PA2 Report

1. Code structure:

---Run.py

- Display _exec_time: Display execution time
- Run_scorer: run scorer.py
- Run:
  - Pad_Trim: Pad and trim sparse matrix
  - Pad_Trim_Feature:
  - Get_glove_dataset:
  - Get_glove_vocabs: Build up the vocabulary
  - Build data iterators and initialize the model
  - Shuffle
  - Batchify
  - Train: train a model
  - Evaluate: Evaluate the performance of a model on dev set
  - Evaluate2: Evaluate the performance of a model on the test set.
- Write_json: write json for evaluation
- run scorers

---Model.py

- Class LR: Logistic Regression model with only 1 layer
  - Reset_parameters: Reset the parameters
  - Forward: Forward Propagation
  - Representation: Naïve one hot encoding using sparse matrix
  - Hyperparameters: NA
- Class LR_Dense: Neuron nets having 3 hidden layers
  - Reset_parameters: Reset the parameters
  - Forward: Forward Propagation

Representation: not pretrained embedding, dense vector
Hyperparameters: Embedding size, the channel number of the hidden layers

Class LR_Glove: Neuron nets having 3 hidden layers
Reset_parameters: Reset the parameters
Forward: Forward Propagation
Representation: dense vector based on pretrained GloVe embedding
Hyperparameters: Embedding size, the channel number of the hidden layers, pretrained embedding

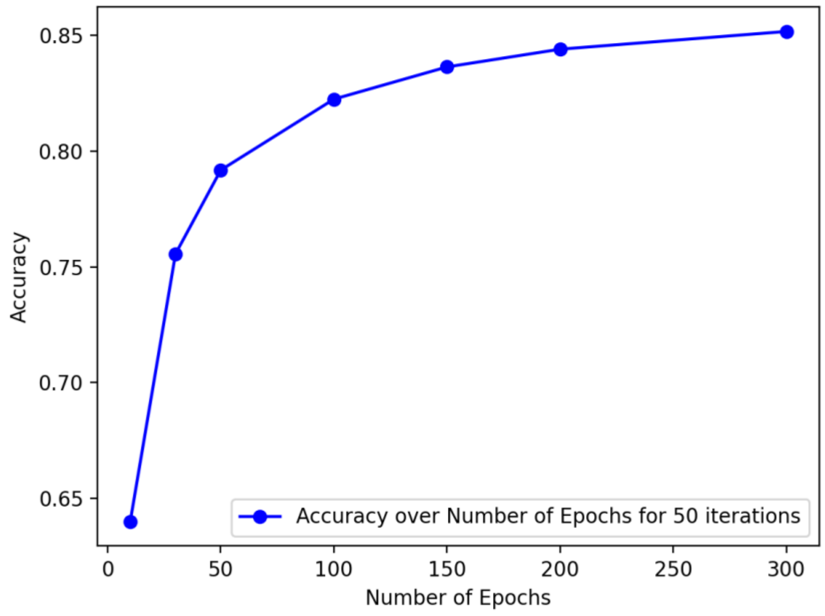
Class LR_CNN: CNN network with 1 convolution layer and 1 max pooling layer
Reset_parameters: Reset the parameters
Forward: Forward Propagation
Representation: dense vector based on pretrained GloVe embedding
Hyperparameters: Embedding size, the channel number of the hidden layers, pretrained embedding

Class Combined_CNN: Concatenate the output of maxpooling of LR_CNN and the output of one layer of neuron net, conduct the multiplication of the combined output
Reset_parameters: Reset the parameters
Forward: Forward Propagation
Representation: dense vector based on pretrained GloVe embedding
Hyperparameters: Embedding size, the channel number of the hidden layers, pretrained embedding

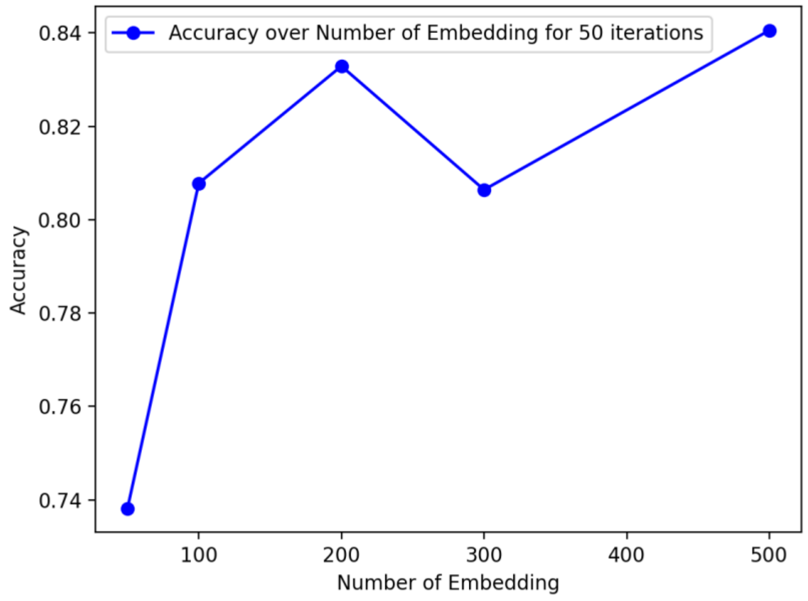
Removed the unnecessary classes and methods of corpus.

How to run:
For the first time of running, comment row 323-328 to execute run.py to get the json files, and the uncomment those lines and run run.py. to score

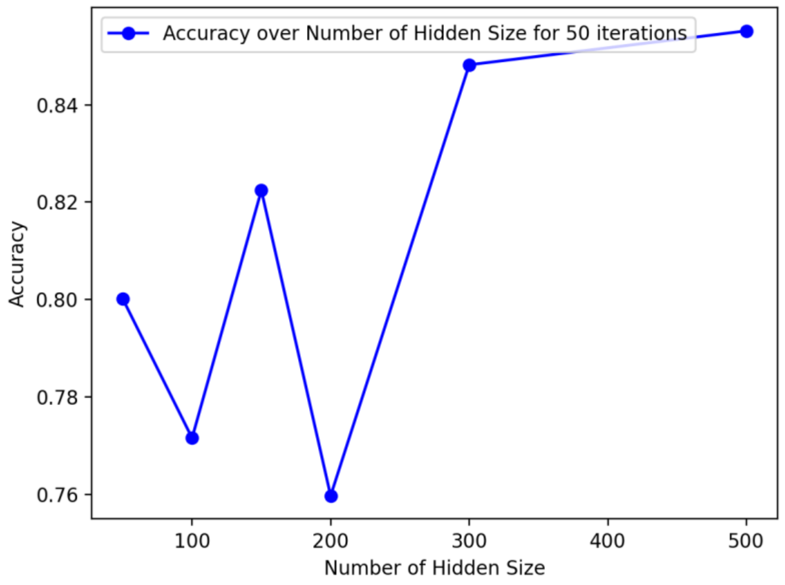
Results:



The accuracy is monotonic increasing upon the time for LR.



The optimal embedding number is 500



The optimum hidden size is 500 for dense representation.

For all models, the non-explicit discourse have the best performance.

Using dense representation and adding hidden layers slightly improve the accuracy.

Using pretrained embeddings does not really increase the accuracy.

CNN has a much better performance on the dataset.