Journal of Computer Science

Original Research Paper

Email Spam Classification Using Gated Recurrent Unit and Long Short-Term Memory

Iqbal Basyar, Adiwijaya and Danang Triantoro Murdiansyah

School of Computing, Telkom University, Bandung, Indonesia

Article history
Received: 09-08-2019
Revised: 06-01-2020
Accepted: 02-04-2020

Corresponding Author:
Danang Triantoro Murdiansyah
School of Computing, Telkom University, Bandung Indonesia
Email: danangtri@telkomuniversity.ac.id

Abstract: High numbers of spam emails have led to an increase in email triage, causing losses amounting to USD 355 million per year. One way to reduce this loss is to classify spam email into categories including fraud or promotions made by unwanted parties. The initial development of spam email classification was based on simple methods such as word filters. Now, more complex methods have emerged such as sentence modeling using machine learning. Some of the most well-known methods for dealing with the problem of text classification are networks with Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU). This study focuses on the classification of spam emails, so both the LTSM and GRU methods were used. The results of this study show that, under the scenario without dropout, the LSTM and GRU obtained the same accuracy value of 0.990183, superior to XGBoost, the base model. Meanwhile, in the dropout scenario, LSTM outperformed GRU and XGboost with each obtaining an accuracy of 98.60%, 98.58% and 98.52%, respectively. The GRU recall score was better than that of LSTM and XGBoost in the scenario with dropouts, each obtaining values of 98.98%, 98.92% and 98.15% respectively. In the scenario without dropouts, LSTM was superior to GRU and XGBoost, with each obtaining values of 98.39%, 98.39% and 98.15% respectively.

Keywords: GRU, LSTM, Spam Classification

Introduction

Since first invented by Tomlinson (Partridge, 2008) in 1997, the email has grown immensely such that it pervades all facets of human life. The email phenomenon was greatly helped by the development of the Internet, which pushed individual communication to become centralized around emails (Tsugawa *et al*., 2010). A survey run by Radicati Group inc. (2013; 2014; 2017) revealed that the average annual growth of email users was 4%. Radicati Group inc. (2017) predicted a total of 319.6 billion of overall email transactions by 2021, with total users numbering 4 billion.

However, despite the massive number of transactions, 40% of email is classified as spam (Harisinghaney *et al*., 2014). Spam email is unwanted email that is sent to a recipient without his or her consent and contains elements of fraud, promotions (Nikam and Chaudhari, 2017). Before the email classification system was invented, email users would manually perform email filtering that used up many resources, known as email triage (Smith *et al*., 2005). Email triage is time-consuming for the user that deals with large email transactions and could fast escalate into a problem if not handled.

Smith *et al*. (2005) observed that email users who received more than 100 emails in their inbox every day would spend 2 h. only managing emails. It has also been recorded that the loss caused by spam email could reach 355 million USD each year (Awad, 2011).

Several developments have therefore been undertaken to solve this problem; starting from a simple approach such as making a priority inbox (Tsugawa *et al*., 2010; Aberdeen, 2010), to a more sophisticated spam classification system using deep learning, which has now become increasingly popular. Before the emergence of deep learning in the last decade, the Artificial Neural Network (ANN) was the dominant method used in most classification systems. Since its first debut in the 90s, ANN has been known as a robust method for representing training data.

The Recurrent Neural Network (RNN) is a type of ANN with a recurrent flow of neutrons, made to handle sequential data that is sensitive to order such as time-series or sentences. Despite it having a great opportunity to take over other normal machine learning methods, RNN research met a dead end towards the end of the

90s. This was because the gradient descent technique that was used at that time could not perform well, resulting in the vanishing gradient problem or exploding gradient problem. To overcome this problem, (Hochreiter and Schmidhuber, 1997) formalized the Long Short-Term Memory (LSTM) network, an advanced form of the classic RNN. In terms of application, LSTM still has some problems concerning time complexity.

Up until now, many studies have been carried out to further develop the LSTM network. One of the most recent is the Gated Recurrent Units (GRU) network, a simplified LSTM model that still retains the performance of the original model. In some cases, this model has proven to be better than previous methods (Cho *et al*., 2014). Therefore, in this paper, the performance of GRU and LSTM in classifying spam emails was compared. It is expected that the GRU method will produce a classification model that is better than other classification methods.

The data used in this experiment are spam emails consisting of subjects and email bodies. The word length used after the vectorized data was 70 words. This word length was chosen for the following reasons: (1) The graph of the length of data distribution supports the use of 70 words as a boundary, (2) the memory capacity is not able to store variables during the training process and the data size is too large.

The performance measurement metrics used are accuracy and recall; because this paper deals with spam, the sensitivity of the model is important. The error or loss metric chosen was the categorical cross-entropy with Adam optimization.

Based on the problems stated above, the focus of this study is to determine the performance of the LSTM and GRU models in classifying spam email data under testing scenarios explained in more detail in Section 3. In addition, the model's ability to overcome overfitting was tested using dropout (with or without).

Related Studies

Since the 90s, machine learning methods have proved very popular for classifying emails. Some of the most commonly used methods are the Bayesian Network (Banday and Sheikh, 2014; Wang *et al*., 2014), the K-Nearest Neighbor (KNN) (Harisinghaney *et al*., 2014), the Support Vector Machine (SVM) (Karthika and Visalakshi, 2013; Song, 2013; Ma *et al*., 2013; Shi, 2012; Zhu, 2008; Xu *et al*., 2014), the K-Means (Elssied *et al*., 2014) and the Decision Tree (Shi *et al*., 2012). After the concept of deep learning was introduced, the trend of research began to change to focus more on deep learning. This is because deep learning opens a very wide opportunity to address difficult problems using simple existing machine learning models.

A. Long Short Term Memory (*LSTM*)

One of the weaknesses of ANN in machine learning is the inability of the model to recognize patterns in data that is sensitive to time sequences. In the year 1990, an ANN architecture was created with the ability to store short-term memory based on ANN, as well as handle the above problem, named the Recurrent Neural Network (RNN). RNN is an ANN network that is able to learn features and long-term dependencies from data (Bengio *et al*., 1994). The main idea of RNN is to create a layer that accepts two inputs and produces two outputs.

At the beginning of its development, the RNN used the Gradient Descent technique in its training model. As a result, the problem vanishing gradient or exploding gradient cannot be avoided. This problem renders the RNN unable to update memory for a long period of time. Hochreiter and Schmidhuber (1997) created the LSTM architecture to deal with this problem. By adding a Memory Cell to each RNN cell, the model will have a memory capacity that is immune to the problem of the vanishing gradient or the exploding gradient. In fact, LSTM is even able to work well with noise (Hochreiter and Schmidhuber, 1997).

The design of the LSTM cell can be seen in Fig. 1. LSTM works by receiving 3 inputs and producing 3 outputs. The input received is Current Data, Memory Cell from the previous cell and Hidden State from the previous cell. Meanwhile, the output of the LSTM cells is Memory Cell and Hidden State for output or the next cell. Equation (1-6) describe the process that occurs in the LSTM cell:

$$f_t = \sigma\left(W_f \,.\left[h_{t-1}, x_t\right] + b_f\right) \tag{1}$$

$$i_t = \sigma\left(W_i \,.\left[h_{t-1}, x_t\right] + b_i\right) \tag{2}$$

$$j_t = tanh\left(W_j \,.\left[h_{t-1}, x_t\right] + b_j\right) \tag{3}$$

$$o_t = \sigma\left(W_o \,.\left[h_{t-1}, x_t\right] + b_o\right) \tag{4}$$

$$c_t = c_{t-1} \,.\, f_t + i_t \,.\, j_t \tag{5}$$

$$h_t = tanh\left(c_t\right).o_t \tag{6}$$

Based on equation above, f_t is the Forget Gate, i_t is the Input Gate, o_t is the Output Gate, c_t is the Memory Cell and h_t is the hidden state. The Forget Gate reduces information from the hidden state's previous cell and current data. The Input Gate is the input data to be processed with the Forget Gate to update the Memory Cell and the Output Gate is the output of the current cell that is a mix of memory and data input. With this method, the LSTM is able to store long-term memories in *C*.

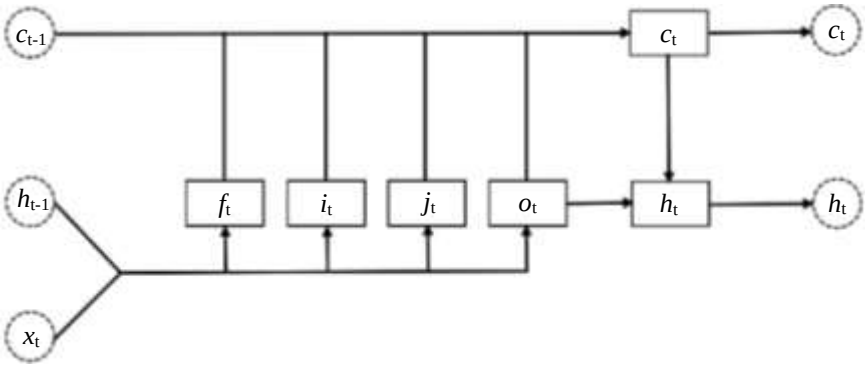

Fig. 1: LSTM cell

LSTM is generally used for time-series data. This is because the LSTM background is intended to handle data that has sensitivity to time sequences. In the problem of text classification, two new problems arise, namely when the text is too short and the text is too long. Text cases that are too short, as in SMSs, Tweets, or captions can be a problem because it is almost certain that the text does not have adequate sentiment value. In the case of very long documents, Liu *et al*. (2015) built Multi Timescale LSTM (MT-LSTM) so that the model will not lose 'memory' in the update process. The main idea of this method is to connect several LSTM layers in a timescale with an LSTM layer in another timescale. In the research, the MT-LSTM model was trained and tested on film review documents. From the results of the tests mentioned, MT-LSTM was able to surpass other LSTM and CNN models at that time, even for short-text cases. Liu *et al*. (2015) mentioned that for further research, the authors should test several feedback mechanisms.

In addition to text classification, LSTM has also been used to address other problems. In the case of sound, LSTM has been used in the problem of speech recognition (Sak *et al*., 2014) and the classification of laryngitis (Guedes *et al*., 2018). In addition, in 2018, LSTM was used to detect earthquakes in Japan (Kuyuk and Susumu, 2018) with good results.

Zhou *et al*. (2015) introduced a new method of classification combining the Convolutional Neural Network (CNN) and LSTM called C-LSTM. In his research, Zhou *et al*. (2015) used CNN to get text data features in the form of arrays. This array then served as the LSTM input for later classification. This method succeeded in surpassing the performance of CNN and LSTM.

Lee and Dernoncourt (2016) used LSTM and CNN to classify short text. By previously converting the text into a vector using Google's Word2Vec, each of the LSTM and CNN models successfully outperformed the machine learning method in general. The same problem was investigated by (Raj *et al*., 2018). In his study, (Raj *et al*., 2018) used LSTM to classify SMS. The proposed model successfully outperformed several other machine learning methods, returning an accuracy of 97%.

B. Gated Recurrent Unit (GRU)

In practice, LSTM is able to handle the problem of vanishing or exploding gradients faced by RNN. Nevertheless, LSTM is considered to have a fairly complicated architecture. Cho *et al*. (2014) created the Gated Recurrent Unit (GRU) as an alternative to LSTM. A simple form of LSTM, GRU has a simpler complexity than LSTM and has proven to outperform LSTM in some cases (Chung *et al*., 2014). Figure 2 describes the design of the GRU cell.

The main difference between GRU and LSTM is that GRU does not have a Memory Cell like LSTM and instead replaces it with a hidden state only. For this replacement, the GRU combines the Input Gate and the Forget Gate in the LSTM mechanism into an Update Gate to change the hidden state from the previous one into the candidate hidden state now. The Reset Gate in GRU serves to forget memories from previous hidden state. The complete GRU cell equation is given by Equation (7-11):

$$r_t = \sigma\left(W_r \cdot \left[h_{t-1}, x_t\right] + b_r\right) \tag{7}$$

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right] + b_z\right) \tag{8}$$

$$\tilde{h}_t = tanh\left(W_{\tilde{h}} \cdot \left[h_{t-1}, x_t\right] + b_{\tilde{h}}\right) \tag{9}$$

$$o_t = \left(z_t \cdot h_{t-1}\right) + \left[\left(1 - z_t\right) \cdot \tilde{h}_t\right] \tag{10}$$

$$h_t = c_{t-1} \cdot f_t + i_t \cdot j_t \tag{11}$$

where, z_t is the update gate, r_t is the reset gate and h_t is the hidden state.

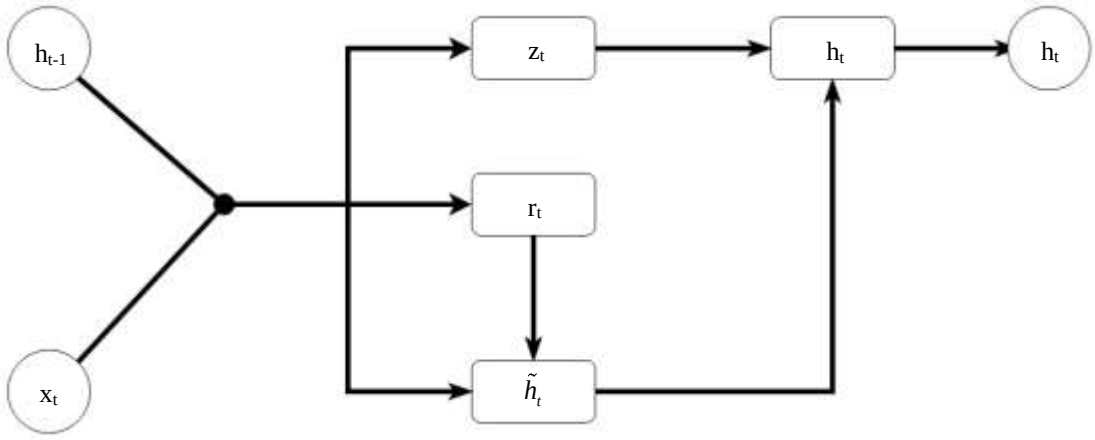

Fig. 2: GRU cell

C. K-Fold Cross Validation

K-Fold Cross Validation is a sample-based model performance validation method (Stone, 1974). Basically, using this method, the data is divided into *K* parts or folds, where $K \in N$. Separate data are then grouped into the *K* scenario with each scenario having training data and validation data at a ratio of (1: *K*-1). In the end, the model is tested against each of the *K* scenarios mentioned above.

D. Word Embedding

In a statistical approach, machine learning and deep learning, texts that will be classified will usually be represented as numbers to enable the processing of the data using mathematical functions known as vectorization. The vectorization method is divided into two, namely frequency-based and predictive methods.

Word embedding was first introduced by Bengio *et al*. (2003) as a language model based on ANN. The advantage of the word embedding over the frequency-based model is its ability to study the context of sentences or data. In many cases, word embedding greatly outperformed the frequency-based model (Baroni *et al*., 2014). The use of word embedding as a standard in language modeling is now prevalent, with the creation of Word2Vec by Mikolov *et al*. (2013) in 2014 and Glove by Pennington *et al*. (2014), a word embedding model often used as a pre-trained model.

E. Dropout

Models that are built using the deep learning approach have a very large number of training parameters, which can result in an increased possibility of overfitting of the model. These problems can be overcome using dropping, a technique that disables a unit or neuron in the neural network.

A large number of parameters in the deep learning model will increase the overfitting potential. Srivastava *et al*. (2017) used the dropout technique to deactivate some neurons in the ANN model and was able to improve the performance of the model. The dropout technique continued its development in 2015 when Tompson *et al*. (2015) used the Convolutional Neural Network to propose a new dropout technique called Spatial Dropout.

In the conventional dropout, each neuron that will be deactivated has a mutually independent opportunity while with the Spatial Dropout, each neuron has the same chance. Spatial Dropout is used if several neurons are considered related to each other. The results of the research showed that using Spatial Dropout could improve the performance of the model in question.

System Design

The data used in this study consists of a spam email dataset owned by the Enron company. This data contains the subject and content of e-mail with a time span of 1999-2005. Total data was 34519 with a spam portion to non-spam of 1:3. Email spam classification is an example of classification of imbalanced data (Chawla *et al*., 2002). The data used in this study was processed data (eliminating duplicates, choosing spam and non-spam emails with a portion of 1:1 (balancing class)). Balancing class is needed to get good machine learning model (Poolsawad *et al*., 2014). In this study, spam data is labeled 1 and non-spam data is labeled 0. The data sample is given in Table 1. The data sample is classified as spam (value: 1).

A. System Design

Overall, the system built was based on neural network. Therefore, the original data in the form of text must be converted into vector form with the word

embedding technique using a pre-trained model, glove840B, which has been trained against 840 million of words via common crawl. After the data is converted into vector form, the model classifies the data through the LSTM/GRU layer and go through a dense or fully connected neural network. The performance of the model was measured using *K*-Fold Cross Validation. *K*-Fold Cross Validation is a resampling procedure used to evaluate machine learning models. The procedure has a single parameter called K that refers to the number of groups that a given data sample is to be split into. In our case, the value of K is 7. A general description of the test scenario with 7-Fold Cross Validation for each model is given in Fig. 3.

Based on Fig. 3, data in the form of text was separated into *K* groups with each group having a portion of training and validation data. After the data was separated, the data was sent to each fold test scenario to train and test the model that has been created in each fold.

The design of the LSTM and GRU models was made as closely as possible to improve the feasibility of the resulting performance comparison. The two designs can be seen in Fig. 4 and 5.

The process flow of data entered into the model starts from the embedding layer where the data in the form of text was tokenized, used as sequences, then vectorized using the Glove word embedding. In this study, the number of words was limited to 70. In addition, the Glove word embedding model produced a vector of 300 values for each of these words.

The data that was separated was then trained into as much as 100 units of LSTM cells, then analyzed into 2 neural network dense networks measuring 1024 with ReLU activation function and finally channelled to dense size 2 with a softmax activation function because only 2 classes (spam and non-spam) were involved. The value after softmax was then compared with the label value of the data in question to produce an error or loss. The loss function used in this model was the categorical cross-entropy with the Adam optimizer.

Table 1: Data sample

Email	Class
Subject: dobmeos with hgh my energy level has gone up ! stukm Introducing doctor - formulated hgh human growth hormone - also called hgh is referred to in medical science as the master hormone. it is very plentiful when we are young, but near the age of twenty - one our bodies begin to produce less of it. by the forty time we are nearly everyone is deficient in hgh and at eighty our production diminished has normally at least 90-95%. advantages of hgh : …	1

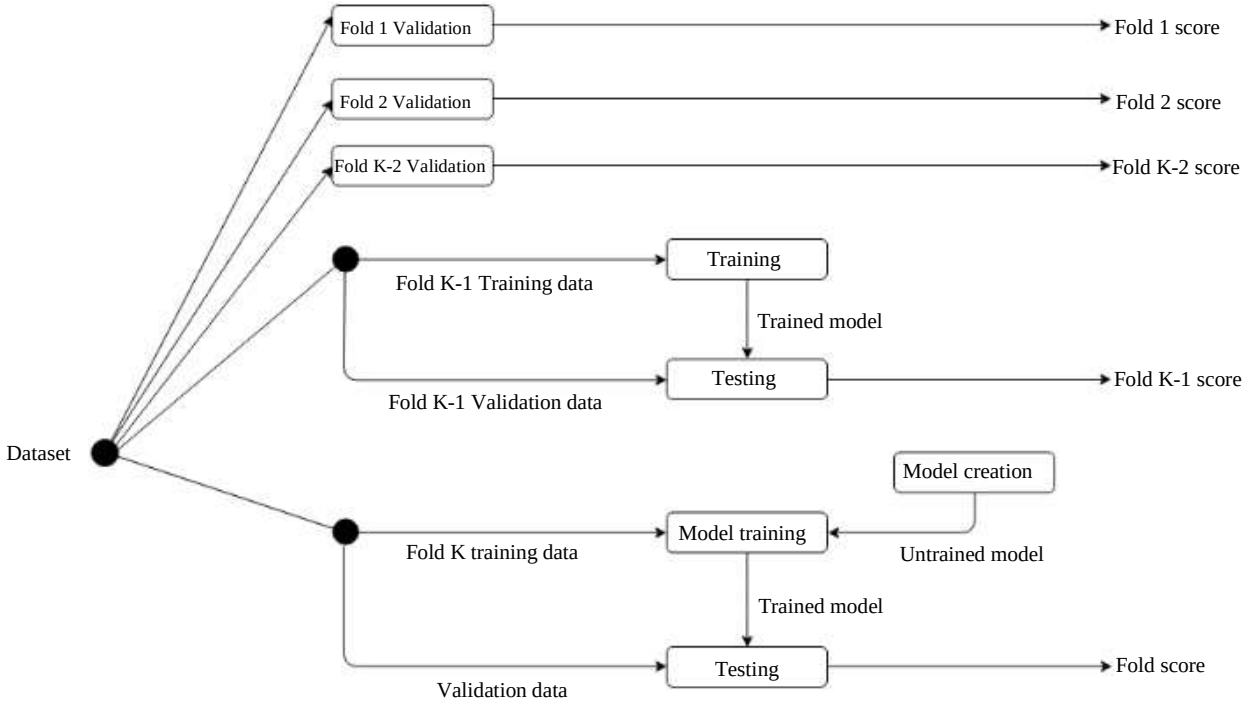

Fig. 3: System design

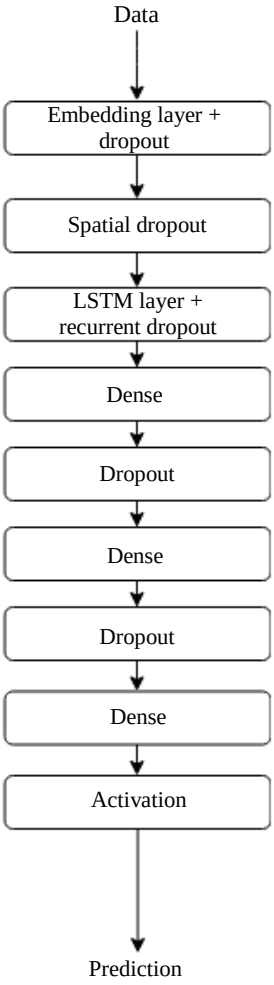

Fig. 4: LSTM model

As a comparison, in this study, the accuracy and recall performance of the two models were also compared with the performance of the XGBoost (Guestrin, 2016) model to determine whether the performance of the two models was able to exceed the existing baseline model. XGBoost is a scalable end-to-end tree boosting system. Tree boosting is highly effective machine learning system and it has been shown to give state-of-the-art results on many standard classification benchmarks (Li, 2010).

The only difference between the designed LSTM and GRU models was the LSTM and GRU units used after the layer embedding. The difference between the two cells was explained earlier in Fig. 4 and Fig. 5. To test the ability of both models in dealing with overfitting, the test was separated into 2 scenarios. The first scenario tested both models with the help of dropouts. The second scenario tested both models without dropout assistance. This scenario was developed to measure the ability of the two models in overcoming overfitting without dropout assistance.

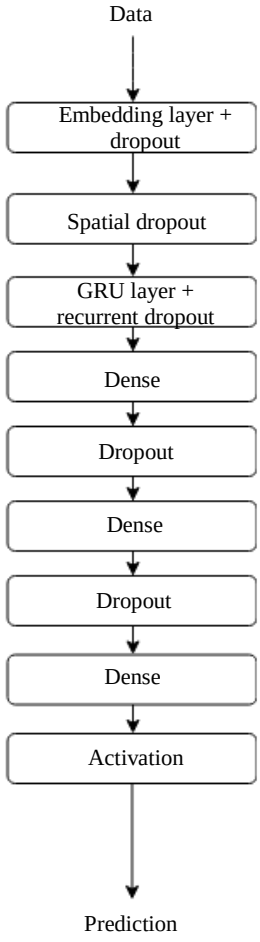

Fig. 5: GRU model

Result

A. Result

Based on the system design built, the main results of this test are the accuracy and recall metrics for each algorithm after going through the validation process using 7-Fold Cross-validation. The results are listed in Table 2 and 3:

In addition to the accuracy and recall metrics, the execution time (in seconds) of the LSTM and GRU in classifying the entire test data during the validation process and using 7-fold cross validation models was also noted. The results show that the GRU took 44.5561 2 sec while LSTM took 45.26729 2 sec to perform the classification.

B. Analysis

The results of the measurement of the execution time state that the GRU had an execution time of 44.5561 seconds while LSTM took 45.26729 2 sec. This is in line with the fact that GRU has a less complicated architecture than LSTM. Broadly speaking, the GRU cell only has four

operations, while the LSTM cell has six operations. In other words, the GRU must be more efficient than the GRU based on a ratio of 2:3. However, from the results of the time measurements obtained, 44.5561:45.26729 was not proportional to 2:3, as it should be.

The not proportional ratio is because the total operations performed were not only done in the GRU cell or the LSTM cell, but also in the entire model. From the built-in model, it is known that the GRU cells had a total of 120, 300 parameters and the LSTM cell had a total of 160, 200 parameters. Both these values are proportional to 2:3, rendering the running time performance of the two methods insignificant, as it is simply another parameter that involves the process of embedding, dropping and dense networks.

Based on the model that was built, the total parameters of the LSTM model were 57, 476, 974 parameters, while GRU had 57, 436, 874 parameters. If these two values were compared, 0.9993023 would be obtained, which approaches one. This is the reason why the run time comparisons of the two methods were less significant.

In terms of performance, the accuracy of the LSTM and GRU models was the same, namely 0.990183 and both outperformed the XGBoost and LSTM models without Dropout and GRU without Dropout. Meanwhile, the recall value of the GRU model outperformed other models, returning a value of 0.989827. That is, GRU had better sensitivity than LSTM. These results prove that the LSTM and GRU models were able to compete in the case of email classification, although GRU performed slightly better than LSTM.

In the test scenario without dropouts, the two models returned similar accuracy and recall and so did not have a difference in performance. This proves that the two models have a tendency to avoid overfitting by the Forget Gate in LSTM and the Reset Gate in GRU. However, from the results obtained, it appears that without dropout, LSTM was able to outperform the GRU in terms of accuracy and recall. Most likely, this result is attributed to the help of LSTM's Memory Cell, which functions as a long-term memory pipeline.

Table 2: Accuracy score

	Model				
Fold	*GRU*	*LSTM*	*GRU (No Dropout)*	*LSTM (No Dropout)*	*XGBoost*
Fold 1	0.98962	0.98921	0.98547	0.98651	0.98443
Fold 2	0.99066	0.99232	0.98609	0.98651	0.98443
Fold 3	0.98734	0.98817	0.98443	0.98588	0.98485
Fold 4	0.99211	0.99107	0.98651	0.98692	0.98526
Fold 5	0.99253	0.99336	0.98713	0.98754	0.98464
Fold 6	0.98837	0.98526	0.98464	0.98256	0.9865
Fold 7	0.99066	0.9919	0.98609	0.98609	0.9865
Average	**0.99018**	**0.99018**	0.98576	0.986	0.98523
Standard deviation	0.001887	0.002817	0.00098	0.001612	0.000913

Table 3: Recall score

	Model				
Fold	*GRU*	*LSTM*	*GRU (No Dropout)*	*LSTM (No Dropout)*	*XGBoost*
Fold 1	0.988907	0.990512	0.982815	0.982456	0.983176
Fold 2	0.988741	0.989956	0.987849	0.984721	0.983877
Fold 3	0.98913	0.989541	0.983974	0.983633	0.980525
Fold 4	0.992126	0.986054	0.983526	0.984336	0.979525
Fold 5	0.99185	0.991863	0.985366	0.98696	0.978261
Fold 6	0.990272	0.988618	0.980777	0.98264	0.984325
Fold 7	0.98776	0.988187	0.982878	0.982878	0.982499
Average	**0.989827**	0.989247	0.983884	0.983946	0.981741
Standard Deviation	0.001651	0.00186	0.002231	0.001581	0.002323

Summary

From the analysis of the results, it can be concluded that for spam email classification cases using the Enron dataset, GRU performed the same as LSTM. Meanwhile, GRU had superior recall than LSTM with a score of 0.989827 compared to 0.989247. In addition, dropout deletion did not have a major impact on the scores of the two algorithms. However, LSTM without dropout was able to outperform GRU without dropout with an accuracy of 0.986001 as opposed to 0.985763; and 0.983946 versus 0.983884 for recall. The two models were able to outperform the XGBoost method for all scenarios.

Overall, the resulting difference was not that significant. This is most likely due to the training dataset that was used, which was too easy to be learned by the models. Because of research time availability, we could only perform with that training dataset. To deal with this research limitation, future research should use other training datasets and the model parameters should be more varied to improve the validity of the results obtained.

Author's Contributions

All authors have equally contributed in this work.

Ethics

There are no ethical issues associated with this research.

References

Aberdeen, D., 2010. The learning behind Gmail priority inbox.

Awad, W.A., 2011. Machine learning methods for spam email classification. Int. J. Comput. Sci. Inform. Technol., 3: 1-1.

Banday, M.T. and S.A. Sheikh, 2014. Multilingual e-mail classification using Bayesian filtering and language translation. Proceedings of the International Conference on Contemporary Computing and Informatics, Nov. 27-29, IEEE Xplore Press, Mysore, India, pp: 696-701. DOI: 10.1109/IC3I.2014.7019788

Baroni, M., G. Dinu and G. Kruszewski, 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, (ACL' 14), Association for Computational Linguistics, Baltimore, Maryland, pp: 238-247. DOI: 10.3115/v1/P14-1023

Bengio, Y., P. Simard and P. Frasconi, 1994. Learning long-term dependencies with gradient descent is difficult. Tran. Neural Networks, 5:157-166.

Bengio, Y., R. Ducharme, P. Vincent and C. Jauvin, 2003. A neural probabilistic language model. J. Mach. Lear. Res., 3: 1137-1155.

Chawla, N.V., K.W. Bowyer, L.O. Hall and W.P. Kegelmeyer, 2002. SMOTE: Synthetic minority over-sampling technique. J. Artificial Intell. Res., 16: 321-357.

Cho, K., B.V. Merrienboer, C. Gulcehre, D. Bahdanau and F. Bougares *et al*., 2014. Learning phrasere presentations using RNN encoder-decoder for statistical machine translation.

Chung, J., C. Gulcehre, K. Cho and Y. Bengio, 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.

Elssied, N.O.F., O. Ibrahim and W.A. Ulbeh, 2014. An improved of spam e-mail classification mechanism using k-means clustering. J. Theoret. Applied Inform. Technol.

Guedes, V., A. Junior, J. Fernandes, F. Teixeira and J.P. Teixeira, 2018. Long short term memory on chronic laryngitis classification. Proc. Comput. Sci., 138: 250-257.

Guestrin, T.C.C., 2016. XGBOOST: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 13-17, ACM, San Francisco, California, USA, pp: 785-794. DOI: 10.1145/2939672.2939785

Harisinghaney, A., A. Dixit, S. Gupta and A. Arora, 2014. Text and image based spam email classification using KNN, Naıve Bayes and reverse dbscan algorithm. Proceedings of the International Conference on Reliability Optimization and Information Technology, (OIT' 14), pp: 153-155.

Hochreiter, S. and J. Schmidhuber, 1997. Long short-term memory. Neural Comput., 9: 1735-1780.

Karthika, D.R. and P. Visalakshi, 2013. Latent semantic indexing based SVM model for email spam classification. J. Sci. Industrial Res., 73: 437-442.

Kuyuk, H.S. and O. Susumu, 2018. Real-time classification of earthquake using deep learning. Proc. Comput. Sci., 140: 298-305.

Lee, J.Y. and F. Dernoncourt, 2016. Sequential short-text classification with recurrent and convolutional neural networks.

Li, P., 2010. Robust logitboost and Adaptive Base Class (ABC) logitboost. Proceedings of the 26th Conference Annual Conference on Uncertainty in Artificial Intelligence, (UAI'10), pp: 302-311.

Liu, P., X. Qiu, X. Chen, S. Wu and X. Huang, 2015. Multi-timescale long short-term memory neural network for modelling sentences and documents. Proceedings of the Conference on Empirical Methods in Natural Language Processing, (NLP' 15), Lisbon, Portugal, pp: 2326-2335. DOI: 10.18653/v1/D15-1280

Ma, T., H. Xu and L. Lifan, 2013. The research on email classification based on q-Gaussian kernel SVM. J. Theoretical Applied Inform. Technol., 48: 1292-1299.

Mikolov, T., K. Chen, G. Corrado and J. Dean, 2013. Efficient estimation of word representations in vector space.

Nikam, S. and R. Chaudhari, 2017. A review paper on image spam filtering.

Partridge, C., 2008. The technical development of internet email. Annals History Comput., 30: 3-29. DOI: 10.1109/MAHC.2008.32

Pennington, J., R. Socher and C. Manning, 2014. Glove: Global vectors for word representation. Proceedings of the Conference on Empirical Method Sinnaturall Anguage Processing, (SAP' 14), ACL, Doha, Qatar, pp: 1532-1543. DOI: 10.3115/v1/D14-1162

Poolsawad, N., C. Kambhampati and J.G.F. Cleland, 2014. Balancing class for performance of classification with a clinical dataset. Proc. World Congress Eng., 1: 1-6.

Radicati Group inc., 2013. Email statistics report, 2013-2017.

Radicati Group inc., 2014. Email statistics report, 2014-2018. f

Radicati Group inc., 2017. Email statistics report, 2017-2021.

Raj, H., Y. Weihong, S.K. Banbhrani and S.P. Dino, 2018. LSTM based Short Message Service (SMS) modeling for spam classification. Proceedings of the International Conference on Machine Learning Technologies, May 19-21, ACM, New York, USA, pp: 76-80. DOI: 10.1145/3231884.3231895

Sak, H., A. Senior and F. Beaufays, 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Proceedings of the 15th Annual Conference of the International Speech Communication Association, (SCA' 14).

Shi, L., Q. Wang, X. Ma, M. Weng and H. Qiao, 2012. Spam email classification using decision tree ensemble. J. Comput. Inform. Syst., 8: 949-956.

Shi, T., 2012. Research on the application of e-mail classification based on support vector machine. Frontiers Comput. Edu., 133: 987-994. DOI: 10.1007/978-3-642-27552-4_129

Smith, M.A., C. Neustaedter and A.B. Brush, 2005. Beyond "from" and "received": Exploring the dynamics of email triage. Proceedings of the Conference on Human Factors in Computing Systems, Apr. 02-07, ACM, New York, USA, pp: 1977-1980. DOI: 10.1145/1056808.1057071

Song, M.H., 2013. E-mail classification based learning algorithm using support vector machine. Applied Mech. Mater., 268: 1844-1848.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, 2017. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Lear. Res., 15: 1929-1958.

Stone, M., 1974. Cross-validatory choice and assessment of statistical predictions. J. Royal Stat. Society. Series B (Methodological), 36: 111-133.

Tompson, J., R. Goroshin, A. Jain, Y.L. Cun and C. Bregler, 2015. Efficient object localization using convolutional networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun. 7-12, IEEE Xplore Press, Boston, MA, USA, pp: 648-656. DOI: 10.1109/CVPR.2015.7298664

Tsugawa, S., K. Takahashi, H. Ohsaki and M. Imase, 2010. Robust estimation of message importance using inferred inter-recipient trust for supporting email triage. Proceedings of the 10th IEEE/IPSJ International Symposium on Applications and the Internet, Jul. 19-23, IEEE Xplore Press, Seoul, South Korea, pp: 177-180. DOI: 10.1109/SAINT.2010.53

Wang, Z.J., Y. Liu and Z.J. Wang, 2014. E-mail filtration and classification based on variable weights of the Bayesian algorithm. Applied Mech. Mater., 513: 2111-2114.

Xu, K., C. Wen, Q. Yuan, X. He and J. Tie, 2014. A mapreduce based parallel SVM for email classification. J. Networks, 9: 1640-1640.

Zhou, C., C. Sun, Z. Liu and F. Lau, 2015. AC-LSTM neural network for text classification.

Zhu, Z., 2008. An email classification model based on rough set and support vector machine. Fuzzy Syst. Knowl. Disco., 5: 236-240.