CSCE 479/879 Homework 2: Sentiment Analysis with Sequential Models

Anh Vo, Junxiao Zhang, Nate Thach, Richard Mwaba

March 13, 2022

Abstract

We developed two sequential neural network models for the IMDB movie review classification dataset and determined which of the two emotions, negative or positive, the reviews belonged to. We implemented both long shot-term memory (LSTM) and gated recurrent unit (GRU) architectures and investigated the performance of these two architectures with added attention layer. The dense layer neurons and adam regularizer learning rate are used as hyperparameters to investigate their effects on the models. We can conclude that the learning rate can greatly affect the accuracy of the model. However, the number of neurons in the hidden layer did not have a big impact on the accuracy. Also suprisingly, GRU model had a longer training time compared to LSTM model.

1 Introduction

The main purpose of this project was to apply recurrent neural networks RNN) to the problem of sentiment analysis of movie reviews. The model predicts sentiments in form of movie reviews in to two categories: positive and negative. We developed two architectures with attention mechanism to address this issue, with one being a bidirectional long short-term memory(LSTM) and the other a bidirectional gated recurrent unit (GRU). In addition, we apply k-fold cross validation and perform hyperparameter tuning in order to observe the effects of chosen hyperparameters on model accuracy. Further, we analyze of models' performances based on validation accuracy, and select the best model. The best model recorded an accuracy of 87.06% and (0.1192, 0.1397) confidence interval on 95% generalization error.

2 Problem Description

The problem of interest in this assignment is to perform sentimental analysis, particularly binary sentiment classification, on the IMDB dataset [4] (accessed via TensorFlow Datasets). The dataset consists of 50,000 movie reviews labeled

as positive (1) or negative (0). Each of the training and testing sets has 25,000 reviews. The reviews are highly polarized, in which the positives have score $\geq$ 7 out of 10, while the negatives have score $\leq$ 4 out of 10. In other words, the dataset does not include neutral reviews. Furthermore, the dataset contains an even number of positive and negative reviews. We can assert that the dataset is highly suited for our binary classification problem. Our goal then is to train a sequential model robust enough to correctly classify these reviews, which could potentially help movie producers save a good amount of time going through feedbacks from viewers.

3 Approaches

We developed two architectures (Figure 1) for the IMDB dataset: gated current unit (GRU) and long-short term memory (LSTM. GRU is a simplified verion of LSTM where short-term memory and long-term memory are merged. Both architectures utilize attention mechanism, drop-out regularity. Besides observing how varying learning rate and number of neurons in hidden layer will affect the training time and accuracy in each model, we also observed how accuracy and training time are traded-off between LSTM and GRU.

We applied Luong-style attention [3] in all architectures, which is an improvement over the Bahdanau attention [1].

4 Experimental Setup

Our initial task to set up this experiment was to understand the data that we were working with. The imdb dataset contains 50000 movie reviews split evenly into 25000 train and 25000 test sets. It is available at [4] and was downloaded through tensorflow datasets.

Post loading the data, we split the training data into four equal folds in preparation for k-fold cross validation. This resulted into four distinct sets of data, such that each set consisted a unique tuple of 75% of training data, and 25% of validation data. We further shuffled each set of data and created batches, each of size 64, for training. Considering the dataset contains text reviews, our models would not make great use of the data in its raw format. We therefore converted the sequence of words into integers where each word was represented by a discrete integer. This is generally referred to as text vectorization. Also, text vectorization accounts for the varying word sequence lengths by padding the shorter sequences with zeroes so that all representations are of the same length. However, in this new representation, we lose the relationships that exist among different words. For example, our model would not be able to determine from the integer representation that "love" and "like" are more similar than "love" and "hate" [5]. To re-establish these relationships, we applied word embedding in which similar words are trained to have similar embedded representations.

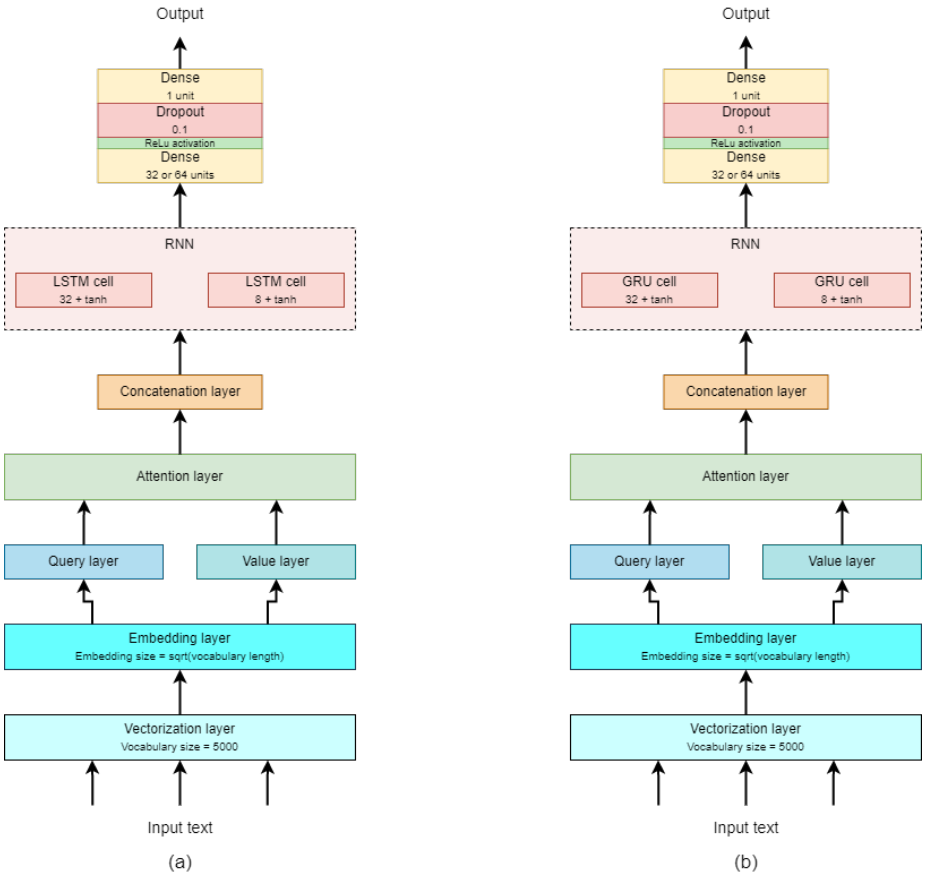

Figure 1: (a) LSTM architecture and (b) GRU architecture. Each block represents the layer as labelled. In the LSTM and GRU cells, the numbers and text below the labels represent the number of units and activation function applied respectively. In the dropout layers, the value below the label represents the dropout rate applied.

Table 1: Hyperparameters tuned while training model

Hyperparameter	**Values**
Neurons	32 and 64
Learning rate	0.01, 0.001

In the training process, we applied k-fold cross validation while tuning hyperparameters to extensively train the models, enabling us to pick with great confidence, the best model to generalize well on unseen data. The hyperparameters tuned were number of neurons in the dense hidden layer and the learning rate for the optimizer of choice, adam. The values used for each of the hyperparameters is shown in Table 1. For each of the two implemented architectures, we trained a model consisting of all the unique combinations of the hyperparameters, resulting in a total of 8 trained models shown in Table 2. In addition, we applied k-fold ($k = 4$) cross validation on each model while keeping track of the average train and validation accuracies of all the folds. The average accuracies for each model were considered as the representative accuracies for that model, and used in the performance comparisons. We also recorded the training time for each model (Table 2).

To choose the best model, we evaluated the performance of each model on the validation accuracy. We chose the model with the highest validation accuracy as the best model. Finally, we validated the best model by applying it on the unseen test data, and recorded the resulting prediction accuracy as a representation of the overall performance of the model.

5 Experimental Results

Table 2 lists all 8 trained models with different combinations of architecture, learning rate and number of neurons in the hidden layer. Despite having a tie between two models, we picked the second model as the best model with a validation accuracy of 0.8594. This is because it had a lower variance between the training and validation accuracy. We applied this model on the unseen test data and recorded the final test accuracy of 0.8706 (87.06%) with 95% generalization error confidence interval of (0.1192, 0.1397). Figure 2 shows the resulting normalized confusion matrix.

Table 2: Result of 8 models, with best model highlighted.

Model No.	Model Name	Learning rate	Hidden Dense Units	Train Accuracy	Valid Accuracy	Training Time
1	LSTM-Attention	0.001	32	0.8918	0.8438	2886.2223 secs
2	LSTM-Attention	0.001	64	0.8694	0.8594	2884.5005 secs
3	GRU-Attention	0.001	32	0.9011	0.8594	3735.2053 secs
4	GRU-Attention	0.001	64	0.8990	0.8125	3754.9640 secs
5	LSTM-Attention	0.01	32	0.5172	0.5312	2945.9520 secs
6	LSTM-Attention	0.01	64	0.5052	0.5000	2899.4235 secs
7	GRU-Attention	0.01	32	0.5171	0.5000	3720.7129 secs
8	GRU-Attention	0.01	64	0.5448	0.5625	3748.4688 secs

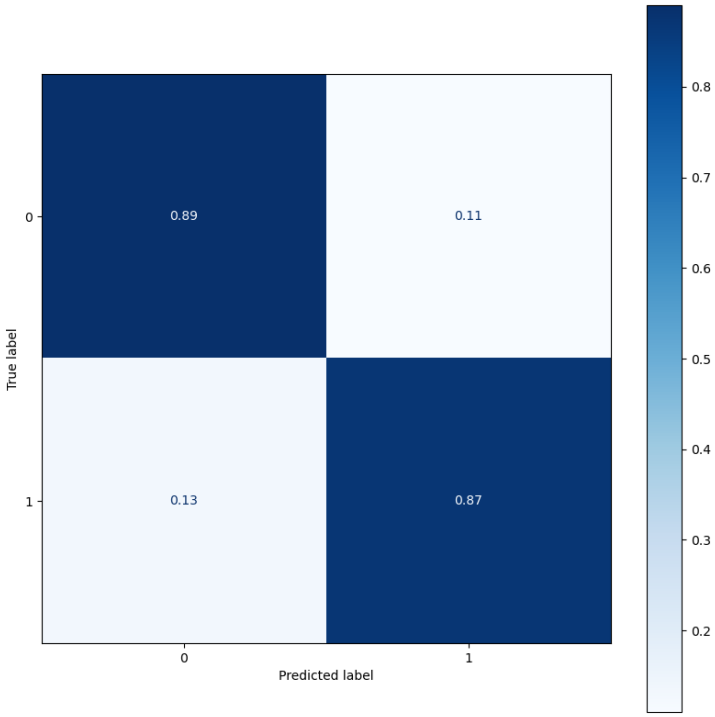

Figure 2: Confusion matrix

6 Discussion

In general, the models performed fairly well as can been seen in Table 2. All the models achieved a validation accuracy greater than 50%. Something apparent right away is the similarity in performance among closely related model configurations. This was expected considering the two implemented architectures (Figure 1) were almost identical, with the major difference being the LSTM and GRU cells applied in the RNN layer of each architecture. The best model which is based on the LSTM architecture has the same validation accuracy as the "second" best model, based on the GRU architecture.

Despite the observed similarities in performance, a significant difference is noted in the training time for each model. LSTM architecture based models commonly retain a shorter training time compared to the GRU based models. This is a very interesting discovery as we expected GRU to outperform LSTM as highlighted in literature [6, 2]. The best models further show this, where the LSTM model trained 56% faster than the GRU model.

For the learning rates, 0.001 generally performs well as shown in Table 2. Learning rate 0.01 produced accuracies 30% lower which hinted to the gradient not converging. This was expected because a larger value causes the gradient to hop around the curve, possibly skipping the minima, and hence leading to non-convergence.

For the number of units in the hidden dense layer, there seems to be no clear distinction in performance between 64 and 32. Generally, a considerably low number of neurons is used to prevent the network from becoming a memory bank that fails to generalize well on unseen data. However, the number shouldn't be very small such that we are unable to learn the weights in the network. A more comprehensive hyperparameter tuning would be needed to determine the right number of neurons. In our case, we can conclude that the number of neurons in the single hidden layer of the fully connected layer had no significant impact on the models' performance.

In both the architectures, we implemented a dropout layer as a regularization strategy to prevent overfitting. A closer look at the differences in training and validation accuracy for each trained models suggests that the models did not experience any overfitting. Models 1, 3 and 4 display variances of 5% to 9% between the training and validation accuracy show, nonetheless, this is not significant enough to demonstrate overfitting. Since we did not implement any architecture without a dropout layer, it is inadequate to fully asses the impact of the applied dropout in overcoming overfitting. Further investigation would be required to draw a well informed conclusion.

Let us now look at the confusion matrix (Figure 2) of our best model. We can see that the model had a balanced performance on both classes (as shown by the darker blues). This can be attributed to the k-fold cross validation applied during the training of models as well as the balance in the data. For each training run, the models were trained on different distributions of data according to the fold under scrutiny. This allowed the models to learn the variances in all data points and therefore avoid any form of bias towards one class of data.

Considering the discussed findings, we will consider implementing a control architecture in future assignments to give us a better perspective on the effects of certain hyperparameters . This will also be useful in drawing more informed conclusions.

7 Conclusions

In this problem, our main goal was to develop sequential models for sentiment analysis of movie reviews on IMDB dataset and classify those reviews in to positive and negative. We designed and implemented two architectures, LSTM and GRU, and applied the attention mechanism on both architectures. We trained a total of 8 models models resulting from these architectures and a variation of two sets of hyperparameters (Table 1). Also, we applied k-fold cross validation in the process of training which can reduce the accuracy bias of the result of the models. After evaluating the accuracy and the training time of the models, we inferred the best model (model 2) on the test set and recorded an accuracy of 87.06% and (0.1192, 0.1397) confidence interval on 95% generalization error. One surprising finding is the LSTM model training 56% faster than the GRU model as opposed to our expectations and literature [6, 2]. This is something that would require further investigation. We also observed that there was not much of a difference in the validation accuracy among models of the two architectures having the same hyperparameters. As a result, we cannot fully conclude which architecture has a better performance. However, based on the current experiment, LSTM based models performed relatively better than GRU based models.

Lastly, our future research direction could focus on exploring more architectures to draw a fair comparison in performance between LSTM and GRU. In addition, we would like perform a more comprehensive hyperparameter tuning by incorporating grid search, random search or Bayesian optimization. Based on that, we can choose the appropriate model architecture by combining model run-time and accuracy.

Table 3: Contributions by team member for this assignment.

Team Member	**Contribution**
Nate Thach	Coding, section 2
Junxiao Zhang	Abstract, section 1 and 7
Anh Vo	Section 3
Richard Mwaba	Coding, sections 4, 5 and 6

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[2] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 2342–2350, Lille, France, July 2015. JMLR.org.

[3] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[4] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[5] Eleanor Quint. Intro-Deep-Learning-Notebooks, February 2022. original-date: 2019-03-14T00:50:31Z. URL: `https://github.com/DrKwint/Intro-Deep-Learning-Notebooks/blob/bafa8cd94aa6c3550c7b8da09a47795dabf8b750/2020S_hackathons/hackathon6.ipynb`.

[6] Shudong Yang, Xueying Yu, and Ying Zhou. LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. In *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pages 98–101, June 2020. `doi:10.1109/IWECAI50956.2020.00027`.