INF721 Final Project

João Vitor Silva de Oliveira
Department of Informatics
Federal University of Viçosa
Viçosa, MG, Brazil
joao.silva.oliveira@ufv.br

1 Introduction

Transformer-based models have revolutionized various areas of natural language processing (NLP), including tasks such as sentiment classification, machine translation, and question answering. This project aims to implement a custom transformer model for sentiment classification using the Stanford Sentiment Treebank v2[1] (SST-2) dataset, widely used as a benchmark in the field.

This work is inspired by the paper "Improving Language Understanding by Generative Pre-Training" by Alec Radford and collaborators, which introduced the concept of generative pre-training followed by fine-tuning. This paper is a landmark in the NLP field, demonstrating how pre-training on large amounts of data can significantly improve performance across various tasks. It also showcased the effectiveness of transformers in learning universal representations that transfer well across different tasks. This project adopts concepts from the paper to create a sentiment classification model.

The goal was to build a transformer from scratch capable of achieving similar results compared to the original model, leveraging the PyTorch framework. The main contributions include:

- Implementing a transformer model tailored for binary classification.
- Creating custom layers and integrating them with standard PyTorch libraries.
- Evaluating results in terms of accuracy.

2 Implementation Details

The code was organized into modules to facilitate development and maintenance. The main classes include:

- **SST2Dataset**: A class for handling the dataset, responsible for tokenizing texts and creating training batches.
- **TransformerClassifier**: Implementation of a transformer model with embedding layers, attention mechanisms, feed-forward layers, and a classification head.

2.1 Used Libraries

- `torch`: Used to define, train, and run deep neural networks;
- `tokenizers`: Helps in the preprocessing of text data by splitting text into tokens;
- `datasets`: Used for loading the dataset;
- `sklearn`: Userd for calculate the accuracy and split the dataset;
- `os`: Used to allow GPU memory segments to grow dynamically, optimizing memory usage.

[1] Stanford Sentiment Treebank v2: `https://huggingface.co/datasets/stanfordnlp/sst2`

Differences compared to the original authors' code:

- Reduced dataset size for experimental training (used 7000 samples).
- Added autocast (mixed precision training) for better hardware utilization.

3 Experimental Setup

This session will detail the scenario in which the model was trained, including hardware specifications and model parameters.

3.1 System Information

- **Python:** 3.12.4
- **Operating System:** Windows 10
- **Architecture:** 64-bit
- **Processor:** Intel Core i5 10400
- **RAM (GB):** 15.87
- **GPU:** NVIDIA GeForce RTX 3060
- **GPU Memory Capacity (GB):** 12.0
- **Number of GPUs Available:** 1

3.2 Parameters

Disclaimer: The project was executed using a GPU, and due to hardware limitations, some parameters had to be adjusted compared to the original paper. The following parameters were used:

- **Optimizer:** Adam with an initial learning rate of 6.25e-5.
- **Scheduler:** Linear warmup during the first 2% of total steps.
- **Embedding size:** 768
- **Number of attention heads:** 12
- **Inner dimension:** 3072
- **Number of layers:** 12
- **Number of classes:** 2
- **Maximum sequence length:** 512
- **Batch Size:** 32
- **Pre-Train Epochs:** 100
- **Fine-tuning Epochs:** 3
- **Activation Function:** GELU (Gaussian Error Linear Unit).
- **Dropout:** 0.1

4 Results

The obtained results were evaluated in terms of test set accuracy. The table below summarizes the results:

Configuration	Accuracy (%)
Implemented model	54.79
Original model (reference)	91.3

Table 1: Test set accuracy comparison

4.1 Pre-training

The pre-training phase consisted of 100 epochs with a consistent decrease in loss over time, reflecting the model's ability to learn representations from the dataset. The loss started at 0.712 and converged to approximately 0.686 by the final epoch. The learning progression during pre-training indicates that the model effectively adapted to the task of generating embeddings for sentiment analysis.

4.2 Fine-tuning

The fine-tuning phase was conducted over three epochs, with losses of 0.6877, 0.6876, and 0.6875, respectively. Despite the relatively stable loss values, the fine-tuned model achieved an accuracy of 54.79% on the test set. This result highlights some limitations in transferring pre-trained embeddings to the specific task of sentiment classification.

5 Discussion

The primary observations and challenges encountered during this project are as follows:

- **Pre-training Effectiveness:** The gradual decline in loss during the pre-training phase demonstrates that the model effectively captured the underlying structure of the dataset. However, the convergence to a relatively high loss value suggests potential inefficiencies in the chosen architecture or training strategy for fully leveraging the dataset.
- **Fine-tuning Limitations:** The test set accuracy of 54.79% indicates a significant performance gap compared to state-of-the-art models for sentiment classification. This gap could stem from limited dataset size, insufficient pre-training epochs, or suboptimal fine-tuning configurations.
- **Hardware Constraints:** The model was trained on an NVIDIA RTX 3060, which necessitated reducing the dataset size and simplifying the model architecture. These adjustments likely limited the model's ability to generalize effectively.
- **Future Improvements:** Addressing the challenges requires several steps:
  - Increasing the dataset size to enhance training diversity.
  - Extending the pre-training phase or incorporating additional pre-training objectives to improve embedding quality.
  - Experimenting with alternative/original fine-tuning hyperparameters or architectures better suited to sentiment classification.
  - Use a setup with greater computational capacity so that parameters can be optimized

In summary, while the project demonstrated the feasibility of implementing a custom transformer model for sentiment analysis, the results indicate substantial room for improvement in both pre-training and fine-tuning stages.

References

[1] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, *Improving Language Understanding by Generative Pre-Training*. OpenAI, 2018.

[2] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, Christopher Potts, *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank*. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA, October 2013. Association for Computational Linguistics, 1631–1642. `https://www.aclweb.org/anthology/D13-1170`