Philip Tam

Project Deliverable 3

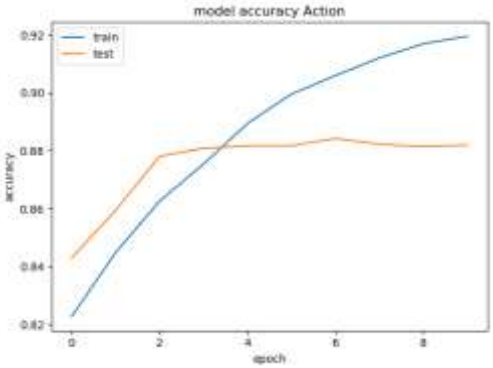
1. Final Results

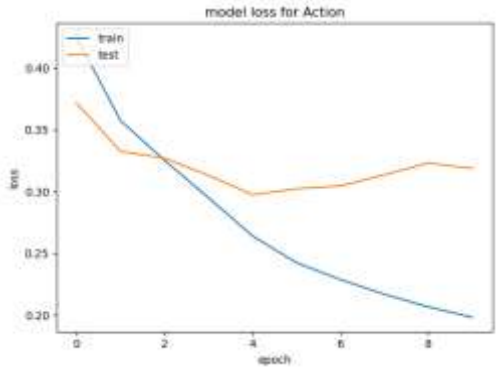
The final results compared to the preliminary results show a vast improvement in terms of accuracy and loss. The results indicate a 5% accuracy improvement for the Action and Comedy responses. Instead of completely changing the model completely, the assumption of the preliminary results was that the model was overfitting. One minor change to the model was the batch size. Instead of 16, a size of 32 was used to speed up the training time because the additional data. To alleviate this supposed problem, 100,000 movie plot summaries were added from IMDB. Moreover, the genres to classify were changed to Action, Adventure, Comedy, Thriller, Family, Crime, Mystery, and Romance because of the new plot summaries changing the original count. The total training time taken was approximately 16 hours to train every model to classify each genre individually. Each genre was trained individually because it offered the best accuracy when tested after where al of their accuracies were over 85% except Comedy which stands at 80%. Albeit, 80% is the lowest accuracy, it is an improvement over the preliminary results. The loss function that the model was trying to minimize was a binary cross-entropy because each genre was trained individually therefore it is a two-label problem. The results show that the loss of every genre acted in very similar fashion only differing in the value of the loss. The loses would fall for the first few epochs and then increase in future iterations because of the data itself was not standardized. Additionally, because of the lack of computational power, a maximum length of 500 was set to the sequences which could also affect the accuracy. The initial goal was to replicate the Bidirectional LSTM model of Ertugrul's (2018). Although the approach taken was different, the model is working and can classify multiple different genres, therefore the initial goal was met of replicating the Bidirectional LSTM model. It is important to note that the higher the count of each

genre is different which affected the training process, and the results. For future iterations, it would be better to make it so that the genres all have an equivalent count, standardized, and trained on cloud platform to handle the computational problems.

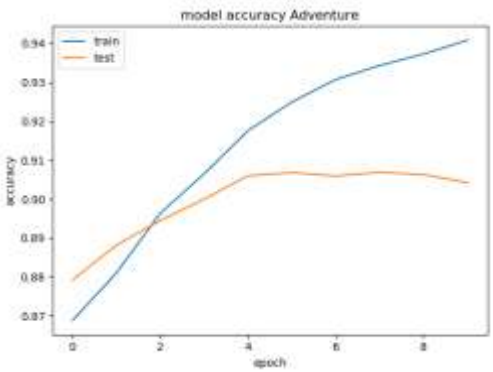
Final Results

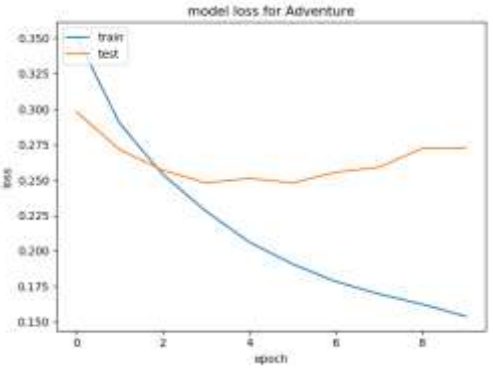
Action



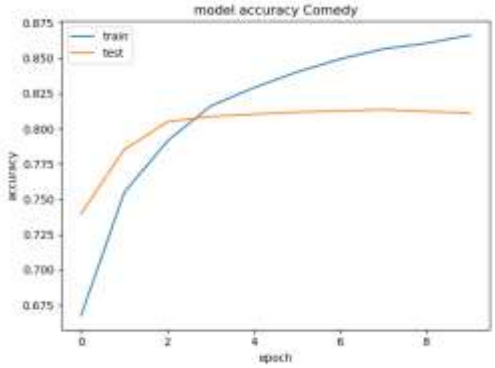


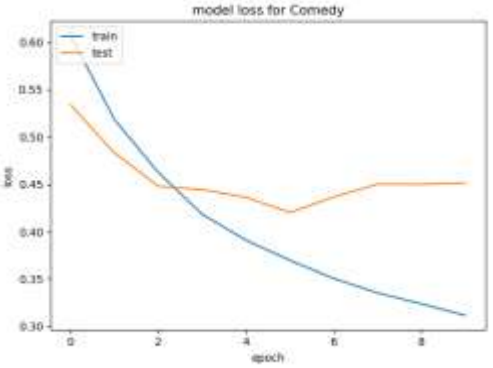
Adventure

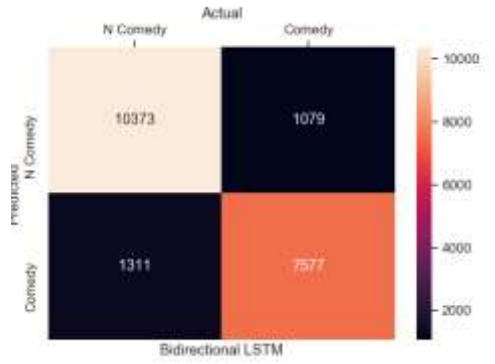




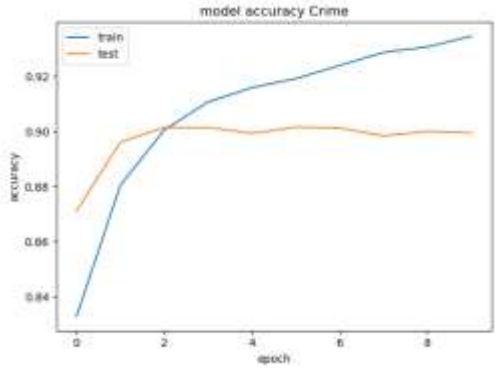
Comedy

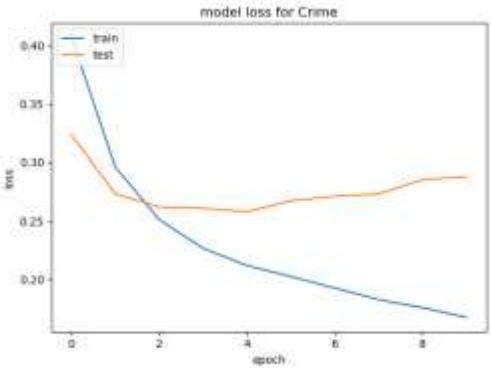


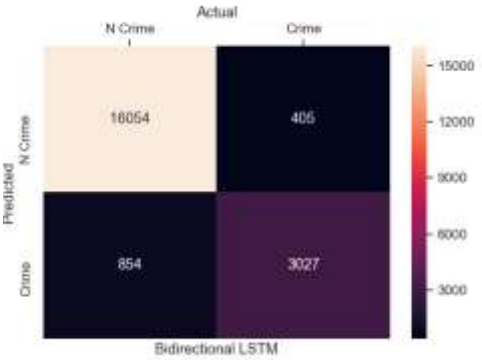




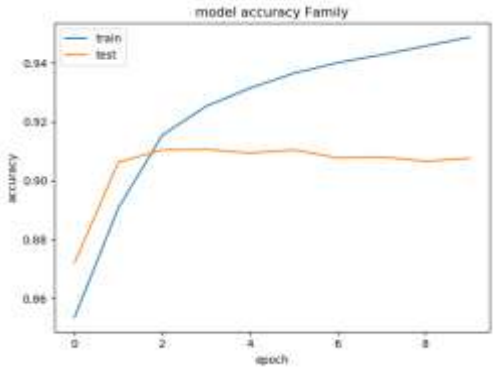
Crime

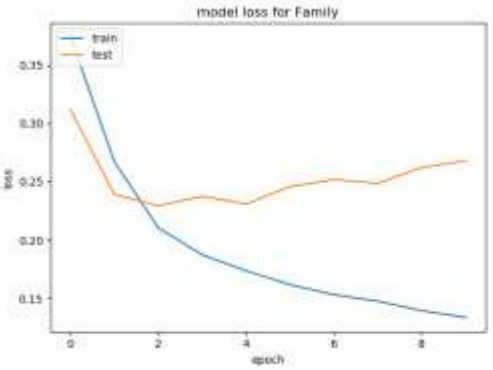




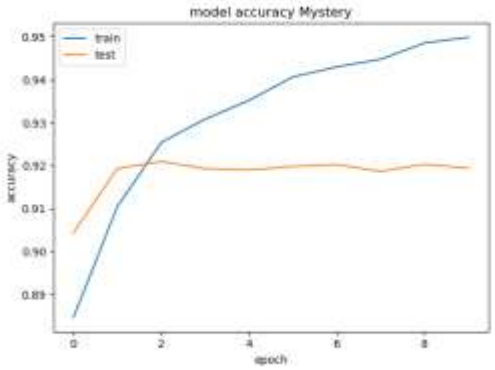


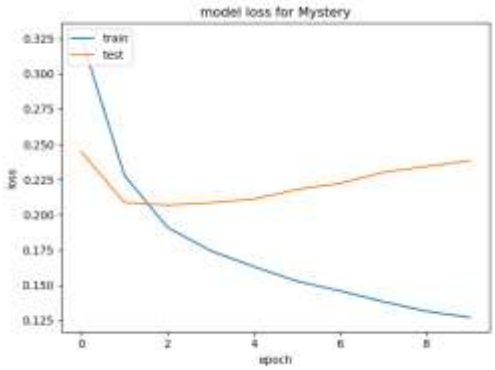
Family



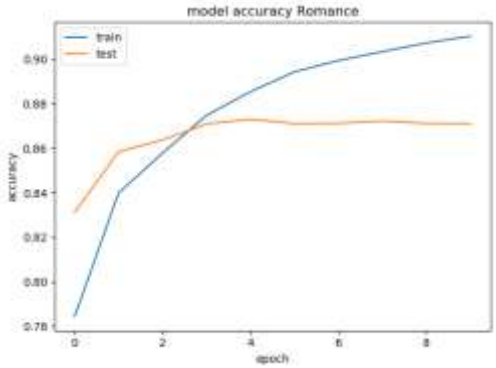


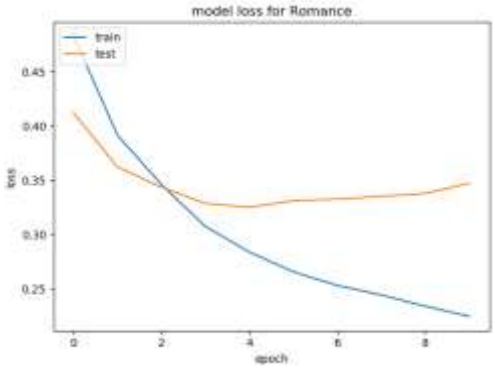
Mystery

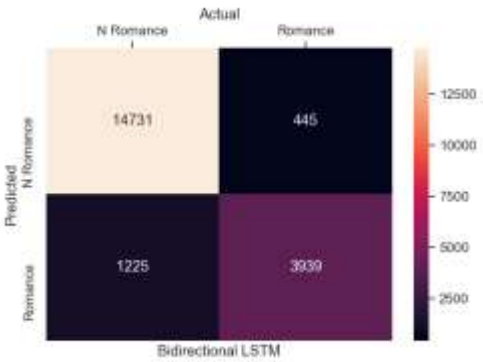




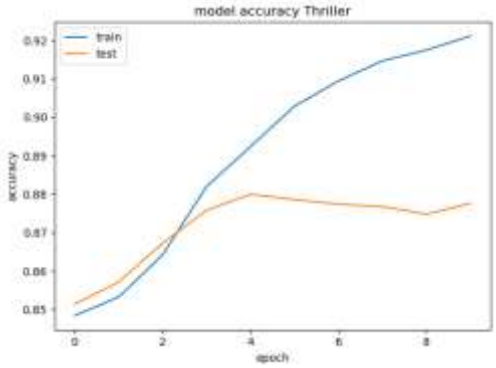
Romance

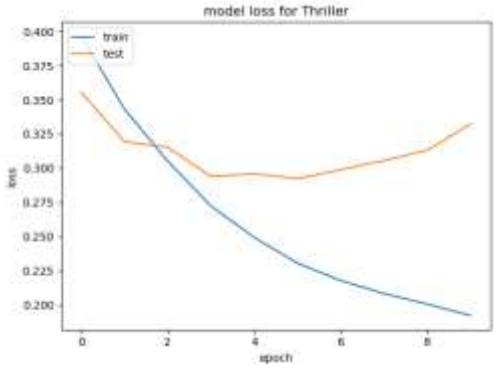






Thriller





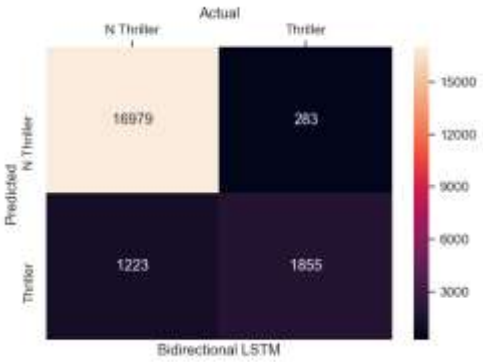


Table: Count of movies

Genre	#movies
Action	18249
Adventure	13493
Comedy	44342
Thriller	15386
Family	15442
Crime	19408
Mystery	12030
Romance	25908

2. Final demonstration proposal

There are three options for final products:

1. A iOS application focused on practicing on writing movie plot sypnosis while also having a section to be able to look at other movie plot summaries and how the model predicts them. The iOS application could be done using CoreML model which will be migrated from the Keras model or using a flask server to create an api that will run the model and then send it to the iOS application. This option sounds the most doable because most of my experience is in iOS development and it is within my comfort zone.
2. A Google Chrome Extension that uses TensorFlow.js where the model will be imported from a keras model. This google chrome extension will be activated on movie related sites and will automatically classify and create a bounding box while browsing that indicate the different genres it is predicting. This option is the one I am most intrigued with, but would have the most difficulty doing.
3. A simple web app using flask that has a text box that takes in an input and displays the genre it has classified, while also having a section to display other examples of predicted

inputs. This type of application could also have an API part where it will interconnect with the iOS application. This option would interests me a lot and will probably be my choice.

References

Ertugrul, Ali Mert & KARAGOZ, Pinar. (2018). Movie Genre Classification from Plot Summaries Using Bidirectional LSTM. 10.1109/ICSC.2018.00043.