Travis Mayer
SCU ID: 00001184297
CSEN 346
April 24, 2024

Sentiment Analysis using Neural Networks: Logistic Regression and LSTM Approaches

Introduction:

In this assignment, I was tasked with performing sentiment analysis on a dataset of single-sentence reviews from various domains. The goal was to classify each review as either positive or negative sentiment using different approaches and techniques learned in class. The assignment required implementing a logistic regression model as a neural network using PyTorch, experimenting with word embeddings for feature representation, and proposing a custom sentiment classifier by creating different neural networks and applying feature selection.

My Approach:

To tackle this sentiment analysis task, I decided to explore two main approaches. The first approach involved implementing a simple logistic regression model as a neural network using PyTorch. For feature representation, I utilized the pre-trained GloVe word embeddings to capture the semantic meaning of words in the reviews. I also applied 5-fold cross-validation to assess the performance of this model. In addition, I experimented with the hyperparameters and optimization methods.

In the second approach, I aimed to create a more complex neural network architecture and incorporate feature selection techniques. I experimented with different numbers of layers, activation functions, and training tips discussed in class. Specifically, I designed an LSTM-based neural network to capture sequential dependencies in the text data. Additionally, I applied feature selection by excluding stopwords, rare words, and considering bigrams to enhance the classification performance.

Methodology:

To implement the logistic regression model as a neural network, I used PyTorch and created a simple architecture with an input layer and an output layer. The input layer consisted of the word embeddings, and the output layer produced the predicted probability of positive sentiment. I trained the model using binary cross-entropy loss and the Adam optimizer.

For the LSTM-based neural network, I designed a more complex architecture with multiple layers. The input layer took the word embeddings, followed by an LSTM layer to capture sequential information. The output of the LSTM layer was then passed through a fully connected layer and a sigmoid activation function to produce the predicted probability. I experimented with different hyperparameters such as the number of layers, hidden dimensions, dropout rate, and learning rate to optimize the model's performance.

In terms of feature selection, I preprocessed the text data by lowercasing, removing stopwords, and excluding rare words that were not present in the pre-trained word embeddings. I also considered incorporating bigrams to capture important phrases or combinations of words that could impact sentiment classification.

Challenges and Solutions:

One of the main challenges I encountered was the limited size of the dataset. With a small number of training examples, it was difficult to train complex models effectively and avoid overfitting. To mitigate this, I applied techniques such as dropout regularization and experimented with different hyperparameters to find the best configuration.

Another challenge was the selection of relevant features for sentiment classification. Initially, I used all the words in the reviews as features, but this led to the inclusion of noise and irrelevant information. To address this, I applied feature selection techniques like removing stopwords and rare words, which helped to focus on more meaningful and discriminative features.

Conclusion:

Through this assignment, I explored different approaches and techniques for sentiment analysis using PyTorch and neural networks. I implemented a logistic regression model as a neural network and experimented with word embeddings for feature representation. I also proposed a custom sentiment classifier using an LSTM-based neural network and applied feature selection techniques.

Interestingly, despite the complexity of the LSTM neural network, I achieved a very similar cross-validation accuracy score with the simple logistic regression model. However, surprisingly, the loss was actually significantly less for the LSTM model than the logistic regression model. However, it is important to note that this could be due to overfitting in the first model, given the limited size of the dataset. The small dataset made it challenging to tune the hyperparameters effectively and assess the true generalization performance of the models. When tuning the hyperparameters, especially for the LSTM model, I noticed that playing with the learning rate in particular could significantly reduce the loss, but at the same time was often correlated with a slight reduction in the cross-validation accuracy. Again, possibly a symptom of overfitting, but there may be other underlying factors in such a complex model.

Overall, this assignment provided valuable insights into the process of sentiment analysis using neural networks and highlighted the importance of feature selection and hyperparameter tuning. It also emphasized the need for larger and more diverse datasets to build robust and reliable sentiment classification models.