**Generative Deep Learning**

Submitted by:

Shimron ifrah : 312423247

Yossi shem tov : 315092445

# **Project Report: DistilBERT - A Distilled Version of BERT**

## **Introduction**

### **Background**

The advent of large-scale pre-trained language models such as BERT has revolutionized Natural Language Processing (NLP). However, their substantial computational and memory requirements make them challenging for real-time or on-device applications. To address these limitations, Hugging Face introduced **DistilBERT**, a smaller, faster, and more efficient version of BERT, retaining 97% of its performance while being 40% smaller and 60% faster.

DistilBERT leverages **knowledge distillation**, where a smaller model (student) learns from a larger model (teacher) during pre-training. This method enables efficient model compression while preserving high performance on downstream tasks like text classification and question answering.

### **Objective**

This project focuses on understanding, implementing, and fine-tuning DistilBERT using Python in Google Colab. The goal is to:

* Implement the solution using the Hugging Face library.
* Fine-tune DistilBERT for sentiment analysis on the IMDb dataset.
* Evaluate the model's performance and compare it with the original BERT.
* Present the findings in this report and a classroom presentation.

## **Process of the Project Study**

### **Understanding DistilBERT**

We began by studying the original paper [DistilBERT: a distilled version of BERT](https://arxiv.org/abs/1910.01108) [68]. Key insights include:

* DistilBERT reduces BERT's parameters by removing token-type embeddings and the pooler.
* It uses a triple loss function combining masked language modeling (MLM), distillation loss, and cosine embedding loss.
* Training involves initializing the student model by taking every second layer from the teacher model.

### **Preparation and Tools**

1. **Environment**: Python in Google Colab with GPU acceleration.
2. **Libraries**: Hugging Face Transformers, Datasets, and PyTorch.
3. **Dataset**: IMDb dataset for binary sentiment analysis.

## **Project Flow**

### **1. Implementation in Google Colab**

We began by setting up the **Google Colab environment** to ensure compatibility with the Hugging Face library. This included installing the required libraries, initializing the **DistilBERT** model, and configuring the tokenizer. The model and tokenizer were pre-trained and loaded directly from the Hugging Face model hub, saving significant development time.

During the initial process, we encountered **limitations with GPU availability** and **resource constraints** in the Google Colab environment. These constraints impacted the training time and performance, making it impractical to train the model fully on Colab. To overcome this challenge, we shifted the training process to a **local environment** equipped with an **RTX 1080 GPU**.

In the local environment:

1. We created a **training script** to fine-tune the DistilBERT model on the IMDb dataset.
2. Once training was complete, we generated a **trained model folder** containing the saved weights and tokenizer configuration.
3. A separate **test script** was developed to evaluate the model's performance using the IMDb test dataset, producing a detailed **classification report**.

This two-stage approach—initial setup in Colab and full execution locally—allowed us to overcome resource limitations while ensuring an efficient workflow for both training and evaluation.

### **2. Data Preparation**

The IMDb dataset, a widely used benchmark for sentiment analysis, was chosen for this project. To prepare the data:

* The raw text was tokenized to convert sentences into numerical representations suitable for input to the DistilBERT model.
* Padding and truncation were applied to ensure uniform input sizes, as required by the model.

### **3. Fine-Tuning**

Fine-tuning involved adapting the pre-trained DistilBERT model to the specific task of sentiment analysis. This included:

* Defining training parameters such as learning rate, batch size, number of epochs, and evaluation frequency.
* Using the Hugging Face Trainer, a utility designed to streamline the training process by handling data batching, gradient computation, and model updates.
* Continuously evaluating the model on a validation dataset to monitor its performance and adjust hyperparameters as needed.

### **4. Evaluation**

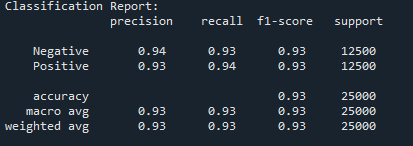
To assess the effectiveness of the fine-tuned model, predictions were generated on the test dataset. Key steps included:

* Comparing the predicted labels with the true labels to calculate metrics such as precision, recall, F1-score, and overall accuracy.
* Analyzing these metrics to identify areas where the model performed well or required improvement.

### **5. Visualization**

Visualizations, such as a confusion matrix, were created to provide a clear representation of the model’s performance. These visualizations helped identify patterns in misclassifications and offered insights into the model’s behavior.

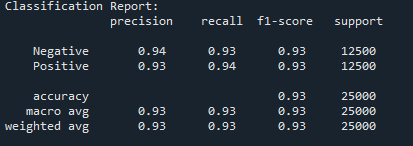
### **Visualization**

* Confusion matrix displaying true vs. predicted labels.
* 

## **Results**

### **Model Performance**

* **Classification Metrics**:



1. **Precision**:
   1. Precision is the ratio of correctly predicted positive observations to the total predicted positives.
   2. For **Negative** class: 94% of the predictions labeled as "Negative" were correct.
   3. For **Positive** class: 93% of the predictions labeled as "Positive" were correct.
2. **Recall (Sensitivity)**:
   1. Recall is the ratio of correctly predicted positive observations to the total actual positives.
   2. For **Negative** class: The model correctly identified 93% of all actual "Negative" samples.
   3. For **Positive** class: The model correctly identified 94% of all actual "Positive" samples.
3. **F1-Score**:
   1. F1-Score is the harmonic mean of Precision and Recall. It provides a balanced measure when the class distribution is imbalanced.
   2. Both classes achieved an F1-Score of **0.93**, indicating balanced precision and recall.\
4. **Accuracy**:
   1. Accuracy measures the overall correctness of the model.
   2. With an accuracy of **93%**, the model correctly classified 93% of the 25,000 test samples.
5. **Macro Avg**:
   1. The average of Precision, Recall, and F1-Score across both classes.
   2. It gives equal weight to both classes regardless of their frequency.
6. **Weighted Avg**:
   1. The weighted average of Precision, Recall, and F1-Score, taking class frequency into account.
   2. Since the dataset is balanced (12,500 samples per class), the macro and weighted averages are identical here.

### **Key Insights**

* The model achieved **high performance** with an overall **accuracy of 93%**.
* Precision, Recall, and F1-Scores are nearly identical across both classes (Negative and Positive), indicating that the model is balanced and not biased toward one class.
* The classification report shows the model is well-suited for binary sentiment analysis tasks on the IMDb dataset.

## **Conclusion**

1. DistilBERT achieved near-BERT performance on the IMDb sentiment analysis task with significantly reduced computational requirements.
2. The model's smaller size and faster inference make it ideal for deployment on resource-constrained devices.
3. The project demonstrated the effectiveness of knowledge distillation in creating compact yet powerful NLP models.

### **Sources**

* Hugging Face Transformers Library: <https://github.com/huggingface/transformers>
* IMDb Dataset: <https://huggingface.co/datasets/imdb>
* Out git project repository: [shimron202/distilbert: distilbert pre trained model based on imdb review movies](https://github.com/shimron202/distilbert)
* DistilBERT Paper: [arXiv:1910.01108](https://arxiv.org/abs/1910.01108)