CS6375 Assignment 1

Submission Due: March 12, 11:59pm

In this project we will consider neural networks: first a Feedforward Neural Network (FFNN) and second a Recurrent Neural Network (RNN), for performing a 5-class Sentiment Analysis task.

Starter Code, Data, and Embedding Download: eLearning

Report Template (Including Grading Rubrics): eLearning

1. Dataset & Embedding

You are given (via eLearning) access to a set of Yelp reviews (train, validation and test). These reviews are associated with a rating $y \in Y := \{1, 2, 3, 4, 5\}$[1]. For this project, given the review text, you will need to predict y. This is known as sentiment analysis in the literature. Moreover, we also provide an embedding file (i.e. word_embedding.pkl) for your use in RNN. For this assignment, we do not anticipate any further preprocessing to be done by you. Should you choose to do so, please report this if it was of interest, but it is not a required aspect of the assignment. The data loading is done with the load_data function.

2 Reference Sources & Tips

- For the project, you will be dealing with basic neural network models. Our slides provide introductions to neural models.
- Alongside the course text, this assignment will introduce you to the PyTorch framework and the well-written PyTorch documentation is likely also to be useful[2]. If you have questions regarding setting up PyTorch or using Python, feel free to come to TA office hours.
- **Advice: The report is important!** The report is where you get to show that you understand not only *what* you are doing and *how* you are doing it. Spend some time doing error analysis for the models. This assignment, at its core, involves training and experimenting with neural models. We have made a very committed effort towards making these models run quickly.
- **Tips:** Although the training of models can be finished locally (within 20 minutes usually), we suggest that you use a smaller portion of data for debugging/implementation. You can also run full experiments with Colab Tutorial for speeding up if necessary.

Although we only require filling the forward() function, you may still encounter bugs when completing. We suggest that you carefully review the slides and Pytorch tutorials for other models. You may look for abnormal values step by step as well. Pytorch version 1.10.1 was benchmarked for this assignment, and you may use any library or toolkit at your leisure.

3 Tasks & Models

You are given a partial implementation (in Pytorch) for a FFNN and another for RNN. You are only

[1] In the assignment code, we use {0, 1, 2, 3, 4} to more naturally align with Python's zero-indexing.
[2] https://pytorch.org/docs/stable/index.html

asked to complete the forward computation part in **forward()** by filling in lines starting with **[to fill].** Layers and parameters are defined in the **__init__(self)** function. The implementations are not perfect (e.g. irrelevant comments and spurious parts), you can edit or add at your will.

3.1 Feedforward Neural Network (FFNN)

As introduced in class, for FFNN, it takes as input a fixed length vector to conduct a feedforward pass. The complete implementation should do computation to obtain:

$$\text{input} : \mathbf{x} \in \mathbf{R}^{d}$$
$$\text{hidden layer} : \mathbf{h} \in \mathbf{R}^{|h|}$$
$$\text{output layer} : \mathbf{z} \in \mathbf{R}^{|Y|}$$

where d is the vocab size, $|h|$ the hidden Σlayer dimension. Then we apply softmax to **z**, and obtain **y** which is the probability distribution (i.e.$\sum_{i \in |Y|} y[i] = 1$)
Apart from the implementation of the data loading function and the class FFNN, we also provide functions that do a bag-of-words vectorization process for each review. After correctly finishing **forward()**, you should be able to train the model by specifying hyperparameters and file path:

```
python ffnn.py --hidden_dim [hparam] --epochs [hparam] --train_data [train data path] --val_data [val data path]
```

3.2 Recurrent Neural Networks (RNN)

RNN takes in a sequence of vectors and computes a new vector corresponding to each vector in the original sequence. It achieves this by processing the input sequence one vector at a time and it computes an updated representation of the entire sequence (which is then re-used for the next vector in the input sequence) along with an output for that position. The complete implementation should do computation to obtain:

$$\text{input: } \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k, \mathbf{x}_i \in \mathbf{R}^{e}$$
$$\text{representations after RNN: } \mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_k, \mathbf{h}_i \in \mathbf{R}^{|h|}$$
$$\text{output layer: } \sum_{i=1}^{k} \mathbf{z}_i, \; \mathbf{z}_i \in \mathbf{R}^{|Y|}$$

where e is the embedding size, h the hidden layer dimension. Finally we apply softmax to obtain **y** which is the probability distribution. More specifically, we sum up the vectors for each token of the output layer, as a representation for the entire sequence.

If you read the other part of the code, you will see this time we use word embeddings for initialization of word representations. After correctly finishing forward(), you should be able to train the model by specifying hyperparameters and file path:

```
python rnn.py --hidden_dim [hparam] --epochs [hparam] --train_data [train data path] --val_data [val data path]
```

4 Guidelines

Development Environment and Third-party Tools We recommend using Python of version 3.8.x. Do not use older or newer version. We strongly recommend working within a fresh virtual environment for each assignment. For example, you can create a virtual environment using conda and install the required packages:

```
conda create -n cs6375 python=3.8 conda
activate cs6375
pip install -r requirements.txt
```

Submission, Grading, and Writeup Guidelines We'll have access to your **Github repo**, you should keep the repo's code updated. **Your submission on eLearning is your source code and the report.** We encourage you to follow the template provided for the report. Please follow the instructions and replace TODOs with your content. The writeup must include at the top of the first page: your name, your NetIDs, and the URL of the Github repository (otherwise -2pt). Try to make the report brief (i.e. page limit is 5).

The following factors will be considered: your technical solution, your development and learning methodology. Our main focus in grading is the quality of your empirical work and implementation. Not fractional difference. **We value solid empirical work and well written reports**