INTERNSHIP: PROJECT REPORT

Internship Project Title	Automate Detection of different Sentiments from Textual Comments and Feedback.
Project Title	To develop a deep learning algorithm to detect different types of sentiment contained in a collection of English Sentences or a large paragraph.
Name of the Company	Tata Consultancy Services
Name of the Industry Mentor	Mr. Debashis Roy
Name of the Institute	Shree Rayeshwar Institute of Engineering & Information Technology, Shiroda- Goa

Start Date	End Date	Total Effort (hrs.)	Project Environment	Tools used
26/06/2020	05/07/2020	15	Google Collab, Chrome, Windows 10	Python 3 (Tensorflow)

Project Synopsis:

Sentiment analysis is extremely useful in social media monitoring as it allows us to gain an overview of the wider public opinion behind certain topics. The applications of sentiment analysis are broad and powerful. The ability to extract insights from social data is a practice that is being widely adopted by organizations across the world. It can also be an essential part of your market research and customer service approach. Not only can you see what people think of your own products or services, you can see what they think about your competitors too. The overall customer experience of your users can be revealed quickly with sentiment analysis, but it can get far more granular too. It can also be used to categorize feedback in movies, products etc.. Hence, developing a tool for automating Sentiment Analysis can be very beneficial for various industries.

Solution Approach:

Deep Learning Algorithms such as Recurrent Neural Network using the Long Short Term Memory (LSTM) can be used for the task. This model is used to perform sentiment analysis on movie reviews from the Large Movie Review Dataset, sometimes known as the IMDB dataset. The text encoder can be used to break down the sentiments into numerical values. On supplying test sentiments to the model, on the outcome it will try to predict whether it is a positive, negative or neutral feedback!

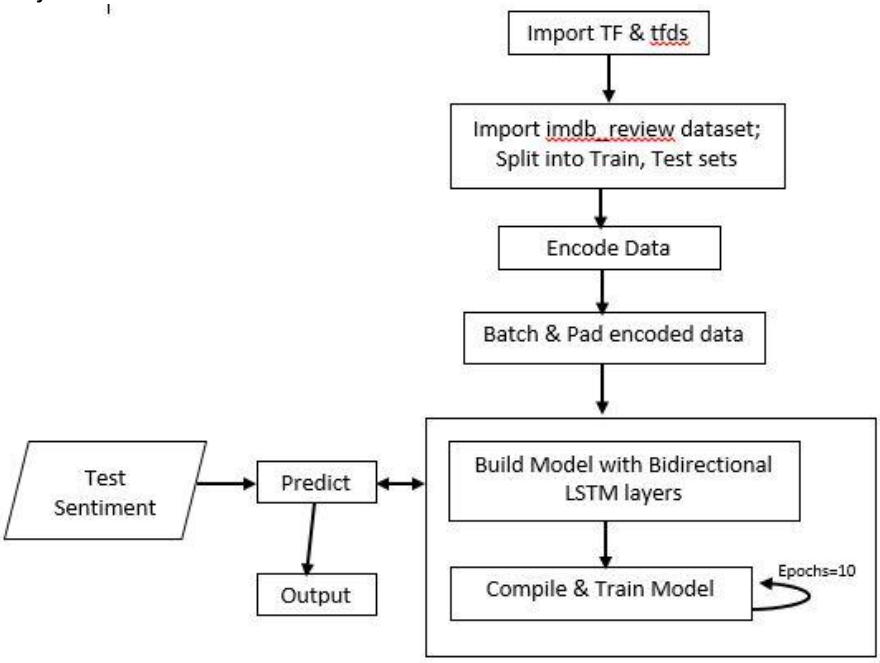
Assumptions:

IMDB dataset that contains the text of 50,000 movie reviews from the Internet Movie Database. These are split into 25,000 reviews for training and 25,000 reviews for testing. The training and testing sets are balanced, meaning they contain an equal number of positive and negative reviews So, its a binary or two-class-classification without 'neutral' label. However, This model does not mask the padding applied to the sequences. The Model will predict the sentiments giving a prediction score. If the prediction is >= 0.5, it is positive. On testing the model, the prediction value of negative sentiments can go up to -4. Therefore, it is assumed that for neutral sentiments (positive-negative combination), prediction values in the range -1 to +0.5 can be assumed (this also depends on accuracy

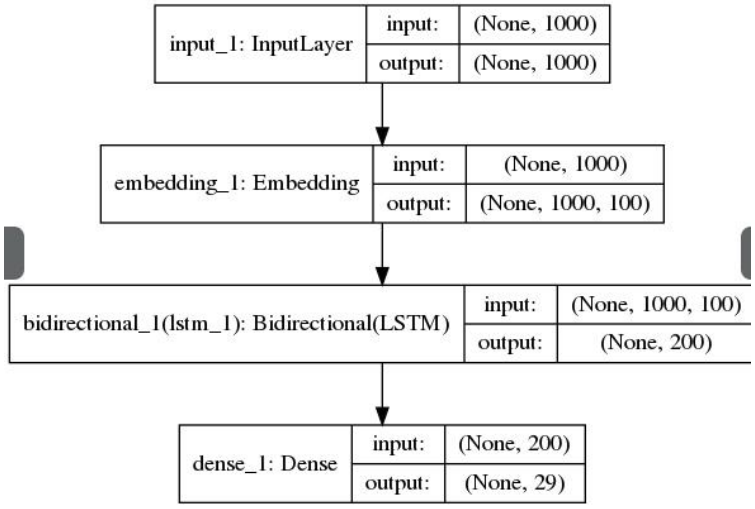
and length of the test sentiment).
Also, the dataset imported is already pre-processed, and is directly splitted in training & test sets by *tensorflow_datasets* library. This library also provides an encoder which breaks down a given string into a list of numerical values.

Project Diagrams:

Project Work Flow:



LSTM Model Flow-Chart Example:



Algorithms:

Recurrent Neural Network (RNN) using multi-layer Bidirectional Long Short Term Memory (LSTM).

Outcome:

For the Following Statement:

"I loved the movie a lot as I am great fan of marvel! Avengers: Endgame, which marks the end of the Infinity Saga, is spellbounding and surely an enthralling experience. The last film of the 'Avengers' franchise is remarkable and doesn't disappoint. Watching all our favourite superheroes in one film is just surpassing. Marvel has been working on this grand culmination ever since they released 'Iron Man'. I'm damn sure that all of their hard work and ambition has paid off. The directors, Anthony and Joe Russo, have made sure that it delivers an unforgettable experience. Christopher Markus and Stephen McFeely have come up with a screenplay full of epic and unpredictable moments. The film has a great balance of humour, emotions and action. The biggest strength of the film is the emotions. This is the most emotional superhero film I have ever seen. It's just perfect. The action sequences were jaw-dropping. The climatic battle left me amazed. It's just filled with memorable moments and cannot be described with words. The visuals are gorgeous and have a great impact on the film. The humour doesn't look exaggerated and manages to entertain throughout the film. The plot twists were very impressive and suspenseful. The film features many cameos of characters from the previous MCU films, which just gives a double dose of excitement. The background score gives me goosebumps, though I've listened to it several times. It was really clever to make changes to the characterization of the Hulk. I enjoyed that a lot. But the show-stealer is Robert Downey Jr, who plays the role of Tony Stark/Iron Man. The man who started it all proves yet again that there's no one else who can perfect his role. Do not miss his powerful moments in the final battle."

The output is POSITIVE.

Whereas for:

"Disappointing storyline - too many sad crying scenes - too much shit swearing compared to other great Marvel movies! Even as an adult I don't appreciate swearing in movies. There are MANY people who don't use cuss words in their lives except maybe in adrenaline traffic moments. To hear Robert Downey jr's 'moment' with his young child using and laughing at the fact that she uses 'adult' language is teaching the new impressionable ages watching this, that it is okay when it isn't. They seemed to want to use their cuss word quota for the rating for this movie. Sad writing when that's how they get their best laughs from audience. Bring back your creative, quirky writers from the two The Defenders of the Galaxy. Now that's smart character development and writing without resorting to desperate shock value. We have loved every movie of that series and eagerly await the next one. The Storyline was soooo boring in this. All of us watching kept hoping it would improve and it didn't. I think the only real laugh we had was the encounter on the ship between Quill (StarLord) and Thor. Subtle, but funny. We all could've cared less if anyone died. That's how checked out we were watching this LONG 3 HRS!!! Why! It was torture and I felt robbed of my time in the end."

The output is NEGATIVE.

"Superhero comics, and much of their adaptations, have long taken an outsized, soap opera-like approach to storytelling. At their best, they can take these fantastical ideas and make them emotionally resonant, even if there's obviously no real-world phenomenon to connect them to. In some respects, Endgame pulls this off beautifully, like how the character Nebula confronts her past self through time travel, giving physical form to her personal growth. But as fun as the movie is, there's an undeniable hollowness at its core induced by its unwillingness to follow through on certain ideas and symbols."

In this case, the output is NEUTRAL/POSITIVE.

Detailed Presentation with Proof of Reasonable Accuracy:

We need to create batches of training data for your model. The reviews are all different lengths, so use padded_batch to zero pad the sequences while batching. Each batch will have a shape of (batch_size, sequence_length) because the padding is dynamic each batch will have a different length. Therefore we initialize BUFFER_SIZE = 10000 & BATCH_SIZE = 64. The training set is shuffled on every code execution, delivering variation in final accuracy.

MODEL: *tf.keras.Sequential* model (6 Layers used)

We selected Keras sequential model here since all the layers in the model only have single input and produce single output.

1. The first layer is an Embedding layer. An embedding layer stores one vector per word. When called, it converts the sequences of word indices to sequences of vectors. These vectors are trainable. After training (on enough data), words with similar meanings often have similar vectors. The resulting dimensions are: (batch, sequence, embedding).

2. The *tf.keras.layers.Bidirectional* wrapper can also be used with an RNN layer. This propagates the input forward and backwards through the RNN layer and then concatenates the output. This helps the RNN to learn long range dependencies. Two Bidirectional Layers are used in this model.

3. This index-lookup is much more efficient than the equivalent operation of passing a one-hot encoded vector through a *tf.keras.layers.Dense* layer. A recurrent neural network (RNN) processes sequence input by iterating through the elements. RNNs pass the outputs from one timestep to their input—and then to the next.

4. The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by 1/(1 - rate) such that the sum over all inputs is unchanged.

5. This fixed-length output vector is piped through a fully-connected (Dense) layer with 64 hidden units. The last layer is densely connected with a single output node. This uses the default linear activation function that outputs logits for numerical stability. Another option is to use the sigmoid activation function that returns a float value between 0 and 1, representing a probability, or confidence level.

The number of layers used in the model along with the hidden units count in maintaining good accuracy.

COMPILING THE MODEL:
The model needs a loss function and an optimizer for training. Since this is a binary classification problem and the model outputs logits (a single-unit layer with a linear activation), we'll use the *binary_crossentropy* loss function, which is better for dealing with probabilities- it measures the "distance" between probability distributions, or in our case, between the ground-truth distribution and the predictions.

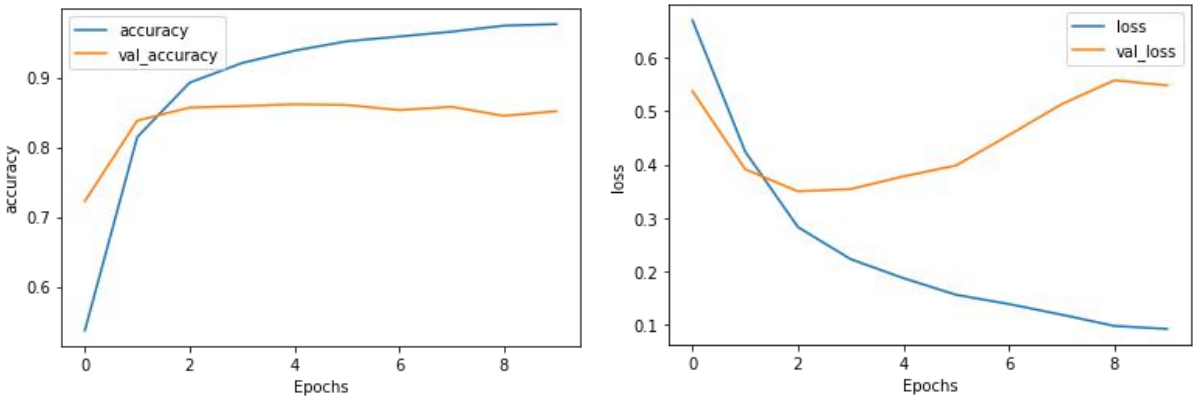
Considering all the above parameters, the final step- when training the model, the number of iterations over entire training set (epochs) with respect to loss function impacts the final accuracy! *model.fit* result with *epochs = 10* gives the following Accuracy:

```
Epoch 1/10
391/391 [==============================] - 1593s 4s/step - loss: 0.6623 - accuracy: 0.5428 - val_loss: 0.4910 -
val_accuracy: 0.7542
Epoch 2/10
391/391 [==============================] - 1618s 4s/step - loss: 0.3803 - accuracy: 0.8460 - val_loss: 0.3598 -
val_accuracy: 0.8578
Epoch 3/10
391/391 [==============================] - 1635s 4s/step - loss: 0.2782 - accuracy: 0.8984 - val_loss: 0.3434 -
val_accuracy: 0.8594
Epoch 4/10
391/391 [==============================] - 1652s 4s/step - loss: 0.2211 - accuracy: 0.9258 - val_loss: 0.3356 -
```

val_accuracy: 0.8547
Epoch 5/10
391/391 [==============================] - 1625s 4s/step - loss: 0.1890 - accuracy: 0.9391 - val_loss: 0.3675 - val_accuracy: 0.8547
Epoch 6/10
391/391 [==============================] - 1668s 4s/step - loss: 0.1649 - accuracy: 0.9505 - val_loss: 0.4221 - val_accuracy: 0.8615
Epoch 7/10
391/391 [==============================] - 1643s 4s/step - loss: 0.1478 - accuracy: 0.9564 - val_loss: 0.4076 - val_accuracy: 0.8552
Epoch 8/10
391/391 [==============================] - 1653s 4s/step - loss: 0.1256 - accuracy: 0.9658 - val_loss: 0.4638 - val_accuracy: 0.8646
Epoch 9/10
391/391 [==============================] - 1648s 4s/step - loss: 0.1198 - accuracy: 0.9671 - val_loss: 0.4462 - val_accuracy: 0.8609
Epoch 10/10
391/391 [==============================] - 1620s 4s/step - loss: 0.1028 - accuracy: 0.9733 - val_loss: 0.5061 - val_accuracy: 0.8542

FINAL ACCURACY & LOSS FUNCTION VALUES:
391/391 [==============================] - 348s 890ms/step - loss: 0.4954 - accuracy: 0.8516
Test Loss: 0.4954008460044861
Test Accuracy: 0.8515999913215637

Graphs:



Exceptions considered:

The Prediction value changes if the padding is True or False; for short sentences, *padding = False* gives better prediction value compared to *pad = True*. However, for large paragraphs, the prediction values remains approximately the same.
Accuracy & Loss Function values vary slightly on every compilation of the model. This may give a slight difference in the prediction value which may impact the neutral sentiments.

Enhancement Scope:

Increasing the accuracy is the biggest enhancement scope. This can be achieved by adjusting the number of layers in the model- it is believed that neither too many nor too less layers can give good

accuracy. Other factors include adjusting the epochs values with respect to the BATCH_SIZE.
Link to Code and executable file: https://colab.research.google.com/drive/1LR0vlfJLbw_e2ECeICkKOqGY7PtiKm7z?usp=sharing https://colab.research.google.com/drive/1peyWG5zP9R3eJdnkw9TZihc6imyenqXo?usp=sharing