Natural Language Analysis

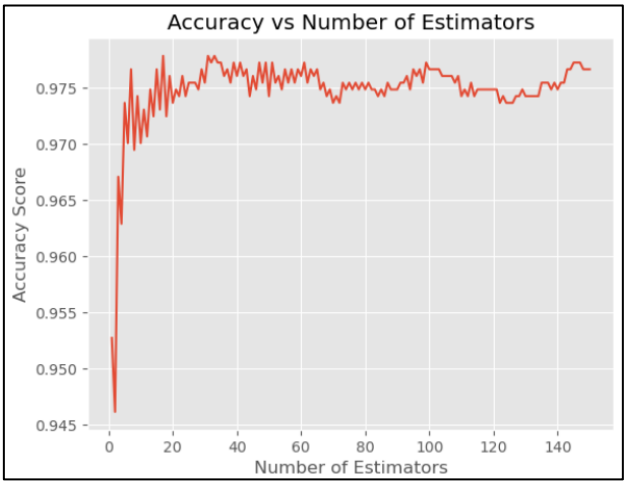
In the realm of natural language analysis, preparing the data properly is important for the success of any model. For the task of SPAM SMS detection, the initial dataset comprised around 85% HAM messages and 15% SPAM messages, reflecting a significant class imbalance. The preprocessing section for these text messages involved converting all text to lowercase, removing punctuation, and eliminating common stopwords. These steps help reduce the noise from the dataset, focusing on the more meaningful content of the messages.

One of the primary feature extraction techniques we applied to the dataset is Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF is an algorithm that counts words' weight by considering the word's frequency and in how many files the word can be found (Hakim et al., 2014). This dual approach helps to amplify the influence of terms that are unique to the documents, with the advantages of distinguishing words that are more likely to be important and are not just random words that commonly appear in texts and handling large datasets with diverse vocabulary very well. However, the algorithm also comes with disadvantages of treating each word independently, ignoring the content and context of the words, and having results that are usually high-dimensional and sparse vectors, which in some situations could be computationally inefficient

On the other hand, Word2vec, is a neural network that includes two learning

models: Continuous Bag of Words (CBOW) and Skip-gram. CBOW predicts the word given its context while Skip-gram do it the opposite (Ma, L., & Zhang, Y., 2015). By training the model on the corpus of SMS messages, each word is viewed as a vector in a defined space where the vector shows similarities to other words. The advantages of Word2vec come with the richness in the semantic area that captures the relationships between words, the efficiency of not having so much dimensional structure compared with TF-IDF, and the ability to fit in and deal with all kinds of natural language problems; meanwhile, it also has the disadvantages of not being able to provide dynamic impact on the work. Once trained, the model cannot interpret any new word without adjustment.

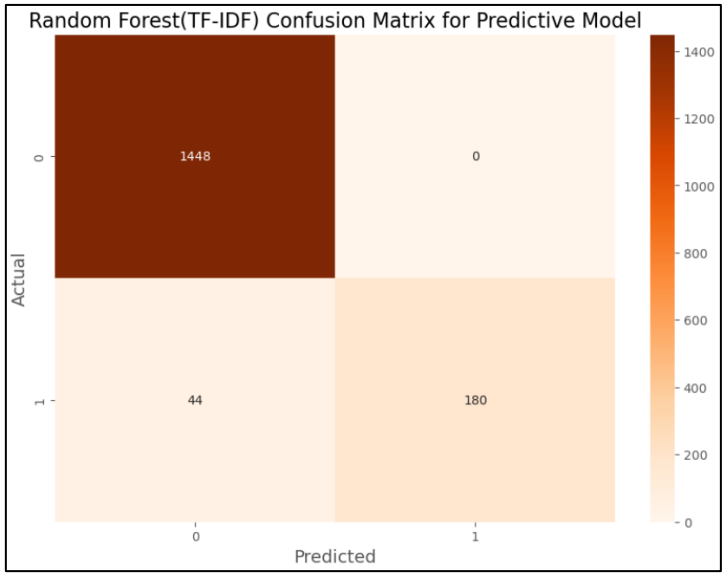
To utilize the features extracted from TF-IDF and Word2vec, the Random Forest classifier was chosen to categorize messages. One key thing to mention is using Grid search to find the best n_estimator number for the best accuracy and efficiency. The result of the model with TF-IDF is as follows:



(Diagram 1: Result of Grid Search of RF with TF-IDF)

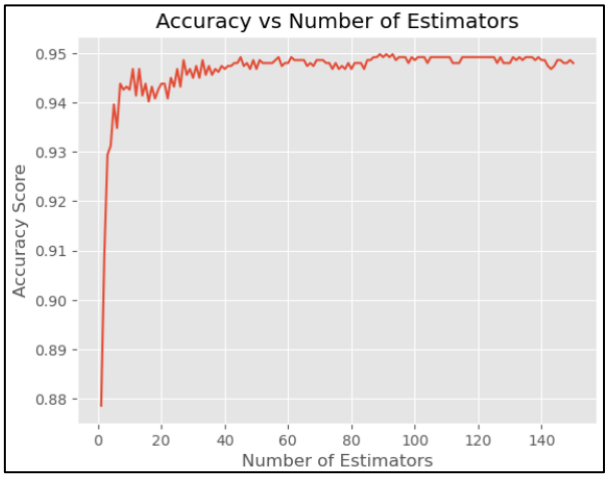
```
Accuracy: 0.9736842105263158

Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99      1448
           1       1.00      0.80      0.89       224

    accuracy                           0.97      1672
   macro avg       0.99      0.90      0.94      1672
weighted avg       0.97      0.97      0.97      1672
```

(Table 1: Result of Random Forest with TF-IDF)

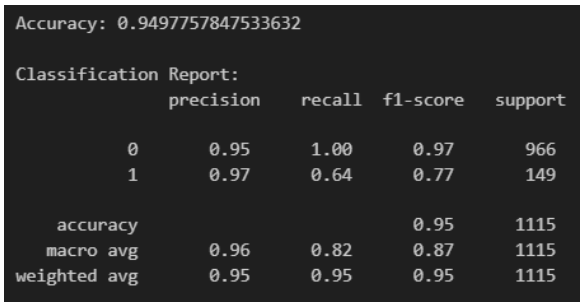


(Diagram 2: Confusion Matrix of Random Forest with TF-IDF)

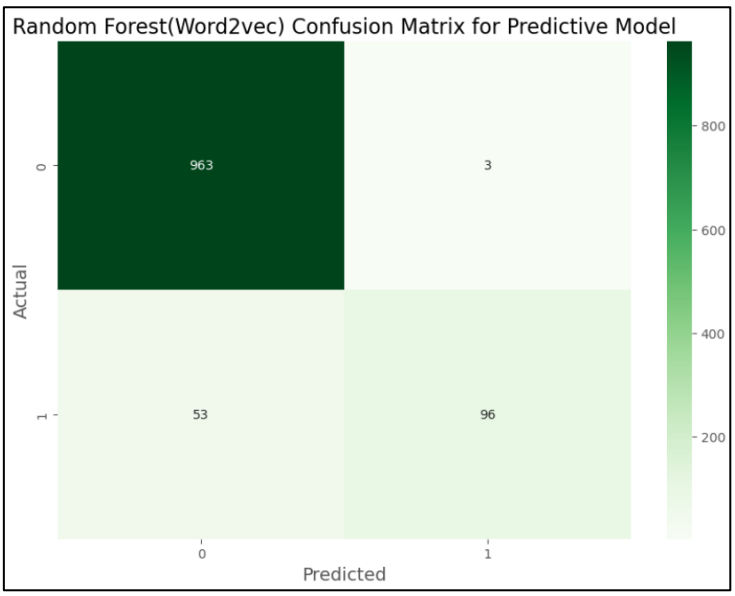
The model achieved a notable accuracy of 97.37% with n_estimators as 37 in 4.0 seconds, demonstrating strong predictive capabilities. Alternatively, the result of the model with Word2vec is shown as follows:



(Diagram 3: Result of Grid Search of RF with Word2vec)



(Table 2: Result of Random Forest with Word2vec)



(Diagram 4: Confusion Matrix of Random Forest with Word2vec)

With an accuracy of 94.98% in 2.9 seconds of training time with n_estimators as 47, the result is just slightly lower than TF-IDF, showing good performances the classifier can have with these two kinds of feature extraction, especially when both the recall rate and f1 score are also coming with a high rate.

Moving on to the deep learning model, Convolutional Neural Networks (CNN) were chosen. For the architecture, both the model with TF-IDF and the model with Word2vec consist of a convolutional layer with 128 filters and a kernel

size of 5, using ReLU activation, and an initial input shape parameter to fit the dimensions, designing to capture the most informative features. A global max pooling layer is added to reduce the dimension of the data, ensuring only significant features are passed through. Two dense layers, with the final layer using a sigmoid activation function to output the probability of being SPAM or HAM. A batch normalization layer and a dropout layer are also added to the model with Word2vec to fight imbalanced problems within the data. Both models were compiled with the Adam optimizer and used binary cross-entropy as loss function, ideal for binary classification problems.

One key thing to mention is to fight the imbalanced problem within the data, both datasets were resampled using SMOTE for oversampling the minority class, SPAM, and Tomek Links for cleaning some samples that may be misunderstood, aiming to provide a more balanced and improved model.

With all models conducted, the results can be listed as follows:

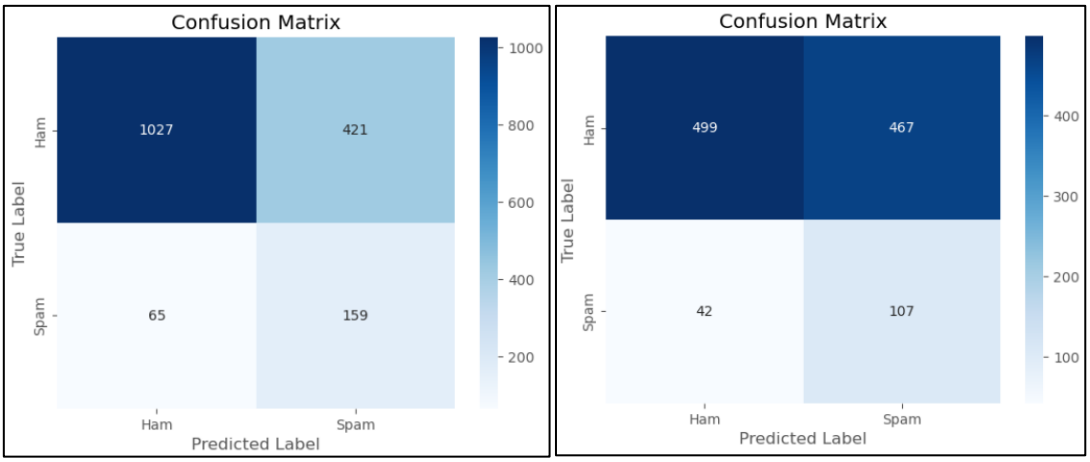
Model	Accuracy	Training Time
RF (TF-IDF)	97.37%	4.0s
RF (Word2vec)	94.98%	2.9s
CNN (TF-IDF)	70.93%	56.1s
CNN (Word2vec)	54.35%	13.4s

(Table 3: Results of Different Models)

The performance of the models varied significantly. The Random Forest models come with a better performance with high accuracy and short training time while also having a high f1 score, showing the power of this model in dealing with imbalanced problems. On the opposite, the CNN model, while may have more flexibility to be adjusted and be better, but the flexibility also presents a strong difficulty in fighting imbalanced data. Methods of forced balancing the weights, using SMOTE to generate more samples from SPAM, and using Tomek Links to minimize the noise are used, which do successfully

solve the imbalanced problem that leads to a guess-HAM-no-matter-what condition since the f1 scores are actually higher, just with the sacrifice of accuracy and training time.

The confusion matrix of CNN can be seen as follows, showing the result of fighting for imbalanced data:



(Diagram 5 & 6: Confusion Matrix of CNN with TF-IDF and Word2vec)

Next, to generate new “SPAM-like” emails, a text generation model using Long Shor-Term Memory (LSTM) is developed. The dataset consists of data directly from the original messages labeled “SPAM” from the SMS collection. With preprocesses like removing punctuation coming first, the data then is tokenized, converting each unique word into a numerical index, which is used to create sequences of a fixed length, serves as the input to the LSTM model.

By adding an embedding layer, two LSTM layers, one dense layer with ReLU activation, and a final dense output layer with SoftMax activation, the model is built, which is compiled with Adam optimizer and categorical cross-entropy loss function, for them being suitable for multi-class classification tasks like predicting the next word in a sequence. The model and dataset are then being used in our text generation function to predict the next word in a sequence

given one of the seed texts we have. The function works with the following progress:

1. Tokenizes the input text to convert it into a sequence of integers.
2. Pads the sequence to ensure it matches the input length expected by the model.
3. Predicts the next word using the model, selecting the word with the highest probability.
4. Repeats the process to generate a chosen number of words to build a new spam text.

This method is used to create 100 “SPAM-like” emails by repeatedly invoking it with different seed texts.

Lastly, the generated emails can be used to test our models, and the results are shown as follows:

Model	Precision	Recall	F1-score	Accuracy
RF (TF-IDF)	1.0	1.0	1.0	1.0
RF (Word2vec)	1.0	0.8	0.89	0.8
CNN (TF-IDF)	1.0	0.7	0.82	0.7
CNN (Word2vec)	1.0	0.7	0.82	0.7

(Table 4: Results of Different Models Testing 100 Generated Texts)

The Random Forest model exhibits perfect precision when using TF-IDF, which may have a problem like overfitting that can be further tested; the outcome of CNN model, on the other hand, performs better than predicting the original data. By successfully solving the imbalanced problem, the recall rate and f1-score are already at a decent level when predicting the original SMS collection, and the result here shows that the work blossoms beautifully.

The accuracy of CNN models, no matter in the original prediction or new spam text prediction, are always lower than Random Forest model, highlighting the challenges deep learning models face without expert-level tuning. The disparity in metrics illustrates the trade-off between precision and recall and the necessity to balance these when optimizing models for imbalanced data detection.

References:

Hakim, A. A., Erwin, A., Eng, K. I., Galinium, M., & Muliady, W. (2014, October). Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach. In *2014 6th international conference on information technology and electrical engineering (ICITEE)* (pp. 1-4). IEEE.

Ma, L., & Zhang, Y. (2015, October). Using Word2Vec to process big text data. In *2015 IEEE International Conference on Big Data (Big Data)* (pp. 2895-2897). IEEE.