International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Soft Voting Ensemble with N-GRAM Vectorization for Accurate News Classification on Twitter Data

Nadiah Jaffreen Shaik[1], Tatavarthy Santhi Sri*[2]

Submitted:11/09/2024 **Revised:** 16/10/2024 **Accepted:** 22/10/2024

Abstract: With the exponential growth in the social media platform, the need for effective tools to accurately classify tweets, according to their news content has become very crucial. This work presents a comprehensive comparison of text classification techniques with various N-gram and vectorization techniques. Using the CybAttT dataset, our methodology employs text preprocessing techniques and then follow the data towards modelling. Six various Machine learning models are used on two vectorizing techniques namely TF-IDF and count vectorization. Following towards efficacy, these baseline six models are further again developed on soft voting ensemble model to leverage the strengths of each individual classifier. And the performance of each model was evaluated based on accuracy, precision, recall, and F1-score performance metrics. The results are rigorously compared on various N-gram configurations. From the experimental results, the soft voting ensemble model achieved an accuracy of 97.02% and 96.56% for count vectorization and TF-IDF for default n-gram. The comparison is also observed on bigram and trigram and out of all models, the ensemble model scores are superior to other machine learning models. These finding reveals that the proposed ensemble model advances text classification dynamics and also proposes a robust framework for researchers and practitioners focusing on social media analytics.

Keywords: *Ensemble Learning, N-gram Vectorization, Social Media Analytics, News Classification, Text Classification Techniques*

1. Introduction

The increase in cyberattacks and data breaches is a pressing concern as technological advancements and digital platforms become more integrated into our daily lives [1]. The consequences of such a threat would greatly affect societal stability and economic security. This is coupled with the critical lack of advanced security measures and mechanisms for detecting threats within software industries and other organizations [2]. As estimated, cybercrimes are on the rise and more dangerous; thus, enormous challenges for businesses lie ahead. However, to meet the new challenge of cyberattacks on business security, machine learning (ML) models, tools, and applications on computer security have been on the upsurge [3]. The difficulty lies in the fact that all planning is usually done secretly by attackers who use online forums to communicate and strategize before they act. Posts in public forums have emerged as an essential source in dealing with such threats. Other recent studies also identified the involvement of social media, including Facebook and what used to be known as Twitter but is now referred to as the X platform, in collecting data relevant to cyber threats [4]. These platforms are widely used globally, providing a vast pool of data as users frequently post about incidents both prior to and during their occurrence. Social media services, with their very high daily data generation, such as the X platform, generate around 500 million tweets daily; therefore, it plays a crucial role [5]. In 2024, it has been estimated that around 63.8% of the population of the world uses social media [6]. The X platform is one of the most prominent ones in terms of speed and ease of connectivity. The API of the platform allows it to support user interactions, thereby fetching and interacting with a myriad of resources, including posts, user profiles, and much more. Users must register their use cases on the platform developer's website for access to data through the API after being approved and receiving an API key. Although basic access to the platform API is free, a paid subscription is required for comprehensive data retrieval to adhere to the platform's guidelines [7]. The X platform remains an essential source of information for experts and celebrities alike in different fields, providing live updates on news, trends, and events, thus becoming an essential tool for up-to-the-minute information on live occurrences and breaking news. The increasing danger of cyberattacks calls for sophisticated analytical tools that decode and predict the potential breach of security.

Natural Language Processing (NLP) emerges as a critical technology in this context due to its ability to analyse vast amounts of unstructured text data, such as that found on social media and online forums, which are frequent hotspots for the exchange of malicious intent and cyber threat planning [8]. Through this process, NLP extracts specific patterns, sentiments, and keywords on security issues related to such datasets, thus identifying threats beforehand [9]. It can help to detect unusual activity or discussions that become early stages of planning for a cyber-attack by reading through the contents in posts while undergoing analysis from language. This is important because cyber-attacks are often not open. Therefore, detecting earlier on, which helps to prevent huge damage. However, NLP has its limitations. One of the most important limitations is that language is ambiguous. Algorithmic interpretations might lead to misunderstanding due to sarcasm, idioms, and phrases dependent on context. One major limitation is the number of sources available and how language keeps changing. Therefore, NLP models are always updated to keep them in effective shape. Despite these limitations, methods such as Term Frequency-Inverse Document Frequency (TF-IDF) and Count Vectorization are a couple of the more fundamental approaches used in this study [10]. They map text data to numerical vectors by measuring word importance or a term's context in each corpus, resulting in the loss of order but preserving meaning content. This enables the conversion so that ML algorithms can handle text data more efficiently when completing tasks such as classification or clustering, which are crucial for discovering potential cyber threats [11]. Notably, TF-IDF represents a sophisticated way to measure the volume of textual information by counting the weights associated with terms using the frequency count for each term against the overall corpus of documents and, hence, is an irreplaceable technique where rapid extraction of significant features in enormous volumes of texts is considered as a prerequisite. This work uses the transformation of text on machines and within the NLP tool to categorize cyberattacks regarding their tweets. This basis is quite strong for further investigations. Through NLP, organizations can monitor and analyze communications through various platforms proactively. Organizations can easily determine and resolve potential cyber threats represented in tweets about hot topics. This proactive application of technology maximizes the capability for responding to threats quickly and effectively, so overall cybersecurity becomes better. The rest of this paper is organized as follows: Section 2 offers a Literature review. Section 3 gives details about the proposed methodology. Section 4 discusses the experimental results, and Section 5 discusses the conclusion's future scope.

2. LITERATURE REVIEW

The research works from the past five years are collected and reviewed in this section related to NLP in detecting fake news posts. The authors, T.D. Jayasiriwardene, et al. developed the methodology to extract keywords from the tweets, and with Stanford Core NLP toolkit, WordNet, and statistical methods scored 67.6% accuracy in detecting fake news [12]. The authors Ari Z. Klein, et al. also developed an NLP pipeline that detected possible under-reported COVID-19 cases from Twitter data [13]. Their work trains the tweets through an application of the BERT-based deep neural network classifier; in this instance, the F1-score it attained was 0.76. Adnan A. Hnaif, et al., focus on detecting the Arabic news posts from social media networks sentiment polarity and analysis has been done of all NLP techniques, including removal of stop-word and stemming as well as normalization steps with multiple different classifiers including, Support vector machine (SVM), Naïve Bayes (NB), K Nearest Neighbor (KNN), Random Forest (RF), Decision Tree (DT) used one by one while training the system and then a test over a given dataset [14]. Among them, the SVM classifier gives the highest accuracy of 83%, which effectively categorizes posts into positive, negative, and neutral sentiments. Another author, M. Senthil Raja, et al. developed a work utilizing Count Vectorizer, TF-IDF Vectorizer, and N-gram models for text analysis along with their other techniques and ML algorithms, including NB, SVM, RF, and Logistic Regression (LR) [15]. The results show that the TF-IDF Vectorizer combined with the SVM classifier has given the highest precision of 93%. Manideep Narra, et al. determined the effects of feature selection and pre-processing in the detection of fake news on COVID-19 [16]. The study reports that the Extra Tree classifier (ET), at best, achieved an accuracy level of 0.9474, with the features combined with TF-IDF and Bag of Words (BoW). It reports that deep learning (DL) models such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Residual Neural Network (ResNet), and InceptionV3 operate less effectively. Lai, et al. examined the use of ML and NLP in the identification of fake news on Twitter based on content-level features [17]. Their approach combined both traditional models in ML with neural networks with an added TF-IDF for ML models and word embeddings for neural networks. The results showed that over 90% accuracy was achieved by the neural network model.

Kumar, et al. use NB, LR, and LSTM classifiers, pre-processed and further enhanced by including TF-IDF for feature extraction to identify the spread of false news within the Hindi news using a dataset having different news resources [18]. Out of these, the LSTM classifier proved the best one, achieving 92.36%. Mishchenko, et al. developed a method for recognizing fake news on the internet by enhancing the TF-IDF algorithm and integrating it with the NB classifier, where the authors used NLP to analyse large volumes of data [19]. This work showed the accuracy of detecting fake news from a previous range of 80% to 90% to a more precise 93%, representing an average efficiency increase of 2.5%. Wani, et al. conducted research on toxic Covid-19 fake news detection on social media by exploiting techniques such as linear SVM, RF, and Bidirectional Encoder Representations from Transformers (BERT) [20]. Their approach, with the help of linear SVM, had a superior performance with an accuracy of 92% and an F1-score of 95%, thereby proving its efficacy in identifying whether the news is toxic or nontoxic. Farooq, et al. studied Urdu language fake news detection to discuss the robustness issues in the multi-domain dataset [21]. The authors used a TF-IDF-based BoW with n-grams as a feature extraction approach and stacking with SVM, KNN, and ensemble methods like RF and ET with LT as a meta learner. The stacked model also showed great performance with 93.39% accuracy and an F1 score of 93.17%. Akhter, et al. developed the DL model, which detects Covid-19 associated fake news applying CNN model to be optimized based on grid search and word embedding [22]. This results in a 96.19% mean accuracy, 95% F1-score, and an AUC of 0.985. This literature review identifies significant limitations within existing studies: the low accuracy of results by traditional NLP tools, limited datasets of certain topical events like the COVID-19 pandemic, inconsistent performance from classifiers, and especially sentiment analysis, as well as low accuracies having less applicability to controlled settings. The other challenges also include the fine-grained underperformance of DL models in certain scenarios, reliance on specific model configurations for achieving high accuracy, language-specific processing limitations, and the need to optimize the vast number of parameters in DL models to be effective. While others have demonstrated robustness in handling multi-domain datasets using such advanced techniques as feature stacking, they often only remain topical with topics like toxic fake news. Based on these challenges, our work focuses on the application of ensemble learning and N-gram vectorization techniques to strengthen multiple classifiers and enhance the robustness and performance of the model in terms of various metrics and configurations. The methodology is presented in the next section.

3. METHODOLOGY

The developed methodology starts from data initialization, followed by text pre-processing techniques and data split for training and testing. Further, various ML modelling techniques are applied as baseline models, and these are compared using different N-gram and vectorization techniques, i.e., specifically, TF-IDF and count vectorization, to determine the most effective approach. In this work, a soft voting ensemble is used to improve efficacy towards the aggregation of different models predictions to provide better classification accuracy based on the strength of each model. At last, it evaluates the performance of each model using crucial metrics like accuracy, precision, F1 score, and recall. This framework is depicted in the Fig. 1.

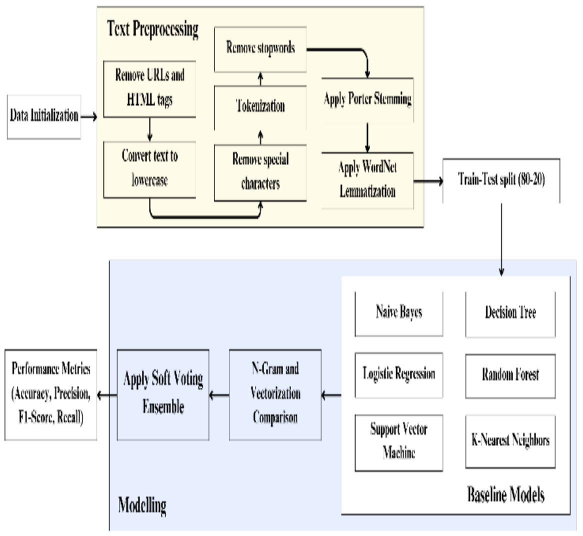

Fig. 1. Depicts the methodology framework.

3.1. Dataset Initialization

The CybAttT dataset, which forms the basis of our study, was carefully curated to allow for a detailed analysis of cyber threat narratives circulating within social media [23]. The dataset comprises 36,080 tweets and was assembled to reflect a diverse range of perspectives on cybersecurity incidents with five classes. Each tweet was classified into one of three categories by a trio of annotators as ‘not_news’, ‘normal_news’, and ‘high_risk_news’, ensuring a robust consensus approach to the labelling process. The dataset very clearly exhibits a significant skew in the classes, where ‘not_news’ constitutes 31,231, ‘normal_news’ makes up 3,948, and ‘high_risk_news’ stands at 892 entries and is presented in Fig. 2. This skew is designed to mimic the real-world reality of generic mentions over actual news content when it comes to cyber threats.

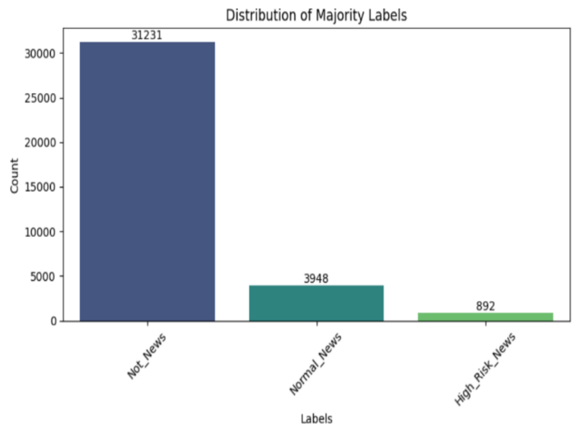

Fig. 2. Depicts the distribution of the dataset.

In preparation for analytical processing, the dataset was refined by removing individual annotations and retaining only the 'majority' label for each tweet. The relevant data was first obtained through initial keyword selection processes based on word cloud, which is shown in Fig 3., where the terms selected having strong relevance to existing cyber security concerns were extracted, including ‘ransomware’, ‘phishing’, ‘attack’, ‘cybersecurity’ amongst others [24]. The frequency of such terms in the dataset was deliberately recorded to help guide the text preprocessing steps subsequently, ensuring that the analysis would be based on the most prevalent terms in cyber discourse.

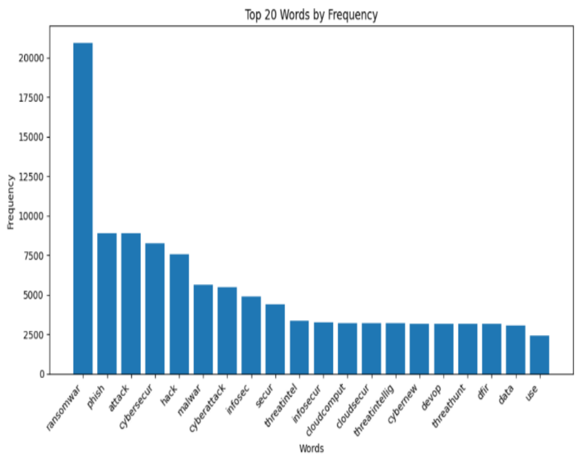

Fig. 3. Depicts the frequence of the key terms used by the data.

3.2. Text Preprocessing

The text preprocessing stage is one of the very important stages where the raw data will be refined to transform into a form that improves the performance of subsequent ML models [25]. This begins with the removal of URLs and HTML tags that will remove all the irrelevant content and reduce the noise in text data. For example, from a raw tweet ‘Read from this link https://example.com for more info!’, the URL ‘https://example.com’ is removed, resulting in ‘Read from this link for more info!’. Further processing includes lowercasing every text because the differences and discrepancies could easily appear in some cases. Tokenization involves breaking down the text strings into individual words or tokens. For example, this text would be tokenized as [‘read’, ‘from’, ‘this’, ‘link’, ‘for’, ‘more’, ‘info’]. Following tokenization, special characters are removed to clean the data and ensure the retention of only the alphabet for further analysis. Lastly, stop words are removed, and the common words have been excluded due to their small value in text class distinction [26]. Also, stemming and lemmatization are finally done in this preprocessing. Porter Stemming is used to reduce words into their base or root forms to generalize different forms of the same word into a single representation. For example, ‘checking’ would get stemmed to ‘check’ [27]. Further, wordnet lemmatization applies morphological analysis to words; it seeks the removal of inflectional endings and brings back to their base or dictionary form of a word, that is, known as the lemma [28]. For instance, ‘better’ would get lemmatized to become ‘good’.

3.3. Train Test Split

In this experiment, the dataset is passed through an important partitioning step to divide it into a training and testing set, a common practice in applying ML models. This is split into 80:20 ratio into training and testing sets sizes are 28,856 and 7,215. This will allow a very comprehensive training on 80% of the data, with the remaining 20% to be used in testing models' ability to generalize to new, unseen data.

3.4. Ensemble Modelling on N-Gram and Vectorization Techniques

Our approach for the improvement of the predictive accuracy in classifying cyber threats via novel paradigm uses ensemble model to leverage the strengths of several baseline ML models, each tuned to categorize the content into three predefined categories of news as not_news, normal_news, and high_risk_news.

3.4.1 B aseline ML models

In our analysis, six ML models are used as baseline models, and each such model contributing to the task at hand will uniquely enhance the relevance of classification for cyber threat communications. The first model, Random Forest (RF), which is an ensemble of decision trees, highly increases the classification accuracy by using bootstrapped versions to reduce overfitting and improve stability, and every tree's output contributes to a majority vote to arrive at the final classification [29]. The second model, Logistic Regression (LR), offers probabilistic interpretations for the binary outcomes by effectively modelling the likelihood of categorization of content with logistic functions [30]. The third model is the Support Vector Machine (SVM), where an optimal hyperplane is built for distinctly separating the different classes while maximizing the margin of classification [31]. K-Nearest Neighbors (KNN), as the fourth model is a non-parametric approach, determines the classifications based on the majority of the labels assigned to the nearest data points [32]. The method leverages local data point similarities to ensure robust multi-class categorization. Decision Trees (DT) are the fifth model that offers a clear visualization of decision paths and outcomes [33]. As such, the model is interpretable and quite good at categorical data. Lastly, the sixth model, Naive Bayes (NB), has been chosen as it is particularly good at text classification, as it assumes features are independent within a class [34]. Therefore, it calculates the probability of every category efficiently. Collectively, these models form a comprehensive baseline ML model framework capable of tackling the complexities inherent in text-based cyber threat detection.

3.4.1. N-Gram and Vectorization Comparison

In order to process and transform text data effectively, N-Gram configurations from unigrams up to trigrams in this study are used to capture various levels of textual context and correlations:

- TF-IDF: The technique for vectorizing weights the importance of a word in a document with respect to the corpus as a whole [35]. The TF-IDF value rises linearly with the word frequency in a document but is corrected by the frequency of the word within the corpus to account for the fact that some words appear more frequently on average. The formula is defined by eq. (1) as

$$TF-IDF(t,d) = TF(t,d) \times IDF(t) \qquad (1)$$

Here $TF(t,d)$ is the term frequency of term t in document $d$and the inverse document frequency of term as $IDF(t) = log(\frac{N}{df(t)})$ where N is the total number of documents in the corpus and $df(t)$ is the number of documents containing the term.

- Count Vectorization: Count vectorization is the process of counting how many times each word gets repeated in any document [36]. Here, a document term matrix was created that stated the frequency occurrence of terms presented in the group of documents, which is defined as eq. (2) as

$$c(t,d) = \text{Frequency of } t \text{ in } d \qquad (2)$$

3.4.2. Apply Soft Voting Ensemble

In order to evaluate each of the model performances on data in terms of TF-IDF and Count Vectorized data for each category, this work implements a soft voting ensemble technique. Soft Voting is an ensemble method whereby the final output class is the one to be decided after obtaining a majority or the highest sum of predicted probabilities from component classifiers [37]. This gives a possibility of combining estimates coming from multiple models in such a way that if the best model provides better estimates, it can, therefore, carry greater 'weight' in the eventual decision. The soft voting mechanism can be mathematically represented by eq. (3) as

$$P(y|x) = \frac{1}{C}\sum_{i=1}^{N} P_i(y|x) \qquad (3)$$

Where $P(y|x)$ is the predicted probability of class y given features x. C is the number of classifiers in the ensemble and $P_i(y|x)$ is the predicted probability of y of x, which is predicted by the ith model.

3.5. Performance Metrics

In this work, to evaluate the effectiveness of the text classification models developed in this study, we employ four critical performance metrics: accuracy, precision, recall, and F1-score [38]. Each of these metrics will give a different view of how well the models are performing so that the results can be compared and analyzed with respect to how well the models classify news content from tweets accurately. And the following metrics are:

- Accuracy is the most intuitive performance measure, and it is simply ratio of correctly predicted observation to total observations. This gives us measure of how many instances were correctly classified is defined by eq. (4) as

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions made}} \qquad (4)$$

- Precision refers the ratio of the correct positively classified observations to the total number of positively classified ones. It is an accuracy measure for that model in classifying a sample as positive, and is defined by eq. (5) as

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \qquad (5)$$

- Recall (Sensitivity) is the number of cases where the model was able to discover all the relevant cases (True Positives) within the dataset. The metric is useful when the cost of false negatives is very high, and is defined by eq. (6) as

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \qquad (6)$$

- F1-Score is the harmonic mean of Precision and Recall. Thus, it incorporates both false positives and negatives. In the case when the class distribution is not equal, it proves to be a useful measure that is defined by eq. (7) as

$$\text{Recall} = 2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \qquad (7)$$

Based on these measures, several ML model's performances are measured by utilizing different N-gram and vectorization settings as well as the ensemble model based on a soft voting strategy. The simulation results are mentioned in section 4.

4. EXPERIMENTAL RESULTS

The experiments were conducted on the high-performance computing setup that featured an 11th Gen Intel Core i5 processor and a 4095MB NVIDIA GeForce GTX 1650 GPU under a 64-bit Windows 11 OS. These experiments used Jupyter Notebook based on Anaconda distribution to execute very robust environments of Python-based analyses. The key libraries involved NLTK for natural language processing, Pandas for data manipulation, NumPy for numerical operations, and Scikit-learn for the implementation and evaluation of ML models [38].This configuration would allow efficient handling and processing of the dataset CybAttT to be used in comprehensive evaluations regarding different text classification techniques to enhance cyber threat detection accuracy on social media platforms.

Table. 1. Comparison of Ensemble Model with various ML Models on TF-IDF Vectorizer based on N-grams.

Model with TD-IDF vectorizer	**Accuracy**	**Precision**	**F1 Score**	**Recall**
Default Ensemble Model	**0.9656**	**0.9643**	**0.9645**	**0.9656**
Default Random Forest	0.9637	0.9623	0.9622	0.9637
Default Logistic Regression	0.9627	0.9615	0.9619	0.9627
Default SVM	0.9626	0.9616	0.9620	0.9626
Default KNN	0.9544	0.9524	0.9528	0.9544
Default Decision Tree	0.9454	0.9454	0.9454	0.9454
Default Naive Bayes	0.9404	0.9365	0.9357	0.9404
Bigram Ensemble Model	**0.9508**	**0.9483**	**0.9481**	**0.9508**
Bigram Random Forest	0.9519	0.9496	0.9497	0.9519
Bigram Logistic Regression	0.9450	0.9418	0.9416	0.9450
Bigram SVM	0.9428	0.9395	0.9400	0.9428
Bigram KNN	0.9416	0.9384	0.9391	0.9416
Bigram Decision Tree	0.9448	0.9426	0.9434	0.9448
Bigram Naive Bayes	0.9389	0.9348	0.9351	0.9389
Trigram Ensemble Model	**0.9346**	**0.9312**	**0.9277**	**0.9346**
Trigram Random Forest	0.9344	0.9309	0.9276	0.9344
Trigram	0.9337	0.9306	0.9266	0.9337

Logistic Regression				
Trigram SVM	0.9339	0.9304	0.9271	0.9339
Trigram KNN	0.9310	0.9266	0.9239	0.9310
Trigram Decision Tree	0.9346	0.9311	0.9278	0.9346
Trigram Naive Bayes	0.9317	0.9277	0.9243	0.9317

In the evaluation of text classification models, the TF-IDF vectorizer across Default, Bigram, and Trigram N-gram configurations are mentioned in the Table. 1, our Ensemble Model, employing a soft voting mechanism, consistently exhibits superior performance. Within all N-gram settings, the ensemble approach consistently achieves the highest metrics across accuracy, precision, F1 score, and recall, highlighting its effectiveness in managing different levels of textual complexity. In the Default (unigram) setting, where the soft ensemble shows tremendous performance in an Ensemble Model, accuracy and recall are 0.966, a slight edge over LR; SVM and RF show robust performances having accuracies around 0.963 with their precision around it. At increased Bigram and Trigram configurations, the performance declines slightly for all models except the Ensemble Model. Despite this, the Ensemble Model remains on top, confirming its capacity for taking in more contextual information as generated by these two configurations. Specifically, in the case of Bigram, while the scores obtained by the Ensemble Model are still relatively lower than those achieved in Default, the model retains effectiveness up to a high accuracy of 0.951. The RF model has shown strength in this setting; this may be taken as some models are more efficient in capturing the nuances of Bigram analysis. In the Trigram configuration, while all models generally show a decrease in performance metrics, the Ensemble Model again shines by maintaining relatively high scores, even with added complexity, with an accuracy of 0.935. The results in trigram also offer a better rich context with which language will be approached; it does give rise to possible overfitting issues in an attempt that is better solved using the integrated cross-model predictions capability of the Ensemble Model.

Table. 2. Comparison of Ensemble Model with various ML Models on Count Vectorizer based on N-grams.

Model with Count vectorizer	**Accuracy**	**Precision**	**F1-Score**	**Recall**
Default Ensemble Model	**0.9702**	**0.9698**	**0.9699**	**0.9702**
Default Random Forest	0.9683	0.9672	0.9674	0.9683
Default Logistic Regression	0.9648	0.9643	0.9645	0.9648
Default SVM	0.9606	0.9612	0.9609	0.9606
Default KNN	0.9579	0.9562	0.9567	0.9579
Default Decision Tree	0.9505	0.9509	0.9507	0.9505
Default Naive Bayes	0.9365	0.9479	0.9404	0.9365
Bigram Ensemble Model	**0.9575**	**0.9558**	**0.9562**	**0.9575**
Bigram Random Forest	0.9543	0.9526	0.9532	0.9543
Bigram Logistic Regression	0.9552	0.9530	0.9533	0.9552
Bigram SVM	0.9559	0.9539	0.9542	0.9559
Bigram KNN	0.9448	0.9417	0.9421	0.9448
Bigram Decision Tree	0.9468	0.9461	0.9464	0.9468
Bigram Naive Bayes	0.9319	0.9400	0.9350	0.9319
Trigram Ensemble Model	**0.9453**	**0.9425**	**0.9412**	**0.9453**
Trigram Random Forest	0.9453	0.9425	0.9413	0.9453
Trigram Logistic Regression	0.9426	0.9398	0.9379	0.9426
Trigram SVM	0.9444	0.9413	0.9406	0.9444
Trigram KNN	0.9389	0.9353	0.9337	0.9389
Trigram	0.9437	0.9405	0.9400	0.9437

Decision Tree				
Trigram Naive Bayes	0.9372	0.9337	0.9348	0.9372

Table. 2. use Count Vectorizer in Default (unigram), Bigram, and Trigram N-gram settings on the models where the Ensemble Model also outperforms in default settings with uniform scores of 0.970 in accuracy, precision, F1-score, and recall, demonstrating high consistency and predictive strength. After that comes Logistic Regression, with a 0.965 score, and then RF, scoring 0.968 - which affirms that these models are great for plain textual analysis. Some more accuracy can be observed in SVM and KNN, with the DT and NB lagging but not too far behind. For Bigram, the Ensemble Model continues to score at the highest with 0.957; although it drops compared to the Default setting, this is still well above most individual models. In the Bigram context, RF and SVM can also be well adapted, with scores around 0.954-0.956. For LR, it still scored at 0.955; the DT and KNN still maintained competitiveness even after dropping slightly in performance. Finally, in the Trigram configuration, the Ensemble Model again demonstrates power, achieving an accuracy of 0.945 with slightly lower precision and F1-score at 0.942 and 0.941, respectively; the performance of the Random Forest matches that of the ensemble, showing the ability of the algorithm in handling the increased textual complexity that Trigram configurations add. SVM and Logistic Regression stay at high levels above 0.940. The smallest decreases are observed by the Decision Tree, KNN, and Naive Bayes, but the latter all reveal the challenges that come with richer linguistic contexts, such as overfitting.

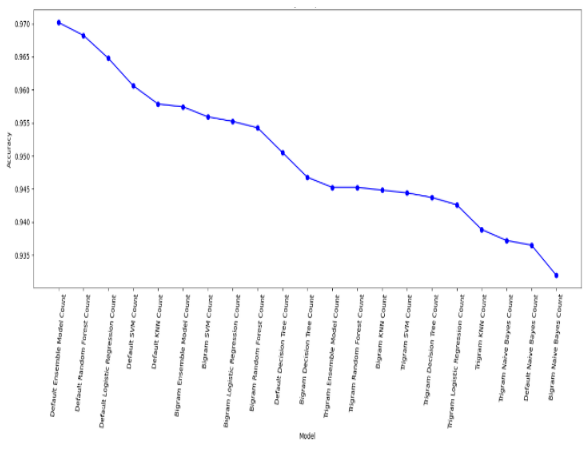

Fig. 4. Model Accuracy Comparison across different N-gram configurations across TF-IDF Vectorization.

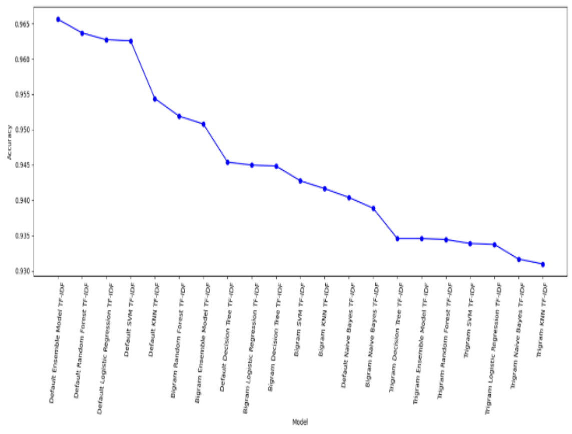

Fig. 5. Model Accuracy Comparison across different N-gram configurations across Count Vectorization.

Observing the model performance in the Fig. 4 and Fig. 5., various limitations are noted in the proposed work. The two vectorization techniques show a general decline in the model accuracy. At the same time, the complexity of the N-gram configurations goes higher, pointing out potential issues in handling high-order N-grams that would introduce noise or irrelevant features in the models. Specifically, count vectorization is often better than TF-IDF, especially in smaller N-gram setups, indicating that Count might be better suited for the frequency features of this data. Still, the leading performance of Count vectorization diminishes with higher complexity within N-gram setups. It may indicate overfitting as models fit the trains too well, getting too snug in the details of the training data and, therefore, losing their generalizing capabilities. Besides this, the almost uniform decline with an increase in N-gram complexity across most models indicates that supplementary syntactic information provided by configurations Bigram and Trigram is not being put to effective usage by the models, possibly owing to limitations either in the currently used modelling techniques or in the inherent properties of the vectorization methods. This will be indicative of a need for better feature selection techniques or more sophisticated model architectures that can better capitalize on the richer contextual information offered by higher-order N-grams without compromising the model's ability to generalize from the training data in the future. Overall, our developed ensemble model achieved superior performance metrics than other baseline models under all N-gram settings signifies the utility of such an approach toward the delivery of reliable and accurate classifications, thus especially important in cyber threat detection against diverse and complex textual scenarios.

5. CONCLUSION AND FUTURE SCOPE

This paper successfully implemented an ensemble model leveraging multiple ML algorithms to enhance the accuracy of cyber threat classification from social media content, utilizing both TF-IDF and Count vectorization techniques across various N-gram configurations. Indeed, though the ensemble outperformed each model individually across various N-gram configurations, exhibiting robustness and efficiency, this approach appears to suffer when moving into the complexity of N-grams, opening avenues for future work. Further study in the advanced deep learning architecture would help bring better classification performance, proper real-time cyber-attack-detecting systems, and social media monitoring tools; some real applications would be mitigating cyber threats through the timely identification and response of potential attacks, enhancing public awareness over risks to cybersecurity, and fortifying efforts from law enforcement agencies in investigating cybercrime.

References

[1] M. A. I. Mallick and R. Nath, "Navigating the Cyber security Landscape: A Comprehensive Review of Cyber-Attacks, Emerging Trends, and Recent Developments," *World Scientific News,* vol. 190, p. 1–69, 2024.

[2] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz and E. Akin, "A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions," Electronics, vol. 12, p. 1333, 2023.

[3] M. Stamp, Introduction to machine learning with applications in information security, Chapman and Hall/CRC, 2022.

[4] V. Chang, L. Golightly, Q. A. Xu, T. Boonmee and B. S. Liu, "Cybersecurity for children: an investigation into the application of social media," Enterprise Information Systems, vol. 17, p. 2188122, 2023.

[5] P. A. Henríquez and F. Alessandri, "Analyzing Digital Societal Interactions and Sentiment Classification in Twitter (X) during critical events in Chile," Heliyon, 2024.

[6] J. M. Nagata, Z. Memon, J. Talebloo, M. P. H. K. Li, P. Low, I. Y. Shao, K. T. Ganson, A. Testa, J. He and C. D. Brindis, "Prevalence and Patterns of Social Media Use in Early Adolescents," Academic Pediatrics, p. 102784, 2025.

[7] Y. Demchenko, J. J. Cuadrado-Gallego, O. Chertov and M. Aleksandrova, "Finding Data on the Web, Data Sets, Web Scraping, Web API," Springer, 2024, p. 417–446.

[8] K. Szabó Nagy, J. Kapusta and M. Munk, "Feature extraction from unstructured texts as a combination of the morphological and the syntactic analysis and its usage in fake news classification tasks," Neural Computing and Applications, vol. 35, p. 22055–22067, 2023.

[9] M. S. M. Suhaimin, M. H. A. Hijazi, E. G. Moung, P. N. E. Nohuddin, S. Chua and F. Coenen, "Social media sentiment analysis and opinion mining in public security: Taxonomy, trend analysis, issues and future directions," Journal of King Saud University-Computer and Information Sciences, p. 101776, 2023.

[10] U. Krzeszewska, A. Poniszewska-Marańda and J. Ochelska-Mierzejewska, "Systematic comparison of vectorization methods in classification context," Applied Sciences, vol. 12, p. 5119, 2022.

[11] M. H. Ahmed, S. Tiun, N. Omar and N. S. Sani, "Short text clustering algorithms, application and challenges: A survey," Applied Sciences, vol. 13, p. 342, 2022.

[12] T. D. Jayasiriwardene and G. U. Ganegoda, "Keyword extraction from Tweets using NLP tools for collecting relevant news," 2020.

[13] A. Z. Klein, A. Magge, K. O'Connor, J. I. Flores Amaro, D. Weissenbacher and G. Gonzalez Hernandez, "Toward using Twitter for tracking COVID-19: a natural language processing pipeline and exploratory data set," Journal of

medical Internet research, vol. 23, p. e25314, 2021.

[14] A. A. Hnaif, E. Kanan and T. Kanan, “Sentiment Analysis for Arabic Social Media News Polarity.,” Intelligent Automation & Soft Computing, vol. 28, 2021.

[15] M. S. Raja and L. A. Raj, “Fake news detection on social networks using Machine learning techniques,” Materials Today: Proceedings, vol. 62, p. 4821–4827, 2022.

[16] M. Narra, M. Umer, S. Sadiq, H. Karamti, A. Mohamed and I. Ashraf, “Selective feature sets based fake news detection for COVID-19 to manage infodemic,” IEEE Access, vol. 10, p. 98724–98736, 2022.

[17] C.-M. Lai, M.-H. Chen, E. Kristiani, V. K. Verma and C.-T. Yang, “Fake news classification based on content level features,” Applied Sciences, vol. 12, p. 1116, 2022.

[18] S. Kumar and T. D. Singh, “Fake news detection on Hindi news dataset,” Global Transitions Proceedings, vol. 3, p. 289–297, 2022.

[19] L. Mishchenko, I. Klymenko and V. Tkachenko, “The fake news recognition method based on Naïve Bayes with improved TF-IDF algorithm,” 2023.

[20] M. A. Wani, M. ELAffendi, K. A. Shakil, I. M. Abuhaimed, A. Nayyar, A. Hussain and A. A. Abd El-Latif, “Toxic Fake News Detection and Classification for Combating COVID-19 Misinformation,” IEEE Transactions on Computational Social Systems, 2023.

[21] M. S. Farooq, A. Naseem, F. Rustam and I. Ashraf, “Fake news detection in Urdu language using machine learning,” PeerJ Computer Science, vol. 9, p. e1353, 2023.

[22] M. Akhter, S. M. M. Hossain, R. S. Nigar, S. Paul, K. M. A. Kamal, A. Sen and I. H. Sarker, “COVID-19 Fake News Detection using Deep Learning Model,” Annals of Data Science, p. 1–32, 2024.

[23] H. Lughbi, M. Mars and K. Almotairi, “CybAttT: A Dataset of Cyberattack News Tweets for Enhanced Threat Intelligence,” Data, vol. 9, p. 39, 2024.

[24] M. Dong, J. Lu, G. Wang, X. Zheng and D. Kiritsis, “Model-based systems engineering papers analysis based on word cloud visualization,” 2022.

[25] L. Hickman, S. Thapa, L. Tay, M. Cao and P. Srinivasan, “Text preprocessing for text mining in organizational research: Review and recommendations,” Organizational Research Methods, vol. 25, p. 114–146, 2022.

[26] S. Sarica and J. Luo, “Stopwords in technical language processing,” Plos one, vol. 16, p. e0254937, 2021.

[27] N. A. Razmi, M. Z. Zamri, S. S. S. Ghazalli and N. Seman, “Visualizing stemming techniques on online news articles text analytics,” Bulletin of Electrical Engineering and Informatics, vol. 10, p. 365–373, 2021.

[28] S. Kundu, “31 An overview of Stemming and Lemmatization Techniques,” 2024.

[29] E. Naresh, B. J. Ananda, K. S. Keerthi and M. R. Tejonidhi, “Predicting the stock price using natural language processing and random forest regressor,” 2022.

[30] A. Shete, H. Soni, Z. Sajnani and A. Shete, “Fake news detection using natural language processing and logistic regression,” 2021.

[31] M. T. H. K. Tusar and M. T. Islam, “A comparative study of sentiment analysis using NLP and different machine learning techniques on US airline Twitter data,” 2021.

[32] L. S. Riza, Y. Firdaus, R. A. Sukamto, Wahyudin and K. A. F. Abu Samah, “Automatic generation of short-answer questions in reading comprehension using NLP and KNN,” Multimedia Tools and Applications, vol. 82, p. 41913–41940, 2023.

[33] S. S. I. Ismail, R. F. Mansour, R. M. Abd El-Aziz and A. I. Taloba, “Efficient E-Mail Spam Detection Strategy Using Genetic Decision Tree Processing with NLP Features,” Computational Intelligence and Neuroscience, vol. 2022, p. 7710005, 2022.

[34] F.-J. Yang, “An implementation of naive bayes classifier,” 2018.

[35] A. Aizawa, “An information-theoretic perspective of tf-idf measures,” Information Processing & Management, vol. 39, p. 45–65, 2003.

[36] A. Wendland, M. Zenere and J. Niemann, “Introduction to text classification: impact of stemming and comparing TF-IDF and count vectorization as feature extraction technique,” 2021.

[37] M. U. Salur and İ. Aydın, “A soft voting ensemble learning-based approach for multimodal sentiment analysis,” Neural Computing and Applications, vol. 34, p. 18391–18406, 2022.

[38] J. Dessain, “Machine learning models predicting returns: Why most popular performance metrics are misleading and proposal for an efficient metric,” Expert Systems with Applications, vol. 199, p. 116970, 2022.

[39] D. Jeet, V. Sharma, S. Mishra, C. Iwendi and J. Osamor, “Twitter Sentiment Analysis and Emotion Detection Using NLTK and TextBlob,” 2023.