11747 Assignment 1

Dahua Gan (dgan@andrew.cmu.edu)

Initial Interest Survey

Our team members are: George Xu (jiayuanx), Dahua Gan(dgan) and Bei Zhou (beizhou). We are currently planning to start our work in the field of Commonsense Reasoning as well as Fake News Detection.

Data preprocessing

I defined a DataProcessor class to do all the preprocessing stuff and the preprocessed data will be sent to the Dataset and DataLoader class defined in PyTorch. Specifically, the DataProcessor will read from given text file, separate the label and text feature. The label will be converted to indexes for classification. As for texts, the strings are tokenized and lemmatized using WordNet, it will also construct a Vocabulary class, mapping each token to a number. Finally, each line will be processed as a label index, as well as a list of token index. Then it will be sent to Dataset and DataLoader. In the self-defined collate function, it will automatically pad the sentences within the batch to the maximum sentence length, which can be used for matrix multiplication in PyTorch.

Model Architecture

I simply adopted the CNN for Text classification baseline [1]. It contains three parts. First part is embedding part. I loaded the pretrained word embedding (Glove 42B 300) and initialize the embedding layer with it (leaving unknown tokens randomly initialized). Next is the convolution and pooling part. The model performs a 1-D convolution in the direction of sentences. There're 3 sets of filters with size 3, 4, 5, each set contains 100 filters. Then, I performed a max-pooling over time to get the most salient feature for each filter. Finally, I concatenate the outputs of the max-pooling layer and connect it with a fully connected layer to perform a non-linear transformation to the final prediction. Dropouts and Batchnorms are added in the FC layer.

Hyperparameter Tuning and Optimizing

The hyperparameters I played with are filter size, CNN layers, dropout rate, learning rate, weight decay, etc. The general idea is that regularization need to be enough, otherwise the model tend to overfit. The optimizer I used is Adam with lr=0.001, weight_decay=0.0001

Result and Discussion

The final accuracy of the model reaches around 83% at the end. The main finding is that freezing the parameter of embedding (initialize it with pretrained embedding) is significantly helpful compared to both initialize it randomly, or using the pretrained embedding as initial value, yet still get updated while training. Probably the CNN classifier can perform well as a N-gram detector when it knows the well-learned semantics of each tokens, yet it's quite hard to learn the semantic from scratch when the dataset is small.

References

[1] Kim, Y. (2014, October). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746-1751).