.com/threat/reportdetail?report_id=294 (accessed on 27 March 2023).
4. Global Email Threat Report for 2022. Available online: http://mailsec.cn/news/html/?539.html (accessed on 31 January 2023).
5. 2023 Email Security Report. Available online: https://cofense.com/blog/phishing-emails-increased-in-2022-according-to-annual-report-from-cofense/ (accessed on 29 March 2023).
6. Verma, P.; Goyal, A.; Gigras, Y. Email phishing: Text classification using natural language processing. *Comput. Sci. Inf. Technol.* **2020**, *1*, 1–12. [CrossRef]
7. Vinayakumar, R.; Soman, K.P.; Poornachandran, P.; Mohan, V.S.; Kumar, A.D. ScaleNet: Scalable and hybrid framework for cyber threat situational awareness based on DNS, URL, and email data analysis. *J. Cyber Secur. Mobil.* **2019**, *8*, 189–240. [CrossRef]
8. Kumar, A.; Chatterjee, J.M.; Díaz, V.G. A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing. *Int. J. Electr. Comput. Eng.* **2020**, *10*, 486. [CrossRef]

9. Niu, W.; Zhang, X.; Yang, G.; Ma, Z.; Zhuo, Z. Phishing emails detection using CS-SVM. In Proceedings of the IEEE International Symposium on Parallel and Distributed Processing with Applications and IEEE International Conference on Ubiquitous Computing and Communications, Guangzhou, China, 15 December 2017.
10. Hamisu, M.; Mansour, A. Detecting advance fee fraud using nlp bag of word model. In Proceedings of the IEEE 2nd International Conference on Cyberspac, Nagoya, Japan, 26–29 June 2020.
11. Junnarkar, A.; Adhikari, S.; Fagania, J.; Chimurkar, P.; Karia, D. E-mail spam classification via machine learning and natural language processing. In Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, Tirunelveli, India, 4–6 February 2021.
12. Castillo, E.; Dhaduvai, S.; Liu, P.; Thakur, K.S.; Dalton, A.; Strzalkowski, T. Email threat detection using distinct neural network approaches. In Proceedings of the 1st International Workshop on Social Threats in Online Conversations: Understanding and Management, Marseille, France, 10 May 2020.
13. Peng, T.; Harris, I.; Sawa, Y. Detecting phishing attacks using natural language processing and machine learning. In Proceedings of the IEEE 12th International Conference on Semantic Computing, Laguna Hills, CA, USA, 31 January–2 February 2018.
14. Unnithan, N.A.; Harikrishnan, N.B.; Vinayakumar, R.; Soman, K.P.; Sundarakrishna, S. Detecting phishing E-mail using machine learning techniques. In Proceedings of the 1st Anti-Phishing Shared Task Pilot 4th ACM IWSPA Co-Located 8th ACM Conference on Data and Application Security Privacy, Tempe, AZ, USA, 21 March 2018.
15. Swetha, M.S.; Sarraf, G. Spam email and malware elimination employing various classification techniques. In Proceedings of the 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology, Bangalore, India, 17–18 May 2019.
16. Chowdhury, M.U.; Abawajy, J.H.; Kelarev, A.V.; Hochin, T. Multilayer hybrid strategy for phishing email zero-day filtering. *Concurr. Comput. Pract. Exper.* **2017**, *29*, e3929. [CrossRef]
17. Harikrishnan, N.B.; Vinayakumar, R.; Soman, K.P. A machine learning approach towards phishing email detection. In Proceedings of the Anti-Phishing Pilot at ACM International Workshop on Security and Privacy Analytics, Tempe, AZ, USA, 21 March 2018.
18. Rastenis, J.; Ramanauskaitė, S.; Suzdalev, I.; Tunaitytė, K.; Janulevičius, J.; Čenys, A. Multi-Language spam/Phishing classification by Email Body text: Toward automated security Incident investigation. *Electronics* **2021**, *10*, 668. [CrossRef]
19. Sharma, V.D.; Yadav, S.K.; Yadav, S.K.; Singh, K.N.; Sharma, S. WITHDRAWN: An effective approach to protect social media account from spam mail—A machine learning approach. *Mater. Today Proc.* **2020**, *12*, 377. [CrossRef]
20. Das, A.; Baki, S.; El Aassal, A.; Verma, R.; Dunbar, A. SoK: A comprehensive reexamination of phishing research from the security perspective. *IEEE Commun. Surv. Tut.* **2019**, *22*, 671–708. [CrossRef]
21. El Aassal, A.; Baki, S.; Das, A.; Verma, R.M. An in-depth benchmarking and evaluation of phishing detection research for security needs. *IEEE Access* **2020**, *8*, 22170–22192. [CrossRef]
22. Gangavarapu, T.; Jaidhar, C.D.; Chanduka, B. Applicability of machine learning in spam and phishing email filtering: Review and approaches. *Artif. Intell. Rev.* **2020**, *53*, 5019–5081. [CrossRef]
23. Bountakas, P.; Xenakis, C. Helphed: Hybrid Ensemble Learning Phishing Email Detection. *J. Netw. Comput. Appl.* **2023**, *210*, 103545. [CrossRef]
24. Dutta, A.K.; Meyyappan, T.; Qureshi, B.; Alsanea, M.; Abulfaraj, A.W.; Al Faraj, M.M.; Sait, A.R.W. Optimal Deep Belief Network Enabled Cybersecurity Phishing Email Classification. *Comput. Syst. Sci. Eng.* **2023**, *44*, 2701–2713. [CrossRef]
25. Clair Collection of Fraud Email, ACL Data and Code Repository. Available online: http://aclweb.org/aclwiki (accessed on 8 June 2008).
26. Magdy, S.; Abouelseoud, Y.; Mikhail, M. Efficient spam and phishing emails filtering based on deep learning. *Comput. Netw.* **2022**, *206*, 108826. [CrossRef]
27. Alhogail, A.; Alsabih, A. Applying machine learning and natural language processing to detect phishing email. *Comput. Secur.* **2021**, *110*, 102414. [CrossRef]
28. Somesha, M.; Pais, A.R. Classification of Phishing Email Using Word Embedding and Machine Learning Techniques. *J. Cyber Secur. Mobil.* **2022**, 279–320.
29. Valecha, R.; Mandaokar, P.; Rao, H.R. Phishing email detection using persuasion cues. *IEEE. Trans. Depend. Secure Comput.* **2021**, *19*, 747–756. [CrossRef]
30. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
31. Qachfar, F.Z.; Verma, R.M.; Mukherjee, A. Leveraging synthetic data and pu learning for phishing email detection. In Proceedings of the 12th ACM Conference on Data and Application Security and Privacy, Baltimore, MD, USA, 24–27 April 2022.
32. Mehdi Gholampour, P.; Verma, R.M. Adversarial Robustness of Phishing Email Detection Models. In Proceedings of the 9th ACM International Workshop on Security and Privacy Analytics, Charlotte, NC, USA, 26 April 2023.
33. Moradpoor, N.; Clavie, B.; Buchanan, B. Employing machine learning techniques for detection and classification of phishing emails. In Proceedings of the Computing Conference, London, UK, 18–20 July 2017.
34. Miller, G.A. WordNet: A Lexical Database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
35. Dietterich, T.G. Ensemble methods in machine learning. In Proceedings of the Multiple Classifier Systems: 1st International Workshop, Cagliari, Italy, 21–23 June 2000.

36. Liang, T.; Xu, J.; Zou, B.; Wang, Z.; Zeng, J. LDAMSS: Fast and efficient undersampling method for imbalanced learning. *Appl. Intell.* **2022**, *52*, 6794–6811. [CrossRef]
37. Vapnik, V.N. An overview of statistical learning theory. *IEEE Trans. Neural Netw.* **1999**, *10*, 988–999. [CrossRef] [PubMed]
38. Wang, Z.; Liang, T.; Zou, B.; Cai, Y.; Xu, J.; You, X. Incremental Fisher linear discriminant based on data denoising. *Knowl.-Based Syst.* **2022**, *237*, 107799. [CrossRef]
39. Egozi, G.; Verma, R. Phishing email detection using robust nlp techniques. In Proceedings of the IEEE International Conference on Data Mining Workshops, Singapore, 17–20 November 2018.
40. Helphed's Data. Available online: https://drive.google.com/drive/my-drive (accessed on 2 September 2021).
41. Enron Email Dataset. Available online: http://www.cs.cmu.edu/~./enron/ (accessed on 2 November 2020).
42. SpamAssassin Public Corpus. Available online: https://spamassassin.apache.org/old/publiccorpus/ (accessed on 2 September 2018).
43. Nazario Phishing Corpus. Available online: https://monkey.org/~jose/phishing/ (accessed on 2 November 2020).