# ICS 661

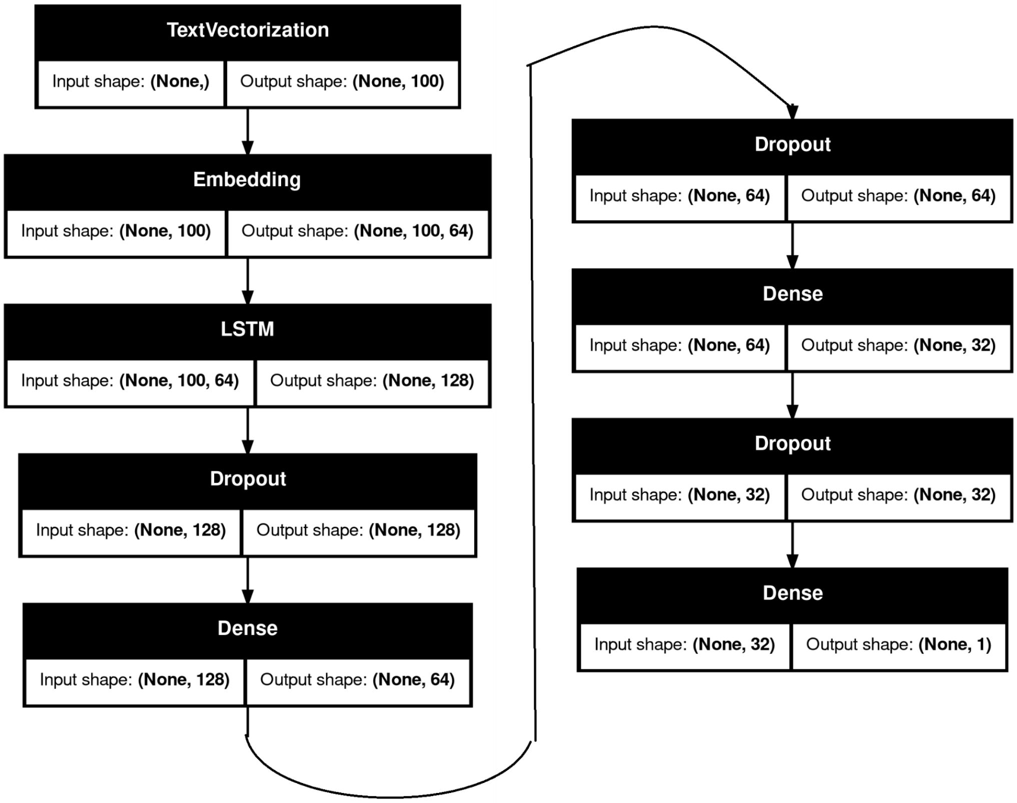
# Advanced AI

Fall 2024

## Section 1: Task Description

The task is to train a recurrent neural network (RNN) model on movie reviews for sentiment analysis to make a binary prediction for whether a given review is positive or negative. The reviews provided are not cleaned, so part of the task was to preprocess the data by removing stop words, punctuation, and normalizing the text data. Afterwards, a vocabulary set should be created that serves as the input to the RNN model.

## Section 2: Model Description



## Section 3: Experiment Settings

### 3.1 Dataset Description

The dataset used consists of 50,000 samples of movie reviews, which are split equally into 25,000 positive reviews and 25,000 negative reviews. There are 25,000 reviews which are split for training, and 25,000 split for the test dataset. The train dataset is further split into 80% train and 20% validation data. Positive reviews are defined as reviews >7 out of 10, and negative reviews are defined as reviews <4 out of 10. Neutral reviews are not included in the dataset, so this is a binary classification problem. Only 30 reviews for each movie is included in the dataset to prevent review bias on popular movies. The dataset is structured as two top-level directories of pos/ and neg/. Each review in its own text file in the format of [reviewId]_[rating].

## 3.2 Detailed Experimental Setups

The data in the training dataset is pre-processed using Tensorflow and NLTK by converting all words into lowercase, removing punctuation, removing stop words, and stripping HTML tags (notably <br />). Afterwards, the cleaned data is passed into a TextVectorization layer in the RNN model. An LSTM was used to learn long-term dependencies between the timesteps of the data. Regular dropout is applied within the LSTM to drop linear transformation of the inputs, while recurrent dropout is applied to drop connections between recurrent units. Afterwards, dropout and dense layers are applied to smoothen each layer dimension to the binary sigmoid activation function. Model settings are defined as follows:

```
model = tf.keras.Sequential([
    vectorize_layer,
    layers.Embedding(input_dim=10000, output_dim=64),
    layers.LSTM(128, dropout=0.5, recurrent_dropout=0.5),
    layers.Dropout(0.5),
    layers.Dense(64, activation='relu'), # 128 to 64
    layers.Dropout(0.25),
    layers.Dense(32, activation='relu'), # 64 to 32
    layers.Dropout(0.1),
    layers.Dense(1, activation='sigmoid')  # prediction between 0 and 1
])
```

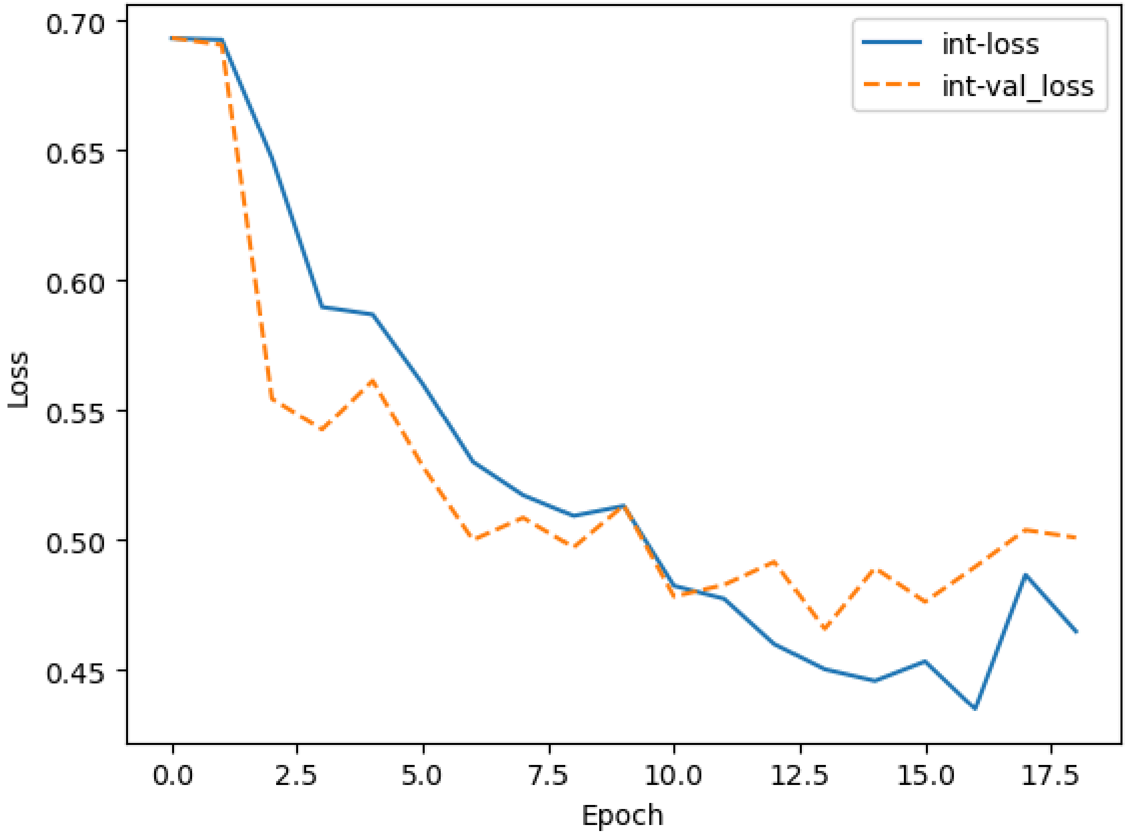
## 3.3 Evaluation Metrics

In the experiment, we will use Accuracy, Precision, Recall, F1 score as our evaluation metrics. Accuracy is the proportion of correctly classified instances out of the total instances  Precision is the proportion of correctly classified positive instances out of the total instances classified as positive, which helps measure the accuracy of positive predictions. Recall is the proportion of actual positive instances out of the total instances correctly classified as positive and falsely classified as negative. F1 score is the harmonic mean between precision and recall. A confusion matrix will also be generated to see the TP/FP/TN/FN rates.

## 3.4 Source Code

https://github.com/echung32/ics661-rnn (Ablation A is the default model).

### 3.5 Training Convergence Plot



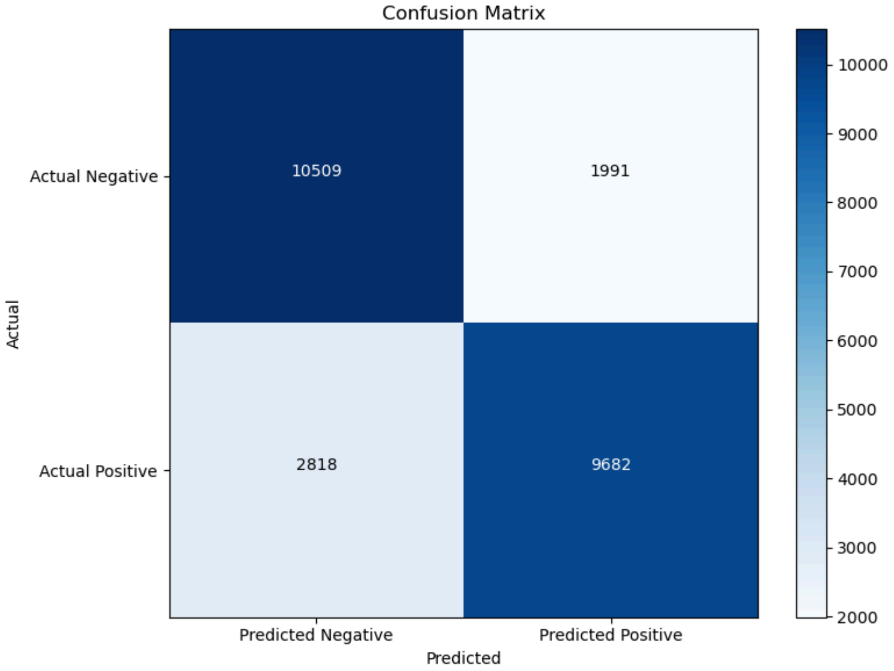
### 3.6 Model Performance

The model was trained using 80% of the training set, validated on 20% of the training set, then evaluated using the testing set. The scores of both models are reported below, as well as the 4 evaluation metrics:

| | |
|---|---|
| Training Score | 0.8322 |
| Test Score | 0.8076 |
| Test Accuracy | 0.8076 |
| Test Precision | 0.8294 |
| Test Recall | 0.7746 |
| Test F1 Score | 0.8011 |

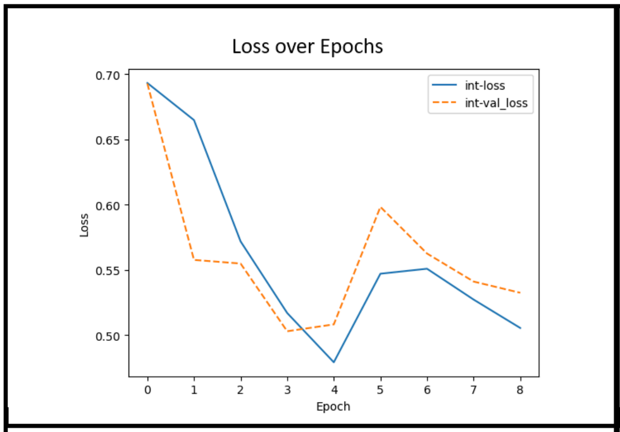
As the Training Score (0.8322) > Test Score (0.8076) by 0.0246 points, this may suggest that the model is slightly overfitting towards the training set. In the future, this could be minimized by utilizing methods like K-fold cross-validation, further experimenting with model layers and dimensions, or preprocessing the data better. However, this difference is not statistically significant because the gap between the training and test scores is relatively small, and still achieves a performance >75%.

A confusion matrix was also generated to visualize the prediction performance of the model, which shows that the model may be better at predicting negative reviews compared to positive ones (10509 vs 9682).

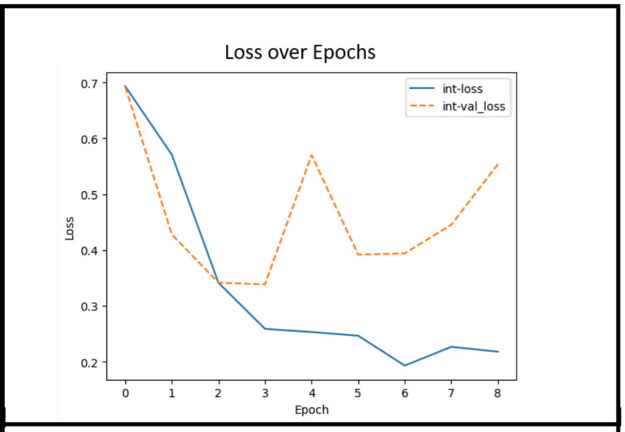


### 3.7 Ablation Studies

Ablation studies were performed, experimenting with dropout and recurrent_dropout values in the LSTM layer. The original model (Ablation_A) was performed with dropout as 0.5 and recurrent_dropout as 0.5. Ablation_B was performed with no dropout and recurrent_dropout as 0.5. Ablation_C was performed with no dropout and no recurrent_dropout.

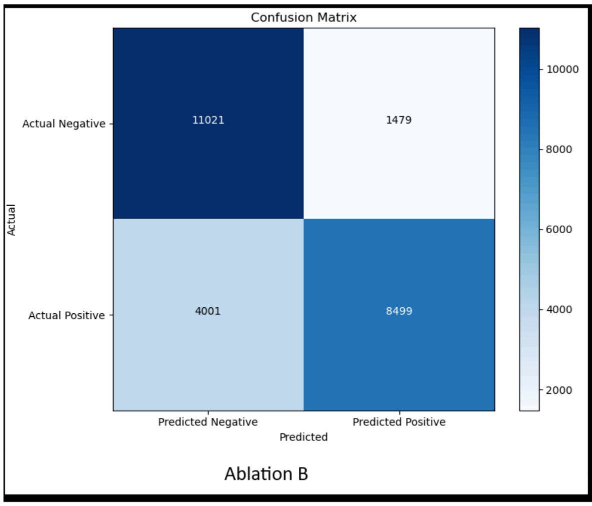


Ablation B results

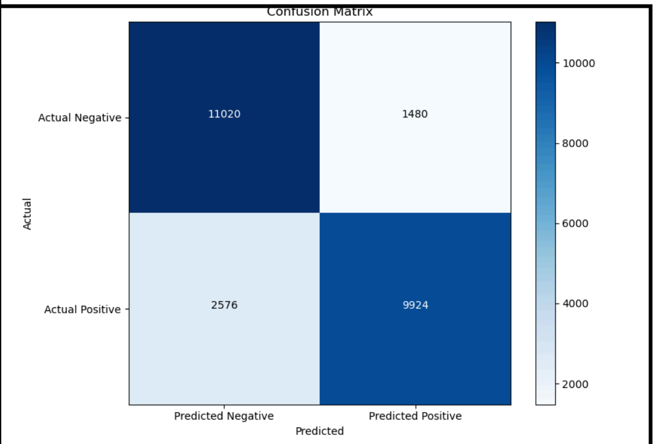


Ablation C results

For Ablation B, the graph shows that the loss went down over epochs, but suddenly went up, and then down again. With more epochs, it is possible that loss between the training and validation set could minimize further, but Ablation A performed well within the first few epochs without a big spike. Ablation C, on the other hand, showed that the model was overfit towards the training dataset, resulting in the validation loss diverging from the training loss.



Ablation B



Ablation C

In regards to the confusion matrix, Ablation B and Ablation C showed similar results to Ablation A, where the model is performing better on predicting true negative results compared to true positive results. Further analysis will need to be completed on the cause of this pattern.