Fall 2018

Matthew Dobbin

Northwestern University

MSDS 458 Artificial Intelligence

Assignment 3 – Recurrent Neural Networks

Contents

1.0 Introduction

This report discusses the implementation and performance of several neural network models for the purpose of classifying text. The architectures of the models include a multilayer perceptron network, several recurrent neural network models (SimpleRNN, Long Short Term Memory and Gated Recurrent Unit) and a one dimensional convolutional neural network model. The computational requirements and classifying accuracy of the networks are compared as well as discussing how each type of model is learning to extract features from the inputs. All tables and images referenced in the report are contained within the Appendix.

2.0 Reuters Data Set

The classification models in the report were fitted on the Reuters data set which can be imported from Keras. The data contains a set of newswires published by Reuters in 1986 which have been categorized into 46 different topics. It contains 8982 training samples and 2246 test samples. The text data has been pretokenized, however the newswire can be converted back into text using a word index. For computational reasons, the imported newswire data was limited to the 10,000 most frequently occurring words. An example newswire is shown in Figure 1.

3.0 One-Hot-Encoding

One-hot-encoding is a common way to turn tokens into a numeric vector so that it is in a format that can be processed by the neural network. The newswires in the Reuters data set have been pretokenized at the word level. The resulting one-hot-encoded vectors are sparse, high dimensional and are hard coded.

3.1 Multilayer Perceptron Model

The first model that was fitted was a multilayer perceptron (MLP) model that used one-hot-encoding. It has two hidden layers and was run for 10 epochs. The structure of the model is shown in Figure 2. The classification accuracy for the training and validation data sets was 0.94

and 0.79 respectively. The performance has been graphed in Figure 3 and it appears the model starts over fitting after approximately 5 epochs.

4.0 Word Embedding

Word embedding is an alternative method to one-hot-encoding for associating numeric vectors with tokens. In contrast to vectors obtained from one-hot-encoding, word embeddings are learned from the data, are low dimensional and are dense. All the models in the following section were trained for 10 epochs and the maximum length of the sequences was restricted to 500 words due to computational processing limitations.

4.1 Multilayer Perceptron Model (EMLP)

The second MLP model that was fitted used word embedding instead of one-hot-encoding. Except for the embedding layer, the model structure, which is shown in Figure 4, is the same as the first MLP model. The classification accuracy for the training and validation data sets was 0.95 and 0.69 respectively. There is a difference of 10% between the accuracy of the validations sets for the two MLP models. This difference may be due to their not being enough training data for the model to learn an appropriate task-specific embedding of the vocabulary (Chollet, 2017). One way to test this hypothesis would be to use a pretrained word embedding such as Word2vec or GloVe and determine if the classification accuracy improves.

4.2 Simple Recurrent Neural Network Model

The second model to be fitted that used word embedding was a simple recurrent neural network (SimpleRNN). Recurrent neural networks (RNN) have internal memory. RNNs consider the current input as well as what it has learned from the inputs it received previously. Due to this internal memory, RNNs are the preferred algorithms for sequential data such as speech, text and other times series data (Donges, 2018).

The architecture of the SimpleRNN model is shown in Figure 6. The classification accuracy for the training and validation data sets was 0.94 and 0.47 respectively. The validation score of 0.47 is poor compared to the word embedding MLP model of 0.69. The poor performance is most likely due to the major issue with the SimpleRNN architecture. It is difficult for SimpleRNNs to learn long term dependencies due to the vanishing gradient problem. The next two models have architectures that have resolved this limitation and the classifying accuracy should improve.

4.3 Long Short-Term Memory Model

Long short-term memory (LSTM) model is a type of recurrent neural network that overcomes the vanishing gradient problem. LSTM networks consist of memory blocks instead of neurons. The forget gate, input gate and output gate are the three types of gates contained within a memory unit. Each memory "unit is like a mini-state machine where the gates of the units have weights that are learned during the training procedure" (Brownlee, 2016). The structure of the model consists of several stacked LSTM layers and is shown in Figure 8. The classification accuracy for the training and validation data sets was 0.66 and 0.60 respectively. A plot of the model's performance is shown in Figure 9 and it could potentially be improved if the model was trained for more epochs. The LSTM model validation accuracy of 0.6 was an improvement from the SimpleRNN validation accuracy of 0.47.

4.4 Gated Recurrent Unit Model

The Gated recurrent unit model works using the same principal as the LSTM but they are more streamlined. Due to this they are less computationally expensive but there is a trade off with classifying performance (Chollet, 2017). The architecture for the model is shown in Figure 10. The classification accuracy for the training and validation data sets was 0.71 and 0.59

respectively. The validation set accuracy was 0.01 less than the LSTM model and it took approximately 60% of the LSTM time to train.

4.5 One Dimensional Convolutional Neural Network Model

One dimensional convolutional neural networks (1DCNN) can be used on certain sequence processing problems such as text classification and time series forecasting. In some cases, they can be competitive with RNNs and are considerably less computationally expensive. The architecture for the model is shown in Figure 12. The classification accuracy for the training and validation data sets was 0.86 and 0.73 respectively. The model performance is graphed in Figure 13 which shows that the model started over fitting at approximately 6 epochs. Its accuracy could potentially be improved by adding dropout layers.

5.0 Model Comparisons

Comparisons of the computational performance and the accuracy of each of the models are shown in Table 1. The models that took the shortest time to train were the MLP models which took approximately 20-60 seconds to train. In comparison, the LSTM was the most computational expensive taking almost two hours to train. As expected, the GRU model took less time to train than the LSTM model but its classifying accuracy was slightly less. The comparatively good performance of the first MLP model in terms of accuracy and computational time highlights the importance of creating a simple baseline model.

For the models that used word embedding instead of one hot encoding, the 1DCNN had the best accuracy of 0.73. It was decided to see if the performance of the LSTM could be improved by using a 1DCNN to pre-process the features of the sequence before feeding them into the LSTM.

5.1 1DCNN LSTM

An advantage of combining a 1DCNN with an RNN is that the 1DCNN can extract down sampled higher-level features from a long sequence and then feed it into an RNN that couldn't have realistically processed the long sequence. A final model was implemented that uses this architecture and it is shown in Figure 14.

The classification accuracy for the training and validation data sets was 0.72 and 0.65. The performance has been graphed in Figure 15 and it appears that the model could be improved if it continued training after the 10 epochs. The validation set accuracy of 0.65 is less than the 1DCNN score of 0.73. However, it is higher than the LSTM score of 0.6 while only taking 38 minutes to train.

6.0 Conclusions

This report reviewed several different modelling techniques that can be used to classify text data. The results have shown that a simple fully connected model outperformed the recurrent neural network models. This highlights the need for data scientists to be familiar with a wide variety of techniques and to tailor their approach depending on the characteristics of the data set.

References

Brownlee, J. (2016). Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras. Retrieved November 16 2018, from https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/

Chollet, F., & Safari, an O'Reilly Media Company. (2017). *Deep Learning with Python*(1st ed.). Manning Publications.

Donges, N. (2018). Recurrent Neural Networks and LSTM, Retrieved November 16 2018, from https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5

Appendix

Encoded Newswire

```
[1, 577, 9, 355, 430, 21, 4, 2222, 5, 4, 83, 181, 73, 418, 171, 1694, 3048, 202, 4596, 11, 15, 6, 750,
4193, 35, 7, 4, 121, 273, 94, 160, 4, 248, 409, 60, 5, 73, 418, 8, 348, 3048, 430, 202, 4073, 11, 15,
6, 566, 158, 35, 577, 910, 335, 6125, 333, 32, 1019, 35, 15, 6, 4459, 557, 35, 232, 218, 377, 563, 55,
772, 6, 30, 1177, 21, 712, 17, 12]
```

Corresponding Decoded Newswire

```
? commercial and industrial loans on the books of the 10 major new york banks excluding acceptances fe
ll 572 mln dlrs to 64 297 billion in the week ended march 11 the federal reserve bank of new york said
including acceptances loans fell 475 mln dlrs to 65 16 billion commercial paper outstanding nationally
increased 2 98 billion dlrs to 339 00 billion national business loan data are scheduled to be released
on friday reuter 3
```

Figure 1 - Example Reuters newswire (16th in training data set).

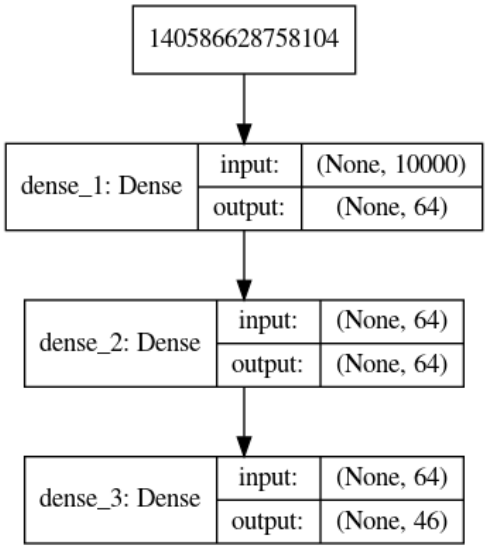

Figure 2 - Architecture for the MLP model.

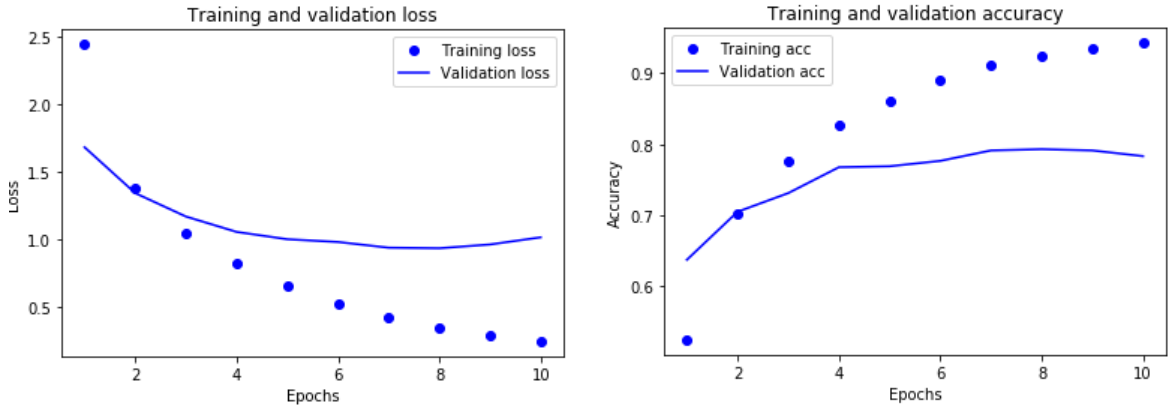

Figure 3 - Training and validation performance for multilayer perceptron neural network.

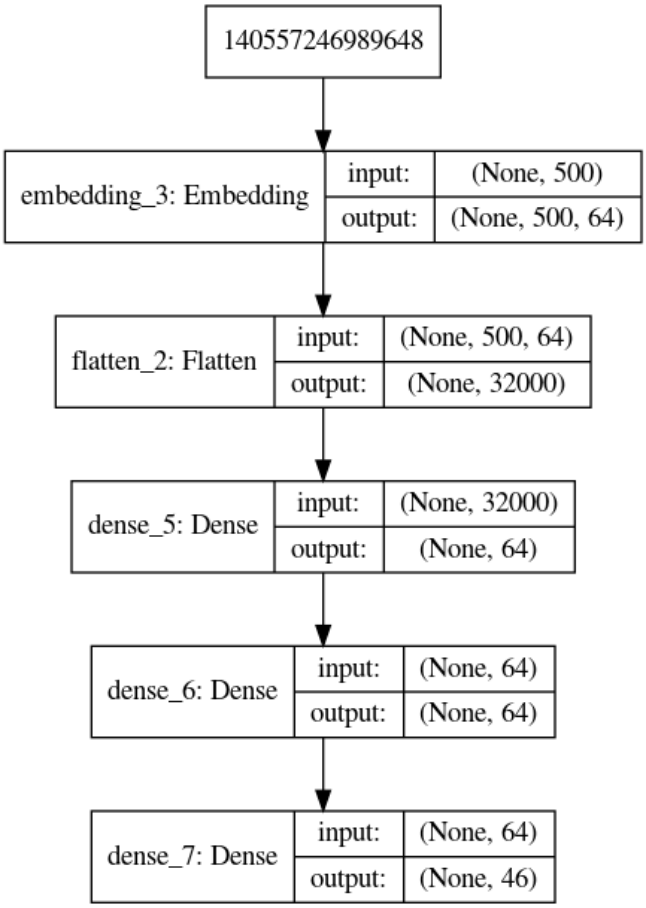

Figure 4 - Architecture for the MLP model that uses word embedding.

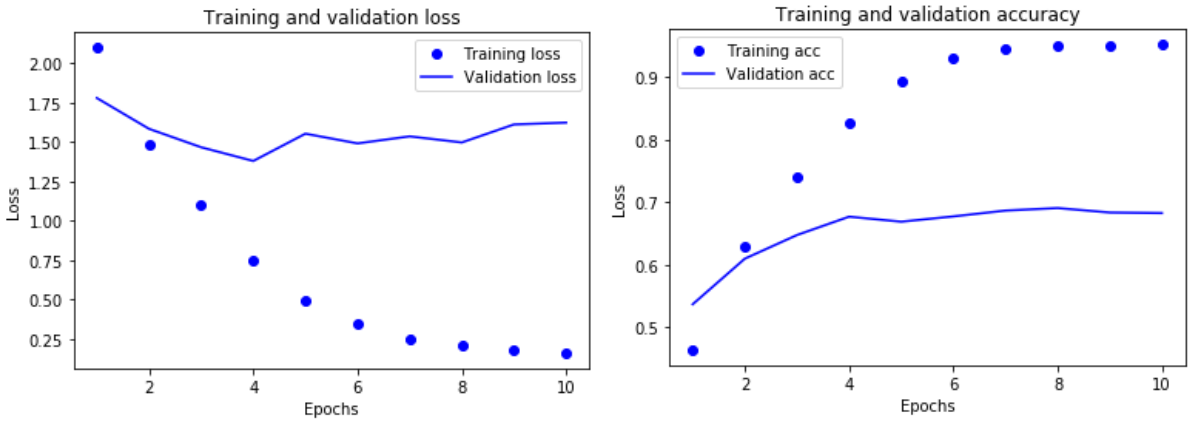

Figure 5 -Training and validation performance for MLP model that uses Embedded layer.

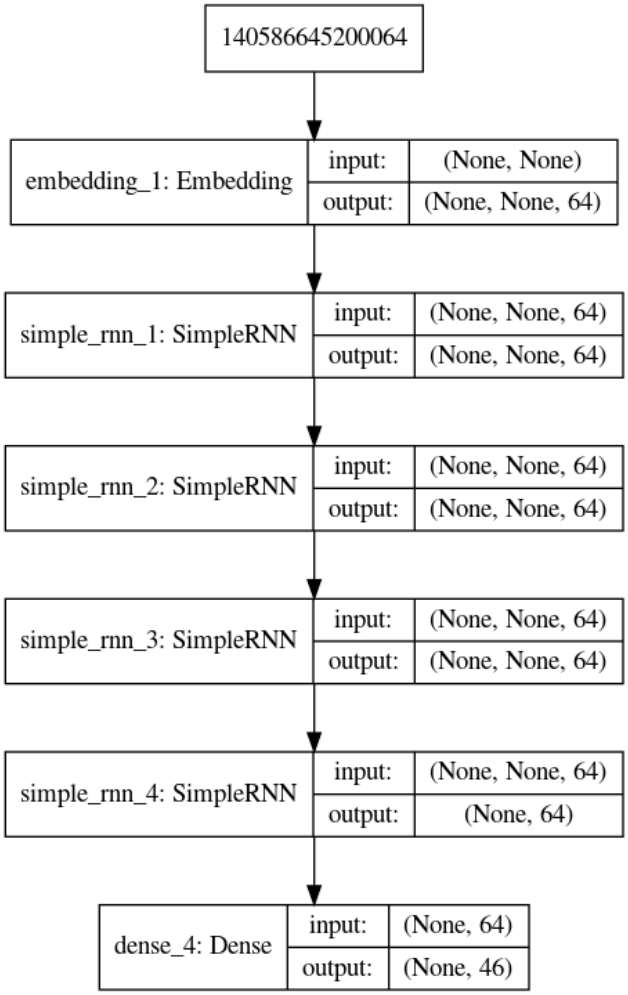

Figure 6 - Architecture for the SimpleRNN model.

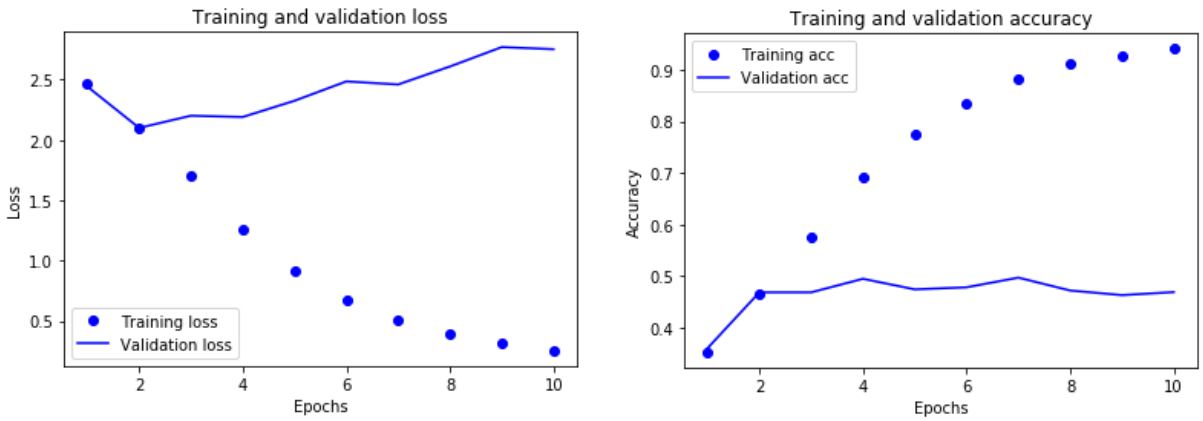

Figure 7 - Training and validation performance for SimpleRNN model.

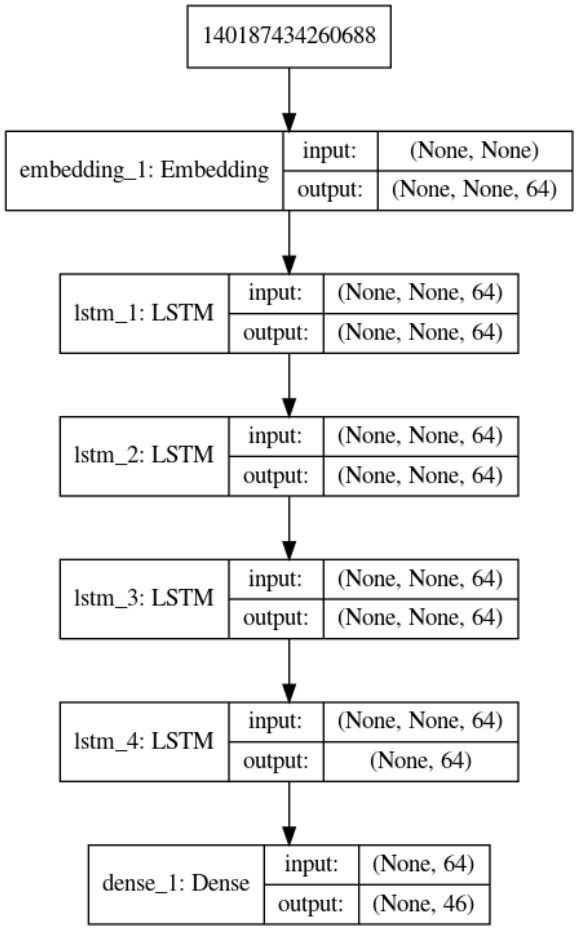

Figure 8 - Architecture for the LSTM model.

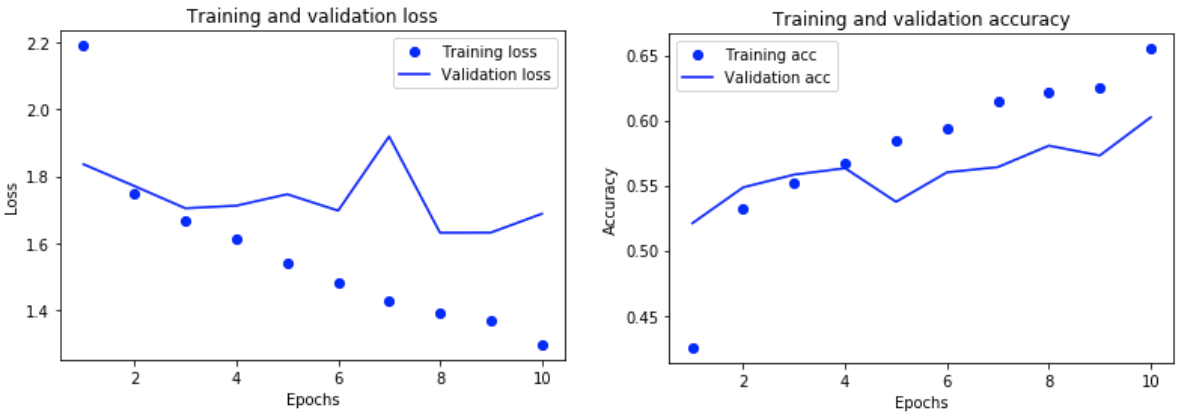

Figure 9 - Training and validation performance for LSTM model.

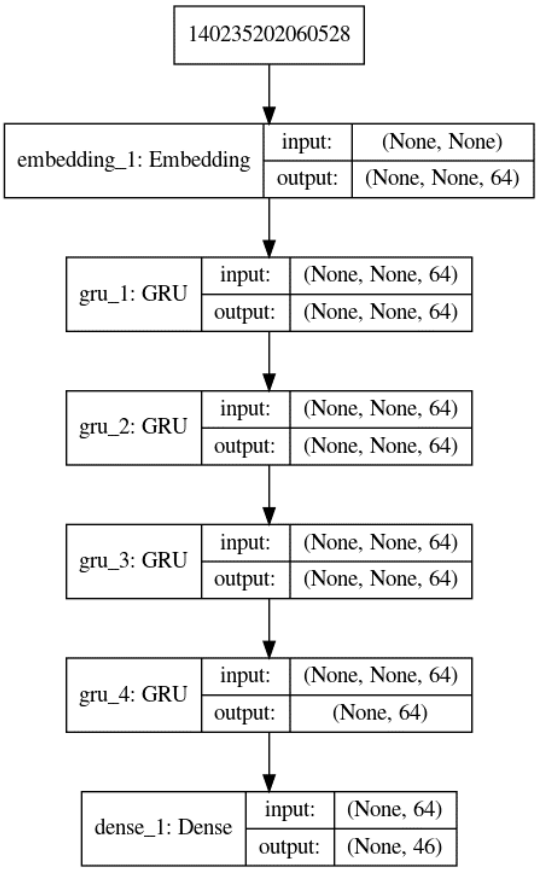

Figure 10 - Architecture for the GRU model.

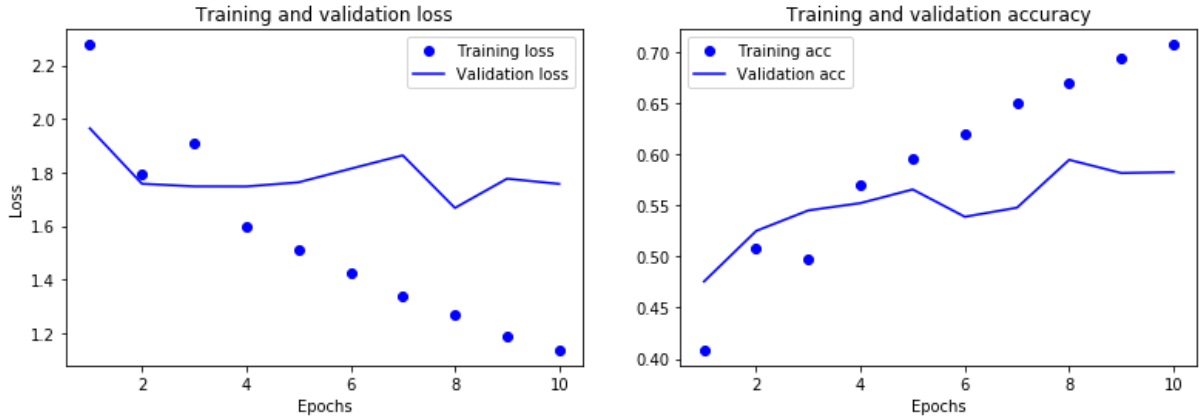

Figure 11 - Training and validation performance for GRU model.

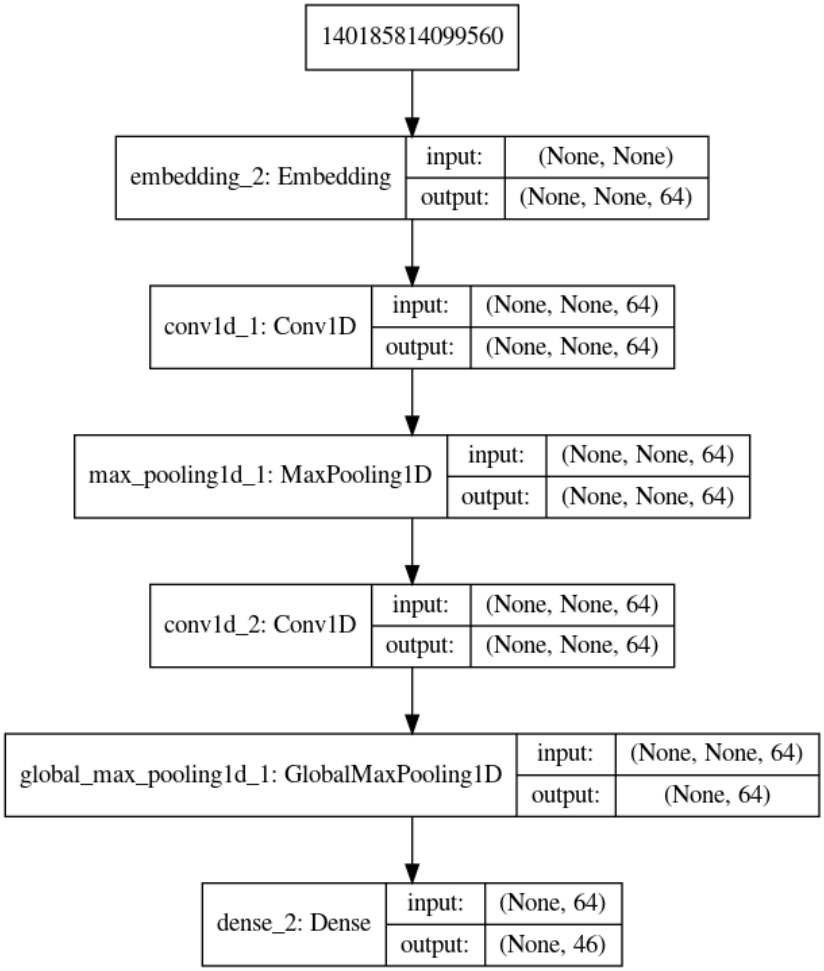

Figure 12 - Architecture for the 1DCNN model.

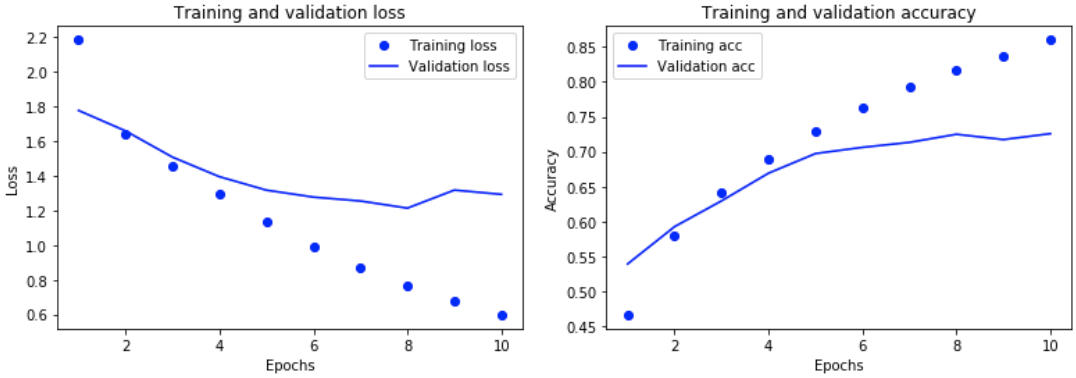

Figure 13 - Training and validation performance for 1DCNN model.

Table 1 - Comparison of overall performance and computational time for each network architecture.

Model	Computational Time	Training Set Accuracy	Test Set Accuracy
MLP	19s	0.94	0.79
EMLP	1min 6s	0.95	0.69
SimpleRNN	13min 12s	0.94	0.47
LSTM	1hr 56min	0.66	0.60
GRU	1hr 5min	0.71	0.59
1DCNN	28min 29s	0.86	0.73

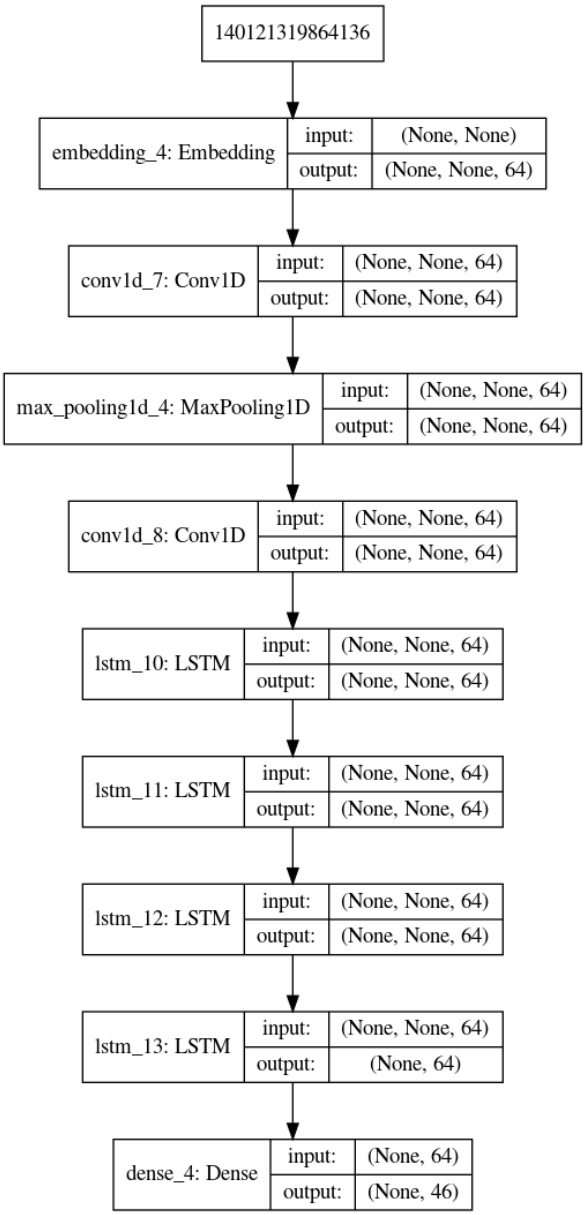

Figure 14 - LSTM model combined with 1D convolutional neural network.

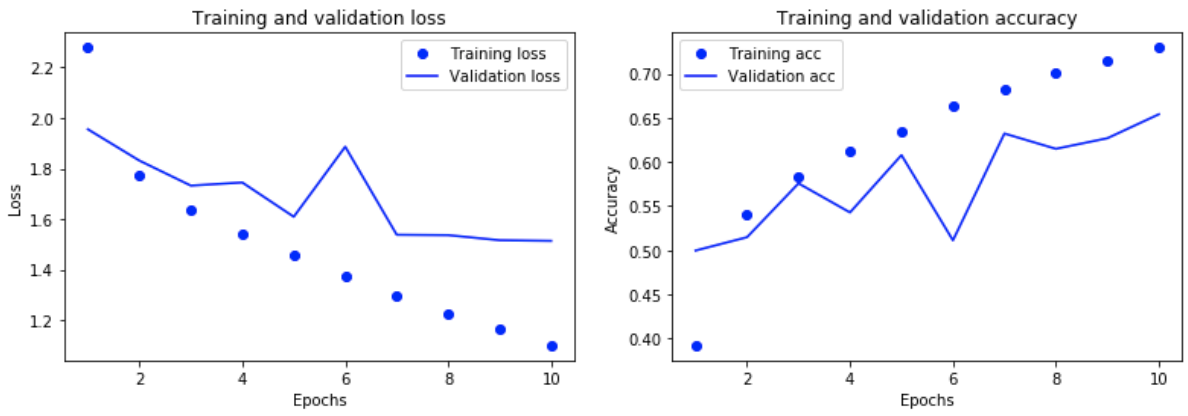

Figure 15 - Training and validation performance for 1DCNN combined with LSTM.