Text classification performance analysis using different supervised machine learning models

Aryan Kumar Singh, Jayprakash, Santosh Kumar Singh

Abstract—Textual data classification is a primary task in many of the Natural Language Processing tasks. Textual data classification is considered as an essential method to manage and process a large number of documents in the digital format that are widespread and continuously increasing in the count. Textual data classification has a vital role in information retrieval and summarisation. Many Supervised Machine learning algorithms can perform text classification. This paper introduces a performance analysis of different machine learning algorithms on the 20 NewsGroup dataset.

Index Terms—CNN, deep learning, GRU,Logistic Regression,LSTM, Machine learning, Naive Bayes, RNN, SVC.

1 Introduction

Textual data classification is and will always be an essential research topic since the start of digital documents[1]. In Today's world, textual data classification is significant because one needs to deal with a large number of text documents and files daily. In Textual Data Classification, a document is classified in a predefined group. Moreover, for classifying text with higher accuracy and prediction, it needs to have a better performing classification machine learning model. Deep learning models have surpassed the classical machine learning models in many different tasks of natural language processing, including text classification, sentiment analysis, news categorization, natural language inference, and question answering [2]. This work will show quantitative analysis of different supervised text classification machine learning algorithms by using the 20 NewsGroup dataset to perform this classification. In the second section, this paper discussed a literature review, the next section explores the methodology applied, and the last section shows the results and conclusion.

2 Literature review

Textual data has a rich source of sequential information [3], but learning and extracting those pieces of information can be lengthy and challenging because of its unstructured nature.

Automatic Textual data classification can be grouped into three categories:

- *Aryan Kumar Singh is currently pursuing a bachelor degree program in computer science engineering in NIT Calicut, India, PH-+917357616638. E-mail: aryansingh98.cr7@gmail.com*
- *Dr. Jayprakash is Assitant Professor in computer science engineering in NIT Calicut, India,E-mail: jayprakash@nitc.ac.in*
- *Dr. Santosh Kumar Singh is Professor in electronics and communication engineering at Jaipur Engineering College and Research Centre Jaipur, India. E-mail: drsantosh.it@jecrc.ac.in*

• **Rule-based methods**: With the help of a set of predefined rules, this rule-based method can classify the textual data into different categories. One needs to perform domain analysis before using this method because in-depth knowledge of the domain is required.

• **Machine learning (data-driven) based methods:** These methods take input data along with their labels, and they learn the correlations based on these pre-labeled examples in case of supervised learning. In the case of unsupervised learning, they learn the hidden patterns and structures in the data without any labels.

• **Deep Learning methods:** These methods require massive datasets to train because they try to generalize the scenario to learn the feature representations which helps them to perform classification and regression tasks as an end to end method. These methods can also be transferred from one application to other because lines and edge detection is a basic need in most of the applications of computer vision.

Natural language processing and textual data classification have progressed rapidly with the help of neural network architecture, and now they are extensively used in the industries, for example, chatbots and speech recognition[1], [3], [4].

3 Methodology

In this work, the first step is to fetch out the dataset. As already mentioned, it has used the 20 NewsGroup dataset for performance analysis using the sklearn library to fetch the dataset. After that, preparing the dataset in X, Y label format, and then splitting the dataset into a training set and test set.

The second step is data cleaning. That includes [5]:

1. Lowercasing every character in the dataset
2. Word Tokenization
3. Removal of stopwords
4. Word Lemmatization

The next step is vectorization, in which textual data is mapped to real-valued vectors so that it can be given as an input to these models which implemented as follows:

3.1 Naive Bayes

It uses Bayes's Theorem for classification [6].

Bayes' Theorem:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)} \quad (1)$$

Here in (1), A refers to the category on which one needs to check the text document, and B refers to the document categories in the given document. So basically, what this model does is it uses its training information to compute this probability. And will label the document with the category which has the highest chance.

Hence, this is computing the conditional probabilities to calculate the probabilities of the occurrence of an event.

Naive Bayes is commonly applied to textual data classification. It simply performs well in many text classification problems. Its advantages include less training time and less training data, which results in less CPU and Memory consumption.

In this case, it has different documents with 20 categories. To help it categorize a new document from a test set, this will train a Naive Bayes classifier.

3.2 Support Vector Classifier

A large margin classifier or SVC is called a binary classifier because it separates only two groups at a time, as shown in fig.1, but it can use one vs. all technique for multiclass classification [7]. By applying the algorithm, the best separating hyperplane in which the margin between the two classes is maximized is figured out.

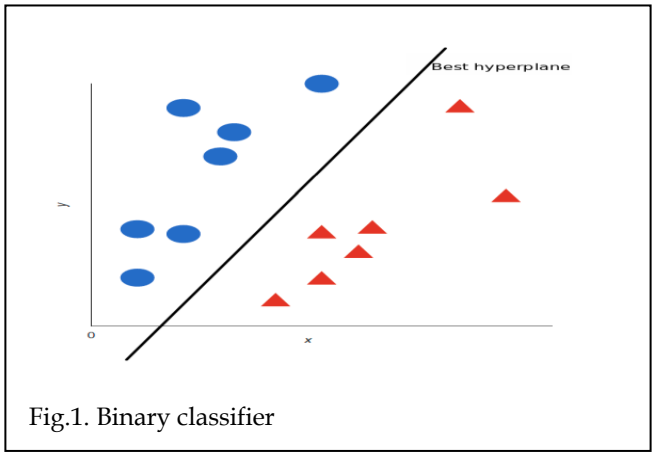

Fig.1. Binary classifier

After the algorithm is trained on the training set and it has determined the best separating hyperplane, then it can simply feed the vectorized input of all the unseen new input text examples and observe on which side of the decision boundary they fall so that it can classify them accordingly.

3.3 Gradient boosting

These classifiers merge many weak algorithms and collectively create a robust model by an internal voting mechanism [9]. Decision trees are a preferred choice when it comes to gradient boosting, as shown in fig. 2.

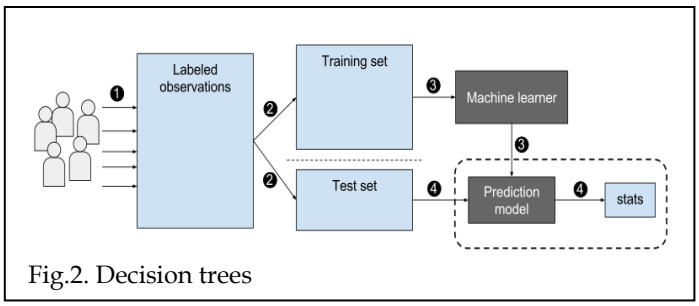

Fig.2. Decision trees

Steps of gradient boosting:

1. Model fitting and training
2. Hyperparameter tuning and changing the parameters of the model.
3. Using the test set and new inputs to make predictions

3.4 Random Forest Classifier

1. Randomly select K data points from 20NewsGroups training set.
2. Now, try to form a forest of decision trees associated with these K data points.
3. Choose the number Ntrees that want to build(at least 500) and repeat steps one and two.
4. To classify a new data point, it needs the votes of each of the N decision trees to predict the label class of the new point. They will assign the average value of all the Ntrees as its new class, as shown in fig.3.

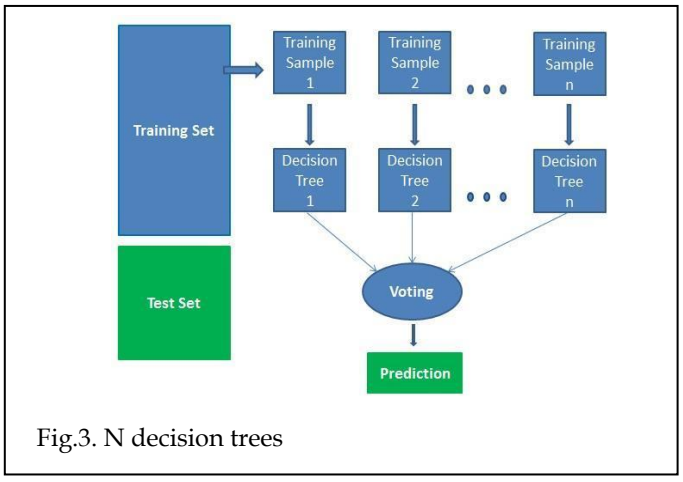

Fig.3. N decision trees

A random forest is just a team of decision trees, each one making some prediction of Y, and the ultimate prediction is simply the average of all the predictions of trees in the forest [10].

3.5 Word Embedding

Word embedding is a popular vectorized representation of the textual data of the document. They convey the word context, the syntactic and semantic similarity of a word, and also its relationship with other words in the document [11].

Suppose there is a term "w" that appears in a textual data of a document, then its context is defined by the adjacent set of words (say +5,-5) that lie around the term "w" in a fixed-size window [8]. They can utilize the context of "w" to form a vectorized representation of "w," which is its word vector. Similarly, a vector for each of the words could be created, and if these vectors are represented on vector space, then it can be observed that similar words cluster together, as shown in fig.4.

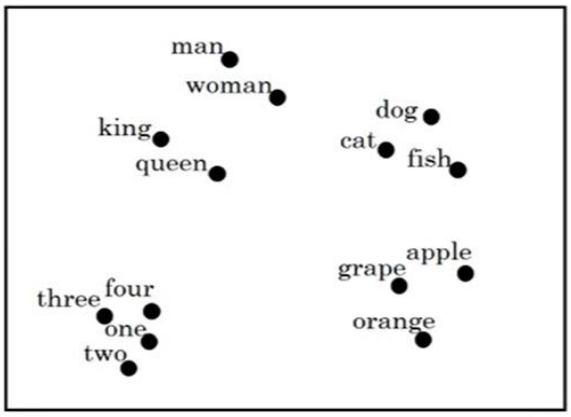

Fig.4. Word Embedding

Pre-trained word embeddings like GloVe can be used for transfer learning. The textual data is converted to an embedding matrix using pre-trained GloVe, and then it is converted into embedding sequences using an embedding layer of Keras. This embedding sequence is fed as input to different Deep Learning models.

3.6 GloVe Embeddings

Global Vectors for Word Representation takes plus points from both word2vec (local context-based) model and statistics of matrix factorization (global context-based) techniques. GloVe incorporates the advantages of both local and global context-based models; hence it performs better than other models that solve the same problem [12].

Advantages of GloVe:

1. Training is fast
2. Can be applied to a large corpus of data
3. Performs well, even when the dataset is small.

Example of GloVe representation as words which appear closest to the **frog**:

Frogs
Litoria
Toad
Eleutherodactylus
Leptodactylidae
Lizard
Rana

3.7 Convolutional Neural Networks(CNN)

CNNs are a type of ANN, and they are famous for analyzing images [13], [14].They have convolution hidden layers to detect patterns using filters.But here, CNN is applied to textual data. The below diagram shows how CNN works on textual data, and it shows all the steps required for making a CNN model, as shown in fig.5.

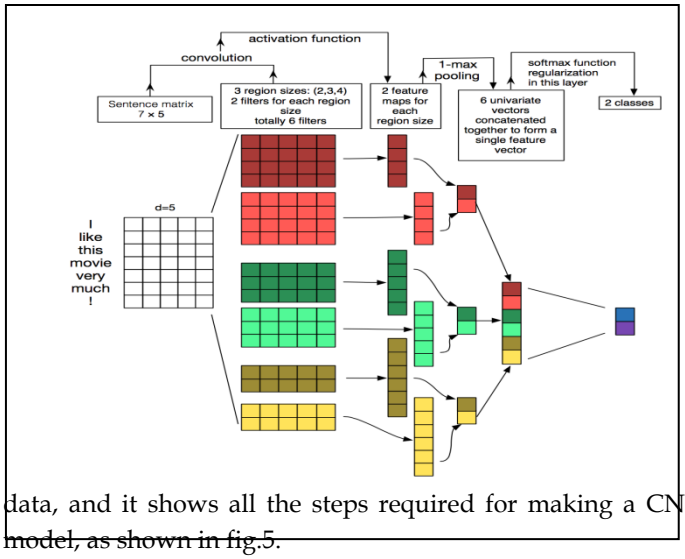

Fig.5. CNN

3.7 Recurrent Neural Network(RNN)

Simple RNNs are used to learn from sequential data, but they cannot capture long sequences.The output of the current state depends on both the current input and the output of the previous state, as shown in fig.6.

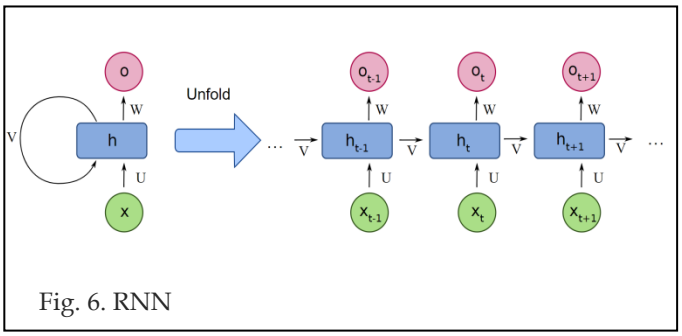

Fig. 6. RNN

In RNNs, if small weightsare provided, then it results in vanishing gradient problem, and usage of large weights resultsin an exploding gradient problem[14]. The earlier in the network, the weight resides, the more the terms are going to be included in the product that calculates the gradient.Multiplying a lot of terms which are less than one resultsin a very small gradient similarly, multiply a lot of terms which are greater than one results in an exponentially large gradient, and since the weights in the layers get updated proportionally to the gradientif the gradient is vanishingly small, then the update is also vanishingly small hence the new weight becomes stuck and

does not reach the optimum value. This hinders the learning process in the RNNs.

4 Proposed Method

The solution to these problems is a variation of RNN: Gated Recurrent Unit and LSTMs.LSTMs can capture long term dependencies and relations. They are able to remember the long term dependencies using the help of gates and cell states.Different units in LSTMs are [16]:

1. **Memory unit**: This helps in remembering what happened many times steps ago.
2. **Selection unit**: When new information and previous predictions arrive, what should be released and what should be kept internal as prediction is decided by this unit using the gating mechanism.
3. **Ignoring unit**: It lets things that aren't immediately relevant be set aside, so they don't cloud the prediction in the memory going forward.

Removal of some of the complexities of the LSTMs results in GRUs,and theyoperate efficiently with the help of only two gates, i.e., update and reset gates, and they don't have the cell states. GRUs usually train faster and are simpler in comparison to LSTMs [17], [18], [19], [20].RNN, LSTMs, and GRUs cells diagram shown in fig.7.

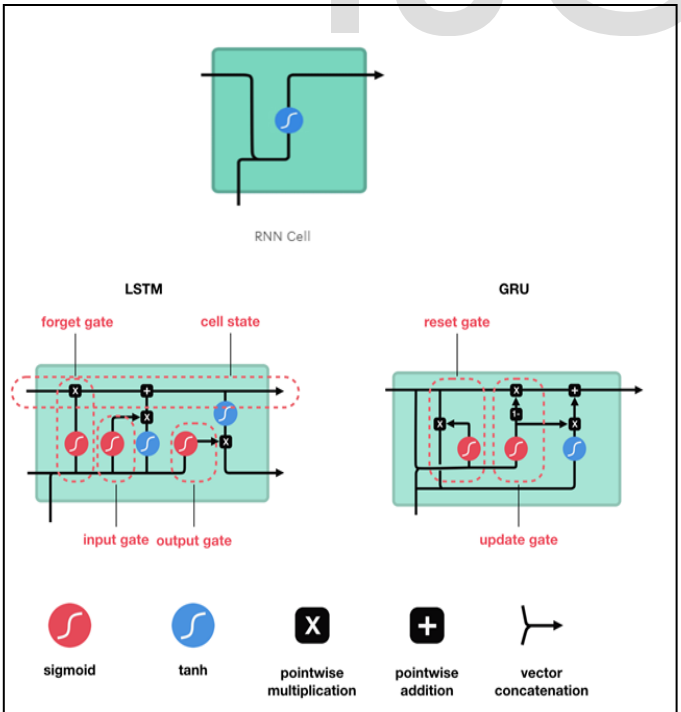

Fig. 7. RNN, LSTMs and GRU cell diagram

4.1Dataset Description

20 Newsgroups: This textual dataset is a collection of newsgroup files and documents [21].

For textual data clustering and textual data classification, this dataset has always been a popular choice for machine learning enthusiasts and researchers.

The dataset was originally collected by Ken Lang, and it consists of 20,000 news documents. There are five main fields that the newsgroups dataset contains, namely: computer, recreation, science, religion, and politics.

5 Results and Discussion

5.1 Supervised Machine Learning Models

In supervised machine learning models, after coding these algorithms in python using the Sklearn library, the following results are depicted in table 1.

TABLE 1
Test set results

	Naive Bayes	SVC	Gradient Boosting	Random Forest	Logistic Regression
Accuracy (test set)	67.3%	65.3%	59.5%	60.1%	67.4%
Run Time	0.06s	86.92s	86.92s	24.96s	6.37s

Both Logistic Regression and Naive Bayes are good for training on this specific 20 newsgroups dataset, but Naive Bayes is slightly more efficient. They both have good accuracies, and their running time is very less compared to svc, gradient boosting, and random forest.

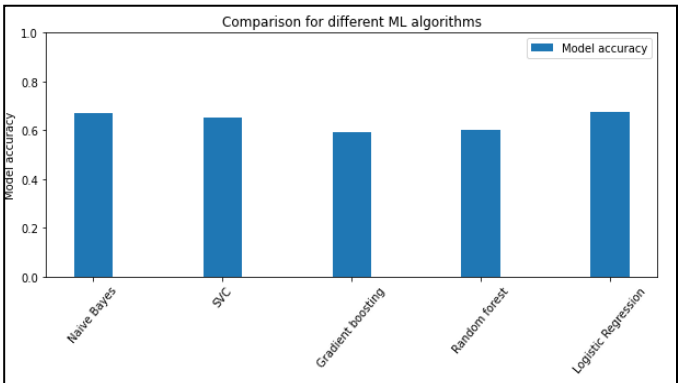

Fig.8. Different ML algorithm model accuracy

However, the top words calculated by the Naive Bayes model seems highly correlated with their corresponding newsgroups, as shown in fig.8.

5.2Deep Learning Model

With the help of the TensorFlow and Keras library, the following results were observed, as shown in table 2.

TABLE 2
Test set results

	CNN	LSTM	GRU	Simple RNN
Accuracy (test set)	56.4%	61.3%	62.5%	15.4%

Both GRU and LSTM perform quite well on this 20 newsgroup dataset with the help of pre-trained glove embeddings shown in fig.17. The accuracy and loss for GRU and LSTM model has been shown in fig. 9 to fig. 12.

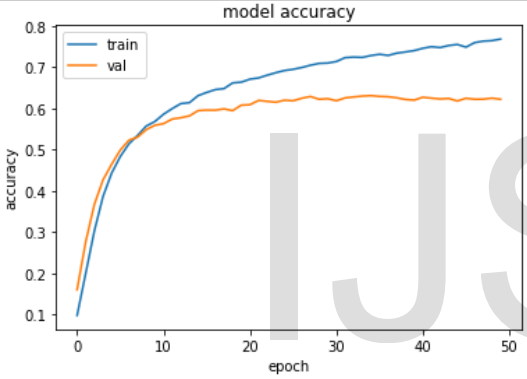

Fig.9. GRU model accuracy

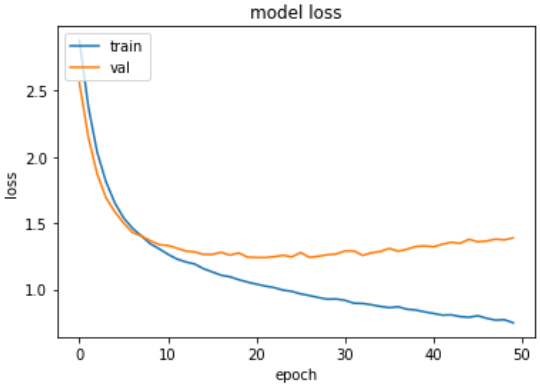

Fig.10. GRU model loss

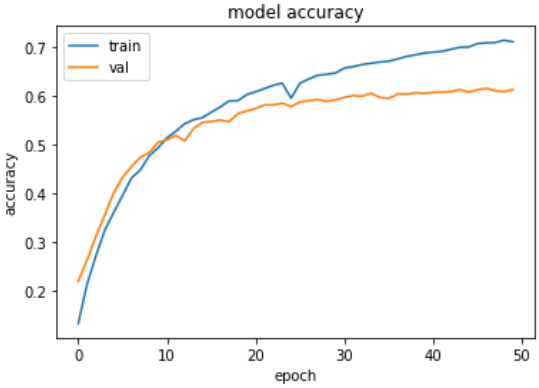

Fig.10. LSTM model for accuracy

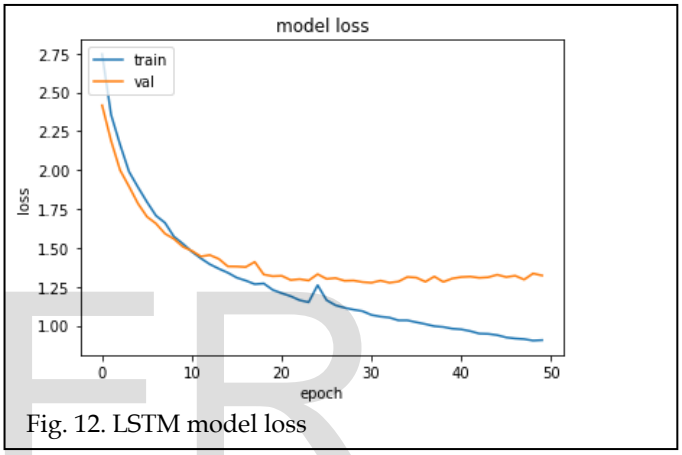

Fig. 12. LSTM model loss

CNN learned the correlations quickly and achieved a fairly good accuracy but started overfitting the training set after 15 epochs, as shown in fig. 13. To fig. 14

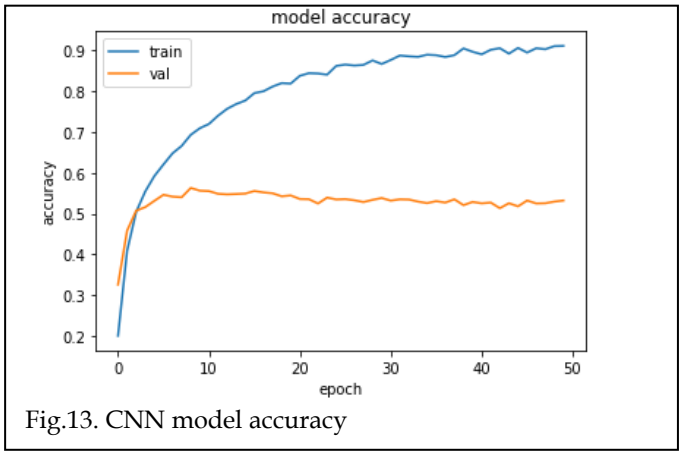

Fig.13. CNN model accuracy

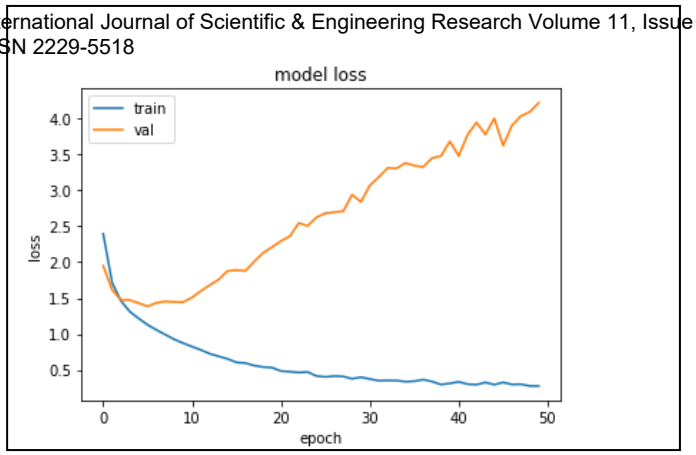

Fig. 14. CNN model for loss

Simple RNN performed worse on this dataset because it could not capture the long term dependencies, as shown in fig. 15 to fig. 16.

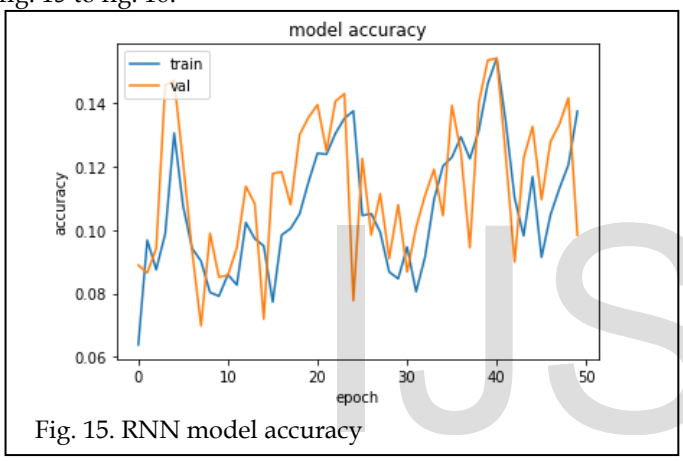

Fig. 15. RNN model accuracy

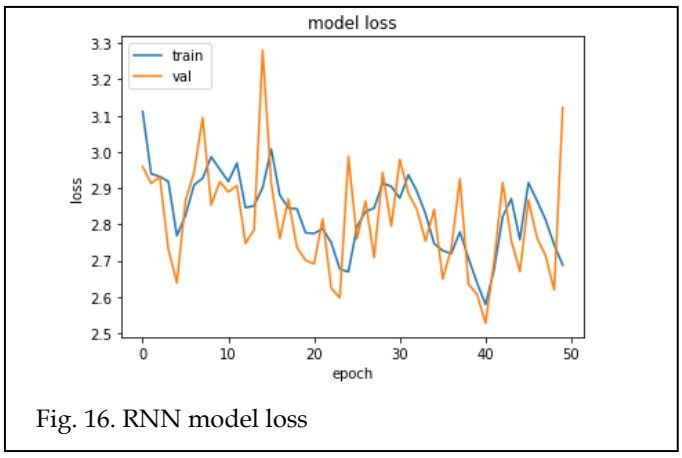

Fig. 16. RNN model loss

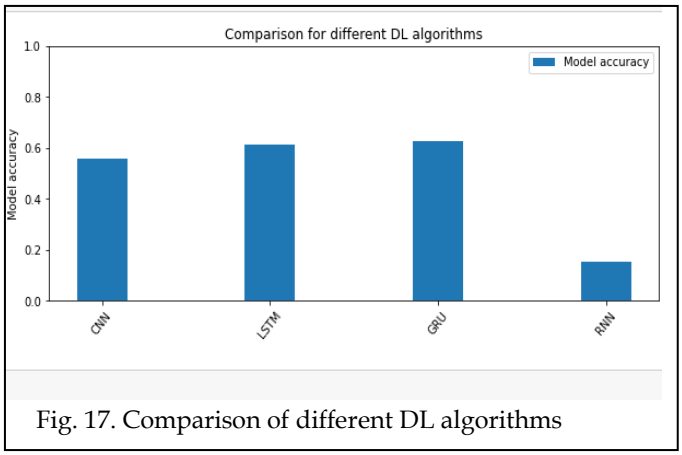

Fig. 17. Comparison of different DL algorithms

6 Conclusion

There seem to be five main fields that the newsgroups are trying to focus on, namely: computer, recreation, science, religion, and politics. Different new groups that report the same fields tend to use very similar words. Therefore it makes the prediction very hard to distinguish different new groups in the same fields.

References

[1] Sebastiani, F., Machine learning in automated text categorization, ACM Computing Surveys, 34(2002), 1-47.

[2] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2020. Deep Learning-Based Text Classification: A Comprehensive Review. 1, 1 (April 2020), 42 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[3] Talabis, M.R.M.; McPherson, R.; Miyamoto, I.; Martin, J.L.; Kaye, D. Security, and text mining. In Information Security Analytics; Talabis, M.R.M., McPherson, R., Miyamoto, I., Martin, J.L., Kaye, D., Eds.; Elsevier: Amsterdam, The Netherlands, 2015; pp. 123–150, doi:10.1016/B978-0-12-800207-0.00006-x.

[4] Dialog Systems and Chatbots: Speech and Language Processing. Daniel Jurafsky & James H. Martin. https://web.stanford.edu/~jurafsky/slp3/24.pdf

[5] *Natural Language Processing with Python*, by Steven Bird, Ewan Klein and Edward Loper, Copyright © 2019

[6] https://nlp.stanford.edu/IR-book/pdf/13bayes.pdf

[7] A Practical Guide to Support Vector Classification Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin Department of Computer Science National Taiwan University, Taipei 106, Taiwan

[8] Friedman JH (2001). "Greedy function approximation: a gradient boosting machine." Annals of Statistics, pp. 1189–1232.

[9] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189–1232, 2001.

[10] Breiman, L.: Random Forests. ML Journal 45(1), 5–32 (2001)

[11] Dinara Aliyeva, Kang-Min Kim, Byung-Ju Choi, Sang Keun Lee, "Combining Dual Word Embeddings with Open Directory Project Based Text Classification", *Cognitive Informatics & Cognitive Computing (ICCI*CC) 2018 IEEE 17th International Conference on*, pp. 179-186, 2018.

[12] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[13] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil. 2014. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In Proceedings of WWW 2014.

[14] A. B. Dieng, C. Wang, J. Gao, and J. Paisley, "Topicrnn: A recurrent neural network with long-range semantic dependency," arXiv preprint arXiv:1611.01702, 2016.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[16] X. Zhu, P. Sobihani, and H. Guo, "Long short-term memory over recursive structures," in International Conference on Machine Learning,2015, pp. 1604–1612.

[17] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," arXiv preprint arXiv:1601.06733, 2016.

[18] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," arXiv preprint arXiv:1605.05101,2016.

[19] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using lstm for region embeddings," arXiv preprint arXiv:1602.02373, 2016.

[20] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional lstm with two-dimensional max pooling," arXiv preprint arXiv:1611.06639, 2016.

[21] http://qwone.com/~jason/20Newsgroups/