Detection of Message Authenticity: A Modern Approach

Gavin Dsa
Data Analytics Engineering
Northeastern University
dsa.g@northeastern.edu

Abhishek Taware
Data Analytics Engineering
Northeastern University
taware.ab@northeastern.edu

Abstract—*With the rapid increase of spam messages being sent every day, there is a need for our devices to classify whether a message that is received is a spam message or not. To avoid these dangerous scenarios and to be safe from theft, correctly classifying these messages becomes even more important. In this paper, we have compared various machine-learning models to see which algorithm does a good job of classifying spam from ham. We have conducted our analysis under different scenarios where in one case we have compared stemmed from lemmatization and in another case, we have compared sampled and unsampled data. We have used accuracy to evaluate and compare our models as it tells us how well the model performed while detecting a spam message.*

I. Introduction

As SMS spammers are getting smarter about the ways they create spam messages, it is essential for our classification algorithms to work with higher accuracy, since it has become difficult for people to self-identify frauds. The aim of our exploration is to challenge Support Vector Machines that are said to be highly accurate at detecting spam messages. To carry out our analysis we have compared Bidirectional Encoder Representations from Transformers (BERT), Support Vector Machines (SVM), Sequential Neural Networks, and Long Short-Term Memory (LSTM) models which are well-known language models that are highly efficient in classifying text. Our approach was to compare each model under four different scenarios where the sampling method and root word generation were changed. We pursued this approach to see if changing the methodology for processing corpus has an effect on these models or not. The accuracy of each model under these four scenarios was compared since we are interested to see how well the model has been able to classify the spam messages from the ham messages. The corpus that was used for this is taken from the UCI Repository which is under the name of 'SMS Spam collection Data Set' and consists of 5574 instances where 13.4% of messages were labeled as spam and the other 86.6% of messages were labeled as ham. Since there was an imbalance in the target variable of our corpus, we decided to under sample our ham messages, to see if the imbalance had an effect on our model performances. Moreover, we have compared both stemming and lemmatization for the pre-processing of our corpus since root word generation can also have an effect on how the language models behave.

II. Background

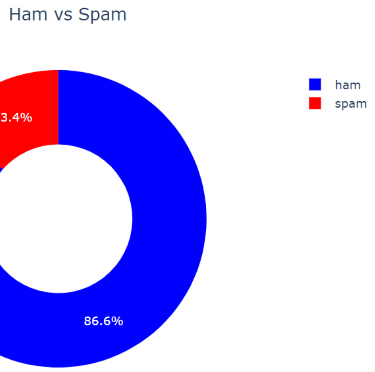
The paper titled "Contributions to the study of SMS spam filtering: new collection and results" which was published in 2011, focuses on the aspect of Support Vector Machines which acts as the best classifier outperforming the rest. The classifiers that were compared in this paper were Support Vector Machine and various Naive Bayes classifiers such as basic, multivariate gaussian, multinomial Boolean, Boolean, boosted, and flexible Bayes. Our goal was to test newer models that are more robust such as BERT, LSTM, and Sequential Neural Networks, to challenge the Support Vector Machine under different scenarios.

III. Approach

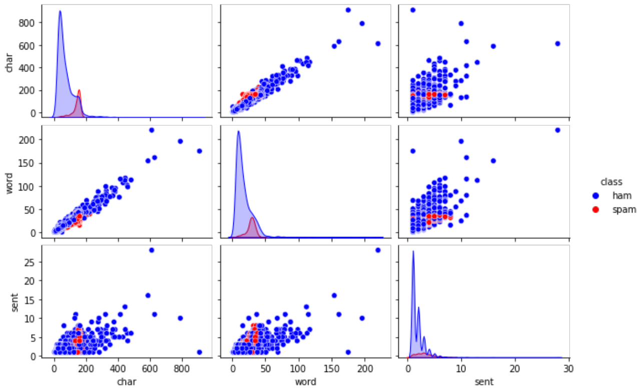
As per the paper that was published in 2011, used techniques such as SVM, Naive Bayes, etc., we felt that a lot of new NLP techniques have been in use since then such as BERT, LSTM, Neural Networks, TFIDF, etc. have been in use and is preferred over a lot of the older techniques.

A. Exporatory Data Analysis

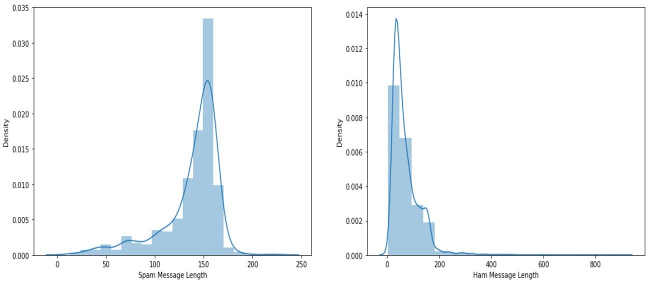
1) ***Figure 1: Ratio of Ham vs Spam***



2) *Figure 2: Distribution of number of characters, words, sentences*



3)Figure 3: *Length of Spam & Ham Messages*



B. Data Preprocessing and Cleaning

To check for the distribution of our spam and ham messages we counted and found that 86.6% of messages were ham and 13.4% of messages were spam as shown in Figure 1 of Exploratory Data Visualizations. From this analysis, we decided to use sampling techniques and compare our models to see if sampling has an effect on model performance. We then dropped the duplicates that were present in our corpus. Along with this, we also visualized the 4 different parameters: The number of characters in a message, the Number of words in a message, the Number of sentences in a message, and the length of the message as shown in Figures 2, 3. This showed us clearly that messages that were spam had a larger text length as compared to spam which was one of the conclusions that we drew from this analysis.

On further exploratory analysis, we found that the most common words that were used in spam messages were 'call', 'free', 'txt', etc. The corpus was then made free of stop words, punctuations, URLs, numbers, etc., and then created into four categories: Sampled Lemmatized, Unsampled Lemmatized, Sampled & Stemmed, and Unsampled & Stemmed. We have tokenized the text streams into tokens as well. The classes which were ham and spam were target encoded in order to convert the text into '0' and '1' for the models to comprehend. We also used word embedding techniques such as CountVectorizer and TF-IDF Vectorizer which essentially helps us convert our corpus to vectors for our models to interpret.

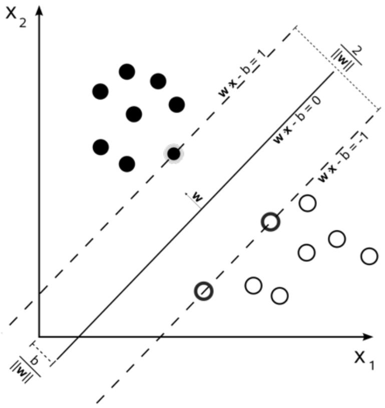
C. Model Implementation

For all our models we have divided our corpus into testing and training sets where training consists of 75% of the corpus and testing consists of 25% of the corpus. The models that we have used are Bidirectional Encoder Representations from Transformers (BERT), Support Vector Machines (SVM), Sequential Neural Networks, and Long Short-Term Memory (LSTM).

1) Support Vector Machine:

Support Vector Machine (SVM) was the classifier that gave the best performance as per the research paper published in 2011. Our goal was to take SVM as a Benchmark and compare the rest of the algorithms to it. SVM chooses a hyperplane such that the distance from the nearest data points on each side is maximized. To use this model, we first used the TF-IDF vectorizer to convert our corpus into numbers for our model to ingest. We also used Grid search for hyperparameter tuning to find the best parameters for our model. The parameters that were considered were kernel, gamma, and C values.

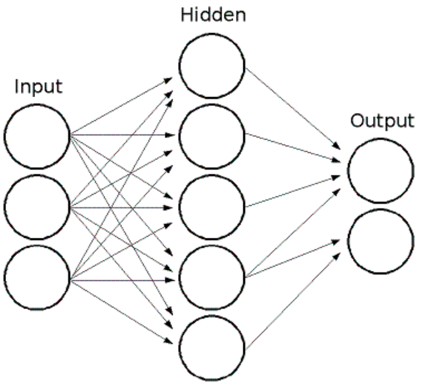
4) Figure 4: *Support Vector Machine*



2) Sequential Neural Networks:

We used this model as our corpus consists of sequential data which gives us more meaning as the sentence is formed. We have used 'ReLu' and 'Sigmoid' as our activation functions with binary cross-entropy as a loss function since our output is binary and accuracy as our metric to judge the model performance. We ran this model for 10 epochs and set the batch size as 10.

5) Figure 5: *Neural Network*



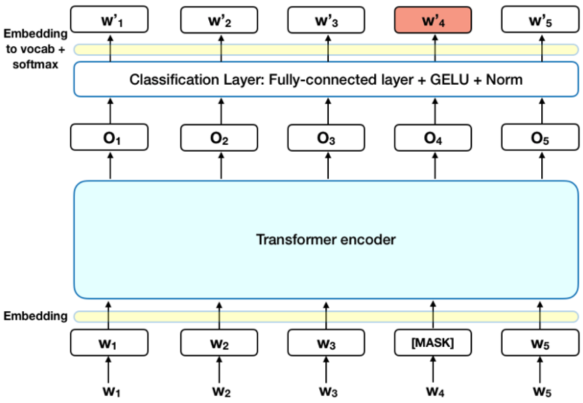
3) Bidirectional Encoder Representations from Transformers:

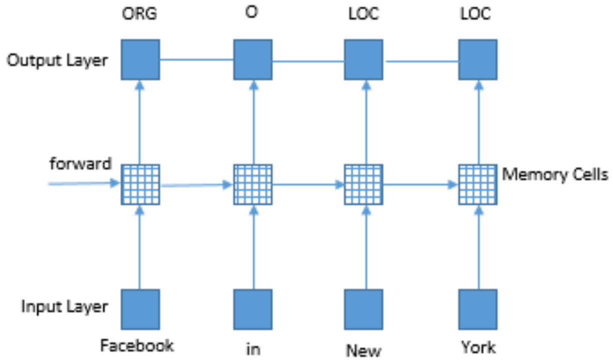
BERT is a language model developed and published by Google in 2018. It is a transformer-based machine learning technique that is used in Natural Language Processing. Since it is a technique that uses Transformers, it has a variable number of encoder layers and self-attention heads. It is now considered as State of the art in language model as it has outperformed all the other algorithms when it comes to NLP benchmarks such as GLUE, SQuAD, Multilingual sentiment analysis, etc.

4) Long short-term Memory:

LSTM is a recurrent neural network (RNN) that is capable of processing multiple sequences of data. Even though it's slightly slow to train on, when we process text data, it can simultaneously process the words as it uses a transformer architecture just like BERT. One major benefit of LSTM is that when it reads a sentence, the context of the words is better understood by the model as it learns them from both directions simultaneously.

6) Figure 6: Long Short-Term Memory


w'₁
w'₂
w'₃
w'₄
w'₅
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O₁
O₂
O₃
O₄
O₅
Transformer encoder
Embedding
W₁
W₂
W₃
[MASK]
W₅
w₁
w₂
w₃
w₄
w₅



IV. Results

As we can see, the table mentioned below contains the accuracy data for the test sets when we executed the 4 algorithms in the 4 different corpora formats.

	Unsampled & Lemmatized	Sampled & Lemmatized	Unsampled & Stemmed	Sampled & Stemmed
SVM	98.22%	92.35%	98.06%	92.33%
BERT	98.84%	96.02%	98.53%	96.94%
Neural Networks	98.07%	95.15%	98.26%	96.43%
LSTM	98.74%	93.51%	98.07%	95.8%

A. Conclusion

As we can see, over the 16 model runs that were done, almost all of the algorithms and the custom corpora have outperformed the cases for SVM. Also, one interesting thing to note over here is the fact that when the corpora were sampled, SVM was taking a major hit in accuracy as compared to any other algorithm. To boost the findings further, BERT which was made public in 2018 by a team of Google's developers has the highest accuracy among all 4 test cases when compared to the other three algorithms. Since Google uses BERT in its search queries as well as Android Operating System to deliver the most accurate results to the users, it is a well-researched model.

B. Future Directions

As discoveries keep on happening, the models mentioned above along with their tuning parameters can be used to test different sets of data to verify the consistency. If this proves to be true, these can be used as spam detection algorithms in devices to remove spam messages as and when detected so that the user isn't bothered by them.

REFERENCES

[1] https://www.researchgate.net/publication/258050187_Contributions_to_the_Study_of_SMS_Spam_Filtering_New_Collection_and_Results_preprint

[2] Neural Networks Explained — Deep Learning 101 | by Vivek Phuloria | Towards Data Science

[3] LSTMs Explained: A Complete, Technically Accurate, Conceptual Guide with Keras | by Ryan T. J. J. | Analytics Vidhya | Medium

[4] BERT Explained: What it is and how does it work? | Towards Data Science