STS task report

Summary

This task is a Semantic Textual Similarity task, a regression problem. In this task, I combine NLP traditional features with DeBerta V3 to do an ensemble.

Instructions

I first did the feature extraction on my PC (as Stanford coreNLP requires a Java environment) and then ran the ensemble part on Colab.

The Python version is recommended greater than 3.6 and lower than 3.10. All python environment requirements are in requirements.txt and Stanford coreNLP has been packaged in the data directory.

code structure: I divided the code into 2 parts. The first part is feature extraction, which is the zip file and the second part is the regression model, which is on the Colab url: https://colab.research.google.com/drive/1FaafgmetDBPwP7ouvYThQBGRfuaEZQWW?usp=sharing.

The structure of feature extraction is shown below

the data folder contains the dataset and Stanford NLP.

the stopwords folder contains nltk stopwords, you could also use `python -m nltk.downloader stopwords` to generate it.

the stst folder is the model folder, which contains the parser data generated by Stanford NLP, the model used to extract the features.

the download.sh contains the way to download stanford NLP, stopwords and so on, but I already download them in this zip file.

the README.md is the readme of the original author, although I modified some codes, I still keep it.

Run the gen_freatures.py to get the features, and train the model.

usage Instructions:

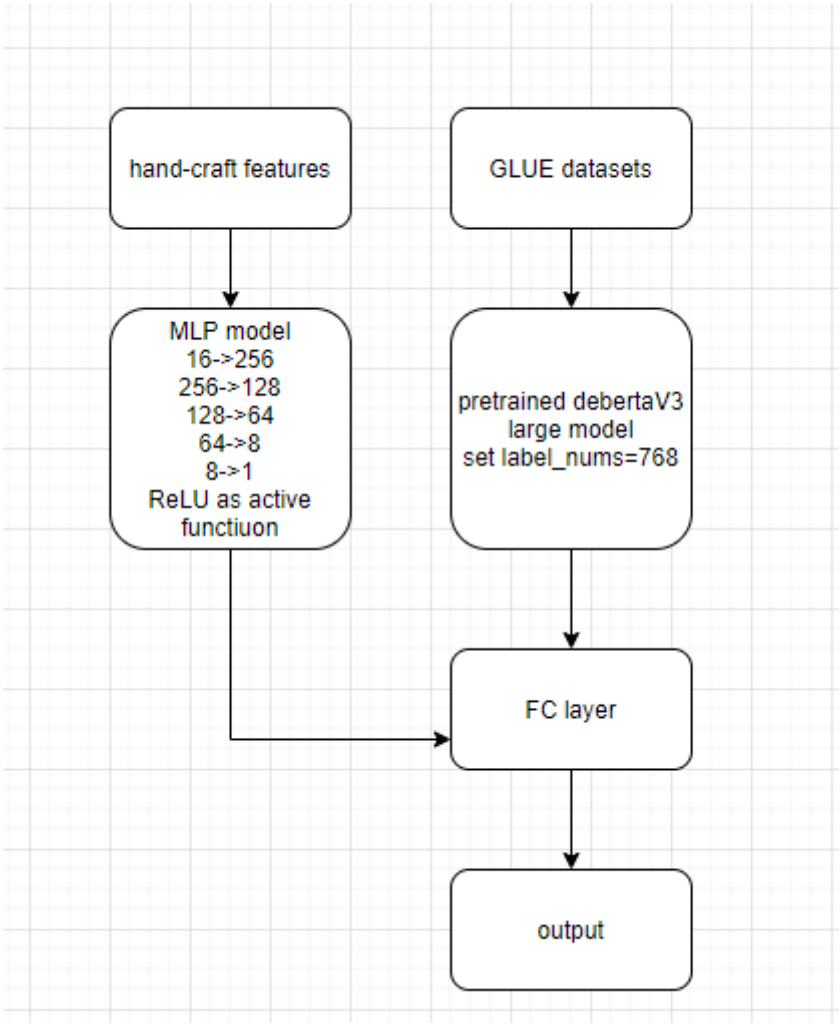
1. Open the NLP server: open the terminal and input the following code:

```
cd cd .\data\stanford-corenlp-full-2015-12-09\
java -mx4g -cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer
```

2. run gen_features.py
3. waiting for training and predictions. there will be a "preds.csv" file after predictions.

Approach

The model diagram of the algorithm is as follows:



Inspired by my MSc project, I think I might do a ensemble with a SOTA model. I chose the debertaV3 as a target since it is a outstanding model, and it is very new, the result of that pretrained model should be good. Also, transfer learning is perform well on NLP.

- the feature part:

  I have modified an open source project in Github, coding by ECNU. The link is: [rgtjf/Semantic-Texual-Similarity-Toolkits: Semantic Textual Similarity (STS) measures the degree of equivalence in the underlying semantics of paired snippets of text. (github.com)]

  In this project, I extracted the Sentence Pair Matching Features and Single Sentence Features.

  - Sentence Pair Matching Features: weighted n-gram overlap feature, Alignment Feature, Idf Alignment Feature.
  - Single Sentence Features: BOW Features and Negative Features.

- the ensemble model:

  The ensemble model is contains a MLP model and a debertaV3 large model, the reason I chose the label numbers as 768 is that the deberta model should be the main model, so the output's number should be higher. I did not choose 1024(the hidden layer size of the deberta model) is because I think the MLP model should have more "weights".

model	spearmanr
feature-based model using RF & XGB boosting	0.7680
BERT	0.8320
ensemble debertaV3 Large (change the sentence sequence)	0.9230
debertaV3 Large	0.9230
ensemble debertaV3 Large (with traditional features)	0.9251

Feature works

I change the different random seeds in my model, the model is stable, but the model should perform better if I have a GPU with higher RAM. Therefore, I may adjust the parameters in the future.

More features should be added. According to the paper https://aclanthology.org/S17-2028, they also have extracted other features(shown below), and I still need to rebuild the whole project.