Josheta Srinivasan
jsriniva@usc.edu

CAIS pre-semester project write up

Choosing a Task type

I used a classification task because the output was a categorical variable: i.e. discrete variable with only two possible values: a positive or negative valency (I know there are three valence categories - positive, neutral, and negative – given in problem statement but looking at the dataset provided I found that there were 0 neutral data samples. So, I chose to restrain my output valence values to only positive and negative).

Pre-processing

I preprocessed my data by first removing all mentions and hashtags off of the tweets, changing all tweets to lowercase, removing punctuations, and removing stop words, and numbers. I tried stemming using PortStemmer but model did not train as well as not stemming at all or Lemmantizing instead. (So, I chose lemmantizing). I them converted all the tweets to word embeddings (using the glove pre-trained word embeddings) and padded them. As for the valence values, I basically converted them to binary values of 0 and 1 (0 for negative and 1 for positive).

Features and Targets

Feature/Target		Domain	Type
Features	Encoded Tweet	array of numbers	Numpy array
Targets	Valence	{0,1}	integer

Table 1

Model

I chose to use a LTSM recurrent neural network because of the nature of the features: they were sentences that were time dependent. As for the architecture, I needed a model that was complex enough to extract enough information but not so complex that it becomes difficult to train. I settled on only using the following architecture with 2 LTSM layer and 2 Dense layers. I found that reducing either number of layers resulted in poorer training and didn't want to add another layer because I didn't want to train my models for hours and hours and also didn't want my laptop to overheat ://. As for the sizes, I just randomly chose a multiple of two for the first three layers (the last dense layer had to be 1 node because of the target being a binary variable). For the activation function, I chose RELU for all the layers except for the last (which I used sigmoid because I wanted a binary value for the target) because it's popular and seemed to work. (I also had three drop out layers of 0.2 dropout between each layer)

I first tried using RMS prop as the optimizer but found there to be excessive gradient explosion (event after adding normalization and dropout layers). I then tried ADAM optimizer which seemed to optimize the network as expected. So, I used the ADAM optimizer. For the loss, I used binary cross entropy because the target was binary.

Josheta Srinivasan
jsriniva@usc.edu

Some hyper parameters used are specified as follows:
Epoch size = 100
Batch size = 512
Validation Test Size = 5%

<u>Splitting data</u>
I split the data the training data set 40-60 (train, test) and validation data set 95-5.

<u>Evaluation</u>
Accuracy = 78.8%