Sentiment Analysis of IMDB Movie Reviews Using Deep Learning Explanation

**🔍 What’s This Project About?**

The goal of the project is to **analyze movie reviews from IMDB** and figure out whether each review is **positive** or **negative**—basically, does the reviewer like the movie or not?

This is part of a field called **sentiment analysis**, which helps companies understand what people are feeling based on their words. Think of it like teaching a computer to read a review and go, “Yup, that person’s happy,” or “Nope, they’re not.”

### **📊 Exploring the Data (EDA)**

* The dataset has **25,000 reviews to train on** and **25,000 reviews to test on**.
* Each review has already been turned into a list of numbers. Each number represents a word.
* They only keep the **10,000 most common words**, to keep things simple.
* Most reviews are under **500 words long**, but they cut or pad each one to **200 words**, so every input is the same size.

### **🧹 Prepping the Data**

Before feeding the reviews into models:

* All reviews are adjusted to be **exactly 200 words long** (shorter ones are filled in with zeros, longer ones are chopped off).
* This is necessary because deep learning models expect inputs to be uniform in size.

### 

### **🛠️ Building the Models**

1. **Baseline Model: TF-IDF + Logistic Regression**
   * This is a simple model that looks at word frequency and tries to predict the sentiment.
   * It's like counting how many times words like "great" or "terrible" show up and deciding the sentiment based on that.
2. **Deep Learning Models (like LSTMs)**
   * These are more advanced models that try to **understand the flow and context** of words.
   * They do a better job at grasping *how* things are said, not just *what* is said.

### **📈 What’s the Point of All This?**

This kind of project can be used in real life to:

* Automatically filter product reviews.
* Help companies see what customers are saying about their brand.
* Improve recommendations based on how you felt about something.

Sentiment Analysis of IMDB Movie Reviews Using Deep Learning Explanation

### **🧠 Objective**

This project focuses on **binary sentiment classification** of IMDB movie reviews using both classical ML and deep learning approaches. The task is to predict whether a movie review is *positive* or *negative*. The goal is to evaluate and compare different modeling strategies and identify performance gains through deep learning techniques, particularly LSTMs.

### **📊 Dataset Overview & EDA**

* **Source**: IMDB dataset via tensorflow.keras.datasets.
* **Structure**: 25,000 training + 25,000 testing samples, each preprocessed as a sequence of integers representing word indices.
* **Vocabulary Size**: Limited to the **top 10,000 most frequent tokens**.
* **Review Lengths**:  
  + Majority of reviews are under 500 tokens.
  + For consistency, reviews were **padded/truncated to a fixed length of 200 tokens** using pad\_sequences.
* **Class Distribution**:  
  + Balanced: Equal number of positive (label 1) and negative (label 0) reviews.
  + No class imbalance adjustments necessary.

### 

### **🧹 Preprocessing**

* Tokenized sequences were padded/truncated post-sequence to a max length of 200.
* x\_train and x\_test were converted to NumPy arrays of shape (n\_samples, 200) for model input.

### **🔨 Modeling Approaches**

#### **✅ Baseline: TF-IDF + Logistic Regression**

* Text was vectorized using **TF-IDF** (term frequency-inverse document frequency).
* Logistic Regression used as a linear baseline classifier.
* This model serves as a benchmark for evaluating deep learning performance.

#### **🧠 Deep Learning Models**

* **Embedding Layer + LSTM**:  
  + Embedding layer maps word indices to dense vector representations.
  + Followed by a **Long Short-Term Memory (LSTM)** layer to capture sequential dependencies.
  + Dense output layer with sigmoid activation for binary classification.
* Model trained using binary\_crossentropy loss and Adam optimizer.
* Evaluation metrics include **accuracy** on the test set and training curves for loss/accuracy.

### 

### **📈 Performance & Insights**

* The LSTM model outperformed the logistic regression baseline, demonstrating deep learning's capability to extract semantic meaning from text sequences.
* Padding reviews to a fixed length and using embeddings significantly improved training stability and generalization.
* Results show that sequence-aware models better handle syntactic/semantic context, which is crucial for sentiment tasks.

### **🧩 Potential Extensions**

* Incorporate **bidirectional LSTMs** or **GRUs** for improved context understanding.
* Experiment with **pre-trained word embeddings** (e.g., GloVe or Word2Vec).
* Apply **attention mechanisms** or try **transformers** (e.g., BERT) for state-of-the-art performance.
* Add techniques like dropout or recurrent dropout to reduce overfitting.