Paraphrase Identification System Description

Eric Nguyen

December 4, 2022

Features

For this project, I used trial-and-error to test different features and their impact on the F1 score. I exclusively used features based on similarity metrics defined in the TextDistance package for simplicity, including:

- Jaccard index
- Sørensen-Dice coefficient
- Cosine similarity (qval=3)
- Cosine similarity (qval=4)
- Cosine similarity (default)
- Levenshtein

Data Preprocessing and Feature Preprocessing

All sentences were lowercased, and the records with positive labels were oversampled to match the number of records with negative labels.

Algorithms and Libraries Used

I construct a simple multi-layer perceptron with one hidden layer and 30 nodes for the hidden layer. The number of hidden layers and nodes for each hidden layer is selected through trial-and-error. I use the Gaussian Error Linear Unit (GELU) activation function for the hidden layer as it is used in popular NLP models such as BERT and I use the Tanh activation function for the output layer as it is best for binary classification. To train the multi-layer perceptron, I use a cross-entropy loss function as it is good for binary classification problems. For the optimizer, I use the AdamW algorithm since I found it yields better results compared to stochastic gradient descent (SGD) and other algorithms through trial-and-error. The optimizer uses the default parameters with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.

I use 4500 epochs to train my multi-layer perceptron as this number yields the best F1 score through trial-and-error.

To construct and train the multi-layer perceptron, I used the PyTorch library. To structure the data into DataFrames, I used the pandas library. To compute the F1 score, I used the scikit-learn library.

Lessons Learned

Through this project, I learned…

- Deep learning takes a great deal of computational power and memory.
- Deep learning requires a lot of tweaking of parameters to obtain good results.
- How to construct and train a neural network and apply it to a classification problem.
- Components of deep learning such as input layer, hidden layer, output layer, activation function, loss function, optimization algorithm, number of epochs, etc.
- Class imbalance greatly impacts model performance.