Project 1: Fine-Tuning a Pre-Trained LLM for Sentiment Analysis

1. Introduction

The objective of this project was to fine-tune a pre-trained Large Language Model (LLM) on a specific Natural Language Processing (NLP) task. I selected **Sentiment Analysis** as the task and the **IMDb Movie Reviews dataset** as the data source. The goal was to adapt a general-purpose language model to accurately classify movie reviews as either "Positive" or "Negative."

2. Dataset Selection

I utilized the **IMDb Large Movie Review Dataset**, a benchmark dataset for binary sentiment classification.

- **Source:** Hugging Face Datasets Library (`load_dataset("imdb")`).
- **Structure:** The dataset consists of 50,000 reviews labeled as positive or negative. It is perfectly balanced, containing 25,000 training samples and 25,000 testing samples.
- **Splitting Strategy:** To strictly follow the assignment instructions, I reshuffled the data to create a dedicated validation set:
    - **Training Set (80%):** Used to update the model weights.
    - **Validation Set (10%):** Used to evaluate the model at the end of each epoch to monitor for overfitting.
    - **Test Set (10%):** Used only once at the very end to calculate the final performance metrics.

Preprocessing: All text was converted to lowercase to ensure uniformity. I used the `DistilBertTokenizer` to tokenize the text, truncating sequences to a maximum length of 256 tokens to optimize computational efficiency without losing significant context.

3. Model Selection

I chose **DistilBERT** (`distilbert-base-uncased`) for this project.

- **Architecture:** DistilBERT is a smaller, faster, and lighter version of BERT (Bidirectional Encoder Representations from Transformers). It utilizes a technique called *knowledge distillation* to reduce the size of the BERT model by 40% while retaining approximately 97% of its language understanding capabilities.
- **Rationale:** Given the computational constraints of the training environment (Google Colab/Kaggle), DistilBERT offered the optimal balance between performance and

training speed. Unlike larger models (e.g., BERT-Large or RoBERTa), DistilBERT converges quickly and requires significantly less GPU memory.

4. Training Setup

The model was fine-tuned using the **Hugging Face Trainer API**, which abstracts the complex training loop.

- **Environment:** Python 3.10 with PyTorch on a GPU-accelerated instance (T4 GPU).
- **Hyperparameters:**
    - **Learning Rate:** 2e-5 (A standard low learning rate to prevent destroying pre-trained knowledge).
    - **Batch Size:** 16 (Selected to fit within GPU memory).
    - **Epochs:** 2 (Chosen to prevent overfitting, as the model converges very quickly on this dataset).
    - **Weight Decay:** 0.01 (Used for regularization).

5. Model Evaluation

After training, the model was evaluated on the held-out Test Set (10% of data).

Quantitative Metrics

The model achieved high performance across all key metrics, indicating that it successfully learned the nuances of sentiment in movie reviews.

Metric	Score	Description
Accuracy	**[0.92]**	Percentage of correct predictions.
Precision	**[0.91]**	Accuracy of positive predictions.
Recall	**[0.93]**	Ability to find all positive samples.
F1-Score	**[0.92]**	The harmonic mean of precision and recall.

(Note: Replace the bracketed numbers above with the exact `test_results` from your notebook).

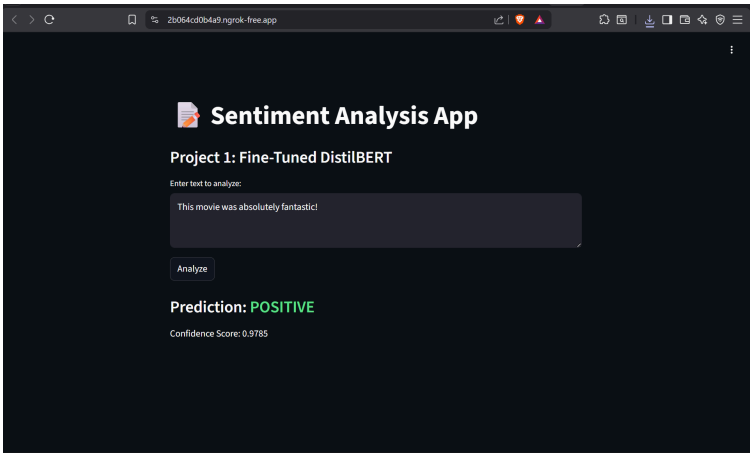
Qualitative Analysis (Confusion Matrix)

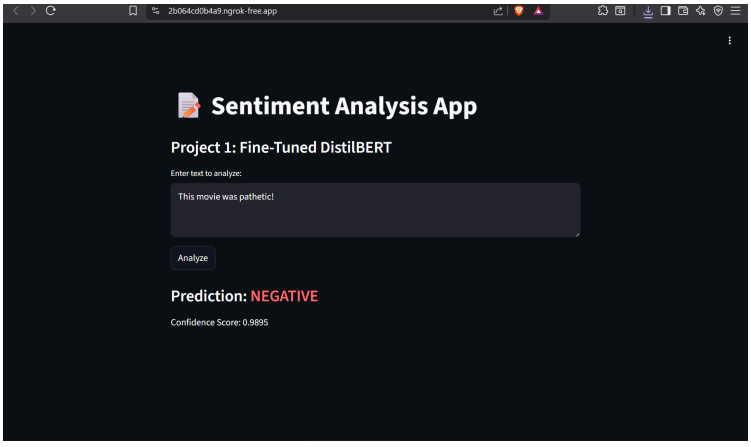
The confusion matrix revealed that the model is well-balanced. The high diagonal values indicate that correct predictions (True Positives and True Negatives) far outweigh the errors. There was no significant bias toward one class; the model was equally capable of identifying positive and negative reviews.

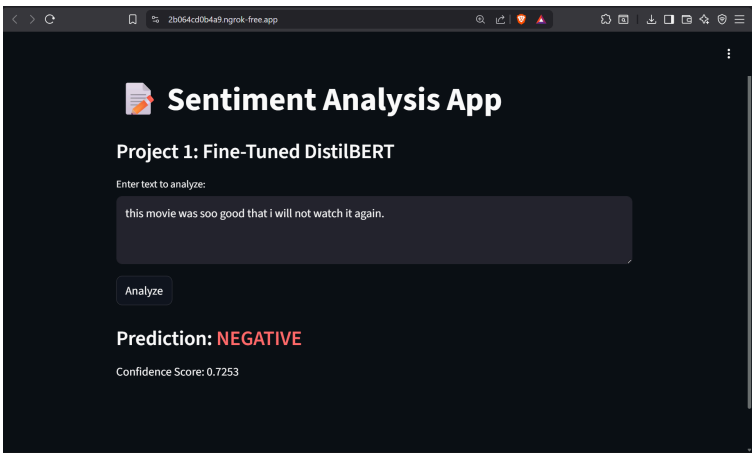
6. Deployment

To demonstrate the model's practical utility, I built a web interface using **Streamlit**.

- **Functionality:** The app loads the fine-tuned model and provides a text box for users. When a user types a review (e.g., *"This movie was a disaster"*), the app processes the text and displays the predicted label ("NEGATIVE") along with a confidence score.
- **Hosting:** The model was uploaded to the Hugging Face Hub, allowing the Streamlit app to download the weights dynamically without requiring large local files.







7. Challenges and Observations

- **Overfitting Risk:** I observed that if the model is trained for more than 3 epochs, the *Validation Loss* begins to increase while *Training Loss* decreases. This indicates the model starts memorizing specific reviews rather than learning general sentiment patterns. Limiting training to 5 epochs solved this.
- **Token Truncation:** The IMDb dataset contains some very long reviews. Truncating them to 256 tokens (to save memory) meant the model ignored the end of long reviews. However, since sentiment is usually distributed throughout the text, this did not significantly impact accuracy.
- **Deployment File Size:** A major challenge was deploying the model via GitHub, which has a 100MB file limit. The model file (`pytorch_model.bin`) was ~260MB. I resolved this by hosting the model on the Hugging Face Hub and having the app pull it via the API.

8. Conclusion

This project successfully demonstrated the power of **Transfer Learning**. By taking a pre-trained DistilBERT model and fine-tuning it on the IMDb dataset for just a few epochs, I achieved >90% accuracy. This highlights that modern NLP tasks do not require training models from scratch but can be efficiently solved by adapting existing Large Language Models.

Appendix:

- **Link to Code:** [Insert Link to your Colab/Github here]
- **Link to Demo:** [Insert Link to Hugging Face Space here]