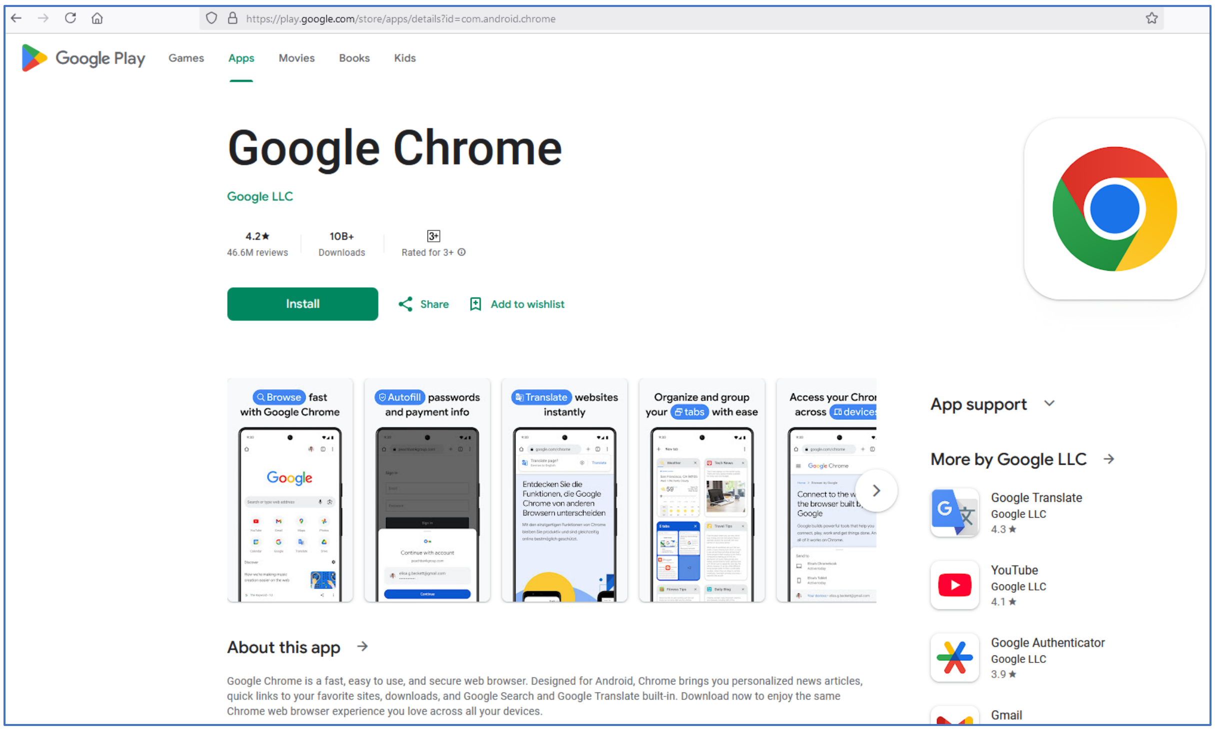
1 Problem 1

1.1 Overview

The objective of the first problem is to build a sentiment analysis model to predict the application review score based on the Google Play Store reviews. The application chosen for problem 1 is the Google chrome browser which has 46.6 million reviews, 10 billion plus downloads and has a rating of 4.2 stars (Figure 1).



Google Chrome is a fast, easy to use, and secure web browser. Designed for Android, Chrome brings you personalized news articles, quick links to your favorite sites, downloads, and Google Search and Google Translate built-in. Download now to enjoy the same Chrome web browser experience you love across all your devices.

Figure 1 – Getting the App ID for Chrome Browser

To collect the review data, we use "**google_play_scraper**", which is a real-time API for crawling the Google Play Store. This allows us to scrape and store the review data into a 'csv' file that consists of 2 columns, a content column which comprise the actual review and a score column which is the rating that was given, ranging from 1(worst) to 5(best).

1.2 Data Loading & Processing

The downloaded dataset, "chrome_reviews_1K.csv", is read into a dataframe. The frequency distribution of the ratings is shown in Figure 2. The rating 5 had the highest number, **103481**, followed by the rating 1, **23717**. Rating 2 has the lowest frequency of **7299**. The dataset is highly imbalanced with reviews skewed towards rating 5.

The implications of training on a dataset with highly imbalanced data can lead to the following issues:

- Biased Predictions: The model tends to predict the majority class more often because it "learns" that this class is more frequent.
- Poor Performance on Minority Class: The model may have high overall accuracy but perform poorly on the minority class, which is often the class of interest.

- Misleading Metrics: Accuracy can be misleadingly high because it reflects the majority class more, masking the poor performance on the minority class.

To mitigate the problem, I will use resampling techniques in particular, oversampling the minority class by adding copies of the under-represented class to the dataset i.e. Random Oversampling. This is achieved through the “imblearn” package, “RandomOverSampler” class.

After performing random oversampling, the frequency of all scores were equalized to **103481**.

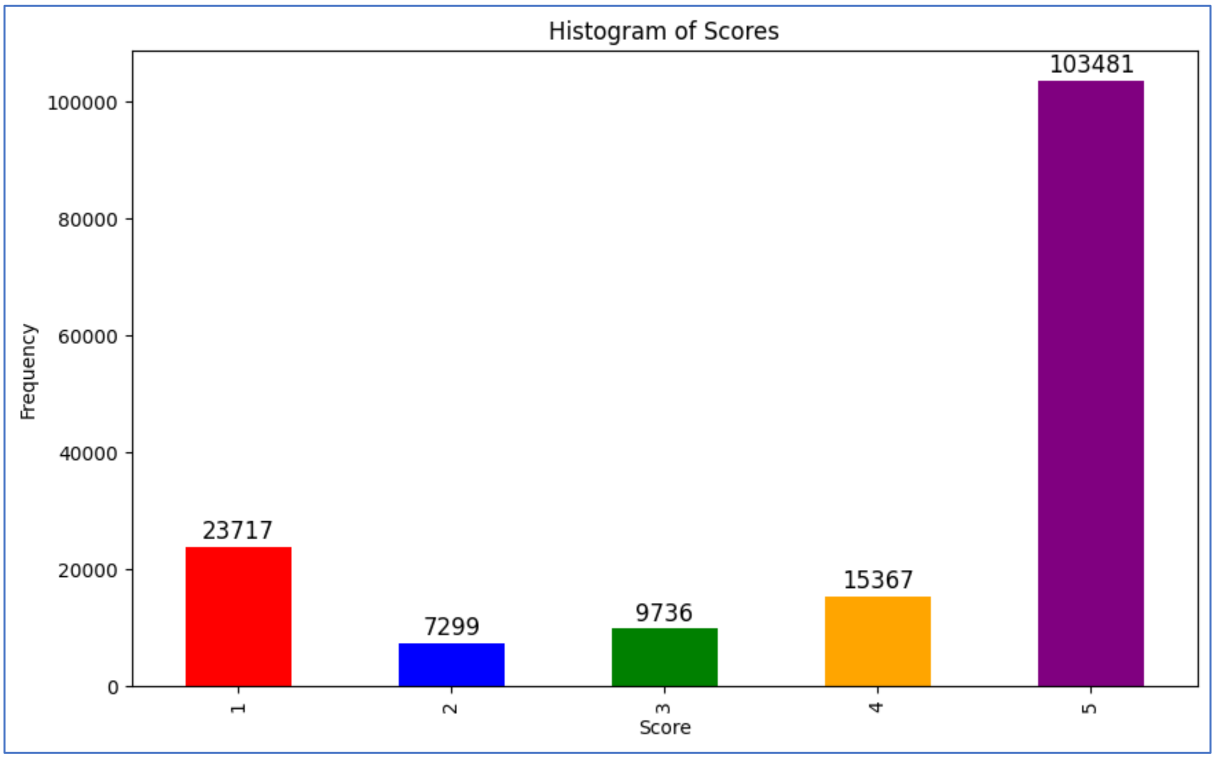


Figure 2 – Frequency distribution for the rating scores

1.3 Data Sampling

We first convert the content and score into numeric tensors (X & y) and also one-hot encodes corresponding labels. The purpose is to prepare textual data and labels for training machine learning models. Split the dataset (X & y) into training set (X_train & y_train) and testing set (X_test & y_test). We use this:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, 
random_state=42)
```

to create a 80:20 split.

1.4 Sentiment Analysis Model Development & Training

Eleven experiments were carried out. They are:

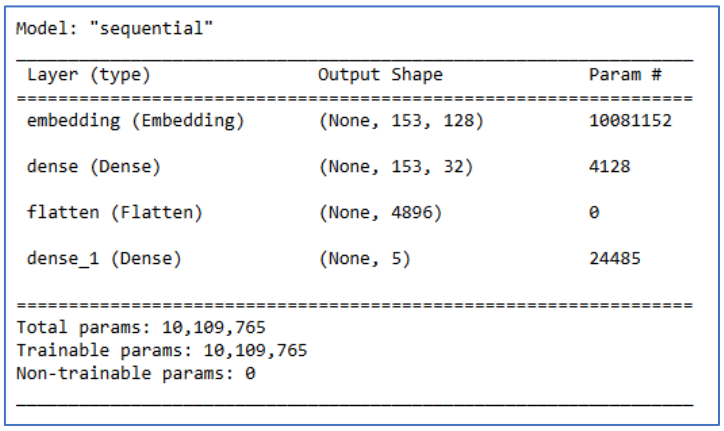
Experiment #	Description
1	Training the word embeddings using the provided dataset, embedding dimension = 128, - # Epochs = 40
2	Using 1D Conv Layer for Sentiment Analysis, embedding dimension = 128, - # Epochs = 25

Experiment #	Description
3	Using 1D Conv Layer for Sentiment Analysis, embedding dimension = 256, - # Epochs = 10
4	Using 1D Conv Layer for Sentiment Analysis, embedding dimension = 64, - # Epochs = 10
5	Using 1D Conv Layer for Sentiment Analysis, embedding dimension = 256, Dropout = 0.3, # Epochs = 10
6	Using 1D Conv Layer for Sentiment Analysis with NO Dropout, embedding dimension = 256, Early Stopping for Sentiment Analysis - # Epochs = 25
7	Using GRU for Sentiment Analysis, embedding dimension = 128, - # Epochs = 25
8	Using GRU for Sentiment Analysis, embedding dimension = 256, - # Epochs = 20
9	Using GRU for Sentiment Analysis, embedding dimension = 256, Dropout 0.3, No recurrent dropout, # Epochs = 20
10	Using GRU for Sentiment Analysis, embedding dimension = 256, No Dropout, recurrent dropout = 0.3, # Epochs = 20
11	Using GRU for Sentiment Analysis, embedding dimension = 256, No Dropout, No recurrent dropout, Early Stopping, # Epochs = 20

Table 1 – Problem 1 Experiments

1.4.1 Experiment 1

For this experiment, we use an embedding layer trained on the dataset downloaded, with an embedding dimension of 128. The summary of the model built is shown in Figure 3.


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 153, 128)          10081152

 dense (Dense)               (None, 153, 32)           4128

 flatten (Flatten)           (None, 4896)              0

 dense_1 (Dense)             (None, 5)                 24485

=================================================================
Total params: 10,109,765
Trainable params: 10,109,765
Non-trainable params: 0
_________________________________________________________________
```

Figure 3 – Experiment 1

In this run, we train the model for 40 epochs. Table 2 shows the training/validation accuracy and loss plots.

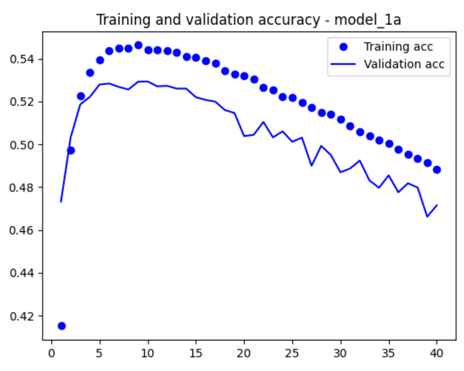
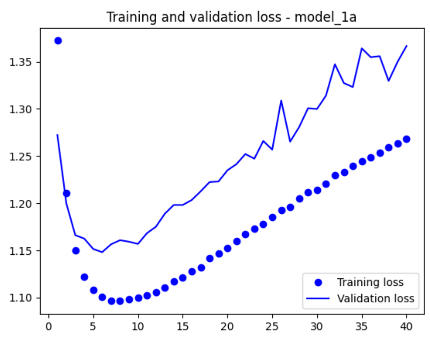
Training/Validation Accuracy	Training/Validation Loss
	

Table 2 – Experiment 1 Training/Validation Accuracy & Loss Curves

Observations

Training Accuracy:

- The training accuracy starts at around 0.42 and increases rapidly, peaking at approximately 0.54 around the 10th epoch.
- Post peak, it gradually decreases, indicating potential overfitting to the training data as the epochs increase.

Validation Accuracy:

- The validation accuracy also starts at about 0.47, peaking around 0.53 around the 10th epoch.
- After peaking, there is a noticeable decline in validation accuracy, suggesting a decrease in the model's generalization capability.
- The validation curve fluctuates more than the training curve, highlighting variability in performance on unseen data.

Analysis

Signs of Overfitting: The divergence between training and validation accuracy post 10th epoch indicates that the model is overfitting.

Generalization Issues: The peak validation accuracy being lower than peak training accuracy is typical, showcasing better performance on the training data.

Variability: The fluctuations in the validation curve suggest inconsistent performance on new data.

Recommendations

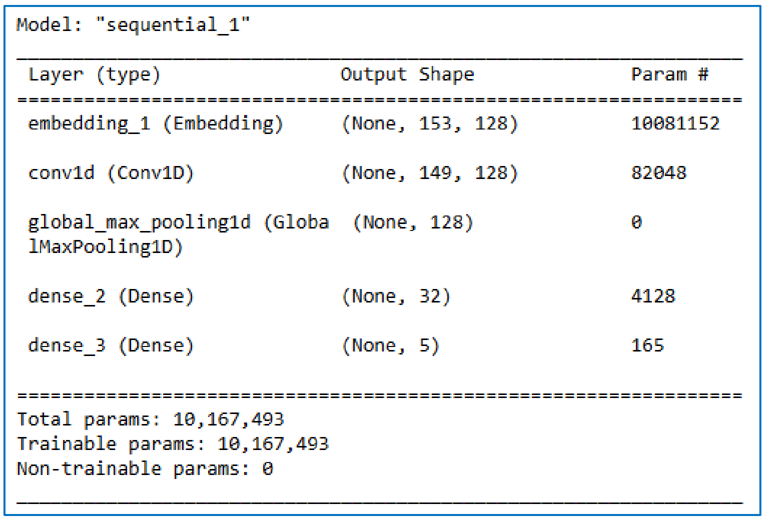
To address these issues, we can consider:

- Early Stopping: Stop training when validation performance stops improving.
- Regularization Techniques: Implement methods such as dropout or weight regularization to avoid overfitting.
- Complex Models: Utilizing more sophisticated models might better capture the underlying patterns without overfitting.

1.4.2 Experiment 2

In this experiment, we build a model using 1D Conv layer for sentiment analysis. There is no dropout used and the embedding dimension was 128. The model summary is shown in Figure 4.

1D Conv is used mainly for sequence data, such as time series, text, or any type of data where the input can be thought of as a sequence of steps or events. It is good for capturing local dependencies and convolution operations can be easily parallelized, leading to faster training times. Local dependencies refer to the relationships or patterns between nearby elements in sequential data.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 153, 128)          10081152

 conv1d (Conv1D)             (None, 149, 128)          82048

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 dense_2 (Dense)             (None, 32)                4128

 dense_3 (Dense)             (None, 5)                 165

=================================================================
Total params: 10,167,493
Trainable params: 10,167,493
Non-trainable params: 0
_________________________________________________________________
```

Figure 4 – Experiment 2

In this run, we train the model for 25 epochs. Table 3 shows the training/validation accuracy and loss plots.

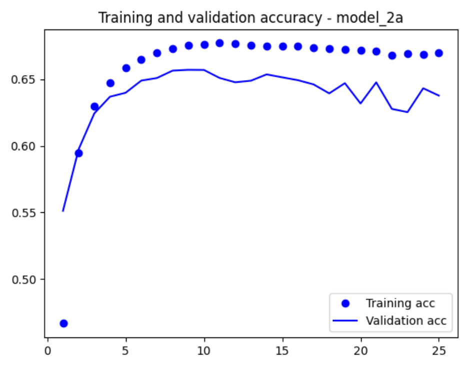
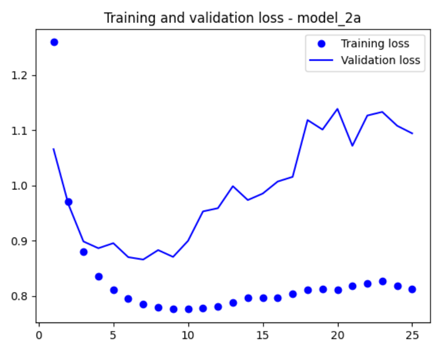
Training/Validation Accuracy	Training/Validation Loss
	

Table 3 – Experiment 2 Training/Validation Accuracy & Loss Curves

Observation and Analysis

Training Loss:

- The training loss starts high, above 1.2, and decreases rapidly in the first few epochs, reaching around 0.8 by epoch 10.

- After this rapid decrease, the training loss stabilizes with minor fluctuations, remaining relatively constant around 0.8.

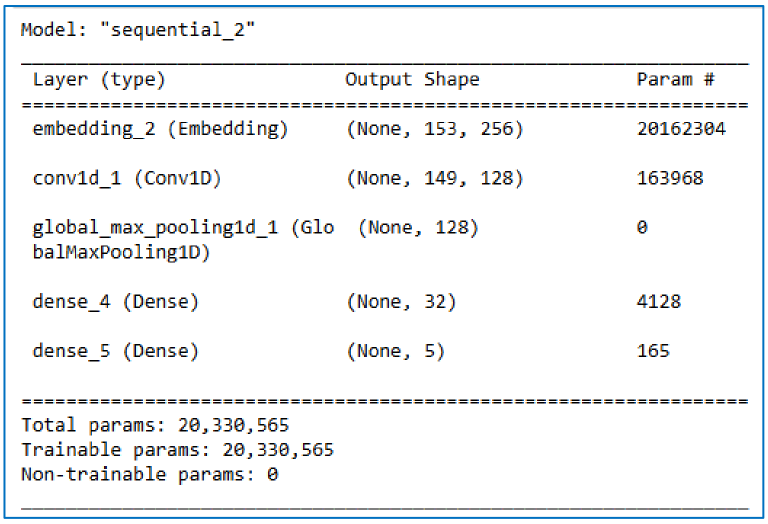
Validation Loss:

- The validation loss also starts high at 1.06, and decreases initially.
- After reaching a minimum around epoch 5, the validation loss starts to increase slightly with noticeable fluctuations.
- By the end of the training period, validation loss is around 1.1.
- The fluctuations in validation loss suggest that the model's performance on the validation set is inconsistent, possibly due to the model's complexity or the nature of the validation set.

In the training/validation accuracy curve, both start at 0.5/0.6 and gradually increase over time. Training/validation accuracy peaks at ~0.67/0.66 at around epoch 10. Post-peak, training accuracy remains relatively stable with minor fluctuations whereas validation accuracy shows some fluctuations around 0.65, potentially decreasing slightly.

1.4.3 Experiment 3

For this experiment, we increase the embedding dimension from 128 to **256**. This model still uses the 1d Conv layer for sentiment analysis. No dropout was used. The model summary is shown in Figure 5.



```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 153, 256)          20162304

 conv1d_1 (Conv1D)           (None, 149, 128)          163968

 global_max_pooling1d_1 (Glo  (None, 128)              0
 balMaxPooling1D)

 dense_4 (Dense)             (None, 32)                4128

 dense_5 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,330,565
Trainable params: 20,330,565
Non-trainable params: 0
_________________________________________________________________
```

Figure 5 – Experiment 3

Table 4 shows the training/validation accuracy and loss plots.

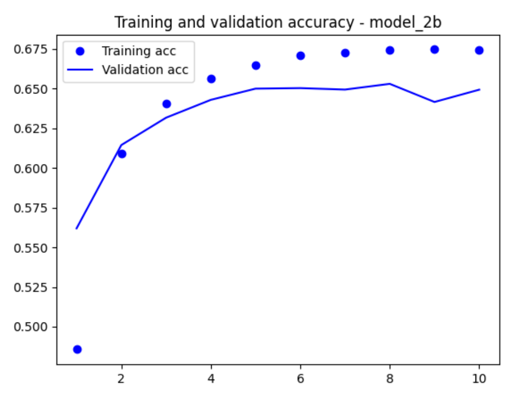
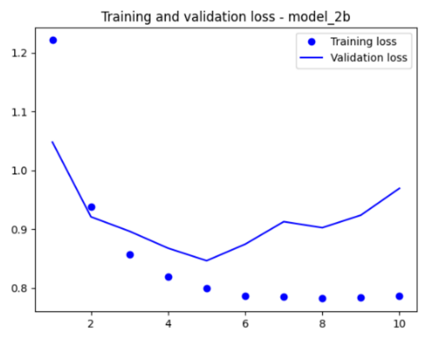
Training/Validation Accuracy	Training/Validation Loss
	

Table 4 – Experiment 3 Training/Validation Accuracy & Loss Curves

<u>Observation and Analysis</u>

Training and Validation Accuracy:

- The training accuracy steadily increases, starting at around 0.50 and reaching roughly 0.67 by epoch 10.
- Validation accuracy peaks around epoch 6, reaching approximately 0.65, then fluctuates slightly but remains mostly stable.

Training and Validation Loss:

- Training loss decreases consistently from above 1.20 to around 0.80 by epoch 10.
- Validation loss shows an initial decrease, reaching its lowest point around epoch 5, then slightly increases, staying below 1.0.

Learning Efficiency:

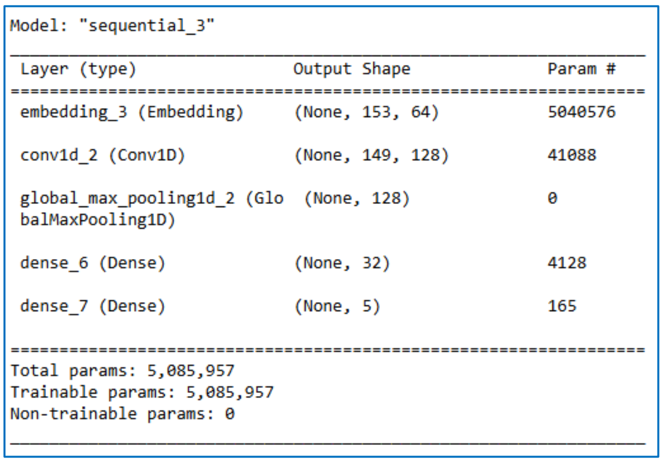
- The steady rise in training accuracy and consistent decrease in training loss indicates the model is learning well from the training data.
- The initial decrease in validation loss and increase in accuracy up to epoch 6 suggests effective learning and good generalization to unseen data at this stage.

Early Overfitting Signs:

- Post-epoch 4, the slight increase in validation loss indicates the model might start overfitting, where it performs excellently on training data but less on validation.

1.4.4 Experiment 4

For this experiment, we decrease the embedding dimension from 256 to **64**. This model still uses the 1d Conv layer for sentiment analysis. No dropout was used. The model summary is shown in Figure 6.



```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding_3 (Embedding)     (None, 153, 64)           5040576   
                                                                 
 conv1d_2 (Conv1D)           (None, 149, 128)          41088     
                                                                 
 global_max_pooling1d_2 (Glo  (None, 128)              0         
 balMaxPooling1D)                                                
                                                                 
 dense_6 (Dense)             (None, 32)                4128      
                                                                 
 dense_7 (Dense)             (None, 5)                 165       
                                                                 
=================================================================
Total params: 5,085,957
Trainable params: 5,085,957
Non-trainable params: 0
_________________________________________________________________
```

Figure 6 – Experiment 4

Table 5 shows the training/validation accuracy and loss plots.

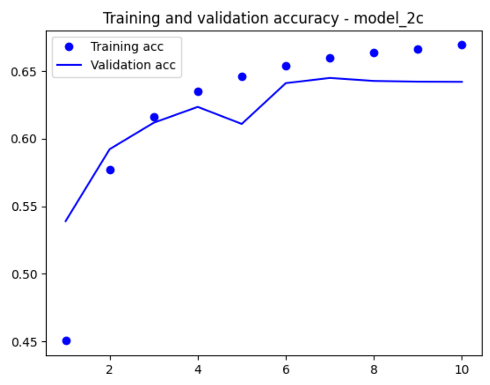
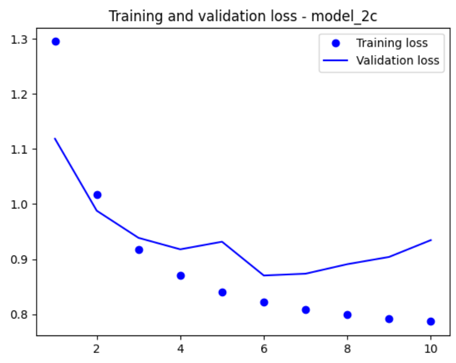
Training/Validation Accuracy	Training/Validation Loss
	

Table 5 – Experiment 4 Training/Validation Accuracy & Loss Curves

Observation and Analysis

Training and Validation Accuracy:

- **Training Accuracy**: Shows a continuous increase from around 0.45 to 0.67 over the 10 epochs.
- **Validation Accuracy**: Also increases, starting from 0.54 and plateauing around 0.65 after the 6th epoch.

Training and Validation Loss:

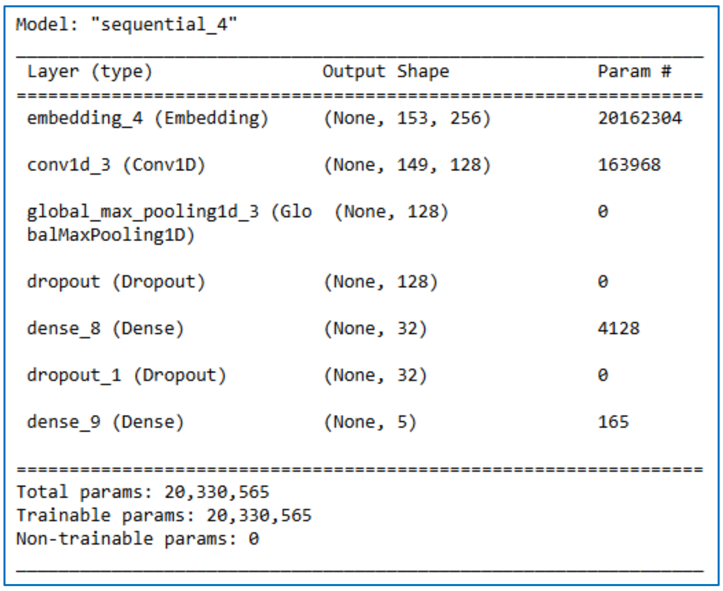
- **Training Loss**: Decreases steadily, starting from 1.3 and reaching approximately 0.8 by epoch 10.
- **Validation Loss**: Initially decreases but starts to increase slightly at the 5th epoch, indicating potential overfitting.

The overall trend of training metrics indicates that the model is learning effectively.

The plateauing validation accuracy and increasing validation loss signal that the model might be reaching its optimal performance around the 6th epoch.

1.4.5 Experiment 5

For this experiment, we use the embedding dimension of 256, with a dropout of 0.3. This model still uses the 1d Conv layer for sentiment analysis. The model summary is shown in Figure 7.



```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 153, 256)          20162304

 conv1d_3 (Conv1D)           (None, 149, 128)          163968

 global_max_pooling1d_3 (Glo  (None, 128)              0
 balMaxPooling1D)

 dropout (Dropout)           (None, 128)               0

 dense_8 (Dense)             (None, 32)                4128

 dropout_1 (Dropout)         (None, 32)                0

 dense_9 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,330,565
Trainable params: 20,330,565
Non-trainable params: 0
_________________________________________________________________
```

Figure 7 – Experiment 5

Table 6 shows the training/validation accuracy and loss plots.

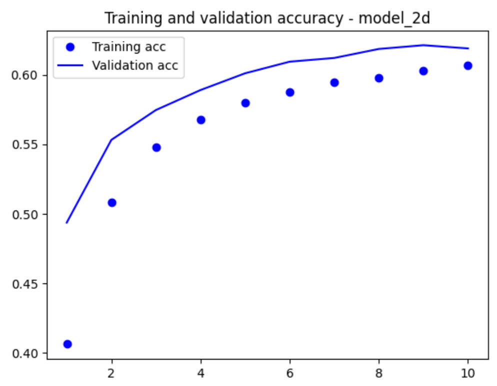
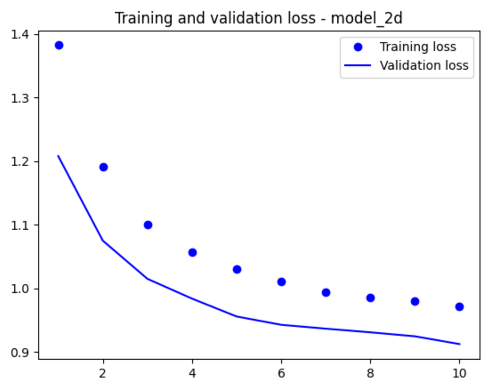
Training/Validation Accuracy	Training/Validation Loss
	

Table 6 – Experiment 5 Training/Validation Accuracy & Loss Curves

<u>**Observation and Analysis**</u>

Training and Validation Accuracy:

- **Training Accuracy**: Starts around 0.40 and increases steadily to approximately 0.61 by the 10th epoch.
- **Validation Accuracy**: Starts around 0.49 and rises to approximately 0.62, showing an overall upward trend.

Training and Validation Loss:

- **Training Loss**: Begins at 1.38 and decreases to around 0.97 by the 10th epoch.
- **Validation Loss**: Starts at 1.2 and drops to approximately 0.9, indicating a general improvement.

Analysis

Learning Efficiency:

- The steady increase in training accuracy and corresponding decrease in training loss suggests that model is effectively learning from the training data.
- The similar upward trend in validation accuracy implies that the model generalizes well to new, unseen data during the earlier epochs.

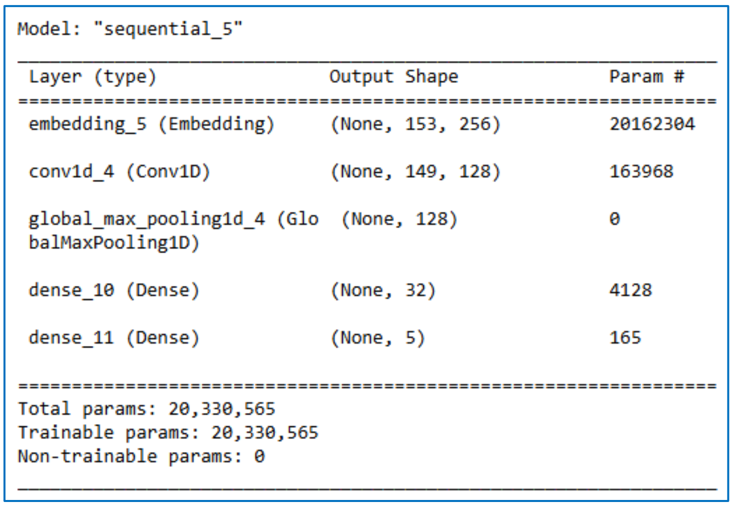
Overfitting Indicators:

- Despite the overall improvement, the minor fluctuations and eventual plateauing of validation accuracy around the 10th epoch indicate potential overfitting.
- Additionally, the gentle decrease in validation loss, followed by stabilization, suggests the model may start fitting too closely to the training data, impacting its performance on new data.

This model demonstrates promising learning capabilities, with clear improvements observed in both training and validation metric.

1.4.6 Experiment 6

In this run, we use the embedding dimension of **256**, with no dropout, **early stopping** and 25 epochs. This model still uses the 1d Conv layer for sentiment analysis. The model summary is shown in Figure 8.



```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_5 (Embedding)     (None, 153, 256)          20162304

 conv1d_4 (Conv1D)           (None, 149, 128)          163968

 global_max_pooling1d_4 (Glo  (None, 128)              0
 balMaxPooling1D)

 dense_10 (Dense)            (None, 32)                4128

 dense_11 (Dense)            (None, 5)                 165

=================================================================
Total params: 20,330,565
Trainable params: 20,330,565
Non-trainable params: 0
_________________________________________________________________
```

Figure 8 – Experiment 6

Table 7 shows the training/validation accuracy and loss plots.

With early stopping, training converged at the **9th** epoch.

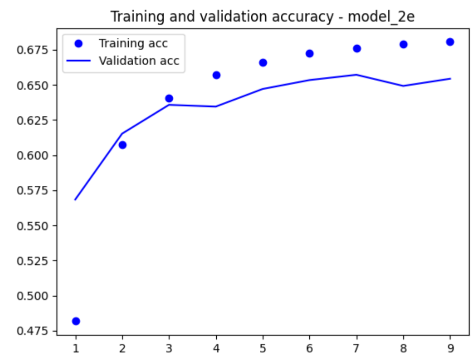
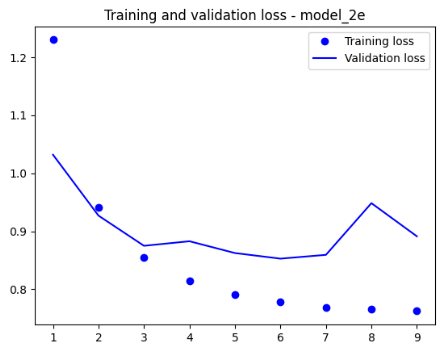
Training/Validation Accuracy	Training/Validation Loss
	

Table 7 – Experiment 6 Training/Validation Accuracy & Loss Curves

Observation and Analysis

Training and Validation Accuracy:

- **Training Accuracy**: Increases consistently from approximately 0.475 to around 0.675 by the 9th epoch.
- **Validation Accuracy**: Increases as well, peaking around 0.65 but with noticeable fluctuations.

Training and Validation Loss:

- **Training Loss**: Decreases steadily from approximately 1.2 to around 0.8 by the 9th epoch.
- **Validation Loss**: Initially decreases but shows fluctuations, peaking at the 7th epoch before decreasing again.

Training Efficiency:

The consistent increase in training accuracy and corresponding decrease in training loss indicate that the model is effectively learning from the training data.

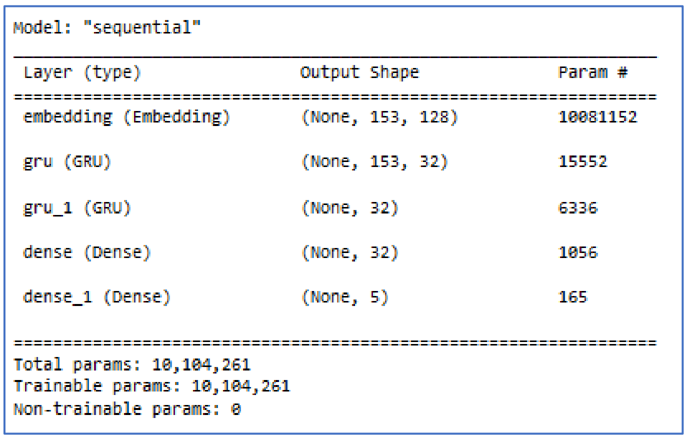
Validation Performance:

While the validation accuracy mirrors the overall trend of the training accuracy, fluctuations suggest variability in the model's performance on unseen data.

The instability in validation loss highlights potential overfitting, where the model fits the training data well but struggles to generalize.

1.4.7 Experiment 7

In this experiment, we build a model using *Gated Recurrent Unit* layer for sentiment analysis. There is no dropout used and the embedding dimension was **128**. **Two** GRU layers with **32** units each was used. The model summary is shown in Figure 9.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 153, 128)          10081152

 gru (GRU)                   (None, 153, 32)           15552

 gru_1 (GRU)                 (None, 32)                6336

 dense (Dense)               (None, 32)                1056

 dense_1 (Dense)             (None, 5)                 165

=================================================================
Total params: 10,104,261
Trainable params: 10,104,261
Non-trainable params: 0
```

Figure 9 – Experiment 7

Table 8 shows the training/validation accuracy and loss plots.

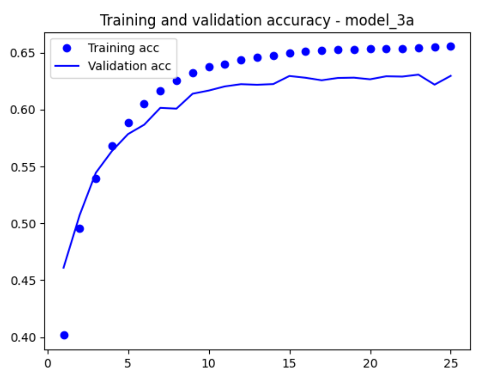
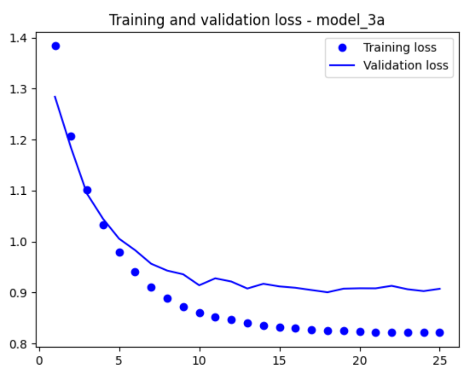
Training/Validation Accuracy	Training/Validation Loss
	

Table 8 – Experiment 7 Training/Validation Accuracy & Loss Curves

Observations

Training and Validation Accuracy

- The training and validation accuracy both increase over the epochs.
- The training accuracy grows steadily and stabilizes around 0.65 after about 13 epochs.
- The validation accuracy mirrors this trend but with more fluctuation, eventually stabilizing around 0.63 as well.

Training and Validation Loss

- Both training and validation losses decrease sharply initially.
- The training losses level off around 0.82 after approximately 20 epochs.
- The validation losses similarly decrease and eventually stabilize around 0.9 after around 16 epochs.

Analysis

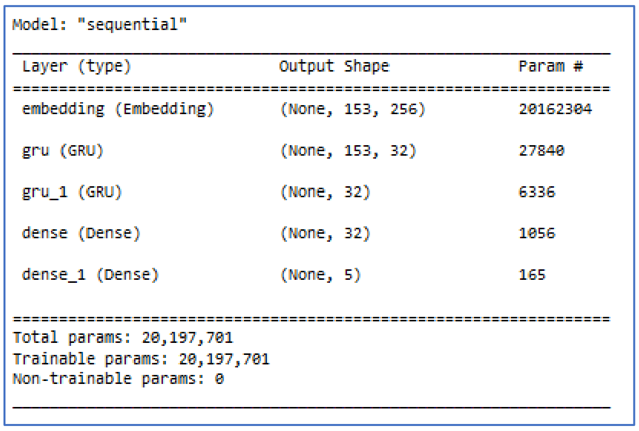
- **Model Performance Stability**: Both training and validation metrics show an initial sharp change and then stabilize. This indicates that the model was rapidly learning during the initial epochs and then reached a plateau as it found an optimal performance zone.

- **Fluctuations in Validation Metrics**: The fluctuation seen in the validation accuracy is quite common as it evaluates the model on unseen data, which naturally leads to some variance. Nonetheless, it does converge parallelly with training accuracy suggesting a decent generalization.
- **Loss vs. Accuracy Dynamics**: The decrease in loss values synchronizes with the increase in accuracy; such tandem movement is desirable and signifies that the model's predictions are becoming increasingly accurate over time.

In the next experiment, I wanted to determine if increasing the embedding dimension would increase the prediction accuracy.

1.4.8 Experiment 8

Here we increased the embedding dimension to **256** from 128, and ran this for **20** epochs. **Two** GRU layers with **32** units each was used. The model summary is shown in Figure 10.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 153, 256)          20162304

 gru (GRU)                   (None, 153, 32)           27840

 gru_1 (GRU)                 (None, 32)                6336

 dense (Dense)               (None, 32)                1056

 dense_1 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,197,701
Trainable params: 20,197,701
Non-trainable params: 0
_________________________________________________________________
```

Figure 10 – Experiment 8

Table 9 shows the training/validation accuracy and loss plots.

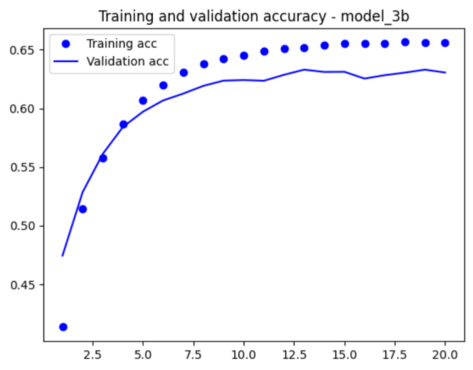
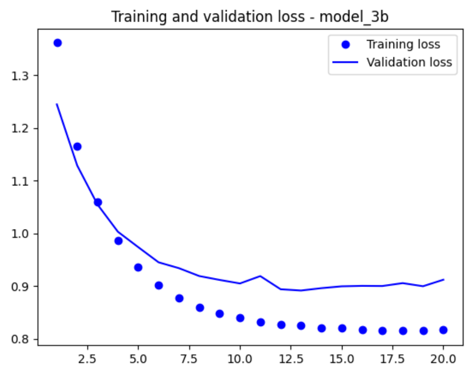
Training/Validation Accuracy	Training/Validation Loss
	

Table 9 – Experiment 8 Training/Validation Accuracy & Loss Curves

Observations

Training and Validation Accuracy:

- Both training and validation accuracy increase over the epochs.
- Training accuracy starts at around 0.41 and steadily rises to plateau around 0.66.
- Validation accuracy follows a similar pattern, beginning at approximately 0.47 and stabilizing around 0.63.

Training and Validation Loss:

- Both training and validation losses decrease sharply in initial epochs and then level off.
- Training loss starts around 1.36 and levels off around 0.82.
- Validation loss follows a similar trend, starting at around 1.2 and stabilizing around 0.9.

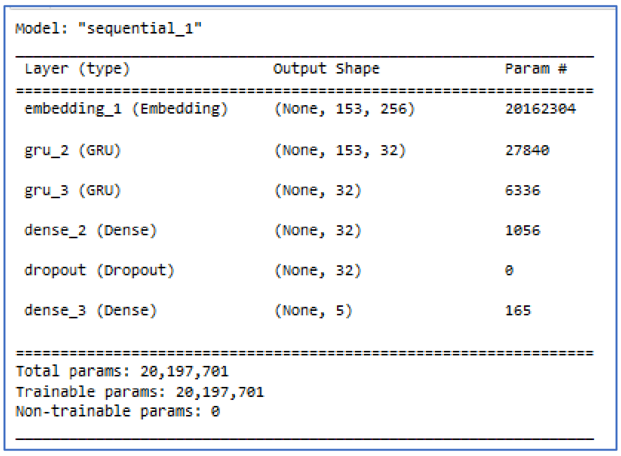
Analysis

Impact of Increased Embedding Dimension: Increasing the embedding dimension from 128 to 256 seems to have contributed to the model's ability to capture more intricate patterns, leading to the observed improvement in both accuracy and loss metrics.

Model Generalization: The gap between training accuracy and validation accuracy is minimal, suggesting that the model generalizes well to new data, though there is minor overfitting evidenced by the slight difference in loss values.

1.4.9 Experiment 9

In this run, we use the embedding dimension of **256**, dropout of **0.3**, no recurrent dropout, and running for **20** epochs. Two GRU layers with **32** units each was used. The model summary is shown in Figure 11.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 153, 256)          20162304

 gru_2 (GRU)                 (None, 153, 32)           27840

 gru_3 (GRU)                 (None, 32)                6336

 dense_2 (Dense)             (None, 32)                1056

 dropout (Dropout)           (None, 32)                0

 dense_3 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,197,701
Trainable params: 20,197,701
Non-trainable params: 0
_________________________________________________________________
```

Figure 11 – Experiment 9

Table 10 shows the training/validation accuracy and loss plots.

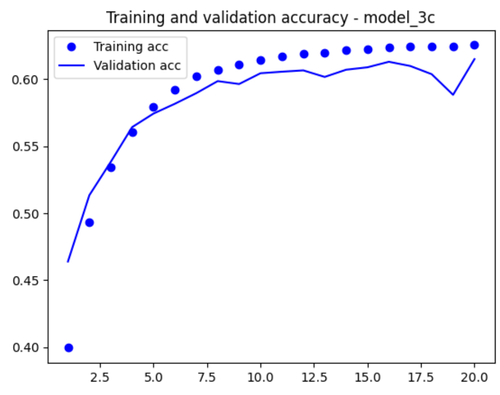
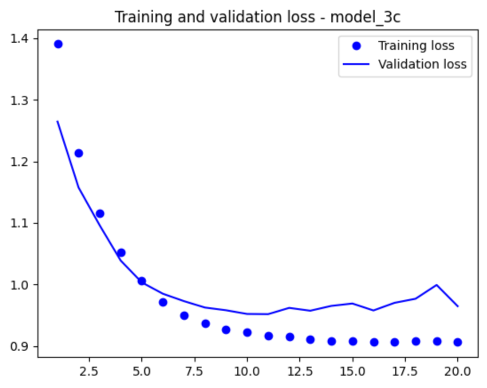
Training/Validation Accuracy	Training/Validation Loss
	

Table 10 – Experiment 9 Training/Validation Accuracy & Loss Curves

Observations

***Training and Validation Accuracy**:*

- Training accuracy increases steadily from around 0.40 to approximately 0.63 over the 20 epochs.
- Validation accuracy follows a similar upward trend but exhibits more fluctuation, ending just below the training accuracy.

***Training and Validation Loss**:*

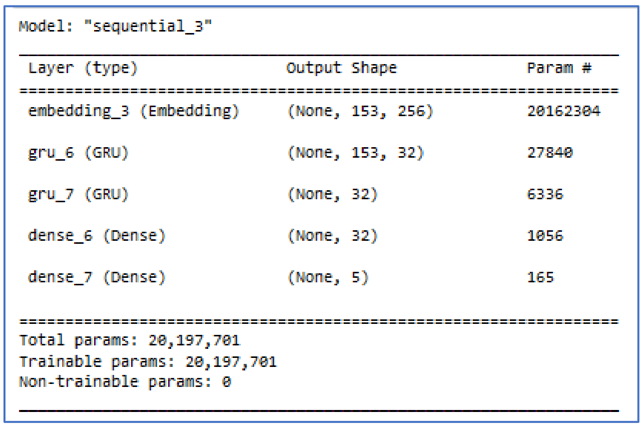
- Training loss decreases sharply from around 1.4 to approximately 0.9 over the 20 epochs.
- Validation loss shows a similar decreasing trend with some fluctuations, stabilizing slightly above the training loss.

Analysis

- **Impact of Dropout**: The inclusion of a 0.3 dropout rate has been effective in mitigating overfitting and enhancing the model's generalization capacity. The observable trend of loss values indicates that dropout has smoothed out learning fluctuations.
- **Model Performance**: Both accuracy and loss metrics indicate that the model is learning effectively. The training metrics show a solid improvement, while the validation metrics suggest a good generalization but with minor variability.
- **Generalization**: The fluctuations in validation accuracy and loss suggest that while the model generalizes fairly well, there is still some room for improvement. Additional regularization techniques or hyperparameter tuning could further stabilize validation performance.

1.4.10 Experiment 10

In this experiment, we use the embedding dimension of **256**, no dropout, a recurrent dropout of **0.2**, and running for **20** epochs. Two GRU layers with **32** units each was used. The model summary is shown in Figure 12.



```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_3 (Embedding)     (None, 153, 256)          20162304

 gru_6 (GRU)                 (None, 153, 32)           27840

 gru_7 (GRU)                 (None, 32)                6336

 dense_6 (Dense)             (None, 32)                1056

 dense_7 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,197,701
Trainable params: 20,197,701
Non-trainable params: 0
_________________________________________________________________
```

Figure 12 – Experiment 10

Table 11 shows the training/validation accuracy and loss plots.

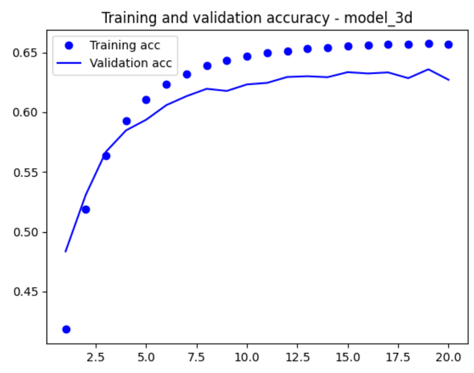
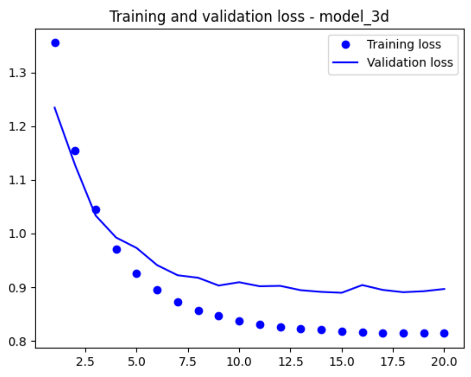
Training/Validation Accuracy	Training/Validation Loss
	

Table 11 – Experiment 10 Training/Validation Accuracy & Loss Curves

<u>Observations</u>

Training and Validation Accuracy:

- Both training and validation accuracy show similar trends, increasing steadily over the epochs.
- Training accuracy starts around 0.42 and stabilizes at approximately 0.66 by the end of 20 epochs.
- Validation accuracy follows a similar pattern, stabilizing around 0.63 with minor fluctuations.

Training and Validation Loss

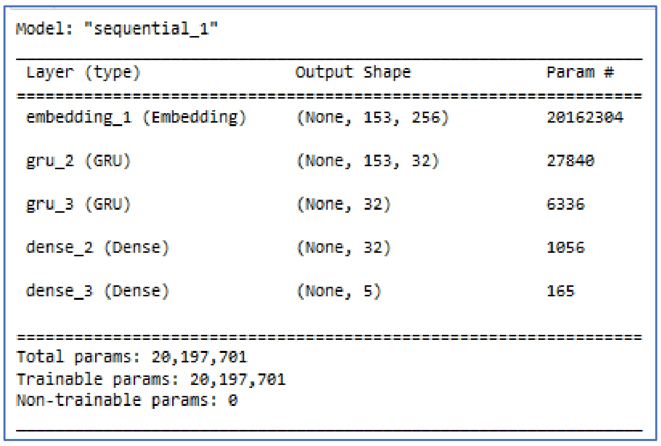
- Training loss decreases significantly from around 1.36 to approximately 0.81 over 20 epochs.
- Validation loss follows a similar trend, starting at around 1.23 and stabilizing around 0.9 with occasional fluctuations.

<u>Analysis</u>

- Effectiveness of Recurrent Dropout: The recurrent dropout of 0.2 appears to have effectively mitigated overfitting by regularizing the model, as evidenced by the close alignment of training and validation metrics.
- Model Learning and Generalization: The similar trends in both accuracy and loss metrics for training and validation indicate that the model is capable of learning effectively from the data and generalizes well to unseen data.
- Minor Fluctuations in Validation Metrics: Some fluctuations in the validation metrics are observed, which is normal and indicates minor overfitting, but overall, the dropout appears to have controlled it well.

1.4.11 Experiment 11

Here we use the embedding dimension of **256**, no dropout, no recurrent dropout, with early stopping and running for **25** epochs. Two GRU layers with **32** units each was used. The model summary is shown in



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 153, 256)          20162304

 gru_2 (GRU)                 (None, 153, 32)           27840

 gru_3 (GRU)                 (None, 32)                6336

 dense_2 (Dense)             (None, 32)                1056

 dense_3 (Dense)             (None, 5)                 165

=================================================================
Total params: 20,197,701
Trainable params: 20,197,701
Non-trainable params: 0
_________________________________________________________________
```

Figure 13 – Experiment 11

Table 12 shows the training/validation accuracy and loss plots.

With early stopping, training converged at the 15th epoch.

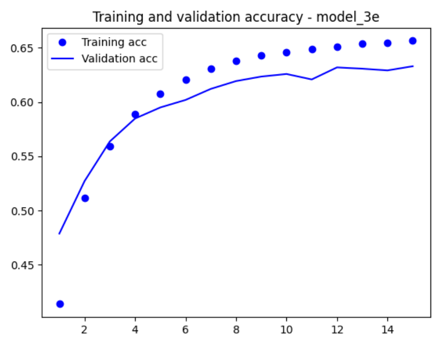
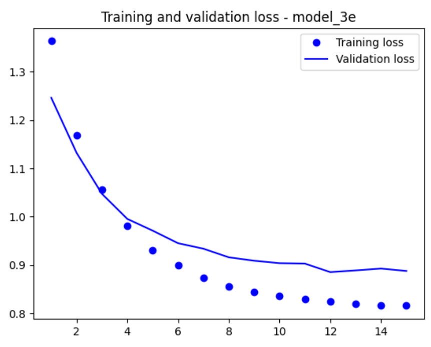
Training/Validation Accuracy	Training/Validation Loss
	

Table 12 – Experiment 11 Training/Validation Accuracy & Loss Curves

<u>Observations</u>

Training and Validation Accuracy

- Training Accuracy: Starts at around 0.41, increases steadily, and stabilizes at approximately 0.66 by the 15th epoch.
- Validation Accuracy: Follows a similar pattern, starting around 0.48, with some fluctuations, stabilizing around 0.63 by the 15th epoch.

Training and Validation Loss

- Training Loss: Begins at around 1.4 and decreases steadily, levelling off at approximately 0.8 by the 15th epoch.
- Validation Loss: Mirrors this trend, starting at about 1.2, with slight fluctuations, stabilizing around 0.9 by the 15th epoch.

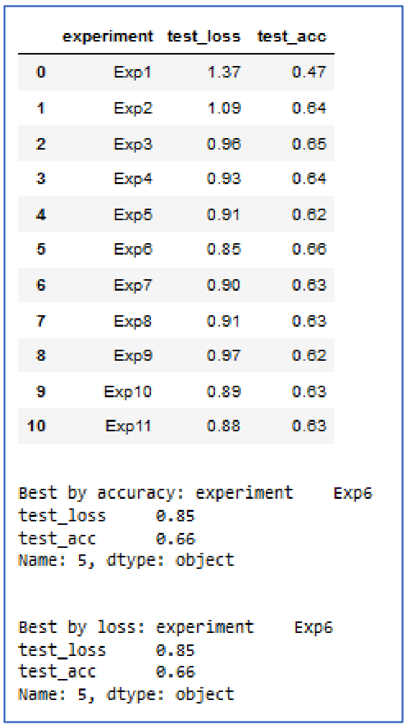
Analysis

- Effectiveness Without Dropout: The absence of dropout did not lead to severe overfitting, as evidenced by the closely aligned training and validation metrics. This suggests that the model's architecture and the use of early stopping helped maintain good generalization.
- Model Convergence: The model converged at the 15th epoch, indicated by the stabilization of both accuracy and loss metrics. This convergence implies that the model reached its optimal learning state at this point, and further training did not provide additional benefits.
- Early Stopping Impact: The implementation of early stopping was highly effective as it halted training once the model's performance metrics plateaued, preventing unnecessary computations and potential overfitting beyond the 15th epoch.

1.5 Model Evaluation

The models were evaluated using the ***evaluate()*** function passing in X_test & y_test. Figure 14 shows a summary of the performance of the six experiments.

The best performing model is from experiment 6, using 1D conv layer for sentiment analysis task, with an embedding dimension of 256, no drop out and early stopping.



	experiment	test_loss	test_acc
0	Exp1	1.37	0.47
1	Exp2	1.09	0.64
2	Exp3	0.96	0.65
3	Exp4	0.93	0.64
4	Exp5	0.91	0.62
5	Exp6	0.85	0.66
6	Exp7	0.90	0.63
7	Exp8	0.91	0.63
8	Exp9	0.97	0.62
9	Exp10	0.89	0.63
10	Exp11	0.88	0.63

```
Best by accuracy: experiment    Exp6
test_loss     0.85
test_acc      0.66
Name: 5, dtype: object


Best by loss: experiment    Exp6
test_loss     0.85
test_acc      0.66
Name: 5, dtype: object
```

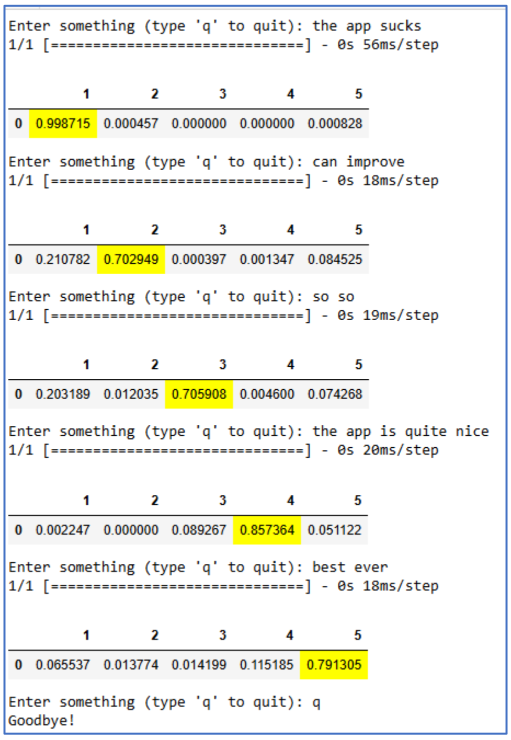
Figure 14 – Model performance summary

1.6 Model Prediction

Table 13 shows some sample text input and the predicted rating given by the best model.

Test Text Input	Predicted Rating
the app sucks	1
can improve	2
so so	3
the app is quite nice	4
best ever	5

Table 13 – Sample text input & predicted ratings



```
Enter something (type 'q' to quit): the app sucks
1/1 [==============================] - 0s 56ms/step

          1         2         3         4         5
0  0.998715  0.000457  0.000000  0.000000  0.000828

Enter something (type 'q' to quit): can improve
1/1 [==============================] - 0s 18ms/step

          1         2         3         4         5
0  0.210782  0.702949  0.000397  0.001347  0.084525

Enter something (type 'q' to quit): so so
1/1 [==============================] - 0s 19ms/step

          1         2         3         4         5
0  0.203189  0.012035  0.705908  0.004600  0.074268

Enter something (type 'q' to quit): the app is quite nice
1/1 [==============================] - 0s 20ms/step

          1         2         3         4         5
0  0.002247  0.000000  0.089267  0.857364  0.051122

Enter something (type 'q' to quit): best ever
1/1 [==============================] - 0s 18ms/step

          1         2         3         4         5
0  0.065537  0.013774  0.014199  0.115185  0.791305

Enter something (type 'q' to quit): q
Goodbye!
```

Figure 15 – Results from the test input samples

Figure 15 shows the complete test runs with the predicted probabilities for ratings 1 – 5.

1.7 Summary

The best performing model uses a 1D Conv Layer, embedding dimension of **256**, with no dropout, **early stopping** and 25 epochs for the sentiment analysis task, converging at the 9th epoch.

These are some strategies that can be used for further improvement:

Data Preprocessing

- **Normalization**: Convert text to lowercase, remove punctuation, and apply stemming or lemmatization to reduce variability.
- **Data Augmentation**: Use techniques like synonym replacement or paraphrasing to artificially increase the size of your dataset.

Model Architecture

- **Embedding Layer**: Use pre-trained word embeddings (e.g., GloVe, Word2Vec) to provide richer representations of words.
- **Multiple Convolutional Filters**: Use multiple convolutional filters of different sizes to capture various n-grams and patterns in the text.

Regularization

- **Batch Normalization**: Use batch normalization to stabilize and speed up training.
- **L2 Regularization**: Apply L2 regularization to the weights to penalize large weights and prevent overfitting.

Hyperparameter Optimization

- **Learning Rate**: Experiment with different learning rates to find the optimal value for faster convergence.
- **Batch Size**: Try various batch sizes to balance training speed and model performance.
- **Number of Filters and Units**: Adjust the number of filters in the convolutional layers and the units in the dense layers.

Advanced Training Techniques

- **Learning Rate Schedules**: Use learning rate schedules to gradually decrease the learning rate during training.

Transfer Learning and Fine-Tuning

- **Pre-Trained Models**: Utilize pre-trained models and fine-tune them on your dataset to leverage existing knowledge and improve accuracy.
- **Fine-Tuning Embeddings**: Fine-tune word embeddings during training to better adapt them to your specific task.