Indian Institute of Technology Gandhinagar

CS-613, Natural Language Processing

Assignment 3

Name:
Nidhin Harilal

Roll Number:
17110092

Contents

1 Task **2**
1.1 Github Repository 2

2 Description of Model **2**
2.1 Pre-Processing 2
2.2 Architecture 2
2.3 Regularizers 3

3 Metrics **3**

1 Task

Build a classifier that can classify the $Eng - Hin$ code-mixed tweets based on their sentiments.

1.1 Github Repository

https://github.com/cryptonymous9/CS-613-Assignments

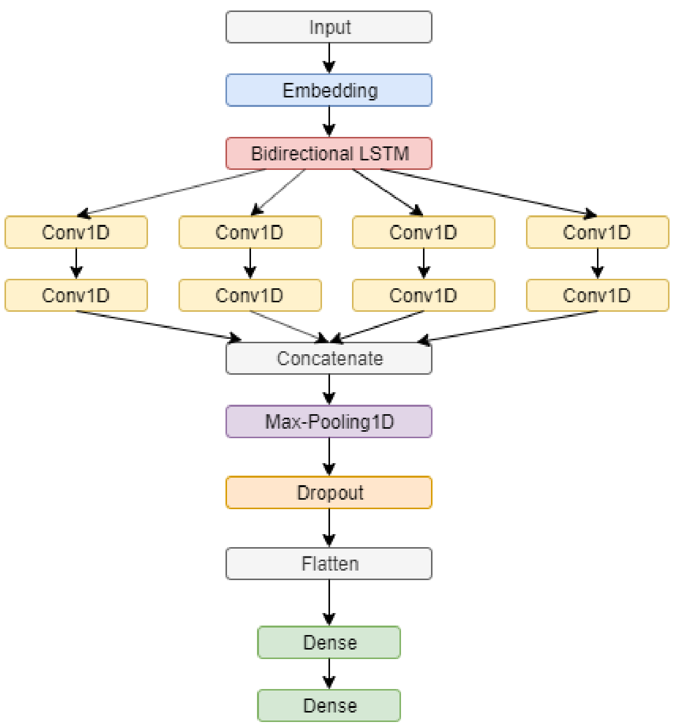
2 Description of Model

2.1 Pre-Processing

Since, the data was of raw Tweets which consisted of English-Hindi texts, a lot of pre-processingwas involved. Some of the major steps in Pre-Processing involved

- Removal of all the Usernames, Hashtags
- Removal of any text which was did not belong to pure alphabets
- Removal all single alphabets and links
- A lookup was performed to translate some of the Hindi words to English using a *Hinglish dictionary*(4)

2.2 Architecture



The Architecture is based on parallel Convolutional 1D layers. First, there is an Input layer with dimensions as $max\_sequence\_length = 31$. The number 31 is actually the longest length of the sentence present in the data. Each sentence (training example) get passed as a one-hot encoding in it. It passes through an Embedding layer which helps in converting the one-hot encodings $shape = (batch\_size, 31)$ to high-order word embeddings, the shape coming from each training instance becomes $shape = (batch\_size, 31, 100)$. After that, it passes through a bidirectional LSTM having 8 LSTM cell.
Then, the output from this gets passed on to 4 identical blocks, each containing 2 Convolutional 1D layers. Hence, These form a 4 parallel Conv1D model. After passing through each of the four blocks parallely, The sequence from these 4 are concatenated to form a single long sequence. After Concatenation, the sequence go through a Max-Pooling1D layer followed by a flattening layer. Since, the sequnce has been flattened to a single dimension, therefore, it is then passed through 2 stacked Dense Layers. The last layer consists of 3 Neurons, responsible for predicting the correct class($classes = 3$).
All the convolutions and the Dence Layers involved except the last one follows an activation of $relu$, whereas the last Dense Layer is followed by an activation of $softmax$ in order to guide the class prediction. $Adadelta$ has been used as optimizer, However, $RMSProp$ also performs very well in this case. The loss function that has been used is $Categorical\_Crossentropy$.

2.3 Regularizers

Different Regularization techniques has been used to avoid Over-fitting and helping in increasing the test accuracy. The LSTM involved has a dropout of 0.3 and a recurrent dropout of 0.3. There is also a Dropout layer just after the Max-Pooling1D layer which has a dropout rate of 0.4. Each of the second Convolutional1D layer in the block and the second last Dense layer has l_2 kernel regularizer with weight decay value of 0.02.

3 Metrics

Train Accuracy: 85.25 **Test Accuracy:** 59.74

```
In [73]: from sklearn.metrics import classification_report
         y_pred = np.argmax(custom_Model.predict(test_input_sequences), axis=1)
         report = classification_report(np.argmax(test_label, axis=1),y_pred)
         print(report)
```

	precision	recall	f1-score	support
0	0.61	0.66	0.63	532
1	0.58	0.50	0.54	754
2	0.60	0.67	0.63	582
accuracy			0.60	1868
macro avg	0.60	0.61	0.60	1868
weighted avg	0.60	0.60	0.59	1868

Figure 1: Precision, Recall and F1-Score of all three classes

References

[1] Laura Barnes, Donald Brown *Text Classification Algorithms: A Survey*

[2] Nishit Shrestha, Fatma Nasoz *Deep Learning Sentiment Analysis of Amazon.com Reviews and Ratings.*

[3] *Keras:* https://keras.io/

[4] *Github:* precog-iiitd/mind-your-language-aaai

[5] *Sk-learn:* sklearn.metrics.classification_report