ROUTERS: Jurnal Sistem dan Teknologi Informasi
Vol. 3, No. 2, Juli 2025, pp. 125-135
ISSN: 2962-1224, DOI: 10.25181/rt.v3i2.4310

Phishing Email Classification Using TF-IDF Method and Random Forest Algorithm

Ismi Rosia Dwianti [1], Dilla Regita Cahyani [2], Titik Khawa Abd Rahman[3], Muhammad Faisal [4], Aedah Abd Rahman[5], Swa Lee Lee [6], Nasir Usman [7]

[1, 2] Internet Engineering Technology, Departments of Information Technology, Politeknik Negeri Lampung
[3, 5] Department of School of Science and Technology, Asia e University
[4] Department of Informatics, Universitas Muhammadiyah Makassar
[6] Department of School of Graduate Studies, Asia e University
[7] Department of Information Systems, STMIK Profesional Makassar

ARTICLE INFO

Received July 19, 2025
Revised July 28, 2025
Published July 28, 2025

Keyword:

Phishing;
Natural Language Processing;
TF-IDF;
Random Forest;
Email

ABSTRACT

Phishing attacks through email are increasingly becoming a serious threat to cybersecurity. Traditional detection methods such as blacklists and pattern matching have proven to be ineffective in addressing increasingly complex attacks. Therefore, a new approach is needed one that can analyze email content contextually and intelligently.This study develops a phishing email detection system by integrating the Term Frequency–Inverse Document Frequency (TF-IDF) method and the Random Forest algorithm. The system is designed to analyze the linguistic structure of email content and recognize common patterns frequently used in phishing attacks.The research stages include text preprocessing, feature extraction using TF-IDF, model training with Random Forest, and performance evaluation using three data splitting scenarios: 60:40, 70:30, and 80:20. The dataset was obtained from Kaggle and consists of 82,486 emails that have been adjusted to be balanced between phishing and legitimate emails. The evaluation results show that the system achieved high accuracy in all scenarios, with the highest score reaching 98.01% in the 80:20 split. The precision, recall, and F1-score metrics also indicate strong and stable performance. In addition, feature importance analysis shows that the model can identify key terms such as "click," "money," and "attached" that frequently appear in phishing emails. This research is expected to serve as a foundation for the development of more adaptive, intelligent, and responsive phishing detection systems in the face of evolving cyber threats.

Corresponding Author:

Coresponding Ismi Rosia Dwianti, Politeknik Negeri Lampung
Email: ismirosia66@gmail.com

1. INTRODUCTION

The rapid advancement of information technology has led to a major transformation in how people communicate, especially through digital media such as email. Email has become a primary medium for both business and personal communication due to its convenience, speed, and efficiency. However, this progress has also been exploited by irresponsible parties to launch cyberattacks, one of which is phishing. Phishing is an attempt to steal personal data by impersonating official institutions through fake emails designed to resemble legitimate ones [1]. In Indonesia, phishing trends have shown a significant increase, with various schemes such as fake lotteries, bank account updates, and fictitious job offers. In fact, according to recent reports, phishing actors are now utilizing AI-based technologies like ChatGPT to create more convincing phishing emails [2].

As the complexity and frequency of phishing attacks increase, traditional detection methods such as blacklists or pattern matching have proven ineffective [3]. Therefore, a new approach is needed that can deeply understand the linguistic context and content of emails. One promising approach is the application of Natural Language Processing (NLP) to email content analysis, combined with machine learning algorithms such as Random Forest for automatic classification. NLP can help extract linguistic features often manipulated in phishing emails, such as sentence structure, vocabulary, and writing patterns. Meanwhile, Random Forest is known to perform well with high-dimensional data such as text [4].

This study references several previous works as the foundation for system development. One such study by Umam et al. [5] analyzed the performance of phishing email classification using the Support Vector Classification (SVC) algorithm and Random Forest. The study used a dataset from Kaggle consisting of 18,650 email entries, comprising 11,322 secure emails and 7,328 phishing emails. Models were trained using three data split scenarios: 60:40, 70:30, and 80:20. Testing results showed that SVC achieved the best accuracy of 97.52% in the 70:30 split, followed by 97.40% in the 80:20 split, and 97.37% in the 60:40 split. Meanwhile, Random Forest achieved its highest accuracy of 96.57% in the 70:30 scenario. These results indicate that although SVC slightly outperformed Random Forest, the latter still demonstrated competitive and stable performance in phishing email classification. Additionally, a study by Bountakas et al. [6] showed that the combination of the TF-IDF method with the Random Forest algorithm achieved an accuracy of 93.41% on a balanced dataset. [6] These results serve as a reference for evaluating the performance of the phishing email detection system developed in this study.

In this research, the Random Forest algorithm is used to demonstrate that it can achieve higher performance when trained on a significantly larger and more balanced dataset. This approach is expected to enhance the effectiveness of Random Forest in handling high-dimensional text data. Although this study does not directly compare Random Forest with other algorithms like SVC, the primary focus is to show that Random Forest can yield excellent classification performance when applied to large-scale datasets. Furthermore, this research aims to demonstrate that the use of larger and more proportionate datasets enables the TF-IDF method to support higher classification performance. This emphasizes that the quality and size of the dataset play a crucial role in optimizing the results of NLP-based feature extraction techniques.

The main contribution of this research is the development of a phishing email detection system based on Natural Language Processing (NLP) combined with the Random Forest algorithm, with a focus on analyzing the text content of emails. Feature extraction is carried out using the Term Frequency–Inverse Document Frequency (TF-IDF) method due to its effectiveness in representing word weights and handling high-dimensional text data. The system is also equipped with an automatic translator feature using the Google Translate API to handle emails in Indonesian and align them with the English-language training data. The dataset used has been adjusted to be balanced between phishing and legitimate data, allowing the model to learn fairly without bias toward one label [7]. Evaluation was conducted using three data split scenarios (60:40, 70:30, and 80:20) to examine the effect of training and testing proportions on classification performance and determine the best configuration. This system is expected to accurately and efficiently classify phishing emails and serve as a practical solution to enhance user awareness and security against cyber threats.

2. METHOD

This study employed several methodological steps, ranging from data collection, text preprocessing, feature extraction, model training and evaluation, to the implementation of a phishing email prediction system using the Random Forest algorithm and Natural Language Processing (NLP). Each stage was carried out sequentially to achieve optimal classification performance. The research methodology workflow is illustrated in Figure 1.

Figure 1. Research Workflow Diagram

2.1. Data Collection

The dataset used in this study was obtained from the Kaggle platform, a public data repository commonly used for training and evaluating machine learning models. The dataset, titled Phishing Email Dataset, is publicly available at: https://www.kaggle.com/datasets/naserabdullahalam/phishing-email-dataset. It contains approximately 82,486 email data entries, divided into 42,891 phishing emails and 39,595 legitimate emails. Each data entry consists of a `text_combined` feature representing the full content of the email, and a label: 1 for phishing, 0 for non-phishing [8].

This dataset can be categorized as relatively balanced since the distribution between phishing and legitimate classes does not show extreme disparity. With approximately 52% phishing and 48% legitimate, the machine learning model can learn from both classes fairly, without significant bias toward either label [9]. Such balance is crucial for achieving stable and accurate classification performance across both classes [10].

2.2. Text Preprocessing

The preprocessing process is a critical initial step in Natural Language Processing (NLP) to clean and prepare textual data before proceeding to the feature extraction and machine learning model training stages [11]. In this study, preprocessing is carried out through several interrelated and sequential main stages, namely lowercasing, cleansing, tokenization, stopword removal, normalization, automatic translation, and concluding with feature extraction using the Term Frequency–Inverse Document Frequency (TF-IDF) method.

The first step is lowercasing, which involves converting all letters in the text to lowercase to ensure consistency in word weighting. This is followed by cleansing, which entails cleaning the text of unnecessary characters such as numbers, symbols, and punctuation marks using regular expression (regex) techniques. Once the text is cleaned, tokenization is performed to break the text into individual word segments (tokens), followed by stopword removal to eliminate common words such as “the,” “is,” and “and” using the NLTK library. The remaining tokens are then normalized into a structured text format, ready for feature extraction.

To handle inputs from users who use languages other than English, automatic translation is applied using the Google Translate API through the Deep Translator library. This translation ensures that all input texts align with the training dataset, which is in English, thereby improving consistency in classification.

2.3. Feature Extraction

After completing the preprocessing stage, the cleaned textual data must be converted into a numerical format to be processed by machine learning algorithms. This conversion process is known as feature extraction, which aims to represent text documents in the form of numerical vectors so that the linguistic characteristics of the text can be analyzed mathematically. In this study, the feature extraction method used is Term Frequency–Inverse Document Frequency (TF-IDF).

TF-IDF is one of the most widely used text representation techniques in Natural Language Processing (NLP). This method calculates the weight of a word based on two main components: Term Frequency (TF) and Inverse Document Frequency (IDF). The TF component measures how frequently a word appears in a document, while IDF measures how important that word is across the entire corpus of documents. Words that frequently appear in a single document but rarely appear in others will have a high TF-IDF value, making them more informative for distinguishing context [12].

In this study, TF-IDF implementation was carried out using the `TfidfVectorizer` function from the Scikit-learn library. This process produces a numerical feature matrix in which each row represents an email and each column indicates the weight of words that have gone through the preprocessing steps. This representation is then used as the primary input for classification algorithms such as Random Forest to determine whether an email is phishing or not. The main advantage of using TF-IDF lies in its ability to reduce the influence of common words (stopwords) and emphasize important words that truly differentiate between phishing and legitimate emails [12].

2.4. Modeling and Training

The modeling and training stage is a critical process in building a classification system capable of distinguishing between phishing and legitimate emails based on the numerical representation of text [13]. In this study, the algorithm used for model training is Random Forest, an ensemble learning method based on decision trees that has proven effective in handling high-dimensional data such as that resulting from the TF-IDF method [14].

Random Forest operates by constructing a large number of decision trees during the training process, then combining the predictions from each tree using a majority voting approach to produce the final decision. The strength of this algorithm lies in its ability to reduce the risk of overfitting and deliver stable classification results, even when applied to complex and unstructured textual data [15]. Additionally, Random Forest can handle feature correlations and perform reliably even in the presence of noise within the data [16].

To train and evaluate the model's performance, three data split scenarios were applied: 60:40, 70:30, and 80:20. The first percentage denotes the proportion of training data, while the second indicates the proportion of testing data. The 60:40 scenario is used to observe how the model adapts to a more limited training dataset. The 70:30 scenario offers a balance between training and testing data, while the 80:20 scenario is used to evaluate the extent of performance improvement when more data is allocated for training.

2.5. Model Evaluation

The purpose of model evaluation is to measure the performance of the classification system that has been developed, in order to determine how well the model is able to distinguish between phishing and legitimate emails. In this study, the evaluation was conducted on the Random Forest model trained using three data split scenarios: 60:40, 70:30, and 80:20. Each scenario was assessed using several standard metrics in the fields of machine learning and Natural Language Processing (NLP), each reflecting different aspects of classification performance.

The accuracy metric is used to measure the proportion of total correct predictions out of all test data. The higher the accuracy score, the better the model's general capability in making correct classifications [9].

Therefore, additional evaluation metrics such as precision, recall, and F1-score are also used to complement the assessment. Precision indicates how many of the emails predicted as phishing were actually phishing. Recall measures the model's ability to detect all actual phishing emails present in the dataset. The F1-score is the harmonic mean of precision and recall, providing a comprehensive overview especially when there is an imbalance between the number of classes in the data.

2.6. Implementation of the Prediction Feature

After the Random Forest model has been trained and evaluated, the next step is to test the model's ability to perform predictions based on direct text input. This stage aims to evaluate the

extent to which the model can be used for real-time phishing email detection using new text inputs not included in the training data.

The process begins by manually entering the email text as input into the system. The text then undergoes the same preprocessing steps as in the training process, which include lowercasing, cleansing, tokenization, stopword removal, and normalization. If the input text is in a language other than English, the system automatically translates it into English using the Google Translate API, ensuring consistency with the language used during model training.

Once preprocessing is complete, the text is transformed into a numerical representation using the Term Frequency–Inverse Document Frequency (TF-IDF) method. This representation is then passed to the previously saved Random Forest model, stored in a serialized format (e.g., using joblib or pickle). The model outputs a binary label: 1 if the email is classified as phishing, and 0 if it is classified as legitimate. Subsequently, the model automatically displays the output for the entered sentence, indicating whether it is predicted to be phishing or not. This prediction output can be reviewed directly to assess the classification accuracy for the given input text. The implementation of this prediction feature demonstrates that the system is capable of handling new text inputs outside the training data and consistently delivering accurate classification results.

3. RESULTS AND DISCUSSION

This section presents the conclusions drawn from the research on the phishing email detection system based on Natural Language Processing (NLP) using the TF-IDF method and the Random Forest algorithm. The discussion is divided into three main aspects: preprocessing and feature extraction results, model performance evaluation, and analysis in comparison with prior research.

3.1. Preprocessing and Feature Extraction Results

This stage involved cleaning the textual data (preprocessing) and extracting features as the initial steps in the phishing email detection system. Preprocessing aimed to reduce noise and simplify the text structure to facilitate processing by the machine learning model. The steps included converting all letters to lowercase (lowercasing), removing non-alphabetic characters (cleansing), eliminating common words (stopword removal), and tokenization.

Table 1. Data Before Preprocessing

index	text_combined	label
0	hpl nom may 25 2001 see attached file hplno 52...	0
1	nom actual vols 24 th forwarded sabrae zajac h...	0
2	enron actuals march 30 april 1 201 estimated a...	0
3	hpl nom may 30 2001 see attached file hplno 53...	0
4	hpl nom june 1 2001 see attached file hplno 60...	0
5	hpl nom may 31 2001 see attached file hplno 53...	0
6	9760 tried get fancy address came back forward...	0
7	hpl noms february 15 2000 see attached file hp...	0
8	fw pooling contract template original message ...	0
9	hpl nom march 28 2000 see attached file hplo 3...	0

As shown in Table 1, the top 10 raw data entries in the `text_combined` column still contain extraneous elements such as numbers, dates, and special characters or punctuation. Since this data has not yet undergone preprocessing, it still contains considerable noise that may interfere with feature extraction by the phishing detection model.

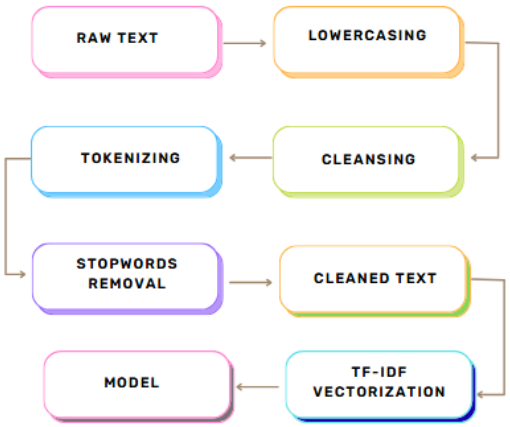

Figure 2. Preprocessing Workflow

Figure 2 illustrates the preprocessing and feature extraction stages in the text-based phishing email detection system. The process begins with raw textual data, which then undergoes a series of cleaning and transformation steps. The first step is lowercasing, which converts all characters to lowercase to maintain word consistency. This is followed by cleansing, which removes non-alphabetic characters such as numbers, symbols, and punctuation marks that are irrelevant to the analysis. Subsequently, tokenizing is performed to break down sentences into individual words or tokens. This process continues with stopwords removal, which eliminates common words that do not significantly contribute to the classification process. The outcome of these steps is a cleaned text, which is then transformed into a numerical representation through the TF-IDF Vectorization method. This representation is subsequently used as input for the model training stage, such as with the Random Forest algorithm, to predict whether an email is phishing or not. All of these stages are designed to ensure that the data fed into the model is in an optimal form and free from noise that could reduce the system's classification accuracy.

Table 2. Data After Preprocessing

index	cleaned	label
0	hpl nom may see attached file hplno xls hplno xls	0
1	nom actual vols th forwarded sabrae zajac hou	0
2	enron actuals march april estimated actuals ma	0
3	hpl nom may see attached file hplno xls hplno xls	0
4	hpl nom june see attached file hplno xls hplno	0
5	hpl nom may see attached file hplno xls hplno xls	0
6	tried get fancy address came back forwarded la	0
7	hpl noms february see attached file hplo xls h	0
8	fw pooling contract template original message	0
9	hpl nom march see attached file hplo xls hplo xls	0

Table 2 shows the top 10 entries after preprocessing. The text has been cleaned and structured—non-alphabetic characters are removed, stopwords eliminated, and all characters converted to lowercase. At this stage, the text is ready for feature extraction using the TF-IDF vectorizer.

3.2. Random Forest Model Performance Evaluation

To evaluate the system's effectiveness in detecting phishing emails, the Random Forest model was tested using three data split scenarios: 60:40, 70:30, and 80:20. The evaluation employed metrics

such as accuracy, precision, recall, and F1-score, which were derived from the model's predictions on test data for each scenario.

Table 3. Model Evaluation Metrics

Split Ratio	Accuracy	Precision	Recall	F1-Score
60:40	0,9784	0,9901	0,9658	0,9778
70:30	0,9800	0,9915	0,9679	0,9796
80:20	0,9801	0,9908	0,9687	0,9796

As shown Table 3, the Random Forest model demonstrated high and consistent performance across all evaluation metrics. For the 60:40 split, accuracy reached 0.9784 with a precision of 0.9901 and recall of 0.9658, resulting in an F1-score of 0.9778. When training data was increased to 70:30, performance improved with an accuracy of 0.9800, recall of 0.9679, and F1-score of 0.9796. The 80:20 split yielded the highest accuracy of 0.9801 and a recall of 0.9687, though precision slightly decreased compared to the 70:30 scenario. Nonetheless, the F1-score remained stable at 0.9796. These results confirm that the TF-IDF and Random Forest combination is effective and robust for accurate phishing email detection, regardless of training/testing split variations.

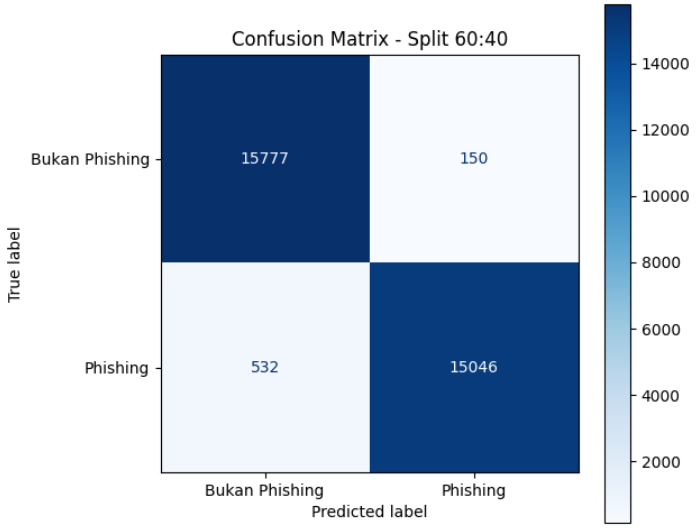

Figure 3. Confusion Matrix 60:40

Figure 3 illustrates the confusion matrix for the 60:40 data split scenario, where the model was able to recognize phishing and non-phishing emails with considerable accuracy. A total of 15,777 instances were correctly classified as non-phishing (true negative), and 15,046 instances were correctly identified as phishing (true positive). Meanwhile, misclassifications consisted of 150 false positives (legitimate emails incorrectly identified as phishing) and 532 false negatives (phishing emails incorrectly identified as legitimate). These results indicate that the model demonstrates strong performance, despite the presence of minor classification errors.

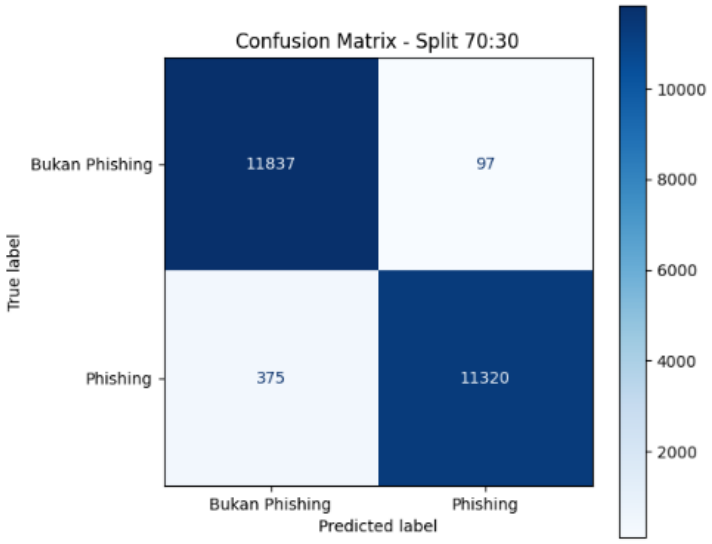

Figure 4. Confusion Matrix 70:30

Figure 4 presents the confusion matrix for the 70:30 data split scenario. In this scenario, the model demonstrated stable results with an increased amount of training data. A total of 11,837 instances were correctly classified as non-phishing (true negatives), and 11,320 instances were correctly classified as phishing (true positives). Meanwhile, there were 97 false positive cases (legitimate emails incorrectly detected as phishing) and 375 false negative cases (phishing emails incorrectly classified as legitimate). These results indicate that the 70:30 data split provides a fairly balanced performance in detecting both classes.

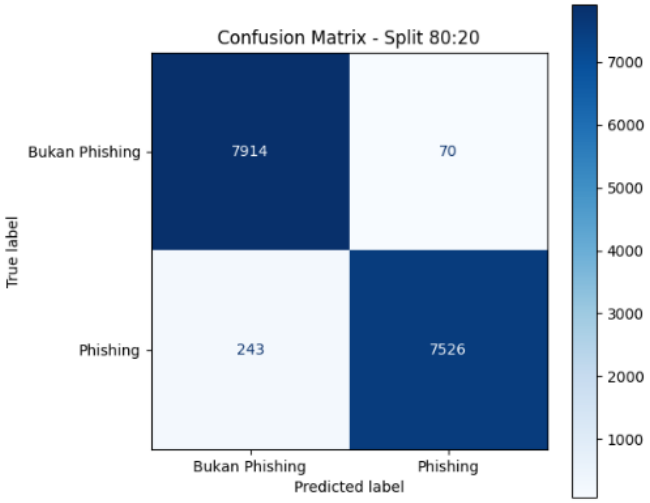

Figure 5. Confusion Matrix 80:20

Figure 5 presents the confusion matrix for the 80:20 data split scenario. In this scenario, with the largest amount of training data, the model achieved the best performance with the fewest classification errors. A total of 7,914 legitimate emails were correctly identified (true negatives), and 7,526 phishing emails were also accurately classified (true positives). The number of misclassifications was lower compared to other scenarios, with 70 false positive cases and 243 false negative cases recorded. These results indicate that the greater the proportion of training data used, the better the model's ability to accurately recognize data patterns.

3.3. Feature Importance Analysis and Linguistic Patterns in Phishing Emails

After training the model using the Random Forest algorithm, the next step is to conduct an analysis of the contribution weights of each feature (feature importance) in determining whether an email is classified as phishing or non-phishing. These features are derived from the transformation of textual data using the Term Frequency–Inverse Document Frequency (TF-IDF) method, which effectively represents the significance of each word within the dataset.

Table 4. Results of Feature Importance Analysis

No.	Feature	Importance
1	enron	0.016738
2	http	0.012702
3	cc	0.009882
4	save	0.008189
5	pm	0.008055
6	vince	0.007732
7	thanks	0.007561
8	money	0.007245
9	attached	0.006751
10	click	0.006711
11	online	0.006699
12	subject	0.006496
13	software	0.005146
14	please	0.005046
15	ect	0.004946
16	life	0.004666
17	sex	0.004516
18	low	0.004434
19	louise	0.004400
20	let	0.004176

To provide a clearer overview of the feature importance analysis results, Table 4 presents the top 20 features based on their contribution to the model's performance. This table not only highlights the most influential features in the classification process but also represents words that consistently appear in the structure and content of the analyzed emails.

Upon further examination, most of the identified features consist of common vocabulary frequently encountered in email communications, whether in personal or professional contexts. For example, the words "click," "online," and "money" often appear in phishing emails designed to lure users into clicking on specific links or performing risky actions, such as providing sensitive information.

In addition, the presence of words such as "attached" and "subject" indicates that structural elements of emails also contribute significantly to the detection process. These findings demonstrate that the model considers not only the message content but also the technical components that constitute the overall email structure.

There are also several words derived from specific contexts, such as "enron," "vince," and "louise." The appearance of these terms suggests that the dataset used may include historical emails originating from particular organizations or individuals, meaning that personal or institutional characteristics within the data also influenced the model's learning outcomes.

Furthermore, the presence of terms such as "sex," "life," and "low" is also noteworthy. These words commonly appear in spam or phishing emails that employ emotionally manipulative strategies to capture the victim's attention. The existence of such features reflects the model's capability to recognize more subtle and not always explicit linguistic contexts.

Overall, the results of this analysis demonstrate that the constructed model is capable of identifying a wide range of linguistic patterns—from general terminology and structural elements to emotional expressions. These findings affirm that the combination of Natural Language Processing (NLP) techniques and the Random Forest algorithm not only yields metrically accurate classifications but also offers a deeper understanding of the linguistic characteristics of phishing emails.

4. CONCLUSION

This study successfully developed a phishing email detection system using a Natural Language Processing (NLP) approach by applying the Term Frequency–Inverse Document Frequency (TF-IDF) method and the Random Forest algorithm. The system demonstrated excellent classification performance, achieving the highest accuracy of 98.01% in the 80:20 data split scenario, along with high and balanced precision, recall, and F1-score values.

The feature importance analysis revealed that the model is capable of identifying key terms and linguistic patterns that frequently appear in phishing emails, both in terms of content and technical structure. This affirms the system's ability to comprehend the linguistic characteristics of malicious messages.

Compared to previous studies, such as Umam et al., which achieved an accuracy of 96.57%, and Bountakas et al., which reported 93.41% accuracy using the TF-IDF and Random Forest combination, the system in this study demonstrates improved performance. This is most likely due to the use of a significantly larger dataset, enabling the model to recognize phishing patterns more accurately.

This research opens up opportunities for the development of more adaptive and responsive detection systems to combat increasingly complex phishing attacks. Moving forward, the system can be further enhanced to meet specific needs, whether in terms of technical aspects, algorithmic approaches, or dataset coverage. Therefore, this study may serve as a foundation for building stronger, smarter, and more responsive cybersecurity systems.

REFERENCES

[1] I. Hadi Ramadhan and E. Kumalasari Nurnawati, "Analisis Ancaman *Phishing* Dalam Layanan E-Commerce," *Pros. SNAST*, pp. E31-41, Nov. 2022, doi: 10.34151/prosidingsnast.v8i1.4169.

[2] S. S. Roy, K. V. Naragam, and S. Nilizadeh, "Generating *Phishing* Attacks using ChatGPT," 2023, *arXiv*. doi: 10.48550/ARXIV.2305.05133.

[3] K. A. Jackson, "A Systematic Review of Machine Learning Enabled *Phishing*," 2023, *arXiv*. doi: 10.48550/ARXIV.2310.06998.

[4] N. Altwaijry, I. Al-Turaiki, R. Alotaibi, and F. Alakeel, "Advancing *Phishing* Email Detection: A Comparative Study of Deep Learning Models," *Sensors*, vol. 24, no. 7, p. 2077, Mar. 2024, doi: 10.3390/s24072077.

[5] C. Umam, L. B. Handoko, and F. O. Isinkaye, "Performance Analysis of Support Vector Classification and Random Forest in *Phishing* Email Classification," *Sci. J. Inform.*, vol. 11, no. 2, pp. 367–374, May 2024, doi: 10.15294/sji.v11i2.3301.

[6] P. Bountakas, K. Koutroumpouchos, and C. Xenakis, "A Comparison of Natural Language Processing and Machine Learning Methods for *Phishing* Email Detection," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, Vienna Austria: ACM, Aug. 2021, pp. 1–12. doi: 10.1145/3465481.3469205.

[7] S. R. Abdul Samad *et al.*, "Analysis of the Performance Impact of Fine-Tuned Machine Learning Model for *Phishing* URL Detection," *Electronics*, vol. 12, no. 7, p. 1642, Mar. 2023, doi: 10.3390/electronics12071642.

[8] N. A. Alam, "*Phishing* Email Dataset." 2024. doi: https://doi.org/10.34740/kaggle/ds/5074342.

[9] A. Alhuzali, A. Alloqmani, M. Aljabri, and F. Alharbi, "In-Depth Analysis of *Phishing* Email Detection: Evaluating the Performance of Machine Learning and Deep Learning Models Across Multiple Datasets," *Appl. Sci.*, vol. 15, no. 6, p. 3396, Mar. 2025, doi: 10.3390/app15063396.
[10] K.-T. Tham, K.-W. Ng, and S.-C. Haw, "*Phishing* Message Detection Based on Keyword Matching," *J. Telecommun. Digit. Econ.*, vol. 11, no. 3, pp. 105–119, Sep. 2023, doi: 10.18080/jtde.v11n3.776.
[11] S. Atawneh and H. Aljehani, "*Phishing* Email Detection Model Using Deep Learning," *Electronics*, vol. 12, no. 20, p. 4261, Oct. 2023, doi: 10.3390/electronics12204261.
[12] A. Al-Subaiey, M. Al-Thani, N. Abdullah Alam, K. F. Antora, A. Khandakar, and S. A. Uz Zaman, "Novel interpretable and robust web-based AI platform for *phishing* email detection," *Comput. Electr. Eng.*, vol. 120, p. 109625, Dec. 2024, doi: 10.1016/j.compeleceng.2024.109625.
[13] N. Palanichamy and Y. S. Murti, "Improving *Phishing* Email Detection Using the Hybrid Machine Learning Approach," *J. Telecommun. Digit. Econ.*, vol. 11, no. 3, pp. 120–142, Sep. 2023, doi: 10.18080/jtde.v11n3.778.
[14] A. A. Tawil, L. Almazaydeh, D. Qawasmeh, B. Qawasmeh, M. Alshinwan, and K. Elleithy, "Comparative Analysis of Machine Learning Algorithms for Email *Phishing* Detection Using TF-IDF, Word2Vec, and BERT," *Comput. Mater. Contin.*, vol. 81, no. 2, pp. 3395–3412, 2024, doi: 10.32604/cmc.2024.057279.
[15] C. Sasirekha, R. Nandhini, K. Mai N L, and Bhuvaneshwari, "email-*phishing*-detection-using-machine-learning." 22-06-2023, 2023.
[16] W. Sarasjati, S. Rustad, Purwanto, H. A. Santoso, and D. R. I. M. Setiadi, "*Phishing* Detection Using Random Forest-Based Weighted Bootstrap Sampling and LASSO+ Feature Selection," *Int. J. Saf. Secur. Eng.*, vol. 14, no. 6, pp. 1783–1794, Dec. 2024, doi: 10.18280/ijsse.140613.