ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING - 6CS012

COURSEWORK 02:

A TEXT CLASSIFICATION WITH RECCURENT NEURAL NETWORK AND ITS VARIANT

“SENTIMENT ANALYSIS OF HOTEL REVIEWS”

Student Name: Saroj Devkota

University ID: 2329255

Group: L6CG6

Module Leader: Mr. Siman Giri

Module Tutor: Mr. Shiv Kumar Yadav

Cohort: 09

Submission Date: 5/15/2025

ABSTRACT

This coursework focuses on sentiment analysis on hotel reviews with the purpose of predicting a rating of the customer from 1 to 5, using deep learning models including Recurrent Neural Networks (RNN), Long Short-Term Memory Networks (LSTM) and the LSTM with pretrained Word2Vec embeddings. The purpose here is to accurately categorize the sentiment in customer feedback for the purpose of obtaining useful business insights.

Along with the high disparity between classes present, manual methods, such as oversampling of underrepresented classes were used to boost the performance of models. Bidirectional architectures were used for better representation of contextual meaning of reviews and the models were made to learn from the preceding and succeeding words of the texts. The collection of methods before training included tokenization, sequence padding, and stop words elimination to make it appropriate for training.

Three models were established and result of evaluation demonstrated that the most accurate one was the LSTM model with Word2Vec embeddings. A graphical user interface (GUI) was used as well to make real time sentiment prediction, which demonstrated the practical use of the models. The results indicate the superiority of LSTM over the usual RNNs, and further improvements may include expanding the dataset to increase generalization.

TABLE OF CONTENT

1. INTRODUCTION

In companies that depend on the feedback received from their customers; the sentiment analysis tasks, for instance, assessing hotel reviews using a method known as Natural language processing (NLP), are quite important. This project attempts to categorise hotel reviews to ratings ranging from 1 to 5 using deep learning models such as SimpleRNN, LSTM, and LSTM with Word2Vec embeddings.

The dataset used for the analysis was also installed from Kaggle and contained 515,000 reviews from 1,493 five stars hotels in Europe. Cleaning of data involved standard data preprocessing steps such as, converting text to lower case, removing punctuation, tokenizing and stopping words.

The study discussed a variety of models such as traditional machine learning methods such as Logistic Regression and Decision Trees, as well as deep learning models such as BiLSTM and GRU. The reviews were put in three forms.

1. 10-class (individual ratings from 1 to 10)
2. 3-class (Negative: 1–4, Neutral: 5–7, Positive: 8–10)
3. 2-class (Negative: 1–5, Positive: 6–10)

The training was performed on a system that had 2 GB of GPU. According to the results, the LSTM model gave the maximum performance in the 2-class setup performance with 97% accuracy-ness, precision-ness, recall-ness, and F1-score of 81.69%. It beat other models such as BiLSTM and Logistic Regression, indicating how LSTM performed well in binary sentiment classification.

2. DATASET

Link: https://www.kaggle.com/datasets/andrewmvd/trip-advisor-hotel-reviews

The dataset used at this project is publicly available at Kaggle (Source: and it was built by Larxel. It contains 20,491 records having two key columns. “Review” and “Rating.” 13,501 characters is the maximum length of review text. Distribution of ratings of the dataset is shown in the following figure:

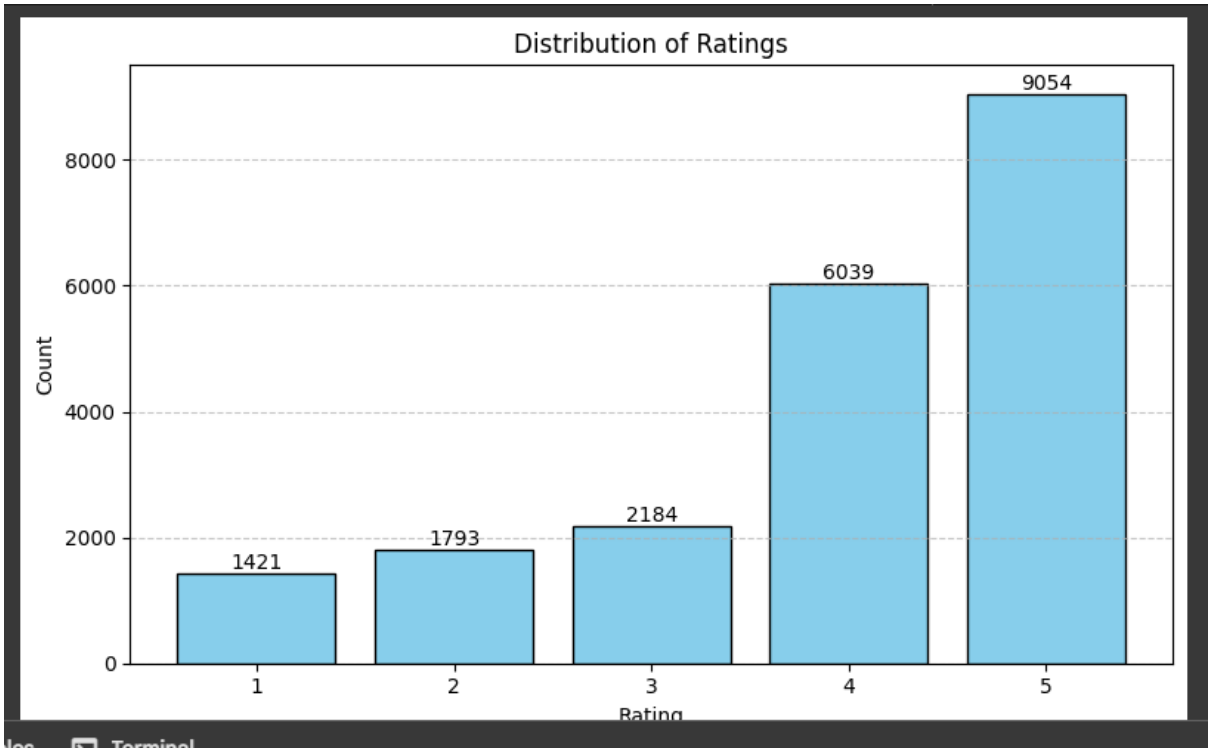


Figure 1 Distribution of Ratings

For purposes of model evaluation, the dataset was divided in two whereby 80% was used in training and 20% were used for testing. Due to imbalance in data along the rating classes, such as oversampling and class weights were employed to pay adequate attention to under-represented classes during the training process.

3. METHODOLOGY

In order to rectify the class imbalance in the dataset, while training the model, we used two different classifications.

- Multi-class classification (1–5 ratings): In this method, the model outputs one of the five possible rating classes; it is achieved with SoftMax activation function in the output layer.
- Binary classification: In here, the ratings were regrouped as two categories; negative and positive categories: ratings 1 and 2 were marked as negative while ratings 4 and 5 were as positive. Such a binomial setup enabled the model to narrow down on identifying clearly negative and positive sentiments.

3.1 TEXT PREPROCESSING

The preprocessing workflow consisted of a number of major steps to prepare the text data to be ready for model training:

- The uniformity involves changing all the texts to lower case.
- The elimination of URLs, user mentions, hashtags, numbers, and special characters to get rid of noises.
- Extending contractions to give a clearer picture.
- Stripping off the stop words while retaining the ones which have sentiment value
- Lemmatization which is reduction of words to their base or root form.
- Tokenizing the text in order to separate the text into single words or tokens
- Padding sequences to make inputs of uniform length for all the reviews

3.2 MODEL ARCHITECTURE

3.2.1 SIMPLE RNN

The first model that is used is a SimpleRNN, which starts with an embedding layer to convert input text to vector form. This enables the model to capture the semantic relationships or meanings existing between words. After the embedding layer, a SimpleRNN layer with 32 units is employed in learning contextual information with regards to the sequence based on past and future dependencies. For better generalization and avoiding overfitting, the algorithms use regularisation methods (L2 regularisation and dropout). The output layer is a dense layer with SoftMax activation that allows classifying the reviews to one of five rating categories.

For the binary classification approach, such an architecture is used with some changes. It employs unidirectional RNN of 64 units that includes a sigmoid activation function in the output layer and binary cross-entropy as a loss function to find out the difference between negative and positive sentiment.

3.2.2 LSTM

The second model that has been used is Long Short-Term Memory Network that has similar embedding layers to SIMPLE RNN. This architecture relies on internal memory and gating to learn context from the irrelevant information, which is contrary to Simple RNN. This model also employs utilization of Bidirectional architecture and using forward and backward direction to process the input. The vector representation from the embedded layer is fed to the bidirectional LSTM layer of 32 unit and the results are classified applying through SoftMax activation. LSTM in the two-approach model uses the architecture of Simple RNN.

3.2.3 LSTM WITH WORD2VEC

The third models are an extension of the second model with an extra LSTM layer with 64 units while holding the same configuration in all other aspect. An already trained Google News Word2Vec model with 300 – dimensional word vector has been used to initialize the embedding layer. This approach will contribute to the model using rich semantic knowledge and enhancing performance.

All three models were trained on a categorical crossentropy loss function and Adam optimizer with learning rates set as 0.0003 and 0.00005 respectively for the first two and the third model. The training was started with 32 batches within 20 epochs with callbacks such as Early Stopping and ReduceLROnPlareau. LSTM with Word2Vec also uses similar architectures in two class approaches as used in the previous two models.

4. EXPERIMENTS AND RESULTS

4.1 RNN VS LSTM PERFORMANCE

Comparison between the two models reveals huge differences with regards to performance, generalization and training efficiency. The training accuracy curve is a continuous upward trend and surpasses 80% but the validity accuracy is almost flat and cannot go beyond 60%. The same trend is visible in the loss curves training loss changes steadily, but validation loss begins at about 1.2 % and keeps this value pretty stable, suggesting possible overfitting.

In the case of the LSTM model, we can see only slight improvements as compared to the SimpleRNN, and slightly higher values of validation accuracy and loss. Apparently, the most critical reason for the overfitting is the high imbalance of data, namely a small number of samples in minority classes. Although the methods of oversampling and regularization are used, the issue of the gap in class distribution continues to be the difficulty that the models fail to address.

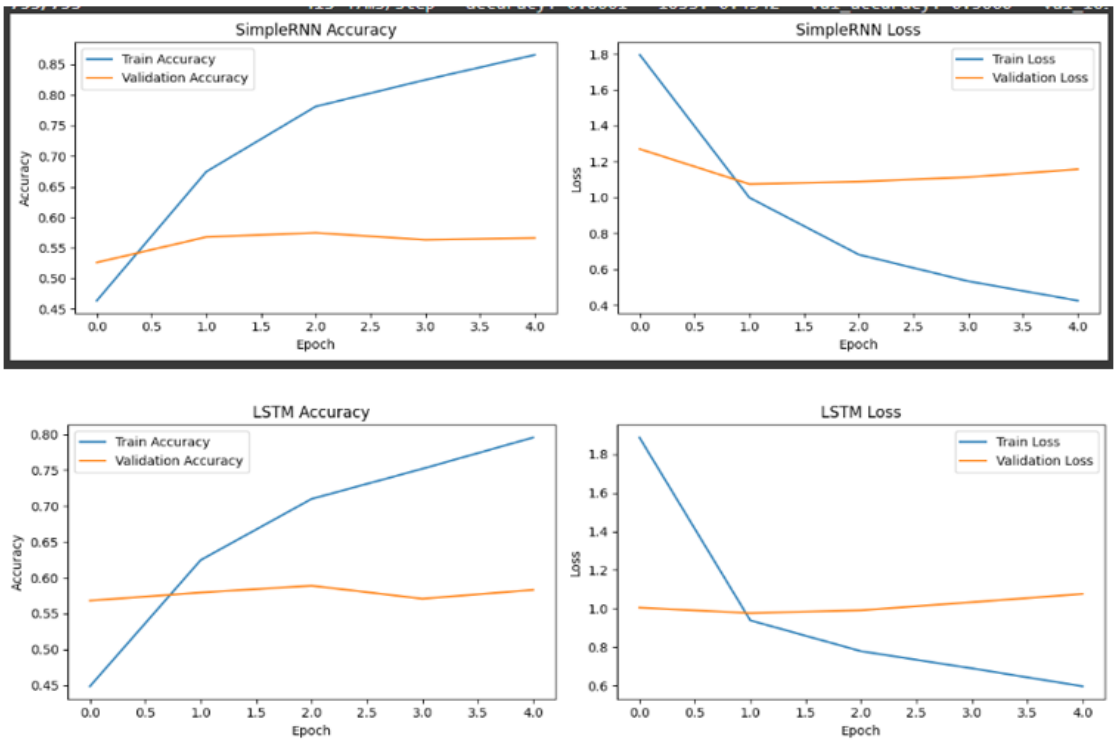


Figure 2 Five Classes Training and Validation Curve

In the context of binary classification approach, the comparison of performances is evident in a way that the LSTM model performs better than RNN model. The training and validation curves for LSTM show stable and consistent learning process, instead of cyclical behaviour

occupying the RNN model. LSTM is able to attain a validation accuracy of about 90%, while the RNN model manages to do just over 70%. Moreover, LSTM has lower loss values than RNN, which implies a better overall performance.

Contrary to the comparatively complex architecture of LSTM, both models consumed about the same amount of time to train because LSTM converged faster during trainings.

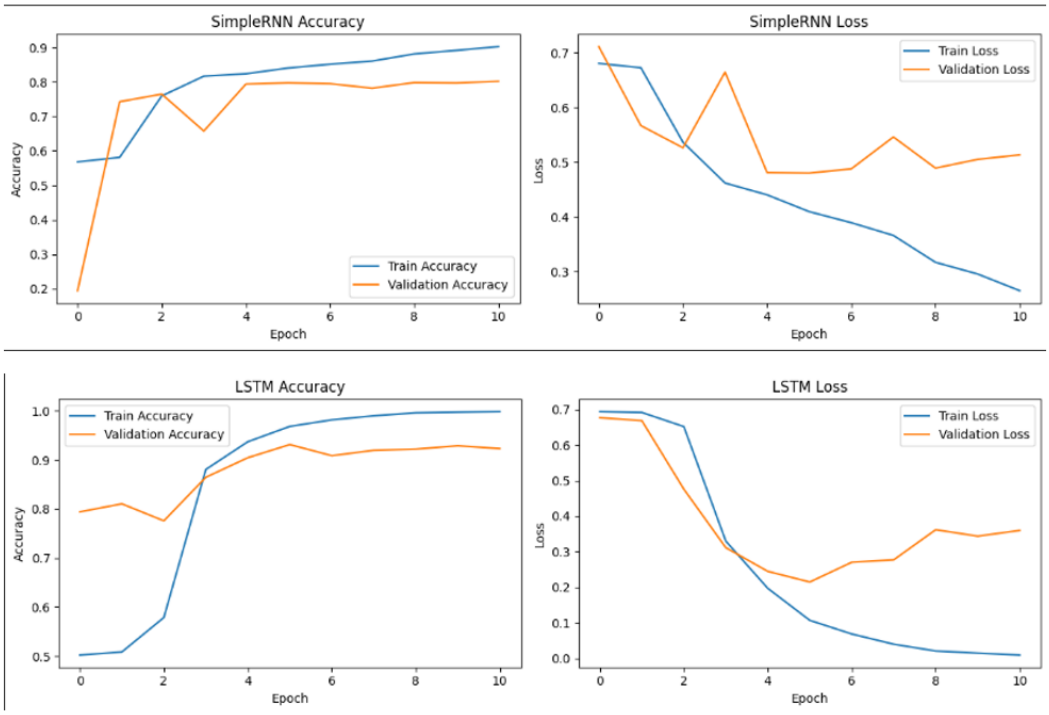


Figure 3 Two Classes Training and Loss Curve

4.2 COMPUTATIONAL EFFICIENCY

Computationally, text classification was not costly when it comes to use of resources because of small datasets. Computation of the training of the models occurred using Google Colab T4 GPU, which is a hardware acceleration for a higher speed of process. In this setup, RNN and LSTM model were trained in 2 minutes, LSTM with Word2Vec in 5 minutes.

4.3 TRAINING WITH DIFFERENT EMBEDDINGS

In the five-class classification task, the results of the LSTM and LSTM with Word2Vec models are not so different only slightly. It has better loss optimization in LSTM with Word2Vec embedding, and acquired better 61% accuracy rate, when compared to 59% of the standard LSTM models. However, the LSTM model is more stable in training with the consistent

learning curves, and the Word2Vec-enhanced model has unstable curves with some minor fluctuation although it was more accurate.

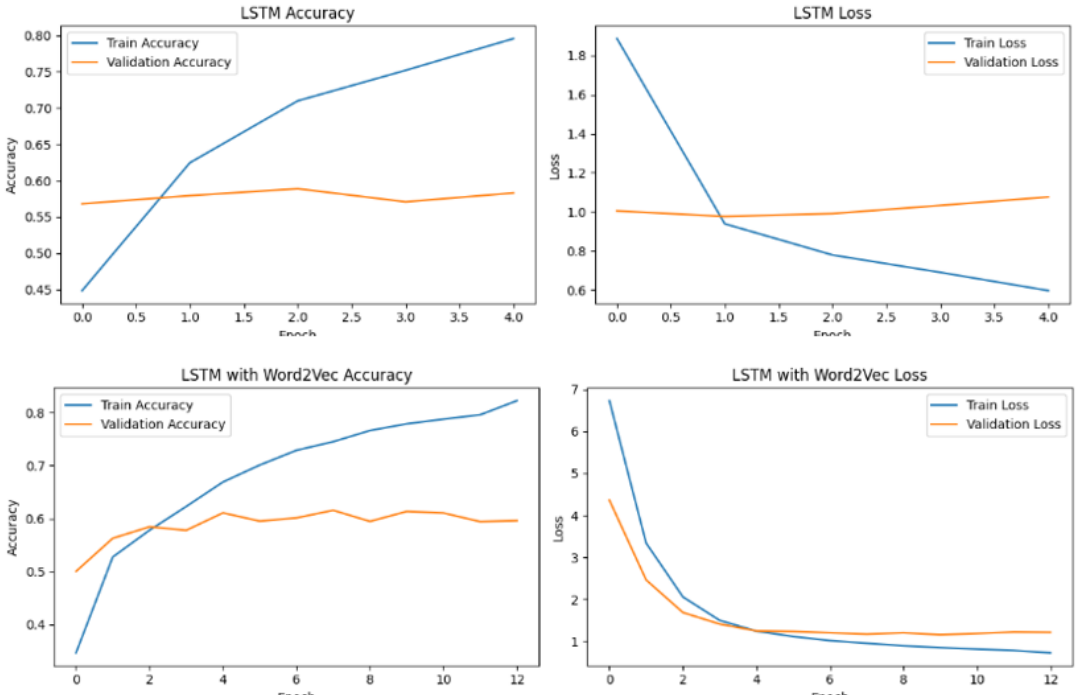


Figure 4 Two Classes LSTM vs LSTM Word2Vec

On a cursory look, the outcome of the LSTM and LSTM with Word2Vec models in the binary classification task looks similar, as both give an accuracy of 93%. A closer glance, though, shows that the Word2Vec-augmented LSTM has better training stability. Furthermore, according to the classification report, the Word2Vec model is better than the standard LSTM in terms of such evaluation metrics as precision, recall, F1-score, which means that the predictions are more trustworthy and balanced. On the whole, the improvement noted in performing word2vec training is that they are stable and minimize loss better than LSTM.

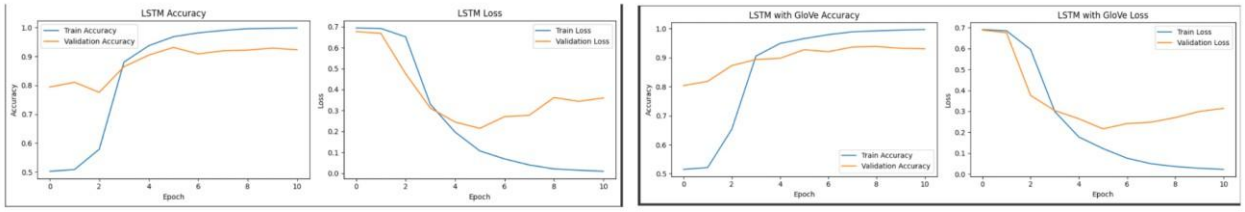


Figure 5 Two Classes LSTM vs LSTM Word2vec

4.4 MODEL EVALUTATION

Out of all the six models, (3 from five classes classification and 3 from two classes classification), word2vec model performs superiorly in terms of accuracy as compared to other models.

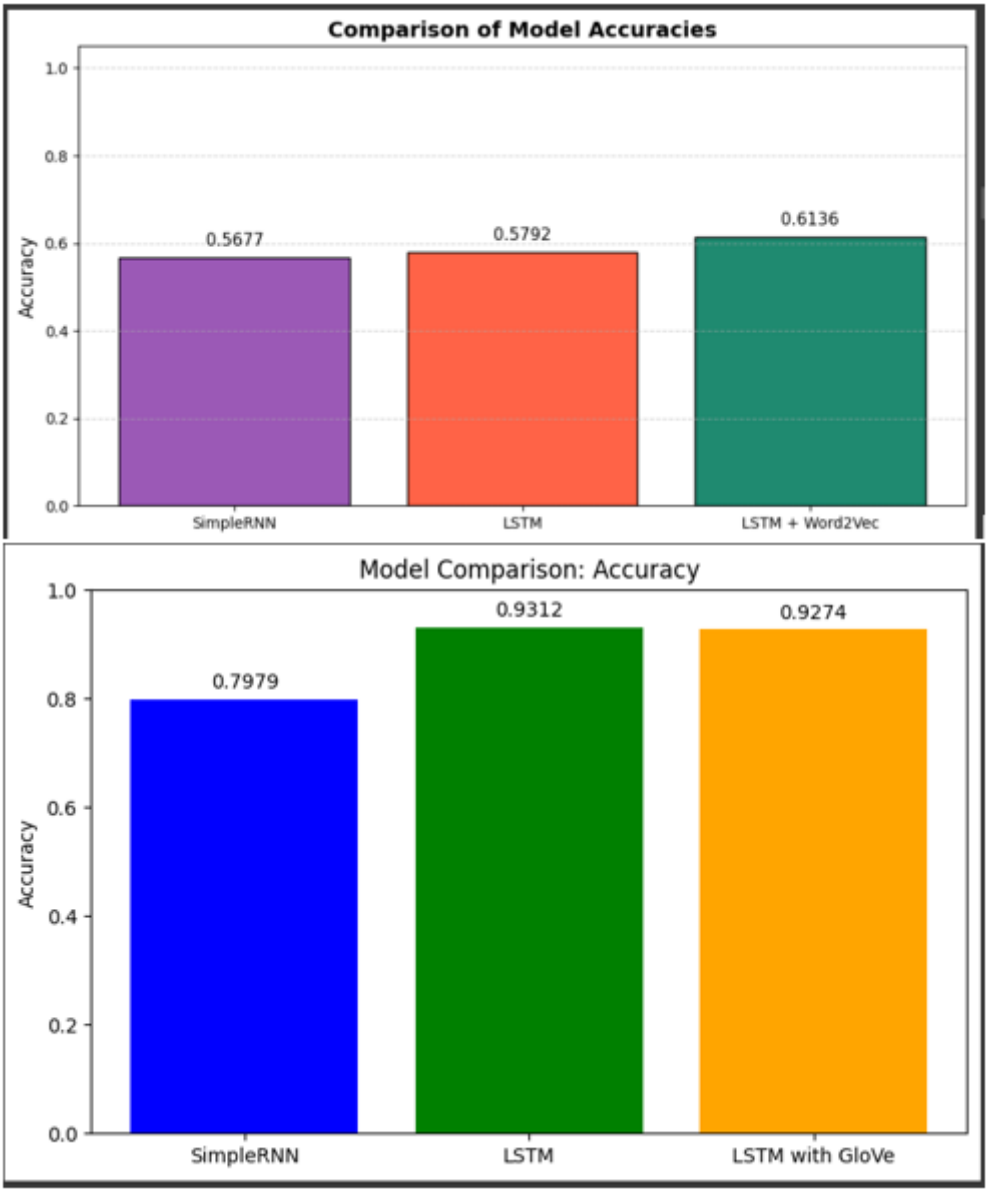
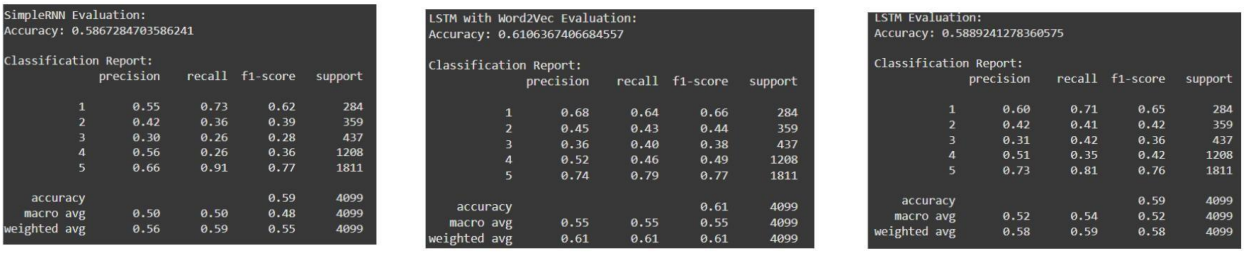


Figure 6 All Six Model Accuracies

According to the performance of the models in five classes computing from a sample of 4,099, the performance of the models differs. The SimpleRNN attains a fair accuracy of 0.5867 on a macro average F1-score of 0.48. A little better performance is confirmed by the LSTM model, which provides accuracy of 0.59 and macro average F1-score of 0.52. The LSTM with Word2Vec model performs the best of all the three models with accuracy at 0.61 and a macro average F1-score of 0.55.



SimpleRNN Evaluation:
Accuracy: 0.5867284703586241

Classification Report:

	precision	recall	f1-score	support
1	0.55	0.73	0.62	284
2	0.42	0.36	0.39	359
3	0.30	0.26	0.28	437
4	0.56	0.26	0.36	1208
5	0.66	0.91	0.77	1811
accuracy			0.59	4099
macro avg	0.50	0.50	0.48	4099
weighted avg	0.56	0.59	0.55	4099

LSTM with Word2Vec Evaluation:
Accuracy: 0.6106367406684557

Classification Report:

	precision	recall	f1-score	support
1	0.68	0.64	0.66	284
2	0.45	0.43	0.44	359
3	0.36	0.40	0.38	437
4	0.52	0.46	0.49	1208
5	0.74	0.79	0.77	1811
accuracy			0.61	4099
macro avg	0.55	0.55	0.55	4099
weighted avg	0.61	0.61	0.61	4099

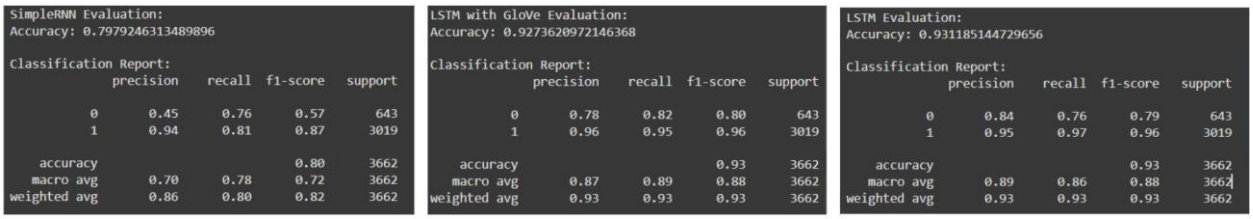
LSTM Evaluation:
Accuracy: 0.5889241278360575

Classification Report:

	precision	recall	f1-score	support
1	0.60	0.71	0.65	284
2	0.42	0.41	0.42	359
3	0.31	0.42	0.36	437
4	0.51	0.35	0.42	1208
5	0.73	0.81	0.76	1811
accuracy			0.59	4099
macro avg	0.52	0.54	0.52	4099
weighted avg	0.58	0.59	0.58	4099

Figure 7 Classification Report of Five Classes

The binary classification task classification report is significantly better than the five-class machine. For the RNN, the accuracy is equal to 0.80 and is accompanied by macro average F1-score, which is equal to 0.80 as well. LSTM and LSTM with Word2Vec models demonstrated identical results with an accuracy of 0.93 and macro average F1-score of 0.88, which is a considerably high performance of the binary classification.



SimpleRNN Evaluation:
Accuracy: 0.7979246313489896

Classification Report:

	precision	recall	f1-score	support
0	0.45	0.76	0.57	643
1	0.94	0.81	0.87	3019
accuracy			0.80	3662
macro avg	0.70	0.78	0.72	3662
weighted avg	0.86	0.80	0.82	3662

LSTM with GloVe Evaluation:
Accuracy: 0.9273620972146368

Classification Report:

	precision	recall	f1-score	support
0	0.78	0.82	0.80	643
1	0.96	0.95	0.96	3019
accuracy			0.93	3662
macro avg	0.87	0.89	0.88	3662
weighted avg	0.93	0.93	0.93	3662

LSTM Evaluation:
Accuracy: 0.931185144729656

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.76	0.79	643
1	0.95	0.97	0.96	3019
accuracy			0.93	3662
macro avg	0.89	0.86	0.88	3662
weighted avg	0.93	0.93	0.93	3662

Figure 8 Two classes Classification Report

Confusion matrices for SimpleRNN, LSTM, and LSTM with Word2Vec draw out the differences of each of them when it comes to the five-class classification task. The SimpleRNN attains its best performance with class 5 (1641 correct) and class 4 (704 correct) but does poor with classes 3 and 1 and misclassifies large number of samples. The LSTM model provides an improvement when it comes to identification of more samples within classes 5, 4, and 3, albeit it still appears to mix class 3 and class 4 and class 4 and class 5. The LSTM with Word2Vec is the one that performs best overall, especially performing better in classes 4 and 5, while making fewer mistakes with class 3 and class 1. Finally, the best class separation is achieved by the LSTM with Word2Vec while the worst performance is from the SimpleRNN.

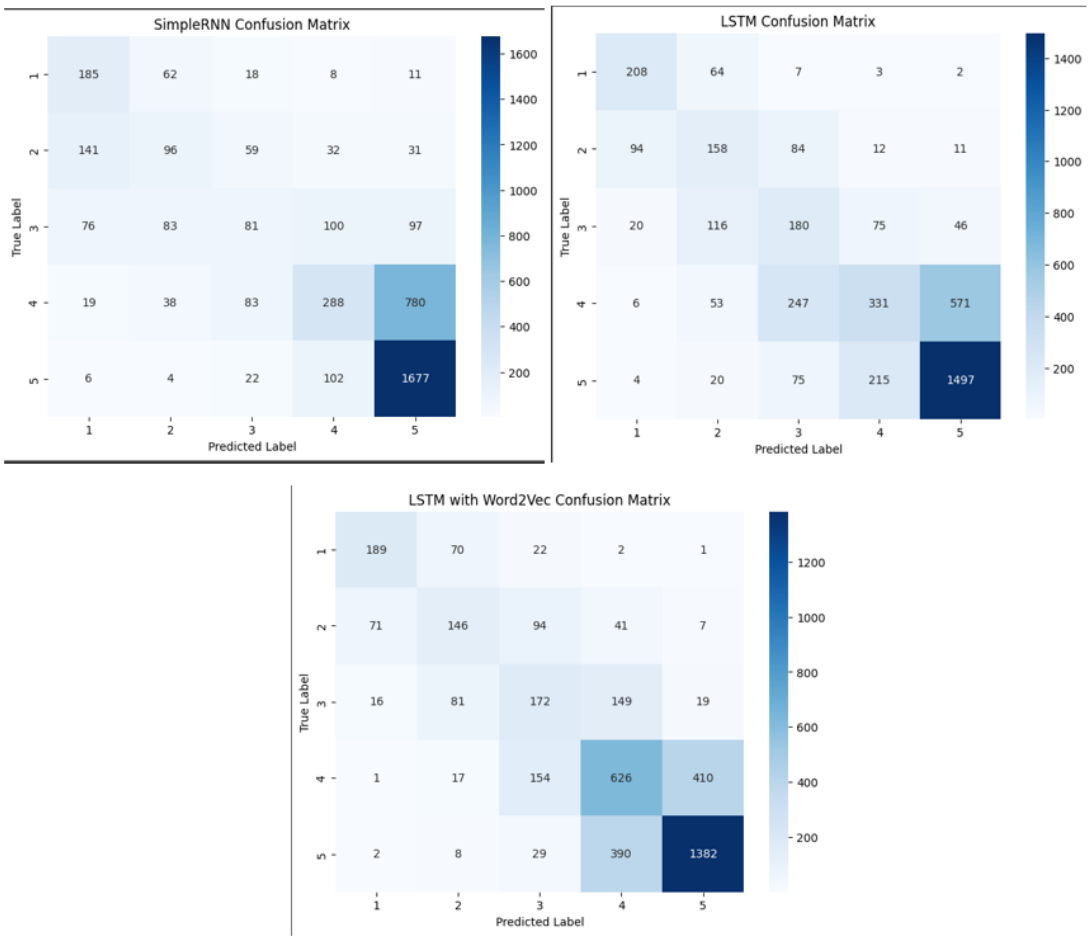


Figure 10 Five Classes Confusion Matrix

Clear distinctions in performance can be seen on the confusion matrices for SimpleRNN, LSTM and LSTM with GloVe on a binary classification task. SimpleRNN classifies 490 class 0 and 2432 class 1 instances correctly but makes a decent number of errors for the class 0, 153 instances, and for the class 1, making 587 mistakes. LSTM performs better with 486 and 2924

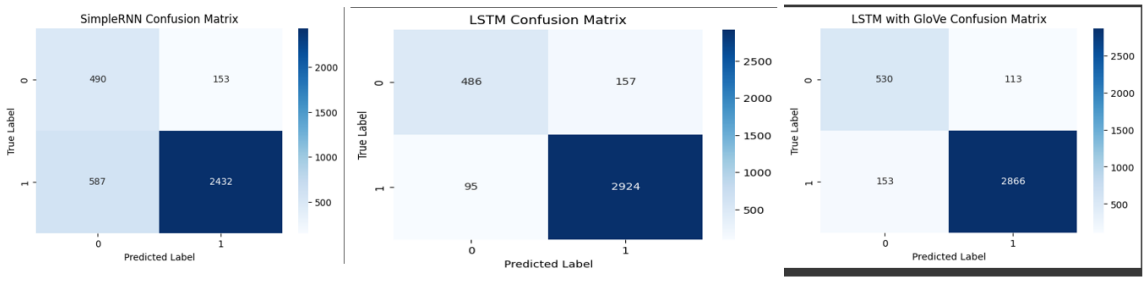


Figure 9 Two Classes Confusion Matrix

correct for class 0 and class 1 respectively and lesser errors. LSTM with GloVe has the best performance with the correct prediction of 530 class 0 and 2866 class 1 instances with

minimum misclassifications. In general, LSTM with GloVe demonstrates the best accuracy and the balance of two classes.

5. CONCLUSION AND FUTURE WORK

In this sentiment analysis project, LSTM with Word2Vec was better than the other ones. It produced 93% of accuracy in binary classification and 61% in five-class classification, beating regular LSTM (93% and 59%) and faring better than the SimpleRNN (80% and 58.7%). LSTM was more stable than SimpleRNN since it was capable of capturing long-term dependencies, as seen in smoother training curves and better validation accuracy (90% – binary classification vs. 70%). Word2Vec was helpful in delivering better results, especially in five-class task while it had a minimum effect on binary classification. Although oversampling was applied to address the class imbalance, overfitting and slow convergence were still the problem. Possible future optimizations may include hyperparameter tuning or an enhancement of the dataset, and possible future directions might be directed at multilingual reviews or real-time business applications. All models were trained easily in a Google Colab T4 GPU that took a training time of 2 to 5 minutes.