NLA Project - Intermediate Report

COMMUNITY QUESTION ANSWERING

Duplicate Question Detection

Paryul Jain (20171083)

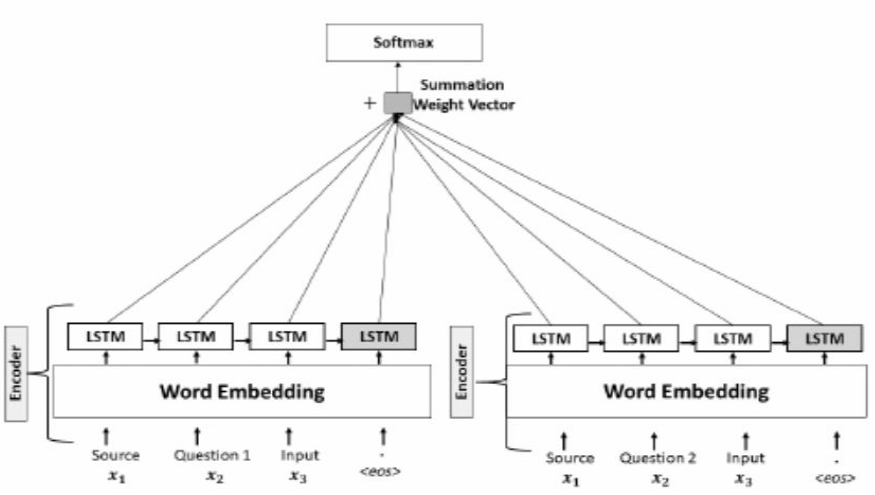
Eesha Dutta (20171104)

OBJECTIVE

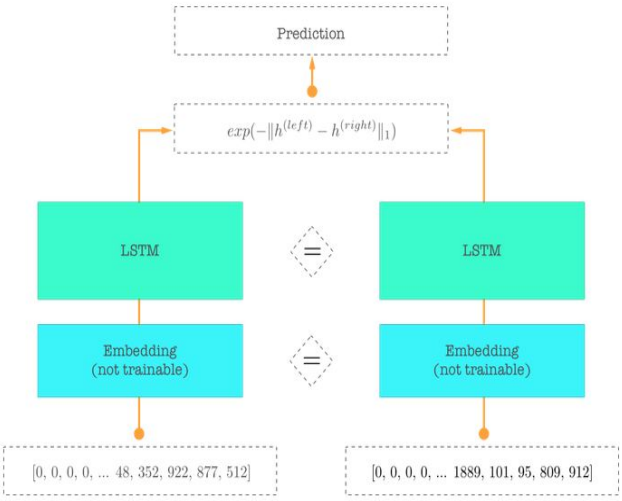
The main goal of this task is to find identical questions pairs. Our model will classify a pair of questions into two categories - duplicate or not-duplicate.

BASELINE MODEL and PROGRESS

The first model that we implemented is Question-Question LSTM given in Question-Question Similarity in online forums by Yallis Chali and Rafat Islam. We first compute the embeddings of the words of both the questions using Word2Vec. The embedding weights are shared across both the questions and these embeddings are pretrained. We then pass these embeddings to our encoder (unidirectional LSTM) which receives the d-dimension word vectors as input to learn the hidden annotations ($h_t = f(x_t, h_{t-1})$). We then aggregate over these vectors by summation for each question and pass this through a **softmax** layer for classification as duplicate or non-duplicate.



We also implemented the MaLSTM Model as given in Siamese Recurrent Architectures for Learning Sentence Similarity by Jonas Mueller and Aditya Thyagarajan. Similar to the previous model, we generate the embeddings, then pass these embeddings to a unidirectional LSTM and obtain the hidden annotations of each sentence. Then, we calculate the **Manhattan distance** between the two vectors as $\exp(-||h^{left} - h^{right}||_1)$ and since we have an exponent of a negative the output (the prediction in our case) will be between 0 and 1.

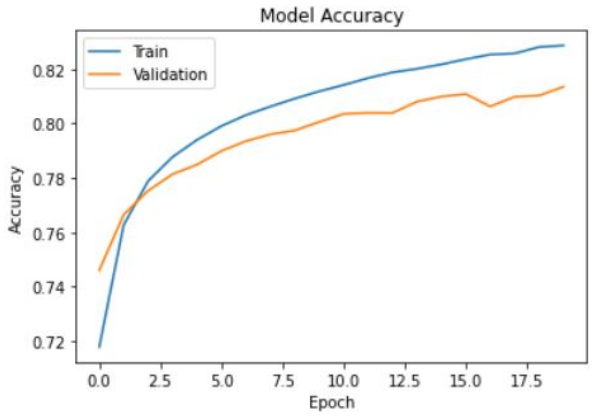


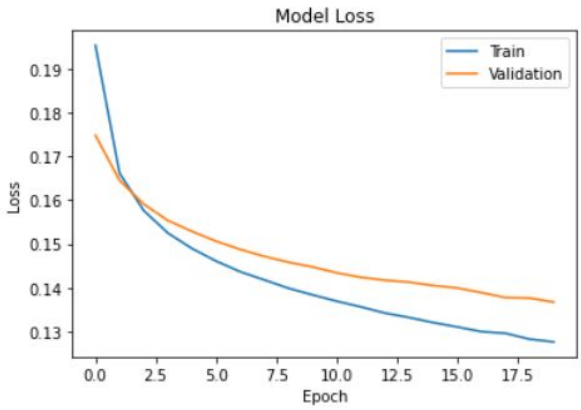
We also tried another method where we used **Cosine distance** between the two sentence vectors instead of manhattan distance.

RESULTS AND ANALYSIS

LSTM with Softmax Classifier gave a low accuracy of 40%. We saw that using Cosine Distance instead of Softmax wass achieving an accuracy of 63%. Using Manhattan distance improved the model's accuracy significantly to 81%. We researched the cause for this difference which led to our finding that using l_2 rather than l_1 norm in the similarity function can lead to undesirable plateaus in the overall objective function. This is because during early stages of training, l_2-based model is unable to correct errors where it erroneously believes semantically different sentences to be nearly identical due to vanishing gradients of the Euclidean distance. We found that utilizing the Manhattan distance outperforms other reasonable alternatives such as cosine similarity.

Model	Accuracy
LSTM with Softmax Classifier	40.03%
LSTM with Cosine Distance	63.22%
LSTM with Manhattan Distance	81.47%





CODE

https://drive.google.com/drive/folders/19KZ8ygnan0MFkBq8jZvBhUsVmg9AI82o?usp=sharing

TRAINED MODELS

https://drive.google.com/drive/folders/1VJvKoi_9B6vbIRba_fyMDNDmBQ8Ad1EM?usp=sharing

FURTHER PLAN

We would be exploring the other three models in Chali et al - LSTM with attention, Bi-directional LSTM stacking with attention and Bi-directional LSTM with attention, CNN and max pooling. We would also be exploring Siamese Convolutional Neural Network to find semantic similarity between the question pair.

We would also like to try out some experiments on these models like - using different Embeddings / training our own embeddings, hyper-parameter tuning etc. After exploring all these models, we will provide a detailed comparison and result analysis of all the models we have implemented.