Aashwin Ranjan

1211977596

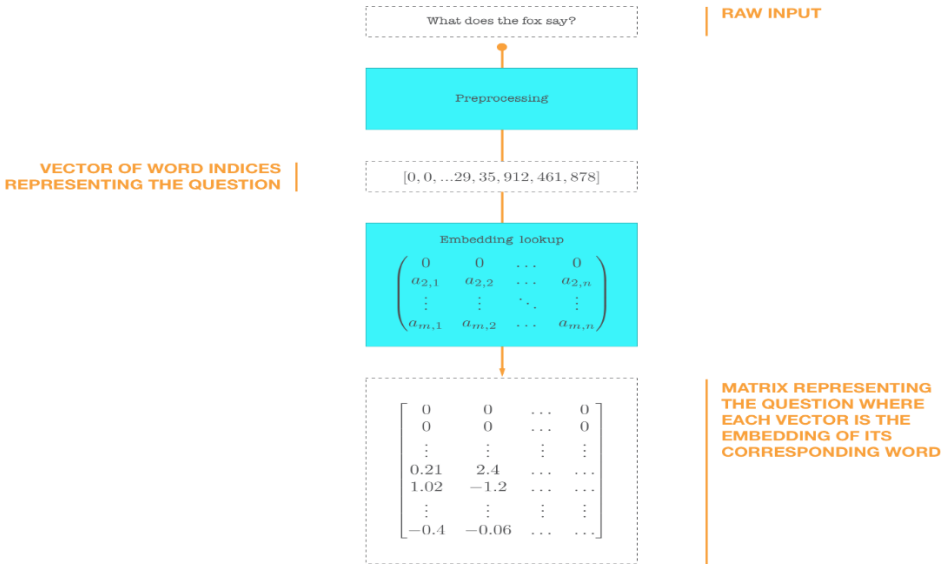
CSE472

Project 2 Report

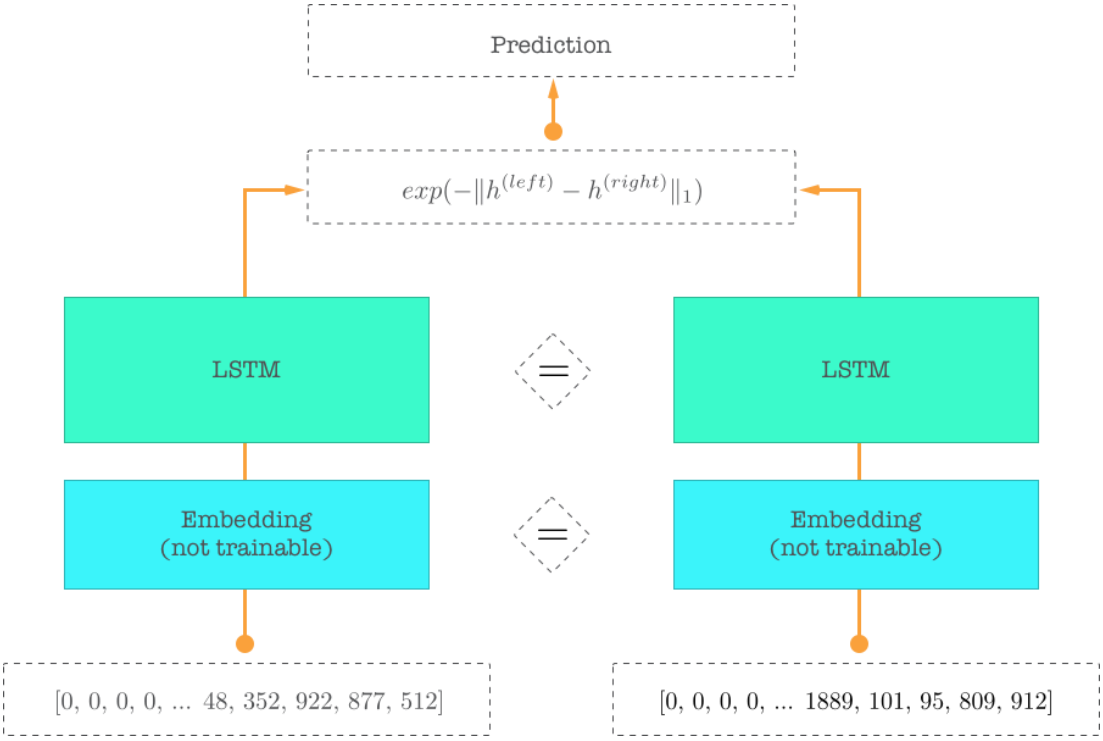
- **Introduction**
  The aim of the project is to train a Machine Learning model to identify fake news using the dataset provided, which contains fake and upcoming news title pairs with their subsequent labels outlining if the articles agree, disagree or are unrelated. The titles also contain their individual id's and the pair id. The model I decided on is a Siamese network LSTM that uses Manhattan distance and has 2 identical subnetworks that share weights on both sides. Siamese networks usually perform well on problems like sentence similarity.

- **Preprocessing and Feature Engineering**
  For preprocessing, in the columns with significant information (titles and label), a function textList() is used. The text is cleaned up so stop words and special characters are removed, and is returned in the form of a list with individual words as entries.
  We use google's word2vec to assign word embeddings to analyze words. title1_en and title2_en are converted into word indices, and subsequently converted to their word2vec embeddings so that we have an embedding matrix for our titles. The labels are converted to 0,1 and 2 using the pandas.factorize.

  Preparing the data: We divide our data into 2 parts for each side of the MaLSTM and then train the model.



- **Model Description:**
  The model is a Siamese MaLSTM (Manhattan distance LSTM). The input is 2 embedded matrices, that represent the fake news article and the upcoming article each. The model calculates and gives put two vectors that encapsulate the semantic meaning of the articles, which are passed through a similarity function that measures the contextual similarity of the inputs.



For the model itself, a similarity function is defined so that after the semantic meaning is caught using LSTM, we can get a value for similarity of the titles. The model is made using LSTM from Keras. The Adadelta optimizer with gradient clipping is used. The model is trained on embedded matrices from the titles and the label.
While training the model, each epoch takes about 30 minutes to completely traverse the data.
This is how the model looks:

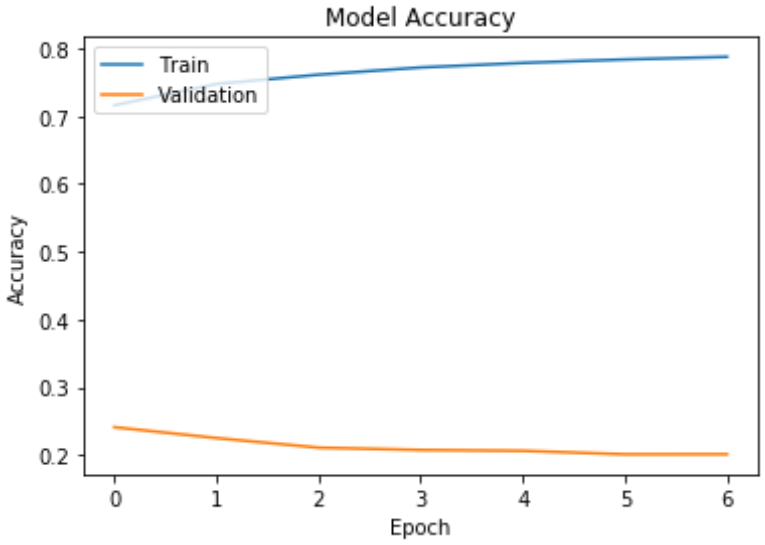
```
Model Saved!
```

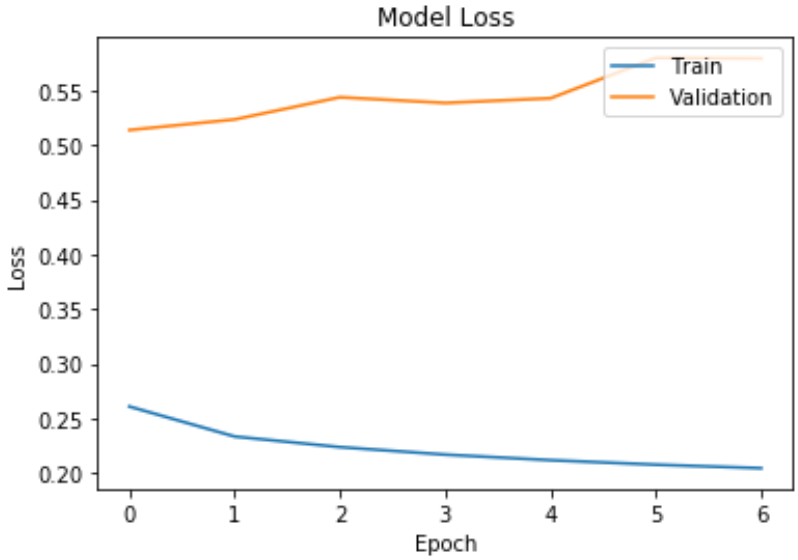
Layer (type)	Output Shape	Param #	Connected to
input_5 (InputLayer)	(None, 500)	0	
input_6 (InputLayer)	(None, 500)	0	
embedding_7 (Embedding)	(None, 500, 300)	12611700	input_5[0][0] input_6[0][0]
lstm_3 (LSTM)	(None, 50)	70200	embedding_7[0][0] embedding_7[1][0]
lambda_3 (Lambda)	(None, 1)	0	lstm_3[0][0] lstm_3[1][0]

```
Total params: 12,681,900
Trainable params: 70,200
Non-trainable params: 12,611,700
```

- **Results**

  The accuracy was low, and loss high due to data being overfitted. The following are the accuracy and loss plots over 7 epochs. The average test and validation accuracy was around .75 and .21 respectively, and the average test and validation loss was about 0.2 and 0.5 respectively.





- **References and Resources**

  Idea, example code, and images: https://medium.com/mlreview/implementing-malstm-on-kaggles-quora-question-pairs-competition-8b31b0b16a07