Aishwarya Lingam

Assignment: Text Data

Git link :https://github.com/alingam9/aishwaryalingam_MachineLearning/tree/main/Assinment5_text

Introduction

This assignment's main goal is to investigate how Recurrent Neural Networks (RNNs) might be used to process and analyze text and sequence data. The assignment specifically focuses on sentiment analysis using the IMDB movie review dataset, which serves as a standard for assessing natural language models. Our goal in this assignment is to examine how RNN architectures function in situations with little training data, especially when enhanced with embedding approaches.

We examine the effects of these differences on model performance by adjusting and testing various configurations, including truncating reviews, limiting training data, and utilizing both learned and pre-trained word embeddings. In addition to enhancing comprehension of RNN principles and text processing, this exercise demonstrates useful methods for enhancing predicted performance in situations where data availability is limited.

This activity is in line with the learning objectives of the course, especially when it comes to comprehending the mathematical foundations of neural networks, deep learning principles, and the specialized uses of RNNs in text-based analysis.

Problem Statement

Because text data is sequential and high-dimensional, it poses special obstacles for machine learning. Conventional models are not appropriate for tasks like sentiment analysis because they frequently fail to capture the context and dependencies inside text sequences. However, because of their capacity to remember and apply sequential information, recurrent neural networks (RNNs) are ideally suited for this field.

The goal of this project is to use the IMDB movie review dataset to assess and improve RNN performance for sentiment categorization. The main goal is to figure out how to increase model accuracy while dealing with a restricted amount of training data. To do this, we alter a baseline RNN model by:

1. Reduce the length of movie reviews to no more than 150 words.
2. limiting the number of training data samples to 100.
3. 10,000 samples are used for performance validation.
4. Just the 10,000 most common terms should be used.
5. before the Bidirectional RNN, compare the results of applying a pre-trained word embedding layer versus a standard embedding layer.

Primary Objective

Applying Recurrent Neural Networks (RNNs) to text and sequence data in order to conduct sentiment classification on the IMDB movie review dataset is the main goal of this project. Building models that can efficiently learn from sparse training data is the main goal of the assignment. It also investigates how various embedding techniques, such as custom embeddings and pre-trained word embeddings, affect model performance.

The purpose of this assignment is to alter a baseline RNN model and methodically assess its correctness using limited data to:

1. Recognize how RNNs manage textual sequential dependencies.
2. Examine the differences in efficacy between learned and pre-trained word embeddings.
3. Determine the bare minimum of training data needed to achieve meaningful model performance.

The ultimate objective is to learn which preprocessing and architectural decisions result in improved prediction performance, particularly in situations with constrained data and computational resources.

Dataset Overview

Data Preparation

The IMDB movie review dataset, a popular standard for sentiment classification tasks, is used for this assignment. Fifty thousand movie reviews with a favorable or negative label are included in the dataset. We use several preprocessing techniques to mimic robustness evaluation and low-resource environment simulation:

- Sequence Truncation: To maintain a constant input length and minimize computational cost, each review is truncated to no more than 150 words.
- Vocabulary Restriction: To increase model generalization and minimize the size of the word index, only the top 10,000 most frequently occurring terms in the dataset are kept.
- Sample Limitation: To provide a meaningful assessment of performance, the validation set has 10,000 samples, whereas the training set is restricted to 100 samples.
- Tokenization and Padding: To guarantee consistent sequence lengths for input into the neural network, reviews are first tokenized into integer sequences and subsequently padded.

These preprocessing procedures are crucial for streamlining the training procedure and evaluating RNN performance with little data.

Architecture Model

A Bidirectional Recurrent Neural Network (RNN), the foundation of the model architecture, processes input sequences both forward and backward to better collect contextual information. The model's two primary versions are examined:

Embedding Model:

- During training, the embedding layer learns word representations from the ground up.
- Bidirectional LSTM: Improves context awareness by processing the sequence in both directions.

Pretrained Embedding:

- Pretrained Word Embeddings: Provides semantic meaning from previous training on a bigger corpus by initializing the embedding layer with external word vectors (such as GloVe).
- Bidirectional LSTM: The custom model's counterpart.
- Final sentiment classification is carried out by the Dense Output Layer.

The assignment compares these two methods to see which performs better, especially in situations where training data is scarce. Important insights into efficient deep-learning techniques for problems involving natural language processing are offered by this architectural experimentation.

Models Built	Training sample size	Validation size	Testing sample size	Model Type	Test accuracy (%)	Test loss
Model 1	100	10000	5000	embedded layer	50.6	0.693
Model 2	10000	10000	5000	embedded layer	49.8	0.691
Model 3	15000	10000	5000	embedded layer	88.6	0.263
Model 4	10000	10000	5000	one-hot	94.9	0.1509
Model 5	15000	10000	5000	LSTM using embedded layer	50.4	0.694
Model 6	25000	10000	5000	LSTM using embedded layer	98.1	0.0596
Model 7	10000	10000	5000	masking enable	87.4	0.412
Model 8	100	10000	5000	Pretrained using GloVe	89.4	0.34
Model 9	10000	10000	5000	Pretrained with 4 LSTM hidden layers	66.7	0.57

Model 10	20000	10000	5000	Pretrained with 2 LSTM hidden layers	78.1	0.47

Graph with highest accuracy and low loss