ASSIGNMENT 4

SBU ID: 112673842

EXPLANATION OF GRU AND ATTENTION IMPLEMENTATION

In the attention function:

I calculated the value of M by using tanh(rnn_outputs) as mentioned in the research paper. Then alpha is calculated by applying softmax on the product of M and omegas. Here omegas is a matrix with random value outputs from a normal distribution. Reduce sum is applied on the product of rnn_outputs and alpha on axis =1. Finally tanh is applied on the final result obtained.

In the call function:

A concatenated word is formed using the word embedding and their pos tags. This is sent as an input to the GRU layer by masking it with a matrix where the values are not equal to 0 in the input matrix. This is then sent as a parameter to the attention function. The output obtained from this is sent to the decoder.

Basic Model F1 score- 0.6061 (word+pos+dep for 5 epochs)

OBSERVATION OF GRU EXPERIMENT 1

When the model is run with only word embedding features:

The f1 score is noticed to be 0.441. This is less than the f1 score obtained by using word+pos+dep (0.6061). This is also less than the case where word+ dep has been used

OBSERVATION OF GRU EXPERIMENT 2

When the model is run with word + pos features:

The f1 score is 0.3709 in this case. This case has the least f1 score out of all.

OBSERVATION OF GRU EXPERIMENT 3

When the model is run with only word + dep structure features:

The f1 score is 0.5829 in this case. This case has the second highest f1 score.

JUSTIFICATION OF ADVANCED MODEL CHOICE

I implemented an advanced model for relation extraction using CNN to improve the accuracy of the basic model. I referred to the research article https://www.hindawi.com/journals/cin/2019/6789520/

To justify that usage of CNNs is better than GRU for relation extraction. CNN can extract locally sensitive information from sentences represented by word vectors and apply to relation extraction.

Most CNN models for Relation extraction use the word vector in the sentence directly obtained from a single training model as the input and extract features. On using a convoluted layer for the same purpose I noticed a significant improvement in the f1 score for the task as compared to when it is performed using GRU. The hidden layers of a CNN consist of convolutional layers, pooling layers.

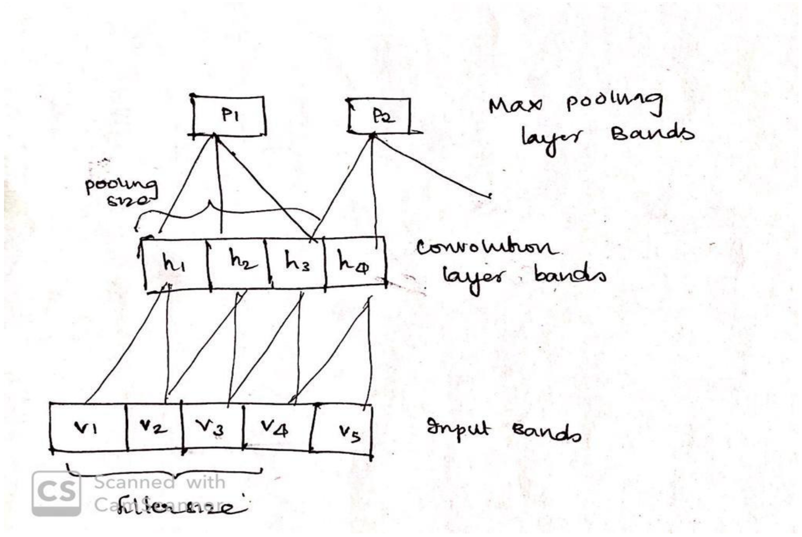

Figure 1: Advanced model implementation using CNNs

IMPLEMENTATION:

- In the advanced model, I created a convolution layer using Keras.Conv2D Class The parameters passed to this layer are the number of filters the convolution layer will learn from and dimensions of the kernel. An activation method 'relu' is also passed as a parameter to this layer
- The word embed and pos embed are concatenated and expanded to 4 dimensions to pass as an input to the convolution layer. This expansion is done because Conv2D expects an input in 4 dimensions.
- I further used max pooling to reduce the spatial dimensions of the output obtained from the Conv2D layer.
- This output is flattened using the flatten() method to obtain the extra dimension
- The output obtained the previous step is passed to dense layer with a softmax activation function. This is also a regular densely-connected NN layer

DIFFERENT CONFIGURATIONS OF THE ADVANCED MODEL:

- **CONFIGURATION 1:**
  Filter size of convolution layer: 64
  Kernel size: (2,2)
  Number of epochs for training: 10
  F1 score: 0.6466
  Only word and pos are considered for this configuration.

- **CONFIGURATION 2:**
  Filter size of convolution layer: 64
  Kernel size: 3
  Number of epochs for training: 10
  F1 score: 0.7273
  Only word and pos are considered for this configuration.
  This is the best configuration for advanced model
- **CONFIGURATION 3:**
  Filter size of convolution layer: 32
  Kernel size: 3
  Number of epochs for training: 10
  F1 score: 0.6377
  Only word and pos are considered for this configuration.