Project Proposal: Sentiment Analysis with PyTorch and Torchtext

By: Your Name

# Introduction

Sentiment analysis is a natural language processing (NLP) task that aims to classify text into positive or negative categories based on the expressed opinions or emotions. It has many applications in domains such as social media, e-commerce, customer service, and marketing. In this project, we propose to build a sentiment analysis system using PyTorch and Torchtext, two popular libraries for deep learning and NLP in Python. We will use a pre-trained word embedding model, a recurrent neural network (RNN) with a gated recurrent unit (GRU) layer, and a binary classifier to predict the sentiment of movie reviews from the IMDb dataset.

# Background

Word embeddings are low-dimensional vector representations of words that capture their semantic and syntactic similarities. They can be learned from large corpora of text using unsupervised methods, such as word2vec or GloVe, or fine-tuned for specific tasks using supervised methods. Word embeddings can improve the performance of NLP models by providing rich and dense features as inputs. In this project, we will use the GloVe embeddings provided by Torchtext, which are trained on 6 billion tokens from Wikipedia and Gigaword.

Recurrent neural networks (RNNs) are a type of neural network that can process sequential data, such as text, by maintaining a hidden state that encodes the previous inputs. RNNs can learn long-term dependencies and capture the temporal structure of the data. However, RNNs also suffer from the vanishing gradient problem, which makes it difficult to learn from long sequences. Gated recurrent units (GRUs) are a variant of RNNs that use gating mechanisms to control the information flow and avoid the vanishing gradient problem. GRUs have two gates: a reset gate and an update gate. The reset gate decides how much of the previous hidden state to forget, and the update gate decides how much of the new input to incorporate. GRUs can learn to selectively remember and forget information over time, which makes them suitable for sentiment analysis. In this project, we will use the GRU layer implemented by PyTorch, which can handle variable-length inputs and outputs.

Binary classification is a supervised learning task that aims to assign a binary label (e.g. positive or negative) to an input (e.g. a movie review). Binary classification can be achieved by using a logistic regression model, which outputs a probability between 0 and 1, and applying a threshold (e.g. 0.5) to make a prediction. Alternatively, binary classification can be achieved by using a neural network with a single output unit and a sigmoid activation function, which also outputs a probability between 0 and 1. In this project, we will use the latter approach and add a linear layer and a sigmoid layer to our GRU model. We will use the binary cross-entropy loss as our objective function and the accuracy as our evaluation metric.

# Methodology

The main steps of our project are as follows:

* Load and preprocess the IMDb dataset using Torchtext. The dataset contains 50,000 movie reviews, half of which are positive and half of which are negative. We will split the dataset into 80% training, 10% validation, and 10% testing. We will also tokenize, lowercase, and pad the reviews, and build a vocabulary based on the training data.
* Load the pre-trained GloVe embeddings using Torchtext and initialize our word embedding layer with them. We will use the 300-dimensional embeddings and freeze them during training.
* Define our GRU model using PyTorch. We will use a single-layer GRU with 128 hidden units, a dropout rate of 0.2, and a bidirectional option. We will also add a linear layer and a sigmoid layer for binary classification. We will use the Adam optimizer and the binary cross-entropy loss.
* Train and evaluate our model using PyTorch. We will use a batch size of 64 and train for 10 epochs. We will monitor the loss and accuracy on the training and validation sets, and save the best model based on the validation accuracy. We will also test our model on the unseen test set and report the final accuracy.
* Analyze and discuss our results. We will plot the learning curves, examine some examples of correct and incorrect predictions, and identify the strengths and weaknesses of our model. We will also suggest some possible improvements and extensions for future work.

# Expected Outcomes

We expect to achieve the following outcomes from our project:

* A sentiment analysis system that can accurately predict the polarity of movie reviews using PyTorch and Torchtext.
* A better understanding of the concepts and techniques of word embeddings, RNNs, GRUs, and binary classification.
* A demonstration of the benefits and challenges of using pre-trained word embeddings and RNNs for NLP tasks.
* A report and a presentation that document our methodology, results, and analysis.