

Email Spam Filtering Methods: Comparison and Analysis

Jun Deng*

School of Engineering, The University of Manchester, Manchester, UK

*Corresponding author: jun.deng-4@student.manchester.ac.uk

Abstract. Email is a common way of communication due to its cheap cost, efficacy, and efficiency. With the emerging of deep learning and machine learning methods, spam filter classification achieves boosting performance with fast inference speed. However, individuals and email servers are affected by spam, which causes issues with wasted time and computer storage space, as well as adverse effects on bandwidth. Even worse, email users are susceptible to scams and fraud that may result in financial loss. Therefore, it is essential to discover an efficient approach to filter spam from the entire number of emails. The purpose of this study is to evaluate and contrast the five most popular machine learning based spam filtering techniques, including Naive Bayes, Supported Vector Machine K-Nearest Neighbor, and XGBoost. We evaluate them based on their performance and efficacy. We hope this paper will help to conclude the current condition and help the researchers to improve better algorithms with higher accuracy.

Keywords: Email spam filtering; Naïve Bayes; SVM; KNN; XGBoost.

1. Introduction

Email is a form of everyday communication that is inexpensive, efficient, quick, and convenient. On the other hand, spam, which consists of large quantities of unwanted and dangerous emails, has become a major concern. The individual who sends unsolicited email is known as a spammer. They get email addresses using websites, chatrooms, and even malware [1]. Spam may waste users' time, hard drive space, CPU power, and network bandwidth. Worse, some individuals have experienced money loss as a result of email scams or fraud. It is also detrimental to the memory space and communication bandwidth of email servers [2]. The ratio of worldwide spam over total email traffic from January 2014 to December 2021 is shown in Figure 1. In April 2014, 71.1% of all email traffic was comprised of spam, the highest level ever recorded. Thankfully, the general tendency has begun to reduce [3]. It is crucial to find a solution to this vexing issue affecting both platform and regular consumers. To combat this dangerous threat, Gmail, Yahoo Mail, and Outlook have used machine learning algorithms to filter spams. These procedures are accurate and effective. They are able to learn, recognise, and categorise spam email via a massive number of messages and machines. There are a few emails for which spam filtering techniques have been used. The objective of this study is to examine and contrast five contemporary methods.

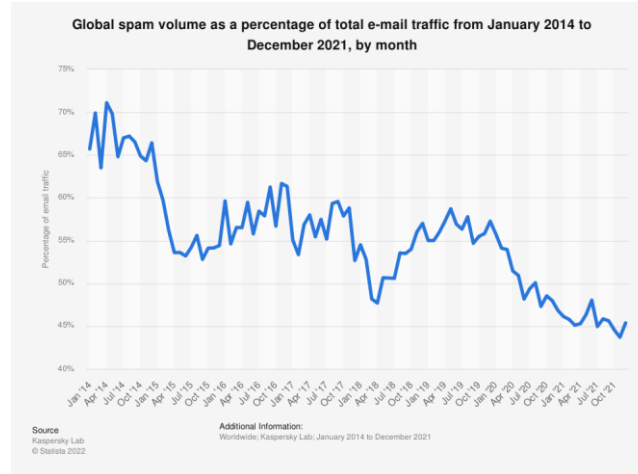


Fig.1 Global spam volume as a percentage of total email traffic from Apr. 2014 to Dec. 2021, by month. (Source: Statista 2022)

2. Spam Classification Algorithms

For spam categorization, decision tree, support vector machine (SVM), deep neural network (DNN) were used. Literature indicates that the aforementioned approaches may also be utilised as a content-based filtering tool by finding certain characteristics (e.g., keywords frequency). The threshold value is the rate used to determine if a message is spam. If the frequency of certain characteristics in an email exceeds the threshold value, the email will be categorised as spam [4].

[5] categorised all spam filtering strategies into the following five categories: content-based filtering, case-based filtering, heuristic or rule-based filtering, prior similarity-based filtering, and adaptive spam filtering methods. They included descriptions of each approach and examples for each category.

2.1. Bayes

The Bayesian algorithm is a probability-based theory of possibility reasoning named after the British mathematician Thomas Bayes (1702-1763). In 1763, he also developed Bayesian statistical theory, which predicts the chance of a test roll happening depending on the amount of time that has passed [6]. Bayesian theory believes that if the result of an event is unclear, the probability of the occurrence may be used to characterise it. If the likelihood of the occurrence of events in previous tests is known, the probability of the occurrence of events in future tests may be predicted using mathematical probability theory [7].

Bayes' theorem is developed from fundamental probability theory notions. The original Bayes theorem may be shown in equation form (1). $P(A|B)$ is the probability that two distinct occurrences A and B occur simultaneously, or, more succinctly, the chance that A and B intersect.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (1)$$

The values of $P(A)$ and $P(B)$ must never equal zero in order for the aforementioned equations to exist and for the proof to proceed with the computational derivation. In order to apply Bayes' theorem, this condition must be satisfied in every instance. Alternative variations of Bayes' theorem are often found when there are two contradictory assertions or hypotheses.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|A')P(A')} \quad (2)$$

In the following equation, $P(A')$ is the probability equivalent to $P(A)$, hence $P(A') = (1 - P(A))$. And if there is a partition of the sample space, an extended version based on Bayesian theory may be used as an alternative (A_i) denotes the partitioning of sample space).

$$P(A|B) = \frac{P(B|A)P(A)}{\sum_i P(B|A_i)P(A_i)} \quad (3)$$

2.1.1 Naive Bayes based method

In testing and experimentation, the Bayesian method computes each unique sample by categorising test samples into classes to which they may belong depending on their features. When the Bayesian method is put to a real-world scenario, it is discovered to be difficult to implement. The Naive Bayes classifier has a reasonably basic structure and is extremely practical, although it uses just test sample data for estimates and classification. Currently, the Naive Bayes classification approach is frequently used in online applications including document classification, spam classification, and comment spam [5]. Based on the conditional independence assumption of the Bayesian network for conditional probabilities, it is the Bayesian network's simplest special instance [8]. The derivation of the Naive Bayes equation starts with the following formula, assuming that each function in the formula is independent of the others:

$$P(X = x|Y = ck) = P(X^{(1)} = x^{(1)}, \dots, X^{(n)} = x^{(n)}|Y = ck) = \prod_{j=1}^n P(X^{(j)} = x^{(j)}|Y = ck) \quad (4)$$

According to the classification theory of Naive Bayes First, an instance x is provided, and its feature vector is determined $(x_1, x_2, \dots, x_n)$. Therefore, it is known that when x belongs to the category ck , the expression:

$$P(y = ck|x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n|y=ck)}{P(x_1, \dots, x_n)} \quad (5)$$

The sample X is subsequently provided, and its posterior probability is computed. Then, based on the following equation, it is divided by the greatest posterior probability.

$$P(y = ck|X = x) = \frac{P(X=x|Y=ck)P(Y=ck)}{\sum_1^k P(X=x|Y=ck)P(Y=ck)} \quad (6)$$

Through this, the analytical formula for Naive Bayesian classification may be derived:

$$y = f(x) = \underset{ck}{agr\ max} = \frac{P(Y=ck) \prod_j P(X^{(j)}=x^{(j)}|Y=ck)}{\sum_k P(Y=ck) \prod_j P(X^{(j)}=x^{(j)}|Y=ck)} \quad (7)$$

Naive Bayes is well acknowledged as a simple and computationally efficient method that solves actual issues satisfactorily. Classifiers based on the Naive Bayes algorithm are well suited for text classification since they generate superior results when presented with multiclass problems and independent rules. And compared to other machine learning algorithms, the Naive Bayes classifier has a better success rate. Due to its evident benefit, it is commonly used in spam filtering systems.

The majority of data in the actual world is incomplete, having aggregates, noisy values, and missing values, therefore [9] separated the method to spam processing into three steps. The author firstly trains the filter to eliminate terms such as conjunctions and articles from emails. Because these terms have no categorization value. The method is then implemented by using the feature selection algorithm.

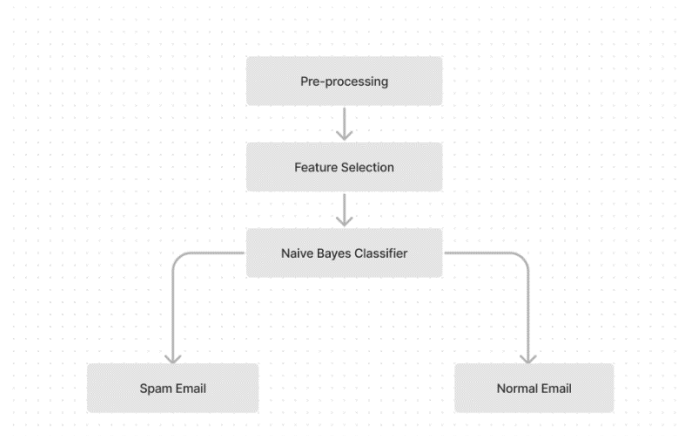


Fig. 2 The stage of Spam filtering.

The preceding flowchart, Figure 2, illustrates how the software operates. Here, the performance of the Naive Bayes method for spam filtering is evaluated using two datasets including spam and non-spam. This dataset is quite small, yet it functions well in this aspect. Then, [10] conducted numerous tests on a test sample of 1,293 emails and discovered that just 22 emails were false positives, leading in a 98% accuracy in Naive Bayes filtering. Conclusion: the classifier utilising the Naive Bayes algorithm plays a significant part in the process of spam filtering. In addition, the quality of a classifier's performance relies on the database's data samples. When the dataset comprises fewer email instances and characteristics, the Naive Bayes classifier may deliver high performance for this data. In addition, if the dataset was acquired from a single email account, the Naive Bayes classifier had the greatest accuracy for spam management block predictions.

2.2. Spam Filtering by Fusing K-Nearest Neighbors(KNN)and Support Vector Machine (SVM)

2.2.1 K-Nearest Neighbors (KNN)

KNN is a distribution-free method that does not assume that the data follows a particular probability distribution [11, 12]. The classification model in knn's model is not data-driven; instead, it compares the test sample to the K training values and classifies each sample based on its resemblance to the K closest neighbours [12]. KNN is classified as a lazy learner since it does not generalise from training data points. Due to the lack of generality, the KNN algorithm's classifier must keep every training data in distinct samples. To be exact, the classifier of the knn algorithm requires all training data for the duration of the testing phase, since all classification judgments are made after sorting and evaluating all training data sets.

In addition, KNN is a sort of supervised learning that has many applications in image classification, data mining, and other areas. The optimal distance metric for calculating KNN is the Euclidean distance, which may be computed as follows [13]. $x_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5} \dots \dots x_{in})$ and $x_j = (x_{j1}, x_{j2}, x_{j3}, x_{j4}, x_{j5} \dots x_{jn})$ distance calculated as follow:

$$D(XI, XJ) = \sqrt{\sum_{k=1}^n ((x_{ik} - x_{jk}))^2} \quad (8)$$

x_i and x_j are the two vectors.

2.2.2 Support Vector Machine(SVM)

SVM is a binary classification model that translates the feature vector of an instance to a set of samples in space. The aim of SVM is to construct a line to differentiate the two kinds of data, such that if more samples are introduced in the future, this line will still be able to classify them accurately. SVM is useful for classification issues involving small to larger samples of nonlinear, high-dimensional data.

Vladimir N. et al initially introduced SVM in 1963 and the current version (soft margin) was presented by Corinna Cortes et al. in 1995. Prior to the introduction of deep learning, SVM was regarded as the most effective and successful algorithm during the last decade.

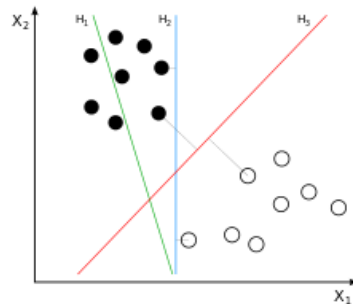


Fig.3 SVM example.

In this illustration, Fig.(3), the item is either of class White (ham) or Black (spam). A divider establishes a border where all items to the right are white and those to the left are black.

Using KNN and SVM algorithms, [14] attempted to filter spam photos. The whole system is essentially split into three parts, the first of which is the preprocessing stage, which consists of removing noisy data and retaining just the necessary information for the following step. Then, the picture format is selected, all photos in the dataset are resized to the same size, and the grey level co-occurrence matrix (GLCM) technique is used to extract the image characteristics. The graphic below illustrates the stages in detail. This research utilises the Dredze database, which comprises 3264 trash images and 1783 normal images.

[14] considers support vector machine to be true. SVM has excellent performance and is an effective classification technique for data classification, which includes the solution of quadratic problems. However, this needs a significant amount of training time and memory to address large-scale issues. When considering training and classification, particularly while the dimensionality between the data is large, it is important to note that when training needs a limited quantity of data, the data needed for training is less than the data required for testing. The KNN classification approach is simple and inexpensive, thus [14] plans to train and tune the SVM algorithm and combine it with KNN to see whether better results can be achieved. Detailed procedure stages are shown in Figure (4).

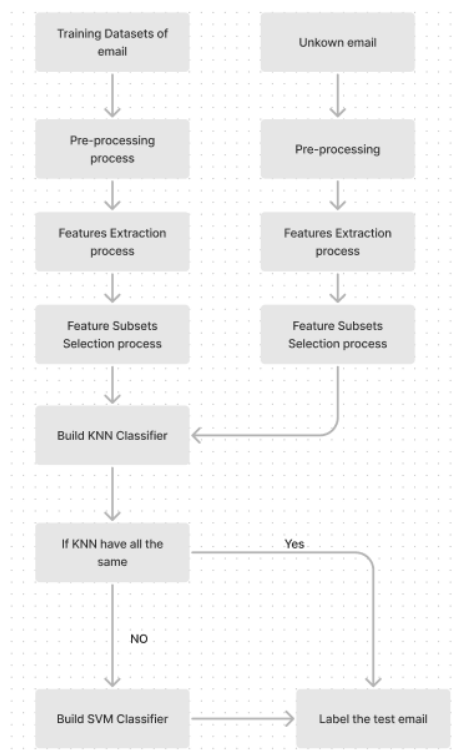


Fig.4 System design proposal for SVM and KNN classifiers.

Subsequently, [14] performed data testing on several algorithms using a test database, and in the first set of tests, the author used SVM as a classifier to validate the classification performance under the accuracy measure. In the second set of tests, the accuracy of utilising KNN as the classifier, and in the third experiment, the classification performance using the KNN-SVM combination as the accuracy metric for the classifier. Lastly, to compare the effectiveness of different strategies

Tab.1 Results of SVM method.

Spam image			Ham image			Average Acc
Train	Test	Accuracy	Train	Test	Accuracy	
50	1494	90.36	50	683	91.51	90.93
100	1494	89.76	100	683	91.95	90.85
150	1494	92.10	150	683	91.80	91.95
200	1494	93.31	200	683	91.95	92.63
1770	1494	0	1100	683	99.56	49.78

Tab.1 shows the average accuracy of SVM for various numbers of training samples. It can be shown that when lacking of training samples, the SVM classifier provides accurate results, otherwise, the accuracy of spam photos decreases to 0.

Tab.2 Results of KNN method with different K-values.

K values	Spam Acc.	Ham Acc.	Average Acc.
16	92.97	95.92	94.45
17	95.92	92.09	94.01
20	96.05	91.95	94
25	95.25	90.63	92.94
30	95.45	91.36	93.41
35	95.18	89.60	92.39
40	94.78	90.92	92.85

Using KNN with varying values of K, 3264 for Spam pictures (1770 training data and 1494 testing images) and 1783 for Ham images were generated using various K values (1100 training data and 683 testing images). Tab.2 demonstrates that the greatest average accuracy is attained for K in the interval (15-20).

Tab.3 Results of method of combining SVM and KNN with different K-values.

K values	Spam Acc.	Ham Acc.	Average Acc.
15	98.80	95.31	97.06
16	98.80	95.61	97.20
17	98.80	95.61	97.20
20	98.93	95.61	97.27
25	98.80	95.61	97.20
30	98.93	95.17	97.05
35	99	95.17	97.08
40	98.19	94.88	96.54

Tab.3 depicts the technique and metrics for performance assessment suggested by [14]. In addition, the system's efficacy may be shown using the accuracy of texture characteristics and other techniques. As shown by the outcomes, the KNN-SVM combo yields the best results.

2.3. XGBoost Classifier

GBDT/lambdMART; system-level enhancements include: weighted quantile sketch, sparsity-aware splitting, cache-aware access, out-of-core computation, feature parallelism, rabbit-based distributed training, etc. The algorithm level enhancements consist mostly of L1/L2 regularisation, application of the second-order derivative of the objective function, etc;

Boosting is based on the concept of combining the findings of numerous weak classifiers into one robust classifier. In the superposition approach, the findings of each base classifier are added together. The objective of constructing the subsequent base classifier is to fit the residual between the sum of previous classifier outputs and the label. When making predictions, the results of each basis classifier are summed, and each base classifier is a weak classifier. The principal base classifiers currently supported by The XGBoost method are rather accurate because The XGBoost technique employs a second-order Taylor expansion (SOTE) on the loss function. XGBoost introduces the second-order derivative in order to boost accuracy and personalise the loss function, respectively. A vast number of loss functions may be approximated by the SOTE. In addition, XGBoost adds a regular term to the goal function to regulate the model's complexity. The regular term includes the L2 normal form of the tree's leaf node count and leaf node weight. The regular term decreases the model's variance, simplifies the learnt model, and prevents overfitting. And after one iteration, XGBoost will increase the weight of each leaf node by this coefficient, mostly to weaken each tree. The tree's impact provides for a broader learning area in the rear. Because XGBoost enables users to create custom optimization targets and evaluation criteria, and because the XGBoost tool supports custom cost functions, so long as the functions may be first- and second-order derivatives, XGBoost provides a high degree of flexibility. XGBoost has guidelines for dealing with missing values. The user must give a value that differs from previous samples and send it in as a parameter, which is utilised as the missing value's value. XGBoost uses distinct processing techniques when various nodes encounter missing values, and it will learn the processing methods in the future. The XGBoost approach is reasonably reliable for samples with missing feature values because XGBoost conducts a SOTE on the loss function. XGBoost introduces the second-order derivative in order to boost accuracy and personalise the loss function, respectively. A vast number of loss functions may be approximated by the SOTE. In addition, XGBoost adds a regular term to the goal function to regulate the model's complexity. The regular term includes the L2 normal form of the tree's leaf node count and leaf node weight. The regular term decreases the model's variance, simplifies the learnt model, and prevents overfitting.

And after one iteration, XGBoost will increase the weight of each leaf node by this coefficient, mostly to weaken each tree. The tree's impact provides for a broader learning area in the rear. Because XGBoost enables users to create custom optimization targets and evaluation criteria, and because the XGBoost tool supports custom cost functions, so long as the functions may be first- and second-order derivatives, XGBoost provides a high degree of flexibility. XGBoost has guidelines for dealing with missing values. The user must give a value that differs from previous samples and send it in as a parameter, which is utilised as the missing value's value. XGBoost uses distinct processing techniques when various nodes encounter missing values, and it will learn the processing methods in the future. XGBoost can automatically learn its split direction for samples with missing feature values, which is analogous to updating and assessing new spam patterns based on spam information in mailbox filtering.

But even if xgboost has such a powerful performance, its shortcomings are obvious. Although using pre-sorting and approximation algorithms can reduce the computational effort to find the best split point, the data set still needs to be traversed during the node split process, which means that the time cost will increase. At the same time, because the space complexity of the pre-sorting process is too high, it is necessary to store not only the feature value, but also the index of the gradient statistical value of the sample corresponding to the feature. This is equivalent to consuming twice as much memory, so it requires a considerable amount of memory space similar to Clouds to run. can automatically learn its split direction, which is equivalent to updating and judging new spam patterns based on spam information in mailbox filtering.

[15] attempted to identify spam using regression tree and poor classification in XGBoost. Using a stratified sampling technique, [15] divides the obtained dataset into a training set and an out-of-sample test set with a 7:3 ratio. While the purpose of the 70% training set is to expose the training model to the type and diversity of spam and non-spam, respectively, the 30% test set is used to evaluate the performance of the proposed model on data samples it has not before seen. Due to the vast array of adjustable hyperparameters, XGBoost models may be constructed. Grid search is thus a systematic method for exhaustively searching through a defined subset of chosen hyperparameters in spam detection models. The steps are as follows:

1. Define a subset or range of given hyperparameters (eta, gamma, minimum sub-weight, etc.)
2. Systematically search for distinct hyperparameter combinations, one at a time. Step 2 involves training using the subsequent set of hyperparameters and recording performance. Return to step 2 if the list of hyperparameter combinations is not exhausted.
3. Determine the optimal mix of hyperparameters and construct the model.

Tab.4 Results of Xgboost method.

Data	Recall	specificity	Precision	F-score	Accuracy	ROC	PR
Test	95.59	97.73	96.47	96.03	96.88	99.08	97.69

Tab.4 shows the experimental data findings of xgboost. The researchers then compared the experimental data of xgboost to the prior old data models and discovered that the suggested model and test is substantially more accurate than the earlier efforts. The highest precision cited in the literature. This paper proposes XGBoost as a spam detector utilising basic computational intelligence techniques, and experimental findings on benchmark datasets demonstrate that it is a worthwhile classifier with improved hyperparameters that are less susceptible to overfitting.

The accuracy of the XGBoost spam detection model constructed using the training set and evaluated using the out-of-sample test set is 96.88%. Gleichzeitig wird angenommen, dass XGBoost's approach of feature reduction/extraction represents a viable future avenue for the investigation of prospective performance enhancements.

3. Discussions

Tab. 5 Summary of previous methods in email spam filtering.

Methods	Email Spam	Average accuracy
Naïve Bayes	√	98
Support Vector Machine	√	92.63
Xgboost Classifier	√	96.88
K-Nearest Neighbors	√	94.45
KNN and SVM	√	97.27

Tab. 5 summarises the data for all the approaches discussed in this article, and based on the data, the average accuracy of all methods is more than 90%. The Naive Bayes Classifier had the maximum accuracy with [10]. Naive Bayes is a classification technique based on probability theory; hence its code implementation and training are very straightforward and use little processing. Even though its premise that the attributes are independent of one another is impractical, it makes it more applicable in reality. The decoupling of the connection between features allows the algorithm to separately examine the distribution of each feature for each category, so avoiding the issue where the quantity of data to be processed rises exponentially with the number of features. The Naive Bayes Classifier is effective for multi-classification issues, and the complexity will not significantly rise. In addition, under the assumption that the distribution is independent, the naïve Bayes classifier performs fairly well while needing a lower sample size. And it works well for the category class input feature variables. For numeric variable characteristics, the default distribution type is normal.

On the other hand, despite the fact that Naive Bayes performs well in the field of classification and is adept at handling classification problems, the algorithm's presumption of the independence of independent variables and the normality of continuous variables will result in a reduction in algorithm accuracy.

The KNN algorithm theory is established, and the concept is straightforward. Moreover, the complexity is less than that of algorithms like support vector machines. It makes no assumptions about the data, has higher accuracy, and is insensitive to outliers. And since the KNN approach depends mostly on the limited surrounding samples, as opposed to the way of distinguishing the category domain, to establish the category to which an item belongs, this method has limitations. Therefore, the KNN method is more appropriate than other methods for multi-class classification. Because the KNN algorithm is better suited for the multi-class classification with numerous samples. Particularly when the amount of features is large, the KNN method will be computationally intensive. And when samples are uneven, the accuracy of predicting uncommon categories is quite poor. The KNN algorithm is a lazy learning approach that effectively does not learn; hence its prediction speed is slower than that of other algorithms.

In light of the aforementioned restrictions, the proposed technique for improvement is to first clip the known sample points and then eliminate the samples that have minimal impact on the categorization. Then, the K adjacent points are weighted, with the point closest to the distance having a greater weight than the point farthest away. This reduces the impact of these constraints. On a limited training set, the SVM method performs much better than other algorithms. The reason support vector machine has become one of the most popular and effective classifiers is due to its exceptional generalizability. The idea of margin provides a formal representation of data distribution, hence decreasing the needs for data size and data distribution. Simultaneously, the SVM algorithm provides a revolutionary, theoretically sound way for learning from tiny samples. It differs from conventional statistical approaches in that it does not use probability measurements, the rule of large numbers, etc. It eliminates the standard transition from induction to deduction, implements efficient "transduction reasoning" from training samples to forecast samples, and considerably simplifies the typical classification issue. A modest number of support vectors decide its ultimate decision function. The

computational complexity is determined by the number of support vectors, not the sample space's size.

However, the SVM technique is difficult to implement for large-scale training samples, and the majority of the storage capacity required by SVM is for training samples and kernel matrices. Since SVM solves the support vector through quadratic programming and solving the quadratic programming requires the computation of the m -order matrix, the m -order matrix must be computed (m is the number of samples). When m is big, storing and calculating the matrix will need a great deal of machine memory and processing time. If there are a high number of data points, SVM training will take longer. It is challenging to simultaneously handle many classification problems using SVM. In the actual application of data mining, it is typically essential to address the multi-class classification issue; however, this problem may be handled by combining numerous two-class support vector machines. It is solved by constructing a combination of multiple classifiers. The fundamental idea is to overcome the inherent limitations of SVM and combine the benefits of other classification methods to improve the classification accuracy of multi-class issues. Therefore, [14] utilised a combination of KNN and SVM.

Because XGBoost performs a SOTE on the loss function, it is relatively accurate. XGBoost introduces the SO derivative in order to boost accuracy and personalise the loss function, respectively. A vast number of loss functions may be approximated by the SOTE. In addition, XGBoost adds a regular term to the goal function to regulate the model's complexity. The regular term includes the L2 normal form of the tree's leaf node count and leaf node weight. The regular term decreases the model's variance, simplifies the learnt model, and prevents overfitting. And after one iteration, XGBoost will increase the weight of each leaf node by this coefficient, mostly to weaken each tree. The tree's impact provides for a broader learning area in the rear. Because XGBoost allows users to define custom optimization objectives and evaluation criteria, and because the XGBoost tool supports custom cost functions, so long as the functions can be first- and SO derivatives, XGBoost has a high degree of flexibility. XGBoost has guidelines for dealing with missing values. The user must give a value that differs from previous samples and send it in as a parameter, which is utilised as the missing value's value. XGBoost adopts distinct processing methods when different nodes encounter missing values and will learn the processing methods in the future. XGBoost can automatically learn its split direction for samples with missing feature values, which is analogous to updating and assessing new spam patterns based on spam information in mailbox filtering.

However, despite XGBoost's impressive performance, its flaws are evident. Despite the fact that employing pre-sorting and approximation methods may minimise the computational work required to locate the optimal split point, the data set must still be traversed during the node split procedure, which will raise the time cost. Due to the excessive space complexity of the pre-sorting process, it is also essential to record the index of the gradient statistical value of the sample corresponding to the feature, in addition to the feature value. This is the same as requiring twice as much memory, therefore it takes a comparable amount of memory space as Clouds to operate.

4. Conclusion

Email is now the most popular method of communication, yet it is susceptible to assault by unwanted and destructive spams. It is vital to regulate the level of spam so that platform and regular users are not negatively impacted by time loss, storage reduction, or bandwidth restriction. Learning-machine approaches can learn, recognise, and categorise email spam. In this paper, Spam Filtering classification is reviewed. First, five types of method are sorted. Then, the author survey 15 papers of those types. Finally, the author gives the prospective and discussions of this area.

The Bayesian algorithm is one of the oldest approaches in use. The central concept of the Bayesian algorithm is that the probability of future tests is proportional to the probability of past events, or posterior probability. However, it is difficult to compute the posterior probability in real-world situations. The emergence of Nave Bayes, the upgraded Bayesian algorithm, increases the

classification rate's precision. KNN is a nonparametric method that does not need a predefined probability distribution assumption. It is known as a lazy learner if there are no or few discrete training periods. KNN must nonetheless store all training data. The SVM is a model for binary classification. The average accuracy of SVM will decrease as data size increases. Combining KNN and SVM algorithms will get the most accurate results. The XGBoost classifier employs tree boosting methods. Boosting is based on the concept of combining the findings of numerous weak classifiers into one robust classifier. Users can define their own optimization objectives, evaluation criteria, and cost function, resulting in a flexible system. It incorporates the second-order derivative to enhance precision and allow for the customization of the loss function. In addition, XGBoost adds a regular term to the goal function to regulate the model's complexity. However, the XGBoost approach is particularly time-consuming because to the necessity to traverse the data set during the node split proves. In the interim, it is necessary to store the feature value and the index of the gradient statistical value. In other words, a substantial quantity of memory space is required.

Overall, all of the aforementioned techniques are effective spam-filtering technologies with an average accuracy of over 90%. The Naive Bayes Classifier had the maximum accuracy with [10]. As Naive Bayes is a classification method based on probability theory, it is a classification algorithm. On the other hand, despite the fact that Naive Bayes performs well in the area of classification and is adept at solving classification difficulties, it has limitations. The KNN algorithm theory is established, and the concept is straightforward. Classification, regression, and nonlinear classification are all possible applications. Moreover, the training time complexity is less than that of SVM and other algorithms. On a limited training set, the SVM method performs much better than other algorithms. Because XGBoost conducts a second-order Taylor expansion on the loss function, it is reasonably accurate.

References

- [1] M. Awad and M. Foqaha, 'Email spam classification using hybrid approach of RBF neural network and particle swarm optimization', *International Journal of Network Security & Its Applications*, vol. 8, no. 4, pp. 17–28, 2016.
- [2] O. Fonseca et al., 'Measuring, characterizing, and avoiding spam traffic costs', *IEEE Internet Computing*, vol. 20, no. 4, pp. 16–24, 2016.
- [3] Statista, 'Spam e-mail traffic share 2021', Statista, 2022. <http://www.statista.com/statistics/420391/spam-email-traffic-share/> (accessed Aug. 18, 2022).
- [4] WRAL, 'New Law Designed To Limit Amount Of Spam In E-Mail', *WRAL.com*, Dec. 30, 2003. <https://www.wral.com/news/local/story/108262/> (accessed Aug. 19, 2022).
- [5] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, 'Machine learning for email spam filtering: review, approaches and open research problems', *Heliyon*, vol. 5, no. 6, Jun. 2019, doi: 10.1016/j.heliyon.2019.e01802.
- [6] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, 'A Bayesian approach to filtering junk e-mail', in *Learning for Text Categorization: Papers from the 1998 workshop*, 1998, vol. 62, pp. 98–105.
- [7] J. Hovold, 'Naive Bayes Spam Filtering Using Word-Position-Based Attributes.', in *CEAS*, 2005, pp. 41–48.
- [8] H. Zhang, 'Exploring conditions for the optimality of naïve bayes', *Int. J. Patt. Recogn. Artif. Intell.*, vol. 19, no. 02, pp. 183–198, Mar. 2005, doi: 10.1142/S0218001405003983.
- [9] P. Rajendran, A. Tamilarasi, and R. Mynavathi, 'A Collaborative Abstraction Based Email Spam Filtering with Fingerprints', *Wireless Pers Commun*, vol. 123, no. 2, pp. 1913–1923, Mar. 2022, doi: 10.1007/s11277-021-09221-5.
- [10] RajKishore Sahni, 'Analysis of Naive Bayes Algorithm for Email SpamFiltering', *IJMTST*, vol. 7, no. 01, pp. 5–9, Jan. 2021, doi: 10.46501/IJMTST0701002.
- [11] T. Saravanan, 'A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm', Retrieved on August, vol. 8, p. 2017, 2010.

- [12] P.-N. Tan, M. Steinbach, and V. Kumar, Introduction to data mining. Pearson Education India, 2016.
- [13] A. Kataria and M. D. Singh, 'A review of data classification using k-nearest neighbour algorithm', International Journal of Emerging Technology and Advanced Engineering, vol. 3, no. 6, pp. 354–360, 2013.
- [14] Y. K. Zamil, S. A. Ali, and M. A. Naser, 'Spam image email filtering using K-NN and SVM', International Journal of Electrical and Computer Engineering, vol. 9, no. 1, pp. 245–254, Feb. 2019.
- [15] I. B. Mustapha, S. Hasan, S. O. Olatunji, S. M. Shamsuddin, and A. Kazeem, 'Effective Email Spam Detection System using Extreme Gradient Boosting'. arXiv, Dec. 27, 2020. doi: 10.48550/arXiv.2012.14430.