# Email spam detection with Lexical information

1Kornelius Aditya Septemedi, 2Darmstater Albertus Albert D. , dan

3Yoel Amazia Prayitno

1,2,3,4,5,6Program Studi Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata

1,2,320k10021@student.unika.ac.id, 4,520k10035@student.unika.ac.id 620k10022@student.unika.ac.id

Abstract

The background of this research is that lexical information has not been further explored for spam email classification. This research is expected to develop an interesting spam email discovery strategy by using a text investigation approach. The main focus of this research is to investigate the substance of the message and recognise the qualities that can distinguish spam messages from genuine messages by adding Stemmer and Post Tagging as preprocessing. There are 7 methods to perform this analysis such as SVM, multinomial naive bayes, KNN, decision tree, random forest, adaboost, and bagging. From the calculation of precision, recall, and f1-score from the training process, SVM and MNB have superior results compared to other algorithms, SVM gets accuracy results at score 2 (0.9848), score 3 (0.9372), and score 4 (0.9834). MNB gets score 1 (0.9380), score 2 (0.9784), score 3 (0.9647) and score 4 (0.9688). KNN has the lowest result after using Post Tagging and Stemmer in the preprocessing process. These results can be used as a basis for developing spam detection systems using more effective lexical information exploration in the future.

**Keywords:** Lexical information, Spam email classification, Precision, Recall, F1-score, SVM, KNN, multinomial Naive Bayes, decision tree, random forest, adaboost, bagging.

## introduction

During this time, email has become one of the most important special tools in many areas of our lives. However, along with the widespread use of email, there is also a growing problem of spam or unsolicited messages. Spam messages are annoying and can compromise your security and privacy. This work will develop interesting strategies for spam detection using text inspection approaches. The main focus of this research is to examine the content of messages and identify characteristics that can distinguish spam messages from real messages. This research uses a combination of machine learning algorithms and natural language processing (NLP) techniques. The sample information used consists of various spam messages and real messages.

Information preprocessing and tokenization are performed during the interaction to cleanse and organize the information. As a result, appropriate order calculus is applied to represent and organize the recognition framework. Model presentation is evaluated using relevant measures such as Accuracy, Accuracy, Validation, and F1 Score. The usual result of this check is to increase the feasibility and productivity of spam email detection, improve security, and inspect lexical information.

## literature review

Fahrur Rozi and Rikie Kartadie’s [1] research, purpose is to develop a method to detect spam and non-spam emails by combining email and spam preprocessing, function selection, and cluster construction with a naive Bayesian classifier. The dataset used in this study consisted of 400 non-spam and 100 spam emails for training and 200 emails (160 non-spam and 40 spam) for testing. increase. Limitations of this study include the small size of the datasets used for training and testing. In summary, a hybrid technique of fuzzy logic and association rule mining shows potential for effective email and spam detection. However, further research with larger datasets and consideration of other email threats are needed to improve the robustness and applicability of this method.

Fitriyah et al.[2] This research aims to detect spam in emails using the Naive Bayes algorithm based on content characteristics. The dataset used is a spam-based dataset accessed via the UCI Machine Learning Repository. This dataset consists of 4,601 records, of which 1,813 records belong to the spam email category and 2,788 records belong to the non-spam email records. This dataset contains 58 functions. One of the limitations of this research is that it does not capture all aspects of spam emails as it relies solely on content-based features. The results showed that using k=9 in k-fold cross-validation gave the highest precision, precision, and recall scores. The average number of correctly classified data was 3903 and the average number of incorrectly classified data was 698. The average precision was 84.8%, the average precision was 0.86, and the average recall was 0.85.

Hartono et al.[3] This research is conducted to identify problems related to research subjects, select methods and algorithms to be used to solve the problems, determine solutions to the identified problems and set research goals. The dataset used was obtained from Kaggle. The results show that traditional machine learning models (SVM and RF) outperform deep learning models (LSTM) in terms of accuracy. The model evaluation shows that traditional machine learning models achieve 98% accuracy.

Firdausillah et al.[4] The purpose of this research is to develop a spam detection system using the SCRUM framework, TF-IDF method, and Support Vector Machine (SVM) algorithm. The SCRUM framework was chosen for its simple, iterative, and incremental approach to product development. Primary data consists of spam and ham mail records obtained from public sources, in particular his Kaggle. These records were used to train and test our spam detection system. Limitations of this study are the limited dataset, limited feature selection, and lack of comparison with other methods. The study concludes that this early warning system can be used by organizations conducting email marketing to ensure that sent emails are safely returned and cannot harm their business. I'm here. The SPAM detection process is a two-step process: domain address verification using DNSBL and email content verification using models developed in SVM. The developed model was validated with 10-fold cross-validation with an average accuracy of 97.54%. This makes it effective enough for use in SPAM detection and prevention systems.

Dada et al.[5] This research provides an overview of research issues related to email spam, its impact on users, and steps that can be taken to reduce its impact. The limitation of this journal is that it does not provide details of machine learning algorithms, simulation tools, publicly available datasets, and the architecture of the email spam environment. The article also does not present the parameters used by previous research in evaluating the proposed techniques. In conclusion, the reviewed journal articles provide insights into research issues, techniques, and actions related to email spam filtering. However, there are limitations in terms of coverage of recent research, comparative analysis, and details of machine learning algorithms, simulation tools, data sets, and email spam archives.

Vutharkar Nagaveni and Dr. Vimal Pandyacture’s [6] research focuses on classifying and detecting spam emails using machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM), and J48 Classifier. Network repository (http://networkrepository.com), d. H. spam email database. The dataset contains 57 attributes and 4601 instances, of which 1813 emails are spam and the remaining 2788 emails are non-spam. The dataset data type is multivariate with real and integer-valued attributes. A limitation of this journal is the lack of more detailed analytical algorithms such as genetic algorithms and classification techniques to spot spam. The journal article concludes that the J48 algorithm, a decision tree classifier, has the best spam classification accuracy. In experiments, an accuracy of 92.68% was achieved. The SVM algorithm also showed good results with 90.48% accuracy and good performance on other parameters. However, the accuracy rate of the Naive Bayes algorithm was 79.56% compared to other classifiers.

Yanhui Guo et al[7]. presents a new approach to email spam detection using pre-trained Transformers Bidirectional Encoder Representations (BERT) models. I'm here. Transfer learning was used in this study, where a pre-trained BERT model was used to extract word embeddings from the email body and used as features for subsequent processing. The journal article uses two of his publicly available datasets. The first dataset used was his Enron spam dataset used by Androutsopoulos et al. has been published. (2006). This dataset consists of 33,716 emails, including 17,171 spam emails and 16,545 hamemailss [4]. The second dataset used is the spam or non-spam dataset published by Raftogiannis (2021). This dataset consists of 2,999 valid samples, including 499 spam and 2,500 ham. A limitation of this journal is that it does not explicitly mention the limitations of the proposed approach to email spam detection using trained BERT models. Finally, this journal article proposes a new approach to email spam detection using pre-trained BERT models and supervised machine learning classifiers. This study uses two publicly available data sets, including Enron's spam data set and spam or not data set. The performance of various machine learning algorithms such as SVM, logistic regression, random forest, and ANN are evaluated using precision, recall, F1 score, and AUC metrics.

Yuliya Kontsewaya et al. [8]. The background of the research is the problem of spam mail and the need for a spam filter system. The purpose of the research is to reduce the amount of spam using classifiers. To achieve accurate spam classification, machine learning techniques were chosen for comparison, namely Naive Bayes, K-Nearest Neighbors, SVM, Logistic Regression, Decision Trees, and Random Forest. The dataset used in the study had a size of (5695, 2) after removing duplicates. The study concludes that machine learning methods, especially naive Bayes and logistic regression, achieved the highest spam detection accuracy of up to 99%. These results highlight the effectiveness of machine learning algorithms in reducing spam volume and highlight the importance of developing intelligent spam filtering systems. This study suggests that combining algorithms or filtering methods can further improve the performance of spam detection classifiers. Overall, the findings help address the spam email problem and provide insights for designing more efficient spam filtering systems.

Hanif Bhuiyan et al. [9]. This journal describes how spam filtering technology has evolved over the years by incorporating machine learning algorithms, feature selection, and metrics to improve the accuracy and effectiveness of filtering unwanted messages. A limitation of this journal is that spam filtering techniques designed for one language or culture may not be effective when applied to another language or culture, and that spammers constantly adapt tactics to bypass spam filters. This is because filtering technology is difficult to keep up with. new spam technology. In summary, spam filtering technology plays an important role in preventing unwanted messages from reaching your users' inboxes. However, these techniques have certain limitations that need to be addressed to improve their efficacy. Limitations include but are not limited to, false positives and false positives, evolving spam techniques, language and cultural idiosyncrasies, excessive customization, computational complexity, privacy concerns, and hostile attacks. yeah.

Craig Beaman and Haruna Isnah [10] research, The background to this research is that email-based attacks such as phishing, email malware, and ransomware pose significant threats to users. Previous research has mainly focused on using email body and subject content to detect anomalies, but little research has been done on the use of email header information. This research aims to fill this gap by using machine learning techniques to detect email anomalies based solely on header information. This study uses two data sets. One is his 2007 dataset consisting of spam and ham emails, and the other is a dataset containing phishing emails. The researchers acknowledged the limitations of using a single server and year for their dataset, and the need for a comprehensive dataset that included emails from multiple servers, different years, and different types of phishing emails. suggesting sexuality. In summary, this research focused on the use of email header information for anomaly detection in emails, especially spam and phishing emails. The study found that especially for supervised learning, email headers alone contain enough information to accurately determine whether an email is anomalous. The research used machine learning techniques such as random forests, SVMs, MLP neural networks, and ANNs to achieve high accuracy in detecting phishing and spam emails. Additionally, unsupervised learning using a one-class SVM model also showed good accuracy in detecting anomalies.

## METHODOLOGY

### Research Methodology

This research was conducted by dividing 2 groups of data, namely test data and training data. This research uses a flowchart that contains an overview of the research conducted from the beginning to the end of the research. The flowchart can be seen in Gambar.1 Flowchart.

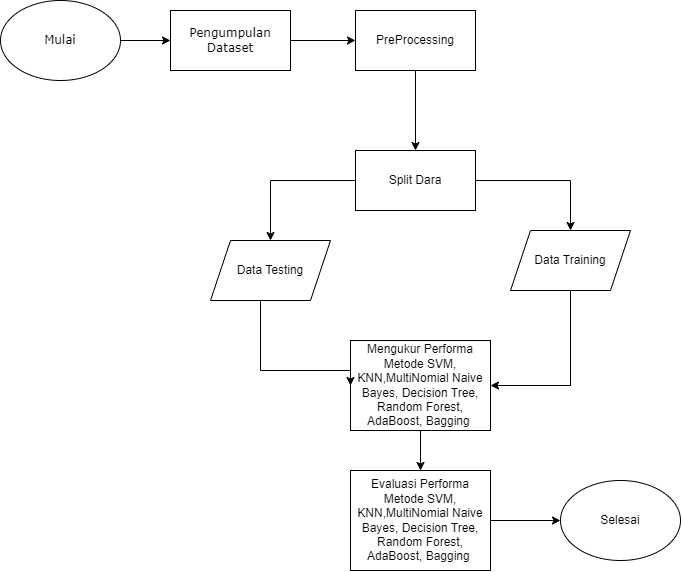


Figure 1

### Dataset Collection

The data for this research was obtained from Kaggle, and the dataset used is Preprocessed TREC 2007 Public Corpus Dataset which can be accessed at <https://www.kaggle.com/datasets/imdeepmind/preprocessed-trec-2007-public-corpus-dataset>.

### PreProcessing

### 

Figure 2

#### Case Folding

At this stage, it is converted to lowercase, removing punctuation, short words, '\n' and '\t', extra spaces, quoted text, and progressive pronouns.

#### Lemmatization

At this stage, the process of converting the inflectional word to its root or root word is performed.

#### Stopwords

The filter phase removes stopwords from the tweet data, stopword removal is done to remove words that still have meaning and also perform spam text detection.

### Split Data

Dividing the dataset into different subsets for training, and testing purposes in modeling or data analysis, which aims to avoid overfitting and ensure that the model or analysis built can be generalized well to data that has never been seen before.

### Mengukur Performa

#### SVM

SVM is used to maximize the distance between classes in order to locate the best hyperplane. A characteristic that can be utilized to divide classes is a hyperplane. The classifying function is referred to as a line in two dimensions, a plane in three dimensions, and a hyperplane in four or more dimensions [11].

A diagram of a graph

Description automatically generated

Figure 3

Figure 3 shows the hyperplanes detected by SVM. Their position is midway between the two classes. This means that the distance between the hyperplane and the data object is different for the adjacent (outermost) class that gets an empty positive round character. In SVM, the outermost data objects closest to the hyperplane are called support vectors. Objects, called support vectors, are the most difficult to classify because they are placed almost on top of other classes. Due to its importance, SVM only considers this support vector to find the optimal hyperplane.

#### KNN

From the image below, some data points are split into two classes, A (blue) and B (yellow). For example, I have new data (black) that I use to predict classes using the KNN algorithm. After calculating the distance between the black point and the other data points, we get the three closest points, consisting of two yellow points and one blue point, as shown in the red box. The new data (black dots) will have class B (yellow) [12].

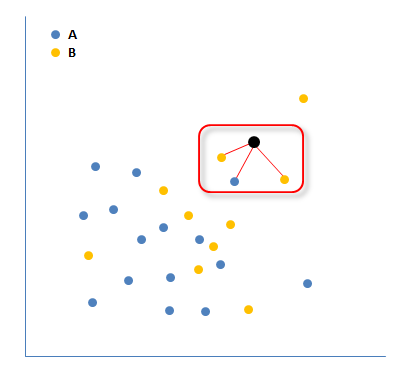


Figure 4

The Euclidean distance measures uclidian distance and identifies who are its neighbors.

#### MultiNomial Naïve Bayes

Multinomial naive Bayes is one of the specialized methods of naive Bayes as a text mining method in the text classification process using class probabilities in documents. The process begins by entering the training data used for learning and then calculating the probability of the occurrence of a class in the training data using Eq.

Further calculation of the probability that word I belongs to a particular category or class can be done using Equation 2.

The addition of the value of one serves to prevent the probability result from being 0, this manipulation is called place smoothing. After the learning stage is completed, the next stage is the classification of new data based on the learning results. For the classification of new data, calculations are carried out using Equation 3.

#### Decision Tree

A decision tree is a method of processing data to predict the future by building a classification or regression model in the form of a tree structure. This is done through continuous decomposition into smaller subsets and, at the same time, step-by-step development of the decision tree. The result of the process is a tree containing decision nodes and leaf nodes.

Decision trees are also useful for data exploration to find relationships between multiple candidate input variables and target variables. Data exploration and decision tree modeling are excellent first steps in the modeling process and are used as final models for several other techniques.

*Gain Calculation*

*Calculating Entropy Value*

#### Random Forest

The Random Forest algorithm is employed for classifying large volumes of data by combining multiple decision trees from a reliable Decision Tree model into a single model. Increasing the number of trees positively impacts the accuracy of the model. The classification decision in Random Forest is made based on the aggregated voting outcomes of the individual trees. This ensemble consists of multiple trees, each containing a diverse set of random variables. In this context, X represents real-valued inputs, while Y denotes real-valued responses. The joint distribution PXY(X, Y) is considered unknown, and the goal is to discover the prediction function f(X) that accurately predicts Y. The prediction function is determined by the loss function L(Y, f(X)). In Random Forests, the base learners denoted as hj(X, Θj), where Θj represents a set of random variables, are independent for each learner j = 1, 2, 3, ..., J. To comprehend the Random Forest algorithm effectively, it is crucial to possess a fundamental understanding of the type of tree utilized as the base learner.

#### AdaBoost

Adaboost is used to classify the data into their respective classes. Adaboost searches for class categories based on class weight values. This process continues to repeat, thus updating the class values. Adaboost keeps increasing the weight value for each iteration of the wrong weight value. Adaboost is a typical ensemble learning algorithm with results showing high accuracy. To form the AdaBoost ensemble, we can use the following formula:

#### Bagging

Bagging, or bootstrap aggregation, is a machine learning technique that uses subsets of the original dataset to train multiple base models. These models are built using unstable learning algorithms and their predictions are averaged or combined during testing. Bagging is applicable to both classification and regression problems, providing more robust and accurate predictions by averaging base model outputs. It enhances stability by mitigating the impact of small changes in the training set on the resulting models. Bagging is particularly useful for handling high-variance datasets and reducing overfitting. By leveraging ensemble learning, bagging improves the accuracy and robustness of machine learning models:

Looping for b = 1, 2, . . ., B

1. Create a bootstrap sample {(𝑋 , 𝑌1∗) 1∗, (𝑋 , 𝑌2∗), … , 2∗(𝑋𝑛∗, 𝑌𝑛∗)}

with random replacement of the training data

{(𝑋1, 𝑌1), (𝑋2, 𝑌2), … , (𝑋𝑛, 𝑌𝑛)}

matching with the Cb classifier turned on the corresponding sample bootstrap.

2. Final classifier output:

### RESULT

In this paper, we conduct a follow-up study on spam email detection using machine learning algorithms with additional postagging. We report the accuracy scores of each algorithm in four different scenarios, namely using default parameters, hyperparameter tuning, stemmer and hyperparameter tuning, and using message length features, stemmer, postagging, and hyperparameter tuning.

*SVM*

Using both Post Tagging and Tf-Idf, the accuracy achieved was 95.40%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy improved to 98.48%. With the combination of Post Tagging and Named Entity Recognition (NER) but without Tf-Idf, the accuracy obtained was 93.72%. Finally, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 98.34%. From the results above, the 2nd test condition is the best of the other 4 conditions, with an accuracy rate of 98.48%.

*KNN*

The accuracy achieved when using both Post Tagging and Tf-Idf was 95.81%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy decreased to 81.86%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy obtained was 90.82%. Lastly, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 83.87%. From the results above, the 1st test condition is the best of the other 4 conditions, with an accuracy rate of 95.81%.

*Naïve Bayes*

The accuracy achieved when using both Post Tagging and Tf-Idf was 93.80%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy improved to 97.04%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy obtained was 96.47%. And, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 96.88%. From the results above, the 2nd test condition is the best of the other 4 conditions, with an accuracy rate of 94.04%.

*Decision Tree*

The accuracy achieved when using both Post Tagging and Tf-Idf was 94.91%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy obtained was 95.64%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy was 94.70%. The last, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 95.63%. From the results above, the 2nd test condition is the best of the other 4 conditions, with an accuracy rate of 95.64%.

*Random Forest*

The accuracy achieved when using both Post Tagging and Tf-Idf was 94.11%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy improved to 96.42%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy obtained was 93.86%. Finally, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 96.22%. From the results above, the 2nd test condition is the best of the other 4 conditions, with an accuracy rate of 96.42%.

*AdaBoost*

The accuracy achieved when using both Post Tagging and Tf-Idf was 87.09%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy improved to 88.44%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy obtained was 88.17%. The final condition, utilizing both Post Tagging and NER along with Tf-Idf, the model achieved an accuracy of 88.91%. From the results above, the 4th test condition is the best of the other 4 conditions, with an accuracy rate of 88.91%.

*Baging*

The accuracy achieved when using both Post Tagging and Tf-Idf was 95.36%. When Post Tagging was not used, but Tf-Idf was applied, the accuracy improved to 96.36%. With the combination of Post Tagging and NER, but without Tf-Idf, the accuracy obtained was 95.35%. Finally, utilizing both Post Tagging and along with Tf-Idf, the model achieved an accuracy of 96.29%. From the results above, the 2nd test condition is the best of the other 4 conditions, with an accuracy rate of 96.36%.

In the default parameter usage scenario, the K-Nearest Neighbors (KNN) algorithm has the highest accuracy score with 0.9581, followed by Support Vector Machine (SVM) with 0.9540. However, after performing hyperparameter tuning, SVM outperformed all other algorithms with a higher accuracy score, reaching 0.9848. When we applied a stemmer to the data and performed hyperparameter tuning, the SVM accuracy score dropped to 0.9372. However, when the message length feature, stemmer, postagging, and hyperparameter tuning were used together, SVM again showed better performance with an accuracy score of 0.9834.

Furthermore, the Multinomial Naive Bayes (MNB) algorithm showed stable performance with high accuracy scores in all scenarios, reaching 0.9380 under default parameter usage, 0.9784 after hyperparameter tuning, 0.9647 after stemmer usage, and 0.9688 after adding message length, stemmer, postagging, and hyperparameter tuning features. Decision Tree and Random Forest also show good performance in spam email detection. Decision Tree achieved an accuracy score of 0.9491 with default parameters, while Random Forest achieved an accuracy score of 0.9411. After hyperparameter tuning, Decision Tree achieved an accuracy score of 0.9564, while Random Forest achieved an accuracy score of 0.9642. Although there is a slight decrease in performance after the application of stemmer, postagging, and hyperparameter tuning, both algorithms still maintain high accuracy.

Other algorithms, such as AdaBoost and Bagging, showed lower performance compared to the other algorithms in terms of spam email detection. However, after hyperparameter tuning, both saw a significant improvement in performance, with an accuracy score of 0.8844 for AdaBoost and 0.9632 for Bagging after the application of stemmer, postagging, and hyperparameter tuning.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Score 1** | **Score 2** | **Score 3** | **Score 4** |
| **SVM** | 0.9540 | 0.9848 | 0.9372 | 0.9834 |
| **KNN** | 0.9581 | 0.8186 | 0.9082 | 0.8387 |
| **MNB** | 0.9380 | 0.9784 | 0.9647 | 0.9688 |
| **Decision Tree** | 0.9491 | 0.9564 | 0.9470 | 0.9563 |
| **Random Forest** | 0.9411 | 0.9642 | 0.9386 | 0.9622 |
| **AdaBoost** | 0.8709 | 0.8844 | 0.8817 | 0.8891 |
| **Bagging** | 0.9536 | 0.9636 | 0.9535 | 0.9629 |

Table 1

Figure 5

### CONCLUSION

In this paper, we conducted an advanced research on spam email detection using machine learning algorithms with additional postagging. We reported the performance results of various algorithms in four different scenarios. The experimental results show that SVM and MNB are the two algorithms that perform best in spam email detection, with SVM achieving the highest accuracy scores in most of the tested scenarios. The use of hyperparameter tuning also proved effective in improving the performance of the algorithms in almost all cases.

In addition, we added stemmer and postagging as additional data pre-processing steps. Although there is a slight performance degradation in some algorithms after the application of stemmer, postagging, and hyperparameter tuning, overall those measures make a valuable contribution in improving spam email detection. This research provides a better understanding of the performance of various machine learning algorithms in spam email detection with postagging added. These results can be used as a basis for the development of more effective spam detection systems in the future.

## Daftar Pustaka

[1] F. Rozi and R. Kartadie, “DETEKSI E-MAIL DAN SPAM MENGGUNAKAN FUZZY ASSOCIATION RULE MINING,” *JIPI*, vol. 2, no. 2, Dec. 2017, doi: 10.29100/jipi.v2i2.348.

[2] N. Q. Fitriyah, H. Oktavianto, and H. Hasbullah, “Deteksi Spam Pada Email Berbasis Fitur Konten Menggunakan Naïve Bayes,” *j.sist.teknol.inf.*, vol. 5, no. 1, pp. 1–7, Feb. 2020, doi: 10.32528/justindo.v5i1.3414.

[3] M. B. Hartono and A. K. Darmawan, “Komparasi Deep Learning Dan Traditional Machine Learning Untuk Email Spam Filtering,” vol. 12, 2023.

[4] F. Firdausillah, M. Hafidz, E. D. Udayanti, and E. Kartikadarma, “Sistem Deteksi Surel SPAM Dengan DNSBL Dan Support Vector Machine Pada Penyedia Layanan Mail Marketing,” *josh*, vol. 3, no. 4, pp. 618–625, Jul. 2022, doi: 10.47065/josh.v3i4.1795.

[5] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, “Machine learning for email spam filtering: review, approaches and open research problems,” *Heliyon*, vol. 5, no. 6, p. e01802, Jun. 2019, doi 10.1016/j.heliyon.2019.e01802.

[6] V. Nagaveni and D. V. Pandya, “VARIOUS\_SPAMS\_AND\_CLASSIFICATION\_ALGORITHMS\_FOR\_ DETECTION\_OF\_SPAM\_EMAIL\_THROUGH\_COMPARING\_J48,\_ SVM\_AND\_NAIVE\_BAYES\_CLASSIFIERS\_USING\_WEKA\_TOOL,” 2020.

[7] University of Illinois Springfield, USA, Y. Guo, Z. Mustafaoglu, University of Illinois Springfield, USA, D. Koundal, and the University of Petroleum and Energy Studies, India, “Spam Detection Using Bidirectional Transformers and Machine Learning Classifier Algorithms,” *JCCE*, Apr. 2022, doi: 10.47852/bonviewJCCE2202192.

[8] Y. Kontsewaya, E. Antonov, and A. Artamonov, “Evaluating the Effectiveness of Machine Learning Methods for Spam Detection,” *Procedia Computer Science*, vol. 190, pp. 479–486, 2021, doi: 10.1016/j.procs.2021.06.056.

[9] H. Bhuiyan, A. Ashiquzzaman, and T. I. Juthi, “A Survey of Existing E-Mail Spam Filtering Methods Considering Machine Learning Techniques,” 2018.

[10] C. Beaman, C. Beaman, and H. Isah, “Anomaly Detection in Emails using Machine Learning and Header Information”.

[11] https://medium.com/@samsudiney/penjelasan-sederhana-tentang-apa-itu-svm-149fec72bd02

[12] https://ilmudatapy.com/algoritma-k-nearest-neighbor-knn-untuk-klasifikasi/