CS388 NLP HW1

1 Part 1

To perform sentiment classification using logistic regression, I implemented the bag-of-words unigram featurization. Initially, I looped through all the words in the training set and generated an all-zero vector of length equal to the number of words in the training set. Then, I iterated through each word in every sentence, incrementing by 1 the corresponding index in the initialized feature vector. Consequently, for each sentence, I obtained a feature vector of the same length. These sentence feature vectors were then concatenated to form a feature matrix, and all label vectors were similarly concatenated. I trained the logistic regression model using the feature matrix and label vector as inputs, employing `CrossEntropyLoss` as the loss function, a fixed learning rate of 0.03, and trained it for 100 epochs. The final accuracy on the development set was 0.77.

1.1 Exploration 1

I experimented with different learning rates, including a fixed learning rate of 0.03, a continuously decreasing learning rate (fixed learning rate $-$ 0.0001 $\times$ epoch), and fixed learning rate/epoch. As illustrated in Figure 1, all three learning rates yielded similar accuracies. However, the accuracy using fixed learning rate/epoch was slightly lower in both the training and development sets compared to the other learning rates. I also observed that the training set accuracy was significantly higher than that of the development set, which may indicate overfitting.

1.2 Exploration 2

For bigram featurization, similar to **Part 1**, I used two consecutive words as elements in the feature vector instead of one. However, the model's performance with bigram featurization was slightly lower compared to unigram featurization. The development set accuracy for the bigram model was 0.72, but the training set accuracy almost reached 1, indicating significant overfitting. I also experimented with unigram featurization, but removed rare words that appeared only once in the entire training set. This approach resulted in a development set accuracy of 0.75, which was still slightly lower compared to using the full feature vector.

2 Part 2

To implement the Deep Averaging Network, I used `data/glove.6B.300d-relativized.txt` to vectorize each word in the sentences. I then averaged each word vector and concatenated these averaged vectors for all sentences in the training set. I also concatenated all labels into a vector and transformed it into one-hot encoding form. The fully connected neural network architecture comprised two hidden layers, each with 256 dimensions. I utilized the **Adam** algorithm to optimize the learning rate and trained the model for 500 epochs. Ultimately, I achieved an accuracy of 0.79, which is slightly higher compared to the bag-of-words unigram featurization.

Figure

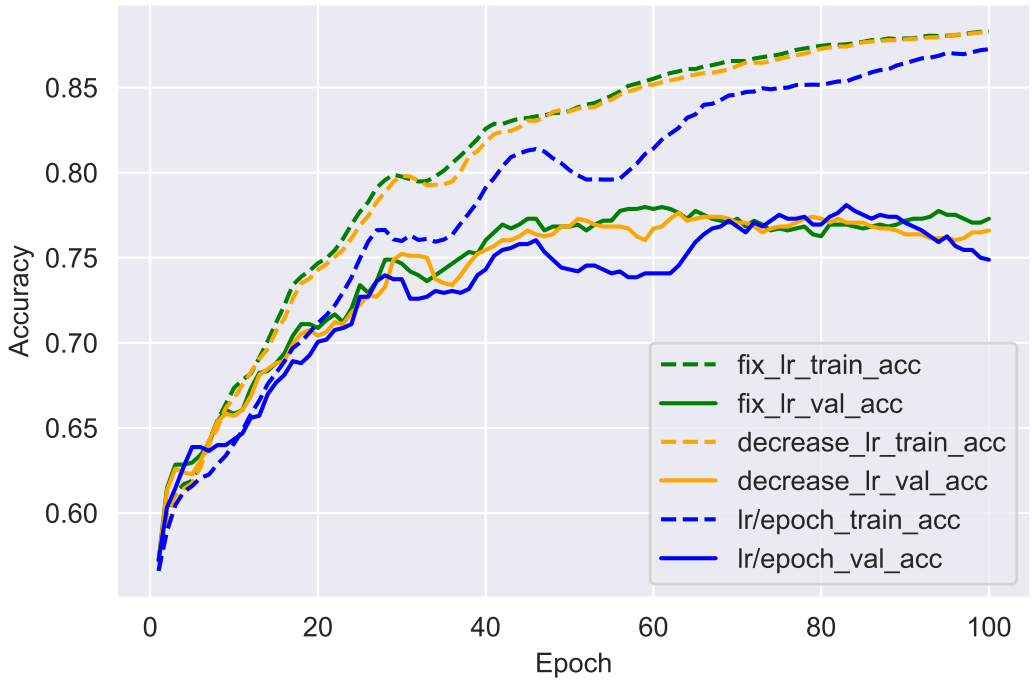

Figure 1: Accuracy