CS-681: Assignment #2

Due date: March 11, 2025 (11:59)

Objective:
The aim of this assignment is to give you hands-on experience in building neural network models for Natural Language Processing (NLP) tasks using different embedding techniques and optimizing hyperparameters for improved model performance.

Instructions:
In this assignment, you will work with NLP datasets. You will create different word embeddings and build an Artificial Neural Network (ANN) model to solve the NLP problem. Additionally, you will explore hyperparameter optimization to enhance your model's performance.

Part 1: Data Selection and Preprocessing

1. **Dataset:**
   - Sentiment analysis (IMDB reviews),
   - Text classification (News Groups)
     - from sklearn.datasets import fetch_20newsgroups
     - cats = [' rec.autos', comp.graphics ']
     - newsgroups_train = fetch_20newsgroups(subset='train', categories=cats)

   - Load the datasets and explore their structures, labels, and features.

Part 2: Implementing Word Embeddings

1. **Embedding Techniques:**
   - Explore different embedding techniques for representing text:
     - Traditional methods: TF-IDF or Count Vectorizer.
     - Pre-trained embeddings: Word2Vec, GloVe, or FastText.
   - Implement at least two different embedding techniques for the same dataset to see how they affect the model's performance.

Part 3: Building the Feedforward Neural Network Model

Models Development and Training:

- Design an Artificial Neural Network (ANN) model using libraries such as TensorFlow or Pytorch.
- Include input layers to accept the different embeddings.
- Add hidden layers with non-linear activation functions (ReLU) and output layers suited for your NLP task (e.g., Sigmoid for binary classification, Softmax for multi-class classification).
- Report accuracy and other metrics on training data and testing datasets
- Train the model using the training dataset.

- Use an appropriate loss function based on your problem (e.g., binary cross-entropy for binary classification, categorical cross-entropy for multi-class).
- Initialize the neural network weights using an appropriate method suitable for the problem and neural network architecture.
- Implement early stopping with all the appropriate configurations (e.g., patience value, loss metric to monitor).
- Report the performance with and without early stopping.
- Implement dropout techniques with an appropriate dropout ratio to prevent overfitting. Report the performance with and without dropout.
- Use weight decay (regularization L2 or L1) to eliminate overfitting and report the performance with and without regularization.
- Use batch normalization and report the results with and without it.

Part 4: Model Evaluation

Evaluate Model Performance:

- Use the test set to evaluate the final model's performance.
- Print the classification report, including metrics such as accuracy, precision, recall, and F1-score.
- Visualize results using confusion matrices and other relevant plots.

Compare Embedding Techniques

- Discuss the impact of different embedding techniques on model performance.
- Reflect on which embedding performed best and why.

Submission Requirement

- A Jupyter Notebook or code script with clear documentation and comments.
- A report summarizing your approach evaluation result