Movie Review Sentiment Analysis with ALBERT

Cathy Gao

May 9th, 2025

Task Description

This project explores the use of Large Language Models (LLMs) for sentiment classification on movie reviews—a fundamental natural language processing (NLP) task. The objective is to train a model that can classify a given movie review as either positive or negative. Sentiment analysis requires the LLM to understand not just individual word meanings, but also tone, nuance, and context within a sentence or paragraph. I chose this task due to its wide applicability in opinion mining and its role as a benchmark for evaluating language understanding capabilities.

Model Description

I used **ALBERT-base-v2**, a lighter and more memory-efficient variant of BERT, as the foundation model. ALBERT-base has approximately **12 million parameters**, making it more computationally affordable to fine-tune than BERT. It uses 12 transformer layers with hidden size 768 and 12 attention heads. The model runs efficiently on standard GPUs and does not require specialized hardware like bfloat16-capable TPUs. I selected ALBERT because of its balance between performance and efficiency, which suits the compute limitations for this project.

Data Description

The dataset used for this task is the IMDb dataset, sourced from Hugging Face. It contains 50,000 movie reviews split evenly between positive and negative sentiment. The training set includes **20,000 examples**, the evaluation set **5,000**, and the remaining **25,000** are used for testing or holdout analysis. Each data point consists of a movie review in natural language and a binary sentiment label (`pos` or `neg`).

Example:

```
{
  "text": "This movie was an amazing journey from start to finish.",
  "label": 1
}
```

Obtaining a Baseline

To establish a baseline, I ran the unmodified ALBERT model on the test set without any fine-tuning. The model was evaluated using accuracy as the metric. The baseline model achieved **0.5014** accuracy and an F-1 score of **0.0281**, which is close to chance level.

Example baseline result:

- **Input**: The plot was dull and the characters were one-dimensional.
- **Prediction**: Positive
- **Ground Truth**: Negative

Method for Fine-Tuning

I fine-tuned the full ALBERT model using the Hugging Face `transformers` library and `Trainer` API. The model was trained for **3 epochs** with a learning rate of `2e-5`, batch size **16**, and early stopping based on validation loss. I used 5,000 validation examples to monitor overfitting and save the best-performing checkpoint. Training was performed using a single NVIDIA Tesla T4 GPU in Google Colab. I chose full model fine-tuning over adapter-based methods to achieve maximal performance improvement, given the relatively small model size.

Results

After fine-tuning, the model achieved an accuracy of **0.9410** and an F-1 Score of **0.9404** on the test set.

Post-tuning example:

- **Input**: The plot was dull and the characters were one-dimensional.
- **Prediction**: Negative
- **Ground Truth**: Negative

This shows that fine-tuning significantly enhanced the model's ability to understand sentiment in context. The results demonstrate that even a relatively small transformer model like ALBERT can perform competitively on standard NLP benchmarks when fine-tuned effectively.

AI Usage

Throughout this project, I used AI tools, specifically ChatGPT, to assist in structuring and refining my project report, clarifying technical documentation, and troubleshooting code implementation steps. While the modeling and training were performed manually, AI support helped streamline the overall workflow and improve presentation clarity.