SMS Spam Detection Techniques

1st Gourav.Aravinda
PES2UG19CS130
PES University
Bangalore, India
gourav.aravinda@gmail.com

2nd Dev Darshan J
PES2UG19CS108
PES University
Bangalore, India
devdj1691@gmail.com

3rd Achyuta Mudhol
PES2UG19CS014
PES University
Bangalore, India
achyuta1210@gamil.com

I. Abstract

This paper will focus on techniques used to detect between spam and real messages. This was done as part of our Natural Language Processing course that was taken as an elective during the sixth semester of our engineering course. We do a comprehensive study on what are the possible techniques that can be utilized to perform spam detection in short messages.The final model in this project is the implemented using modern Recurrent Neural Networks that gives it the edge over other methods.

II. Introduction

This project was done to ensure that any important message does not get lost in the sea of spam messages. The current scenario in the world leads to people getting at least hundreds of messages on a daily basis. As part of many companies marketing schemes, they send messages at random to a plethora of people to market their product and increase their product's profits. This is not wrong by any means but it can inadvertently cause people to miss out on responding to messages that are actually important.

Human nature tells us that after seeing something that we are not interested in, we tend to skip out on events that occur after it. This is what happens when there are many spam messages and there is an important message in the midst. It is difficult to look through all the messages that we have received when we are actually searching for a message that is important to us. This not only puts our mind of the task that we are currently doing but also leads to us losing important time.

Therefore, there is a need to ensure that spam messages are filtered out from real messages as we receive them with a high accuracy to ensure that people do not have to check them through manually. It is important that the model being built should be able to classify spam messages as spam but not even a single message that is real should be classified as spam. This would only compound the issues faced by people of looking through their messages as this message would be in spam.

Before building the model that was capable of performing such a task, a study had to be done on what were the existing methods that were being employed to tackle these problems and what were their weaknesses and strengths which could be very useful towards our study. There were many possible methods that are currently being used to filter out spam messages from real ones

III. Survey of Existing Methods

A detailed study of existing methods was performed for us to find out what could be possible methods that we could optimize and implement as part of our project

A. Support Vector Machine

The support vector machine algorithm has been a long standing algorithm that stood the test of time and has managed to keep itself relevant with the help of many techniques and the best of them being kernel tricks. An implementation of the technique was performed as we used Count-Vectorizers as the vectors that are used to get hyper-planes. However, we noticed that due to the message being too small, using any vectors that are of very high dimensions lead to overfitting in out model. Support Vectors are already known for their notorious ability to over-fit to problems and in this case, it was very evident as we tried to use features that were more relevant and be of higher dimensions. Hence, we could determine that even though we were able to achieve a high accuracy for this type of model, overfitting is a cause of concern.

B. Term Frequency-Inverse Document Frequency with Multinomial Naive Bayes

This is a common natural language processing technique that has been utilised to perform tasks that relate to classifying text to labels. This method depends on utilising the counts that the words appear in each of the following labels and the documents that they appear in to predict the best possible label for a given message. They are utilised with the assumption that the number of times a word appears under a label holds the influence of whether a message is a spam message or not. This is used in conjunction with Naive Bayes, which is based upon the Bayes Theorem. The probability of each word appearing as part of a spam or ham message is calculated and those are multiplied to give a probability of a message being part of a spam or ham messages. Given these probabilities the highest probability is chosen as the tag for the message. However, we found a critical flaw in this method. Even though it has proven to be effective, it does not consider the placement of the words in the sentence. The algorithm assigns each word independently and assumes that the co-occurrence of these words are independent of each other.

C. Decision Tree Classifier

This is rule generating algorithm that generates probable rules to classify the message on the basis of certain words being present. It will generate Boolean conditions that are checked to lead to a certain outcome after going to the optimal depth. This has also been implemented using random forest methodology that utilises common ensemble learning and uses multiple small decision stumps to take a decision. Even though this method has been proven to be effective while handling the classifications, they still suffer from the fact that the location of the words are not taken into accord for a classification to be made.

D. Ensemble learning

A very popular machine learning technique that overcomes the failures of many methods and finds the best of all the algorithms by taking the weighted average of all the methods to give a final label to given input. This method is the best method that is available to classify these messages into their categories. However, a fallacy of this method is that it will take very long to train all the models and find a way to make them all accurate enough to be combined and give us a vector space where all the possible answers are given a label. It also takes a longer time to give us a prediction for a given message. Given the influx of messages being received at the moment it is not possible for us to take too long to train or predict using our model. Hence we have to look at better alternatives.

IV. Motivation

This model is being built to ensure that messages can be filtered in an effective manner to ensure that real messages are not lost among the sea of spam messages. It also makes sure that this task is done at a quick pace and does not take too long which will lead to messages not being delivered at real time. Therefore, it was decided to go with Glove Vector Embedding and Recurrent Neural Networks that have the capability to not only perform this task reliably but also quickly present us with a label for a given input.

V. Proposed Methodology

The model being presented has been made using Glove Embedding and Recurrent Neural Networks. It has been further enhanced for performance with the help of LSTM and GRU cells. The model will be broken down and explained with the need of each component and the purpose they serve in ensuring that the model has been trained to its best ability.

A. Glove Embedding

GloVe stands for Global Vectors and is used for word representation. It is an unsupervised machine learning technique that has been developed after intensive research at Stanford. It has been made with the general intuition that the ways that word affect each other can be derived from the word-word co-occurrence matrix. It makes use of both the count based and prediction based methods and thus has the advantages of both methods. This allows for embeddings for words to be created using their occurrence with other words in the corpus. This vectorization of the words acts as our input layer to the model.

They work on the intuition that words that appear near each other are related to each other one way or the other. It is a log-bi-linear regression model that has a weighted least squares objective function. The embeddings it creates is given meaning by the fact that ratio of word-word co-occurrence probabilities that are taken from the word-word matrix for a corpus will be able to give us meaningful vector representations. If the ratio is greater then 1, the numerator utilised has a closer meaning and if the ratio is lesser then 1, the denominator utilised has a closer meaning to the word in question. If the ratio is 1, we can treat these two words to be noise and not consider them while obtaining our embedding vectors. Glove learns word vectors that when dot product is performed, the logarithm of the word's co-occurrence is returned. These are then trained so that it is possible for all the words in an unsupervised manner.

Glove embeddings have proven to be very effective for small corpora and classification tasks. This model will be utilising a pre-trained Glove embedding that has been performed and placed on GitHub for use by Stanford. Glove has shown to be useful vector representations of words in the vector space then any other vectorization techniques such as Word2Vec and Count-Vectorizer. They also allow for us to effectively pass input to our neural network that represent the input as a distribution that links the co-occurrence of the words.

Hence, GloVe Vector Embeddings have been the first layer that are utilised in the models that we are created. Words are first converted into their GloVe embeddings to be passed as inputs to the neural network. These embeddings have been taken from the Stanford GitHub repository that is open to public usage and have been trained to give 100-d vectors for all the words that was present in their corpora and also extended to work when a new word that was not present in their corpora is encountered as well.

B. Recurrent Neural Networks

These are deep learning architectures that have been proven to be very effective to solve NLP classification problems such as the one we are trying to solve currently. Recurrent Neural Networks have become popular deep learning techniques because they allow for memory to be kept about our past input and can be extended to also consider the future input to come up with a label for a given message.

The key features of a Recurrent Neural Networks is the fact that the weight parameters are kept the same throughout each cell of the RNN. The model we are building makes use of thirty-two cells and one hidden layer in the RNN. The model utilises 32 cells as we assume that the optimal number of words in a message is 32 words. The model is however capable of handling more or less then 32 words and we chose 32 as it was found to be optimal. The Re-LU activation function is the activation function used in our RNN cells.

The RNN cells have also used the RMS Prop Optimizer that helps to converge faster and arrive at a classification label

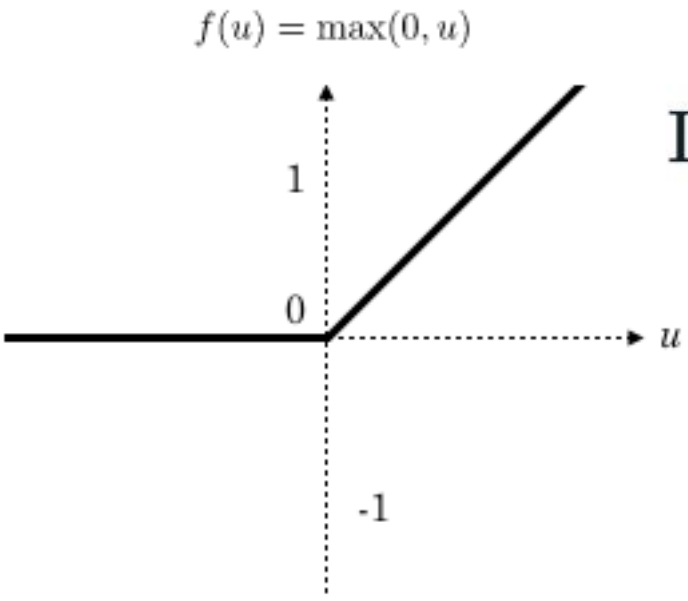

Fig. 1. Re-LU Activation Function.

quicker by controlling the learning rate of the model. It allows for larger learning rates when the gradient is higher and allows for faster training which in turn leads to faster prediction. The comparison between the gradient descent and RMSProp is shown below.

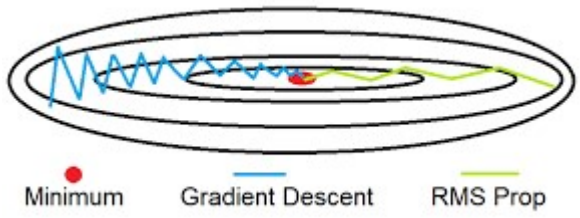

Fig. 2. Performance of RMSProp Optimizer

The model utilised binary cross entropy loss to train the model during back-propagation and tune the weights of the RNN. This loss function is perfect for the scenario at hand as the need is to find out what is the probability of us classifying the spam as spam and the probability of classifying the real message as real. This will help us calculate how well that has been done. The formula for the loss is given below.

The model has a final output layer that utilises sigmoid activation to achieve an output between 0 to 1 that acts as the probability of the message being real. If closer to 1 it means it is real message and closer to 0 means that the message is spam.

The model has been structured as explained above. The final architecture diagram is provided below.

$$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$$

Fig. 3. Binary Cross Entropy Loss

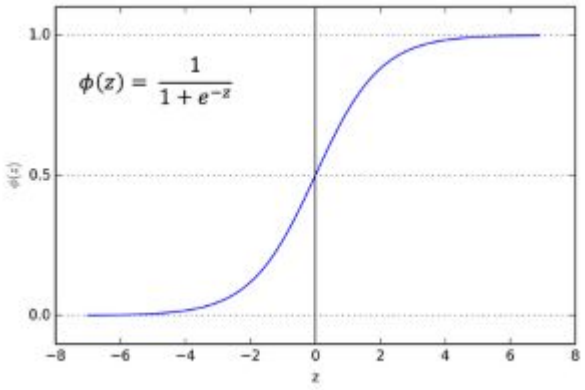

Fig. 4. Sigmoid Activation Function

C. Long Short Term Memory [LSTM]

The RNN model developed above has shown to be very successful. It could differentiate between spam and real messages very well. However, it was taking too long to train and give us a prediction that could affect the real-time delivery of the message that would compound the issue of spam messages. Therefore, the decision was taken to utilise Long Short Term Memory Cells instead of basic RNN cells. The idea behind LSTM was to move towards the concept of gates and allow for saving cell state using another value. There would be three inputs being the hidden state of the previous, previous cell state and the current input and three outputs being the label, hidden state and cell state. The architecture comprises of three gates known as the input, output and forget gate. The input gate is responsible for choosing important input features. The

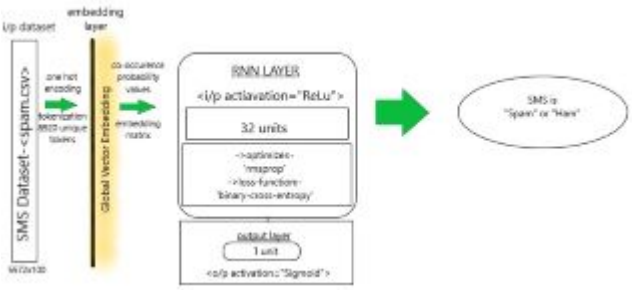

Fig. 5. Final Architecture

forget gate is responsible for forgetting unnecessary details of the past. The output gate is responsible for using the output of both input and forget gate to get the new hidden state and output.

The LSTM makes use of both the ReLU and Tanh activation functions for different purposes. The ReLu activation is used to retain important information and the tanh activation is used to squash the input to between 0 to 1.

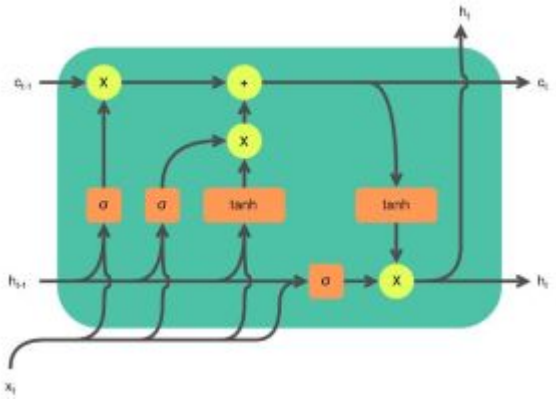

Fig. 6. LSTM Architecture

Using the LSTM Architecture did reduce the time needed for training and prediction but it had taken up too much storage. It would require too many parameters to keep track off which was an overhead that could be a hassle in the future. Therefore, the model moved to utilising GRU units.

D. Gated Recurrent Units[GRU]

This architecture still utilises the same gates idea-logy but instead of maintaining a cell state and using 3 gates, it moves towards using only 2 gates and not maintaining a separate cell state and have only a hidden state to do both. By reducing the number of gates by combining the forget gate and input gate into a single update gate. The other gate is called reset gate. The responsibility of the gates do not change. However, the way the hidden state is maintained and updated is changed. Since the cell state has been removed and the hidden state now has to store details of the past inputs on its own, they gates are combined in a different manner as shown in the architecture below to ensure that necessary details are retained. This responsibility is taken up by the update gate. The reset gate determines what information from the past should be neglected. The output of both gates is combined to get the hidden state of the current cell and the output of the cell. These causes a reduction in parameters as we only need to tune the weights of two gates

This helped with reduction in parameters with only a minimal loss in accuracy and time taken to train and predict. Both the LSTM and GRU were still trained with 32 cells in 1 hidden layer and the output was determined with the help of the final output layer. Both gave better accuracy and reduced training and predicting times.

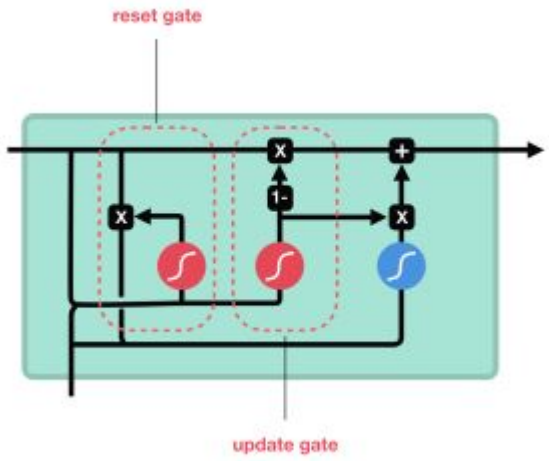

Fig. 7. GRU Architecture

VI. Experimental Results

The comparison of all three accuracy's and training time of the three models with their summaries are shown below.

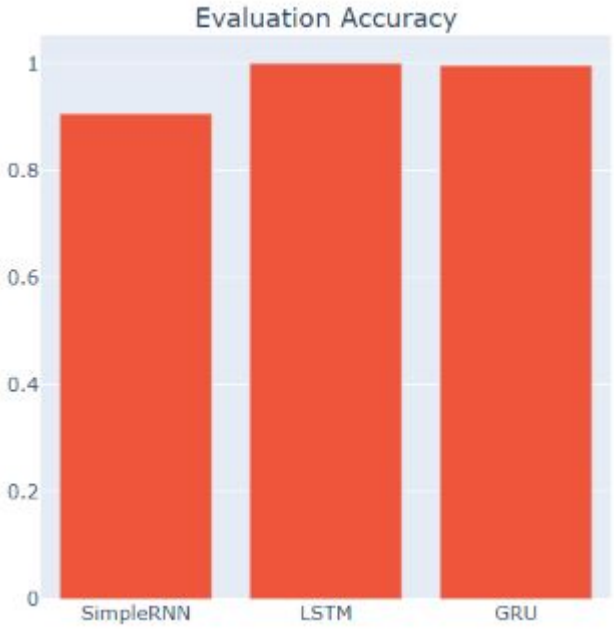

Fig. 8. Accuracy

The comparisons above show us that the accuracy of the LSTM is the highest and also takes the least time to train. However, it occupies more space in memory and when a trade-off can be performed for memory and performance, then GRU can be selected as we can see that it performs reasonably well and very close to the performance of LSTM.

Fig. 9. Loss

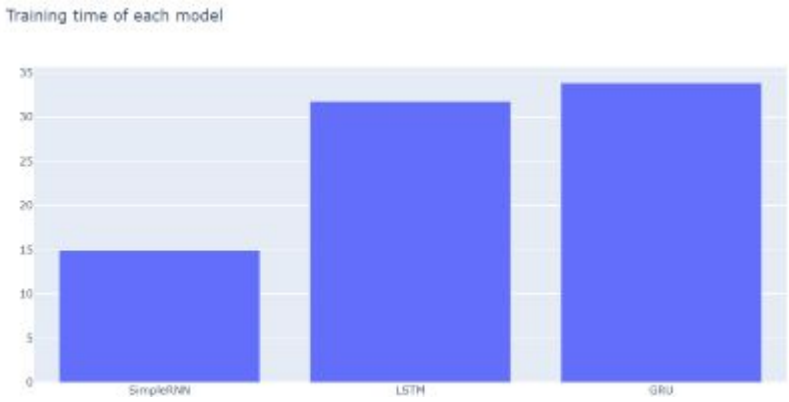

Fig. 10. Time taken from training

VII. Possible Extension

The model can be extended to be able to perform incremental learning which is an upcoming architecture with the rise of Big Data technologies that can perform incremental learning. The model will keep learning from the messages that users ignore or respond to and can then be personalised to each user and ensure that it performs well for each different user and not only for a particular group of users. This would allow for higher perfromance and usage of data more effectively. The arhcitecuture used here can be reused for that purpose as well.

References

[1] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. [pdf] [bib]

[2] J. Fattahi and M. Mejri, "SpaML: a Bimodal Ensemble Learning Spam Detector based on NLP Techniques," 2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP), 2021, pp. 107-112, doi: 10.1109/CSP51677.2021.9357595.

[3] Ora, Anchal (2020) Spam Detection in Short Message Service Using Natural Language Processing and Machine Learning Techniques. Masters thesis, Dublin, National College of Ireland.

[4] Spam Message Detection Using Logistic Regression NIKHIL KUDUPUDI1 , SHILPA NAIR2 U.G. Student, School of Engineering, Ajeenkya DY Patil University Pune, India -4121051

[5] An Efficient Spam SMS Analysis Model based on Multinomial Naïve Bayes model Using Passive Aggressive AlgorithmJ. Shobana and D. Kanchana 2021 J. Phys.: Conf. Ser. 2007 012047

[6] Spam Detection using NLP Techniques BollamPragna, M.RamaBai(NGCT). 4. Gupta, M., Bakliwal, A., Agarwal, S., Mehndiratta, P. (2018). "A Comparative Study of Spam SMS Detection Using Machine Learning Classifiers." 2018 Eleventh International Conference on Contemporary Computing (IC3)

[7] Spam Detection using Natural Language Processing 1Rohit Giyanani, 2Mukti Desai 1 Thadomal Shahani Engineering College P. G. Kher Marg, (32nd Road), TPS-III off Linking Road. Bandra (West), Mumbai - 400050. 2D. J. Sanghvi College of Engineering Plot No.U-15, J.V.P.D. Scheme, Bhaktivedanta Swami Marg, Vile Parle (West), Mumbai-400 056