Advances in Parallel Computing Algorithms, Tools and Paradigms
D.J. Hemanth et al. (Eds.)

doi:10.3233/APC220058

Classifications of E-MAIL SPAM Using Deep Learning Approaches

Anshumaanmishra[a,1] and VigneshwaranPandi[a]
[a]*Department of Networking and Communications,*
SRM Institute of Science and Technology, Kattankulathur, Chennai, India

Abstract: Spamming is an art used to deceive people and, in most cases, send unwanted messages that can be used by cybercriminals to trick victims and get confidential credentials from the victim. Spamming occurs via email, SMS, social networking websites calling the victim's phone number, etc. Spamming can exist for various reasons, but it can be used for malicious purposes mainly like trying to forge the victim for gathering the personal information, bank details card details, passwords, and other confidential data about the intended user. To overcome the security breach, the spam messages are classified for the understanding of spam has been done using different methods like Machine Learning and Natural Language Processing. We propose a method to identify the email spam messages and also to classify the message using deep learning approaches such as Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM), and Bilateral Long Short-Term Memory (BLSTM). We have used an email spam dataset with over 5000 spam text samples are used to train our deep learning models. The proposed method separated the sentences from the email into words followed by their root words, and each word is assigned an index number before training. We have used regularization and dropout layers in our models to reduce the chances of overfitting. The proposed method is evaluated and compared with other existing models based on the history of the loss curve, Precision-Recall, and accuracy. Based on the results, it is observed that our Bi-LSTM model produced higher accuracy than other existing systems.

Keywords: Spam Detection, Bi-LSTM, LSTM, RNN

1. Introduction

Spam is commonly defined as an unwanted message sent to the receiver by a group or a person. Spamming has existed for many years now and is currently mostly used for promotional activities such as advertising, marketing, etc., and collecting confidential information from legitimate users to perform various cybersecurity attacks. Spamming occurs via email, SMS, social networking websites calling the victims over the phone, etc. Examples of spamming include sending multiple messages regarding their products to pressurize the victim into buying their product and clicking the ads to gather information about the users. Spamming can exist for various reasons, but it is mainly used for malicious purposes like trying to deceive the victim by gathering personal information such as bank details card details, passwords, and other confidential data about the intended user in recent days. Once the data is obtained, the victim's data is

[1] AnshumaanMishraam, SRM Institute of Science and Technology, Kattankulathur, Chennai, India; E-mail: 2747@srmist.edu.in.

compromised and leads to further harm inflicted by the attacker on the victim like selling the victim's confidential credentials on the dark web [1]. Spamming is present for generating fake reviews [2] on e-commerce applications also to obtain significant financial profit for organizations and potentially hurt their competitors. There are many different types of spam, E-mail Spam - If the sender sends unwanted or impulsive e-mails directly or with implications to the consumer and the customer seems to have no connection with that E-mail. E-mail spam is included in the electronic spam classification. Phishing emails [3][4] are an illustration of such spam, Webspam — Webspam (also known as search spam) refers to the work of a web crawler to obtain information present online[5]. SMS Spam - SMS spam is called when someone sends unwanted and spontaneous messages on correspondence media (i.e., PDA). It is in the electronic spam category. In recent years, the artificial intelligence strategy is utilized to detect spam messages. Our proposed method is based on the Deep Learning approaches to detect spam messages effectively.

2. Related Works

Sairamesh et al. [6] have examined credit card in online shopping using machine learning algorithms. Carlos et al. [7] utilize a technique that includes extracting link-based and content-based features from private and public datasets and utilizing the Web Topology by exploiting link dependencies between web pages, one limitation is that their accuracy is obtained without regularization methods applied to classifiers. Minoru et al. [8] proposed another spam recognition method that includes the utilization of a spherical k-means clustering algorithm, which distinguishes spam mail by using centroid vectors of the groups for extracting features of that cluster. They give a label to every centroid and a calculation to check similarity is done among a new mail cluster and the marked centroid. They can demonstrate their technique to be powerful, However, they don't consider a dynamic update of the spam and non-spam clusters. Qian et al. [9] made use of content-less features. They used network and temporal features from SMS messages and joined them to train an SVM algorithm. They additionally looked at the performance among SVM and KNN-based algorithm(s). Since they have utilized non-content provisions, they can't compare their current spam discovery techniques as these strategies utilize content-based features to decide spam messages. Faraz et al. [10] present a non-exclusive measurable which distinguishes spam profiles via online media. They found 14 features that are relevant to the ID of spam profiles from which they choose 7 features. They do acquire an exceptionally high identification rate and a low false-positive rate, however, the dataset they used to prepare their model just comprises spam profiles from Twitter and Facebook. Many spam profiles are existing on various online web-based media, spam profiles from Facebook and Twitter don't give complete insights into spam profiles.

3. Methodology

In our work, we have trained neural networks such as Artificial Neural network (ANN), Recurrent neural network (RNN), Long short-term memory (LSTM), and bilateral LSTM to predict spam data from a dataset containing spam and legitimate messages. By applying complex neural networks, we aim to achieve a very high accuracy rate in

the detection of spam messages. We have trained artificial and recurrent neural network algorithms, to predict the spam messages, using a dataset that contains legitimate, and spam messages combined. During the data cleaning process, the null values are removed from the dataset and the words which do not add much meaning to a sentence (also known as stop words) using the stop word [12] library of python. After that, we used the data stemming (words brought down to their base version) process to reduce the words to their root versions. We divide the long lines of sentences into words to reduce the complexity and an index for each word, which would provide vectorized text data. The words are provided with a fixed dimension of each sample or sequence that we have created, which would be useful to process the input in the sequential models used in this work. Our dataset contains 5171 samples of legitimate and spam samples out of which we have used 75% for training and 25% for testing purposes. Approximately 3878 samples for training the model and 1293 samples for testing.

3.1 Recurrent Neural Network (RNN)

Normal neural networks cannot reason about events that occurred in the past, because each layer of computing in this type of network is independent and does not influence each other. Therefore, they are "stateless" and cannot learn information from past sequences, which is their main disadvantage. A recurrent Neural Network (RNN) is a promising solution to the problem of sequential learning in traditional neural networks. The basic structure is like a feed-forward neural network, but they do not have a unidirectional connection between their neurons, they have directed cyclic connections. Our model for the RNN consists of two hidden layers consisting of 50 neurons each. After processing the text data from the samples each word is fed as input to the network, and each output is determined by the input, weight, and bias associated with it along with the activation value of the previous time step as shown by the equation (1).

$$a^i = g(x^i W_X + W_a a^{i-1} + b_x) \quad (1)$$

Where $a^{i-1}$is the activation passed by the previous time step W_x is the weight associated with input x^i and b_x is the bias for input x^i. Our vectorized input has been passed through this network and goes through the feedforward layer present in the network to the output neuron, which uses the sigmoid activation function as shown in equ. (2).

$$\theta(a) = \frac{1}{1- e^{-a}} \quad (2)$$

The final output is computed through the weight, bias, and activation value computed by the hidden layers, and the activation function is given in equation (3).

$$y_i = \sigma (W_y a^i + b_y) \quad (3)$$

Where y_i is the final output which is between one and zero, Wy is the weight associated with the activation function b_y, being the bias. Our output contains values between 0 and 1, and if the values are above 0.5 they would be considered spam, or else it would be legitimate.

3.2 Long Short-Term Memory (LSTM)

LSTM is a variant of RNN and can sustain memory for longer periods, unlike RNN. In RNN the backpropagation of error requires the network to calculate the gradient for the updating weights and causes the RNN, to remember long-term memories with a lot of difficulties, the evidence for this is the gradient becoming too small or large. LSTM consists of repeating LSTM units, which have special memory units that can store information for a long time without degradation. The memory unit (cell state) with three logic gates controls the output to the following memory unit: forget gate, the entry gate, and the exit gate. These gates determine which memory to keep and which to forget. Since our training data is vectorized text data with a huge vocabulary, we have used an embedding layer [11] after the input layer to be represented by dense vectors, where a vector represents the projection of the word into a continuous vector space. The vector space location of a word is learned from the text and is dependent on the words that surround the word when it is employed. After the embedding layer, the input X_t and the information from the previously hidden state $h_{t-1}$are fed into forgetting gate f_t of the LSTM unit which removes non-relevant data which is further passed to the cell state C_{t-1}, which is based on the equation (4).

$$\mathrm{f_t} = \sigma(\mathrm{W_f} \times \mathrm{h_{t-1}} + \mathrm{W_f} \times \mathrm{X_t} + \mathrm{b_f}) \tag{4}$$

The output of f_t multiplies with the previous cell state, the values between 0 and 1 and closer to zero are dropped. The $X_t$from the embedding layer and $h_{t-1}$is also passed through the input gate which uses the sigmoid and provides output i_t and the tanh activation function provides $\underline{C_t}$ as given in the equation (5) & (6).

$$\mathrm{i_t} = \sigma(\mathrm{W_i} \times \mathrm{h_{t-1}} + \mathrm{W_i} \times \mathrm{X_t} + \mathrm{b_i}) \tag{5}$$

$$\underline{\mathrm{C_t}} = \tanh(\mathrm{W_c} \times \mathrm{h_{t-1}} + \mathrm{W_c} \times \mathrm{X_t} + \mathrm{b_c}) \tag{6}$$

The output from both gates is multiplied, and the value is added with the cell state value using the equation (7).

$$C_t = f_t \times C_{t-1} + i_t \times \underline{C_t} \tag{7}$$

The output gate equation is given below which takes X_t and the h_{t-1} as input and previous cell state-input with the weight W_o and bias b_o

$$o_t = \sigma(W_o \times h_{t-1} + W_o \times X_t + b_o) \tag{8}$$

The final cell state will be passed to the other LSTM unit and to the activation function, whose output will be multiplied with X_t, and h_{t-1} is passed via the sigmoid function. This provides the hidden state output, which is passed to the next LSTM unit.

3.3 Bidirectional-LSTM

Bi-LSTM is approximately two RNNs put together which leads to information being shared in a bidirectional way. The output of the Bi-LSTM will be dependent on past

and future information, unlike LSTM which is only unidirectional. Bi-LSTM can store information from both the past and future meaning that forward propagation occurs twice. In the proposed method, the Bidirectional LSTM contains one embedding, hidden, and output layer. The arguments for input in the embedding layers are the same as the ones we used for LSTM. As compared to the LSTM, Bi-LSTM has three gates called forget, update, and output gates which have the same equations as the LSTM. Hidden state value $a^{<t-1>}$, cell state value $C^{<t-1>}$, and input $X^{<t>}$are used to compute the equation for the three gates the same way an LSTM does. The output of the Bi-LSTM $y^{<t>}$requires the value of the previous hidden state and current hidden state namely $a^{<t>}$and $\leftarrow a^{<t>}$ as forward propagation is applied twice here. The output is as follows.

$$Y^{<t>} = \sigma(W_y \times\leftarrow a^{<t>} + W_y \times a^{<t>} + b_y) \quad (9)$$

Where W_y andb_y are the weights and bias.

4. Results and Discussion

In this section, we have discussed the results obtained from our experiment and draw meaningful insights from them. We also compared the results of the models we used in our work based on the history of network learning, the Precision-Recall Curve with the area under the curve, and accuracy. Learning curves are plotted to understand the learning performance over time in terms of the learning experience. This curve contains the recorded training metrics for each epoch. This includes the loss for classification problems as well as the loss for the validation dataset. These records help us gain a better understanding of our model.

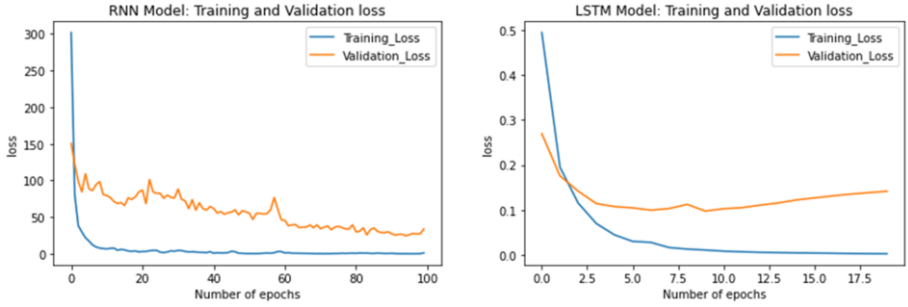

Figure. 1: History of Loss during training and validation in our RNN(left) and LSTM (right) models

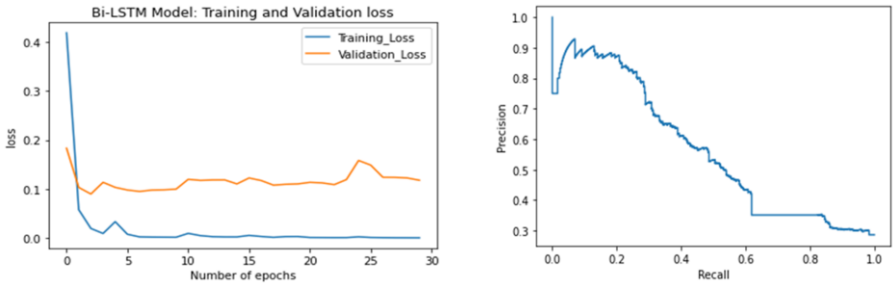

Figure 2: History of loss during training and validation in our LSTM model (left) and ROC curve for RNN (right)

From figures 1,2 & 3 we can see that the gap between the training and validation set is large which indicates that our models are overfitting [13] even after using regularization techniques such as L2 regularization [14] to avoid overfitting. In terms of learning our RNN model has shown a better output compared to LSTM and Bi-LSTM. The dataset we used to train our models is imbalanced and therefore we use a metric called Precision-Recall Curve that would help us provide insights on the performance of the classifiers taking into consideration the imbalance. To calculate the precision, and recall values we need

True Positive (TP): The prediction correctly identified as spam

True Negative (TN): The prediction was correctly identified as legitimate

False Positive (FN): The prediction was wrongly identified as spam

False Negative (FN): The prediction wrongly identified as legitimate

Precision-Recall Curve (PRC) graph is the comparison between the number of correct predictions made from the correct and false predictions for spam versus the number of correctly identified spam predictions out of wrongly identified legitimate and correctly identified spam predictions. The graph shows the performance of a classification model at all classification thresholds using precision and recall. Precision and Recall are calculated using these formulae

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

$$\text{Recall} = \frac{\text{TP}}{\text{FN} + \text{TP}} \tag{11}$$

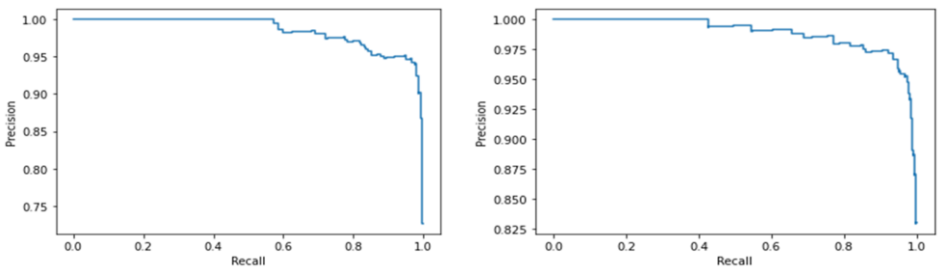

Figure 3: ROC curve for LSTM (left) and Bi-LSTM (right)

We have computed the PRC for all the models we have used in our work. We calculate the Area under the curve (AUC) to identify the model with the highest performance to easily identify the model performance by the Precision-Recall AUC summarizes the curve with a range of threshold values as a single score rather than creating random classification thresholds which can also be used to compare the model performance. AUC helps us by displaying the area under the curves portrayed in figures 2 and 3 The score can then be used as a point of comparison between different models on a binary classification problem where a score of 1.0 represents a model with perfect prediction. The lowest would be 0 indicating that the model is ineffective. The scores for AUC are given in Table I. From the table, we can infer that the Bi-LSTM model has shown the highest performance followed by LSTM and RNN being the least performing model. Accuracy is an indicator that generally describes the performance of the model in all categories. It is useful when all classes are equally important. It is calculated as the ratio of the number of correct predictions to the total number of predictions. Accuracy is calculated using the following formula

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP+TN+FP+FN}} \tag{12}$$

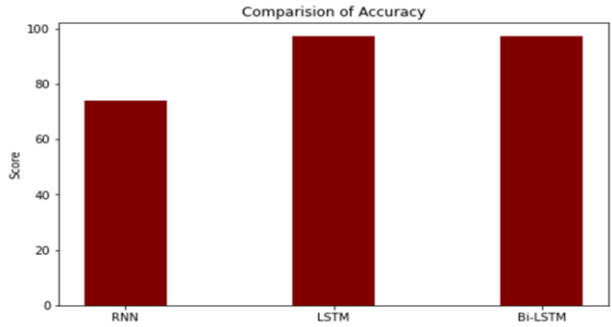

Figure 4: Accuracy of different models used

Table 1 Comparison of the proposed system with existing algorithms

Model	AUC score	Accuracy
RNN	0.569	73.24
LSTM	0.984	96.90
Bi-LSTM	0.988	97.60

From figure 4, the highest accuracy is achieved by Bi-LSTM followed by LSTM and RNN. After the comparison of our work with existing algorithms, we believed that Bi-LSTM effectively remembers more information, improving the content available to the algorithm which helps them perform a better evaluation of other models. This has made it the highest performing algorithm among other algorithms we have used in our work.

5. Conclusion

In our work, we have trained complex deep learning models to predict spam mails that arise in the inbox of everyone. We have compared our model's performance based on loss curves, ROC-AUC score, and Accuracy and it was found that Bi-LSTM outperformed other models. It is observed that Bi-LSTM had more capacity to hold information which helped it make better predictions than other models. We have trained our models using an email spam dataset which contains modern spam emails work regarding other ways of spamming like comments in the comment box, social media spamming etcetera is not considered presently in our work and in the future, these spam messages are also to be considered to devise and predict the better system for spam detection.

References

[1] Weimann, Gabriel. Going dark: Terrorism on the dark web. Studies in Conflict & Terrorism 39.3 (2016): 195-206.

[2] Ott, Myle, et al. Finding deceptive opinion spam by any stretch of the imagination. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

[3] Fette, Ian, Norman Sadeh, and Anthony Tomasic. Learning to detect phishing emails. Proceedings of the 16th international conference on World Wide Web. ACM, 2007.

[4] Li, Wenbin, Ning Zhong, and Chunnian Liu. Combining multiple email Filters based on multivariate statistical analysis. Foundations of Intelligent Systems. Springer Berlin Heidelberg, 2006. 729-738.

[5] Spirin, Nikita, and Jiawei Han. Survey on web spam detection: principles and algorithms. ACM SIGKDD Explorations Newsletter 13.2 (2012): 50-64.

[6] SaiRamesh, L., E. Ashok, S. Sabena, and A. Ayyasamy. "Credit Card Fraud Detectıon in Retaıl Shopping Using Reinforcement Learning." In International Conference On Computational Vision and Bio Inspired Computing, pp. 1541-1549. Springer, Cham, 2018.

[7] Castillo, Carlos, et al. Know your neighbors: Web spam detection using the web topology. Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. 2007

[8] Sasaki, Minoru, and Hiroyuki Shinnou. Spam detection using text clustering. 2005 International Conference on Cyberworlds (CW'05). IEEE, 2005.

[9] Xu, Qian, et al. Sms spam detection using noncontent features. IEEE Intelligent Systems 27.6 (2012): 44-51.

[10] Ahmed, Faraz, and Muhammad Abulaish. A generic statistical approach for spam detection in online social networks. Computer Communications 36.10-11 (2013): 1120-1129.

[11] Hochreiter, Sepp. Recurrent neural net learning and vanishing gradient. International Journal Of Uncertainity, Fuzziness and Knowledge-Based Systems 6.2 (1998): 107-116.

[12] The library can be found at :https://pypi.org/project/stop-words/

[13] Hawkins, Douglas M. The problem of overfitting. Journal of chemical information and computer sciences 44.1 (2004): 1-12.

[14] Cortes, Corinna, Mehryar Mohri, and Afshin Rostamizadeh. L2 regularization for learning kernels. arXiv preprint arXiv:1205.2653 (2012).