**<https://github.com/ay190323/CS4375-HW2.git>**

**Akif Yorulmaz**

**AXY200028**

**1 Introduction and Data (5pt)**

In this project, FFNN and RNN neural network models have been established to analyze Yelp reviews for five sentiment classes. The goal was, therefore, to categorize each review into one of five general categories depending on its sentiment score of 1 – 5 stars.

The dataset consists of three parts: Three collections of source image data: training data set, validation data set, and test data set in JSON form. The training set is used to train the models, and this set is used to tune hyperparameters and the test set is used to evaluate the final model’s performance. The sizes of the datasets are summarized below:

|  |  |
| --- | --- |
| Dataset | Number of Examples |
| Training Set | **16,000** |
| Validation Set | **800** |
| Test Set | **800** |

**2 Implementations (45pt)**

**2.1 FFNN (20pt)**

The FFNN model takes each review as its fixed-length input vector and applies dense layers after the vector. The last layer applies softmax to assign the review to one of the five sentiments.

***Implementation Highlights:***

* **Data Loading:** For that, we utilized PyTorch DataLoader for batch size and shuffling.
* **Forward Pass:** The proposed model has an input layer, one or more hidden layers and an output layer that uses ReLU and softmax activation respectively.
* **Optimizer and Stopping:** The model utilizes stochastic gradient descent and stops based on the set number of epochs.

A computer screen shot of text

Description automatically generated

**2.2 RNN (25pt)**

To increase the model’s ability to take into account sentiment nuances, the RNN model applies each review as a sequence of word vectors, taking into account the sequence of words.

***Implementation Highlights:***

* **Sequential Processing:** The RNN takes each word at the input at a time and updates the hidden state in each step of processing the word.
* **Differences from FFNN:** In contrast, FFNN requires a fixed number of input features; however, for sequences of different lengths, RNN applies the word embeddings.
* **Softmax Output:** The final hidden state produces a probability distribution over sentiment classes via a softmax layer.

A screen shot of a computer program

Description automatically generated

**3 Experiments and Results (45pt)**

**3.1 Evaluations (15pt)**

Both the models are measured by accuracy which is the percentage of the reviews accurately classified to the total number of reviews in the validation or test data set. This metric is deformation, and it is good as far as giving an idea of how well a model does when it is used in multi-class classification.

**3.2 Results (30pt)**

Following are the results of the training and validation accuracy after each epoch. We also tried varying the size of hidden units and the final performance is presented in learning curves.

***Learning Curves:***

A graph of the same size and the same size

Description automatically generated with medium confidence

**4 Analysis (bonus: 10pt)**

***Learning Curve***

Learning curves represent the change in the model accuracy over epochs, demonstrating that the models’ performances are constantly progressing. To this end, the RNN in particular is well suited for sequential processing, as it was shown to exhibit better performance when the temporal structure of the text is taken into account.

***Error Analysis***

An example of misclassification is when a model forecasts a 3-star when it should have foreseen a 1-star because the two are semantically similar. Further enhancements could be done by utilizing attention mechanisms or by going deeper into an architecture.

**5 Conclusion and Others (5pt)**

This assignment showed a general exploration of using RNNs for sentiment analysis. While training the system, FFNNs were faster as compared to the feedback one but these were not able to capture the context effectively.

***Feedback:***  
This project took around 8 hours and was challenging but insightful, especially regarding sequential data processing with RNNs.