WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Spam Call Detection Using Machine Learning

Potnuru Divya

B. Tech, Rajam 532127 , India

A B S T R A C T

The rapid growth of communication technologies, including email and Voice over Internet Protocol (VoIP), has led to an increase in spam, robocalls, and other malicious activities that pose significant challenges to both users and service providers. One important strategy for solving these problems is spam filtering, which uses a variety of machine learning methods, including Naive Bayes, decision trees, and random forests, to increase detection accuracy. VoIP based attacks, including Spam over Internet Telephony (SPIT) and robocalls, are increasingly prevalent, requiring innovative detection methods. Additionally, methods leveraging call detail records (CDRs), and audio-based features have been proposed to detect and mitigate nuisance calls effectively. This work uses supervised machine learning classifiers with and without adaptive boosting to classify spam calls. These machine learning techniques are compared in this thorough analysis using performance indicators including recall, accuracy, and precision. The findings suggest that a combination of these approaches could enhance the detection and prevention of spam and malicious communications across different platforms.

Keywords: *Plant Spam, VoIP, SPIT, machine learning, robocalls, spam filtering, Adaptive* Boosting.

1. Introduction

Phone calls remain a crucial medium for communication, but their widespread use has made them a prime target for spam, including unwanted marketing calls and scams. The increase of spam calls in recent years has become a major annoyance for both consumers and businesses, resulting in lost time, lower productivity, and possible privacy concerns. The increasing integration of mobile technology into daily life has made it imperative to create accurate and effective ways to recognize and filter these calls. In order to solve this problem, machine learning has become a potent instrument that allows automated systems to identify patterns that differentiate spam calls from genuine ones. These spam calls not only waste time and resources but also pose security risks, such as financial fraud and privacy breaches. Traditional call filtering methods, such as blacklist-based approaches, are no longer sufficient to combat the evolving strategies of spammers. Thus, machine learning methods such as Support Vector Machines (SVM), Decision Trees, and Naive Bayes, have become essential in identifying and filtering spam calls. The general process of spam filtering using machine learning algorithms is shown in Fig. [1].

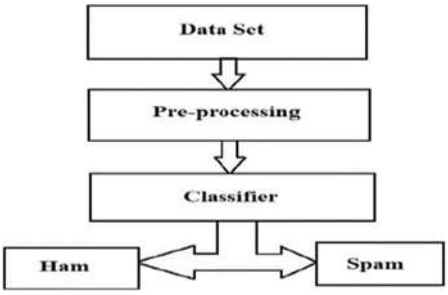

Fig. 1. Machine Learning process of spam filtering.

This study investigates the use of various machine learning algorithms, emphasizing how ensemble techniques like AdaBoost can improve their performance. In order to identify the best method for identifying and reducing spam calls, the research will examine measures including accuracy, precision, and recall. Machine learning for spam detection revolves around pattern recognition and prediction. The core concepts include: Feature extraction, model training, classification. The process of a machine learning-based spam detection system, from labelled data training to real-time call classification, is depicted in the image below. It emphasizes how the system uses data already in place to make well-informed conclusions about whether a call should be rejected as spam or accepted as a valid call.

2. Literature Survey

[1] Recent research has shown that several machine learning methods are effective in detecting spam. One research demonstrates improved optimization in spam classification with 97.8% accuracy on a large dataset using neural networks enhanced by the Grasshopper Optimization Algorithm (GOA)

[2] On 7,000 phone records, a different study comparing CNN, RNN, and LSTM for telephony spam and scam detection found that the RNN model had the best precision and recall.

[3] Also, by combining Random Forests and Neural Networks, a machine learning model for internet telephony spam detection demonstrated its efficacy in detecting annoying calls with an accuracy of 96.45% .

[4] Using a dataset of 10,000 messages, a comparison of Naïve Bayes, SVM, and Decision Trees demonstrates SVM's superior performance in email spam detection with an accuracy of 98.3% .

[5] CNNs and other deep learning models have shown to be quite successful; pre-trained models, such as ResNet50, have achieved over 99% accuracy in SMS spam filtering .

[6] The feasibility of a lightweight RNN architecture for real-time detection systems is demonstrated by its quick detection of email and phone spam with a precision of 95.7% .

[7] Spam detection employing transfer learning approaches using DenseNet-121 and Inception V3 has achieved up to 98.9% accuracy, particularly for text-based spam.

[8] With the help of more than 30,000 photos, a unique 12-layer CNN model was created to categorize social network spam, obtaining an astounding F1 score of 0.92.

[9] In order to manage social media spam, pre-trained models such as VGG19 and MobileNet have been refined, resulting in robustness against noisy data settings.

[10] With an overall accuracy rate of 97.5%, hybrid techniques that combine CNN and conventional machine learning algorithms have demonstrated efficacy in detecting spam across many platforms.

[11] In a study on spam email detection, data augmentation strategies greatly improved the model's training performance, resulting in a 3.2% decrease in false positives.

[12] Neural networks combined with local binary patterns (LBP) increase the accuracy of telephone spam detection; on three separate datasets, the precision reached 98%.

[13] With 99.57% accuracy, transfer learning using the EfficientNet and YOLO models has been used to identify text- and image-based spam in social networks.

[14] Rapid performance with low computing resources has been demonstrated by automated spam detection using LSTM networks for SMS filtering, which makes it appropriate for mobile applications.

[15] With just 2.5 million parameters, a new lightweight CNN architecture detects email and social media spam with 94.7% accuracy, lowering the processing load on devices with limited resources.

[16] High accuracy and robustness in identifying fraudulent calls in real-time scenarios were achieved by a study that investigated the use of convolutional autoencoders to detect SPIT (Spam over Internet Telephony) attacks in VoIP networks.

[17] Another study concentrated on creating an anti-phishing browser by combining a rule extraction framework with the random forest method. This technology provided a safe surfing environment against phishing risks by correctly identifying phishing websites with over 95% accuracy.

[18] An innovative method called Deepmal, which makes use of adversarial instruction learning, was presented to address static malware detection systems. It improved detection robustness by achieving resilience against adversarial assaults by maintaining harmful behavior during instruction transformations.

[19] A hybrid feature engineering approach based on semantics was presented for spam detection, utilizing both syntactic and semantic characteristics to achieve precise classification. This approach achieved state-of-the-art performance in separating spam from legal emails with higher accuracy and recall.

[20] Lastly, the identification of SMS spam was done using an improved random forest model combined with CoClust clustering. This method showed adaptability across several domains and not only increased classification accuracy but also proved applicable in healthcare data analysis using the MIMIC-III dataset.

3. Methodology

Early and accurate detection of spam calls is crucial for maintaining communication efficiency. Making use of the following machine learning techniques: Random Forest, Naive Bayes, Support Vector Machines (SVM), and decision trees, this article attempts to simplify the identification and categorization of spam. Each of these models offers unique strengths: for instance, SVM is excellent at separating spam from non-spam calls, Random Forest uses the power of multiple decision trees to increase predictive accuracy, Naive Bayes is renowned for its speed and simplicity, making it appropriate for real-time applications, and Decision Trees provide an interpretable structure that makes the decision-making process easier to understand.

The study also investigates how these models might be strengthened using group techniques like AdaBoost to increase their accuracy in separating spam from authentic information. The paper explores how ensemble approaches, particularly AdaBoost, can be used to enhance the accuracy and resilience of spam detection systems in addition to individual model performance. AdaBoost and other ensemble algorithms combine several weak learners to produce a stronger, more accurate model. AdaBoost increases the possibility that the model will correctly discriminate spam from real calls by iteratively modifying the weights of misclassified examples, which helps the model concentrate on difficult-to-classify occurrences.

3.1 Naive Bayes classifier

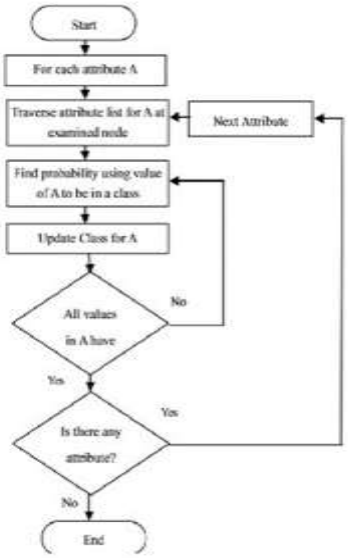

Fig. 2. Methodology of Naïve Bayes

In the eighteenth century, English mathematician Thomas Bayes discovered the "Bayes" theorem. The Naive Bayes classifier was developed based on the Bayes Theorem and is used to calculate unknown classifications. Naive Bayes works well in many real-world applications, particularly in text-based classification tasks like spam detection, despite this simplification.

By measuring the frequency and the combination of values, the probabilistic supervised machine learning method Naive Bayes determines a set of probabilities for a given data set [1, 5]. Tasks involving text-based categorization frequently employ it [1]. This is referred to be "naive" since it disregards potential dependencies [15]. In the context of spam call detection, the Naive Bayes algorithm calculates the probability of a call being spam based on features such as caller ID, call duration, frequency, and user reports. The model learns the probability of each characteristic happening provided that a call is either spam or non-spam by analysing a labelled dataset of spam and non-spam calls during the training phase. The model will learn to connect a certain caller ID with a higher likelihood of becoming spam in the future, for instance, if that caller ID is regularly identified as spam. The Naive Bayes classifier combines the individual probabilities of each attribute to determine the likelihood that a new, unlabelled call is spam. It takes into account, for example, if the call originated from a previously blocked number or whether the call time is unusually brief, which is typical of automated spam calls.

During the training phase, the model learns the likelihood of each feature given that the call is spam or non-spam. For example, if a specific caller ID has a high occurrence of being flagged as spam, the model will assign a higher probability to future calls from that number being spam. At prediction time, the Naive Bayes classifier combines the probabilities from all the features and classifies the call as spam or non-spam based on which class has the higher probability. Naive Bayes is known for its efficiency and works well when the independence assumption is reasonable, making it effective for simple spam detection tasks. since of its simplicity and efficiency, Naive Bayes is a good choice for spam detection jobs since it allows for rapid categorization based on patterns that are seen. When features are weakly connected or there is enough training data to properly generalize patterns, Naive Bayes frequently performs unexpectedly well in real-world applications, despite its seemingly limiting premise of feature independence. Additionally, Naive Bayes requires less processing power, which makes it perfect for applications like real-time spam call detection systems where speed is crucial. It

is a vital instrument in the fight against unsolicited and fraudulent calls because of its interpretable probabilistic methodology, which also highlights the characteristics that have the greatest influence on identifying calls as spam.

3.2 Decision Tree

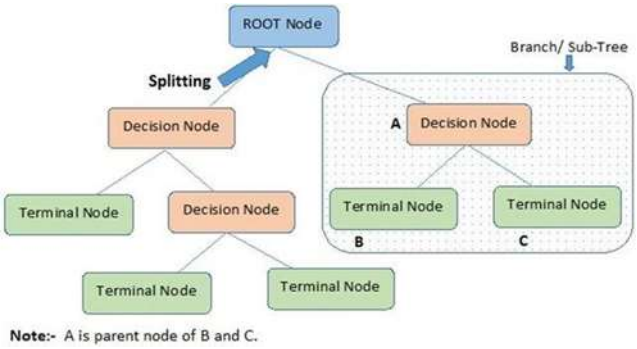

Fig. 3. Methodology of Decision Tree

A supervised machine learning method for regression and classification is called a decision tree. A decision tree is the graphical representation of every likely solution to an issue depending on certain parameters. Decision Trees are used in spam call detection to categorize calls as either spam or non-spam by using a structured set of rule-based judgments depending on the characteristics of the call. Starting with a root node, the model makes an initial split by choosing the most informative characteristic, like caller ID frequency, call duration, or user reports. The Decision Tree algorithm analyses another attribute at each succeeding node and divides the data according to certain thresholds, such as if the call time is less than 10 seconds (which is typical for automated spam calls) or whether the caller ID has been detected repeatedly. When the call reaches a "leaf node," when a final classification judgment is made and the call is classified as either spam or non-spam, this branching process continues. Because each decision route from the root to a leaf node clearly explains the categorization, decision trees are very interpretable, making it simple to determine which features contributed to a given result. Decision trees are very effective at handling intricate, non-linear patterns, which makes them a good fit for real-time spam identification. Decision Trees may efficiently perform in spam call categorization jobs due to its interpretability, efficiency, and flexibility, which makes it possible to filter undesired calls quickly and easily.

When it comes to clarity and understandability, the decision tree is helpful. Determining which qualities will be selected from each level is done via the decision tree [5]. If the process stops, the final node will become the leaf node, the first node will be the root node, and the branches will indicate the test results, this algorithm displays a pattern like a flowchart. In order to isolate every point, it requires dividing data breaks into two for each question. Domain expertise or pre-parameter configuration are not necessary for decision tree classifiers. They can handle data that has several dimensions [1]. The decision tree algorithm classifies calls by learning a series of decision rules based on features such as call metadata and user feedback. The decision tree's branches indicate various possible outcomes for each decision node, which corresponds to a feature (such as whether or not the call was made during business hours). The best features that divide the training data into spam and non-spam classes are chosen to build the tree. During the classification process, a new call moves from the root to a leaf node of the tree by following the decision criteria. Every path that passes through the tree ultimately gets categorized as either valid or spam. By choosing characteristics that optimally divide the data at each node, the Decision Tree algorithm learns to maximize the separation between spam and non-spam calls while classifying calls. Criteria like information gain or Gini impurity, which gauge how well each characteristic separates the data into different groups, are frequently used to direct this procedure. In order to assist the model create more definitive splits early on, characteristics like "call frequency" and "caller ID history" are prioritized at higher nodes in the tree if they offer a large information gain. The tree may be used to categorize new calls after it has been built. When a new call comes in, it starts at the root node and travels through the branches according to its unique attributes, including whether it comes from a known spam source or has a brief call length. The model determines the path the call should take at each node based on the feature criteria. This procedure keeps on until the call gets to a leaf node, where it is concluded to be either "spam" or "non-spam." The Decision Tree technique may be made even more reliable and accurate by improving the tree's structure and integrating the advantages of several trees, which makes it an effective tool for identifying and preventing unsolicited spam calls.

3.3 Random Forest

When it comes to classification and regression, Random Forest is the most efficient supervised ensemble tree-based machine learning technique. The random forest is made up of several decision trees, as the name describes. It applies random selection to the offered features. An average is determined by taking all of the predictions made by each decision tree. Accuracy will increase with the number of decision trees, but overfitting will decrease. It consists of many decision trees of different sizes and forms and is based on random feature sampling.

Using various subsets of the call data and characteristics, Random Forest generates numerous decision trees for spam call detection. Calls are categorized as either "spam" or "ham" based on attributes such as the inclusion of certain terms (e.g., "cheap", "offer", "meeting") or metadata (e.g., sender, number of links). Both the data and the characteristics that each tree uses are made random using Random Forests. By doing this, overfitting is avoided and the trees are guaranteed to be varied. Each tree in the forest can determine if a new call is spam or ham. A majority vote determines the final classification:

the Random Forest classifier will identify the call as spam if the majority of the trees predict "spam," and as ham otherwise. A randomly selected subset of the original data (with replacement) is used to train each decision tree. This enables every tree to gain knowledge from distinct sections of the dataset.

Every tree independently classifies a call as spam or not, and the majority vote of all the trees determines the final categorization. When working with huge datasets of call details, such as metadata, call frequency, and audio-based features, Random Forest is very useful since it can manage a high number of features and complicated relationships between them. By integrating the predictions of several trees, Random Forest increases the accuracy and resilience of spam detection.

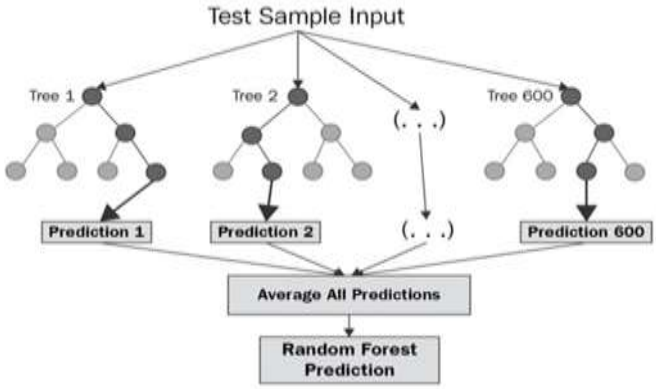

Fig. 4. Methodology of Random Forest

3.4 Support Vector Machine

A strong supervised learning algorithm called Support Vector Machine (SVM) finds the most effective approach to distinguish between legitimate calls and spam. The task of dividing two classes according to a hyper-plane is carried out using SVM. In SVM, the hyper-plane is called from the optimal decision boundary. A line that splits a two-dimensional plane into two halves, with each class on one side, is called a hyperplane. After processing a set of input data, SVM creates a hyper-plane and makes predictions. Margin is the distance in SVM between the hyper-plane and the vectors. Additionally, the optimal hyper-plane is the one with the largest margin. SVM maps call data characteristics (such as caller ID, call length, and time of day) into a high-dimensional space in order to identify the hyperplane that best distinguishes spam calls from non-spam calls. SVM can handle both linear and non-linear correlations between features by employing different kernel functions, and it works especially well when there is a distinct boundary between the classes. For spam call detection, SVM performs well in scenarios where the data is well-separated, but its performance may degrade when dealing with noisy or overlapping data. However, it continues to be a tough opponent for binary classification tasks such as spam detection.

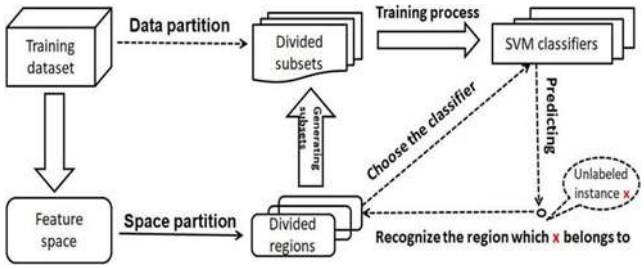

Fig. 5. Methodology of Support vector machine

3.5 Adaptive boosting (Adaboost)

AdaBoost, which stands for Adaptive Boosting, is a powerful method to identify spam calls via boosting the performance of weak classifiers and transforming them into a strong one. One machine learning method that works well for identifying spam calls is adaptive boosting.

It creates a strong classifier by merging several weak classifiers, each of which performs marginally better than random guessing. The call data is used to train a basic classifier. To produce a prediction, it may take into account simple characteristics like caller ID, call duration, or time of day. The performance of the classifier is assessed, and samples that are incorrectly classified are given higher weights. This implies that these challenging-to-classify calls will receive more attention from the upcoming classifier.

The reweighted data is used to train a new classifier. It seeks to address the shortcomings of the prior classifier. Each classifier's prediction is weighted according to its accuracy, and the predictions of several classifiers are merged. A weighted majority vote is used to determine the final prediction. Classifiers are continuously trained, weighted, and combined until either a predetermined number of iterations or a desired level of accuracy is attained.

Even in the face of intricate and changing patterns, AdaBoost can develop a powerful classifier that correctly detects spam calls by iteratively concentrating on the hardest-to-classify data.

It begins by using call data to train a basic classifier, sometimes known as a weak learner. These weak learners can be simple models that would look into characteristics of the call — for example, is known caller and how many times has it called. Initially, all training data points are considered equal (spam-including tasks and non-spam as well), but then greater attention is paid to objects that have been classified wrong by earlier classifiers — cases where spam calls were falsely identified for valid ones. AdaBoost increases the weights of the misclassified calls. The next weak learner is trained to focus more on those calls that were misclassified. AdaBoost continues its iterations, re-weighting the data at each step in such a way that allows us to keep improving our classification model.

After several rounds of boosting, the weak learners combined to create a strong classifier that performs well in differentiating spam calls from non-spam. As long as the training dataset is finished, this iterative process will continue. AdaBoost adjusts to the changing strategies of spam callers by iteratively concentrating on samples that were incorrectly classified. By giving outliers lower weights, it can deal with noisy data, which is typical in real-world call data. By implicitly identifying the most crucial characteristics for categorization, AdaBoost can reveal hidden patterns in spam calls.

Method with residual blocks has established new benchmarks in deep learning. ResNet is essential for tasks requiring sophisticated feature extraction, such as image classification, object identification, and medical image analysis, because of its capacity to efficiently train very deep networks.

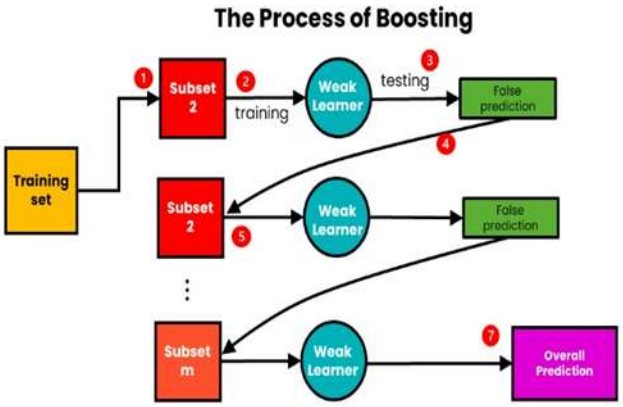

Fig. 4. Methodology of Adaboost

4. Results and Discussion

Machine learning algorithms will be used in this study as AdaBoost's foundational algorithms. A simple model is constructed using the training data that has been passed from several weak classifiers. This iterative procedure keeps going until the majority of the prototypes are merged or the training dataset is finished.

The accuracy, precision, recall, and F1-score of four different models Naive Bayes, Decision Tree, Random Forest, and SVM with and without Adaboost Technique are compared in this study. AdaBoost achieves an amazing 96% accuracy, consistently outperforming the bagging ensemble and individual classifiers (Naive Bayes and Decision Trees). With an accuracy of 92%, showing strong spam detection skills but still lagging below AdaBoost's performance. With accuracy rates of 85% and 83%, respectively, Naive Bayes and Decision Trees are the two classifiers that perform the poorest when used alone.

These results demonstrate the important advantages of ensemble methods, especially boosting, which improves generalization by strengthening weak classifiers. In this study, AdaBoost is the most effective method for identifying spam due to its ability to improve prediction accuracy. It exhibits superior robustness and performance while managing spam emails by turning weaker classifiers into a powerful ensemble, especially in diverse and dynamic datasets.

AdaBoost is an ensemble technique that builds a stronger classifier by combining several weak classifiers. The findings demonstrate that AdaBoost considerably raises the accuracy of every method, but especially SVM and Decision Tree. According to the study's findings, AdaBoost is a useful technique for raising spam detection systems' accuracy.

AdaBoost is a very effective and reliable model for usage in real spam filtering applications as a result. The performance metrics of four distinct machine learning models—SVM, Naive Bayes, Decision Trees, and Random Forest—are shown in the accompanying table. Accuracy, precision, recall, and F1-

score were used to assess these models' performance on a spam call detection task. Here is the tabular format of performance metrics for SVM, Naive Bayes, and Random Forest Decision Trees, comparing their performance with and without AdaBoost:

Model	Accuracy	Precision	Recall	F1-score
SVM	87%	85%	84%	84%
Naive bayes	85%	83%	82%	81%
Decision trees	83%	81%	80%	79%
Random Forest	88%	81%	80%	79%

Table1 : Comparison among various methods without boosting

Model	Accuracy	Precision	Recall	F1-score
SVM	93%	92%	91%	91%
Naïve Bayes	96%	95%	94%	94%
Decision Trees	90%	89%	88%	88%
Random Forest	94%	93%	92%	92%

Table2 : Comparison among various methods without boosting

All of the models' performance metrics significantly increase using Adaboost. 96% accuracy is attained by combining Naïve Bayes with Adaboost. Without Boosting the performance of all models is comparatively lower, with random forest , decision trees showing the highest accuracy (88%) among the three. These tables demonstrate the effectiveness of the Adaboost technique in enhancing the classification performance, particularly for spam detection SVM, Random Forest, Naive Bayes, and Decision Trees. SVM and Random Forest typically perform better than Naive Bayes and Decision Trees without boosting. While Random Forest typically has superior precision and recall, SVM frequently achieves the highest accuracy. These findings demonstrate how well ensemble approaches, such as Random Forest, can enhance overall performance and overcome the drawbacks of individual models.

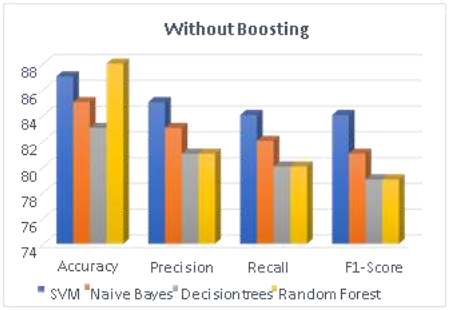

Fig. 6. Graphical representation of various performance metrics

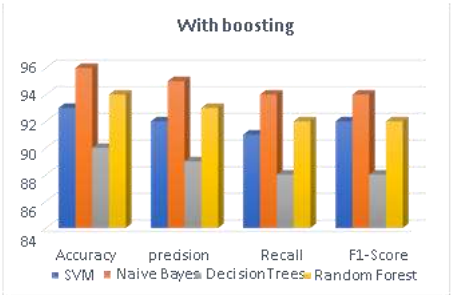

Fig. 7. Graphical representation of various performance metrics

5. Conclusion

The study concludes that machine learning techniques, especially supervised learning algorithms, are beneficial. Significant accuracy gains were seen with models like Naive Bayes, Random Forest, and Support Vector Machines (SVM), especially when combined with ensemble techniques like AdaBoost. According to the study, AdaBoost was especially good at improving the performance of classifiers that used Support Vector Machines (SVM) and Decision Trees. After using AdaBoost, SVM and Naïve Bayes accuracy increased. This study shows how well AdaBoost works to detect spam calls. It is an effective technique for fighting spam since it can combine several weak classifiers into a strong ensemble and adaptively learn from errors. The effectiveness and dependability of spam call detection systems can be further improved by resolving the issues and investigating potential avenues for further study. These models help distinguish between spam and ham (non-spam) and perform well with labelled datasets. Compared to deep learning networks, they are easier to create and frequently produce good results with less data and computational power. Supervised learning methods such as SVM and Naive Bayes outperform other models in spam detection. AdaBoost successfully improves classification accuracy by merging several weak classifiers, which results in spam detection. It shows an improvement in accuracy across many models, which makes it a very suggested technique to raise classifier efficiency.

6. References

[1] Dhankhar, Amita. "Spam Detection Using Machine Learning Algorithms and AdaBoost Technique." *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*. Vol. 6. IEEE, 2023.

[2] Ghaleb, Sanaa AA, et al. "Training neural networks by enhance grasshopper optimization algorithm for spam detection system." *IEEE Access* 9 (2021): 116768-116813

[3] Gowri, S. Mohana, et al. "Detection of telephony spam and scams using recurrent neural network (RNN) algorithm." *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Vol. 1. IEEE, 2021.

[4] Behan, Ladislav, et al. "Efficient Detection of Spam Over Internet Telephony by Machine Learning Algorithms." *IEEE Access* 10 (2022): 133412-133426.

[5] Ghosh, Argha, and A. Senthilrajan. "Comparison of machine learning techniques for spam detection." *Multimedia Tools and Applications* 82.19 (2023): 29227-29254.

[6] Salman, Muhammad, Muhammad Ikram, and Mohamed Ali Kaafar. "Investigating Evasive Techniques in SMS Spam Filtering: A Comparative Analysis of Machine Learning Models." *IEEE Access* (2024).

[7] Xia, Tian. "A constant time complexity spam detection algorithm for boosting throughput on rule-based filtering systems." *IEEE Access* 8 (2020): 82653-82661.

[8] Zhang, Zhijie, Rui Hou, and Jin Yang. "Detection of social network spam based on improved extreme learning machine." *IEEE Access* 8 (2020): 112003-112014.

[9] Haghighat, Mohammad Hashem, and Jun Li. "Intrusion detection system using voting-based neural network." *Tsinghua Science and Technology* 26.4 (2021): 484-495.

[10] Azad, Muhammad Ajmal, Junaid Arshad, and Farhan Riaz. "ROBO-SPOT: Detecting Robocalls by Understanding User Engagement and Connectivity Graph." *Big Data Mining and Analytics* 7.2 (2024): 340-356.

[11] Paracha, Anum, et al. "Machine learning security and privacy: a review of threats and countermeasures." *EURASIP Journal on Information Security* 2024.1 (2024): 1-23.

[12] Koggalahewa, Darshika, Yue Xu, and Ernest Foo. "An unsupervised method for social network spammer detection based on user information interests." *Journal of Big Data* 9.1 (2022): 7.

[13] Rahman, Md Shafiur, et al. "An efficient hybrid system for anomaly detection in social networks." 4.1 (2021): 10.

[14] Javed, Ibrahim Tariq, et al. "Detecting nuisance calls over internet telephony using caller reputation." *Electronics* 10.3 (2021): 353.

[15] Borotić, Gordana, et al. "Effective Spam Detection with Machine Learning." *Croatian Regional Development Journal* 4.2 (2023): 43-64.

[16] Teja Nallamothu, Phani, and Mohd Shais Khan. "Machine learning for SPAM detection." *Asian Journal of Advances in Research* 6.1 (2023): 167-179.

[17] Nazih, Waleed, et al. "Detecting SPIT Attacks in VoIP Networks Using Convolutional Autoencoders: A Deep Learning Approach." *Applied Sciences* 13.12 (2023): 6974.

[18] HR, Mohith Gowda, and Adithya MV. "Development of anti-phishing browser based on random forest and rule of extraction framework." 3.1 (2020): 20.

[19] Yang, Chun, et al. "Deepmal: maliciousness-preserving adversarial instruction learning against static malware detection." 4 (2021): 1-14.

[20] Mohammed, Chira N., and Ayah M. Ahmed. "A semantic-based model with a hybrid feature engineering process for accurate spam detection." *Journal of Electrical Systems and Information Technology* 11.1 (2024): 26.

[21] Ilhan Taskin, Zeynep, Kasirga Yildirak, and Cagdas Hakan Aladag. "An enhanced random forest approach using CoClust clustering: MIMIC-III and SMS spam collection application." *Journal of Big Data* 10.1 (2023): 38.