CS613 Assignment 3 Report

Ribhu Vajpeyi (17110124)

Date of Submission : November 17, 2019

1 Problem Statement

Propose an architecture for stance detection in Hindi-English code-mixed tweets data.

2 Introduction

Presented is a system to detect stance in Hindi English code-mixed data. The workflow is as follows:

1. Process given Test and Train data to clean the tweets (remove hashtags, usernames, URLS etc)
2. Transliterate the cleaned tweets dataset through the the three step decoding method described in [1]
3. Translate the transliterated tweets using Google Translate API
4. Generate Embeddings via BERT-cased using 'flair' python library

I then utilise three architectures to detect stance - **BiLSTM + Self Attention and GRU + Self Attention**.

3 BiLSTM + Self Attention

BiLSTM is an RNN architecture with both forward and backward pass. This means that the input is passed in two ways -

1. From backward to forward thus retains information about past
2. From forward to backward thus retains information about future

This is possible because it uses two hidden states combined at any point in time to preserve information from both past and future. This is different from an LSTM architecture which passes input from only backward to forward and thus only retains information from the past. This has an advantages in NLP based applications since then the entire context of the data is taken into account. I also use a self attention layer to prioritise the inputs given in each iterative step. The architecture is shown in Figure 1.

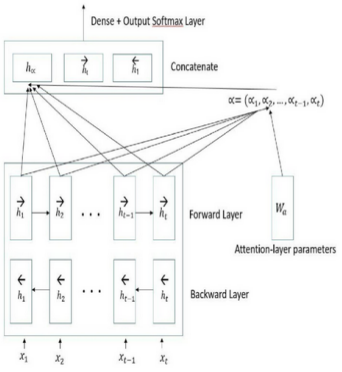

Figure 1: BiLSTM + Self Attention

4 GRU + Self Attention

GRU is also an RNN architecture. It utilises two gates- reset and update gate in caparison to LSTM which utilises three gates- input, output and forget gates. GRU infact couples the forget and input gates of LSTM. As a result, it has no memory unit to control the flow of information like LSTM. Hence it has less training parameters and hence is faster.

GRU has two values at output- output and hidden. I also use a self attention layer to prioritise the inputs given in each iterative step.

5 Observations and Conclusion

The results obtained are shown in the python notebook attached. Kindly refer to it for the same.

6 References

1. Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Manish Shrivastava, and Dipti Misra Sharma. 2018. Universal Dependency Parsing for Hindi-English Code-switching. CoRR abs/1804.05868 (2018). arXiv:1804.05868 http://arxiv.org/abs/1804.05868