Implementing GPT-2

Stanford CS224N {Default}

Kristine Ma, Isaias Martinez, Varsha Saravanan
Department of Computer Science
Stanford University
kmayu@stanford, isaiasm@stanford.edu, vsaravan@stanford.edu

Abstract

This project implements a decoder-only Transformer based on the GPT-2 architecture and evaluates how the same backbone adapts across three downstream settings: sentiment analysis, paraphrase detection, and conditional sonnet generation. For our milestone, we completed and improved the sentiment analysis and paraphrase detection baselines, constructing an end-to-end training and evaluation pipeline for classification tasks. Next, we will implement and evaluate the sonnet generation baseline. Building on these baselines, we will compare standard fine-tuning against three extensions: LoRA and ReFT for parameter-efficient adaptation, and DPO for preference alignment in sonnet generation. Overall, we measure the tradeoff between task performance and trainable parameter efficiency, comparing fine-tuning methods to parameter-efficient updates.

1 Key Information to include

- TA mentor: Tristan Thrush
- External collaborators: No
- External mentor: No
- Sharing project: No

2 Approach

Core Model Architecture: Our model is a decoder-only Transformer in the GPT-2 style [1]. We sum learned token and positional embeddings, apply dropout, pass them through a stack of Transformer blocks, and apply a final layer normalization to produce contextualized hidden states. Each block uses causal masked multi-head self-attention[2], as defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V,$$

. This was implemented by projecting inputs to per-head Q, K, V, applying causal and padding masks, softmax normalization with dropout, and a final output projection. Blocks follow a pre-layernorm design with residual connections and a two-layer MLP with GELU and dropout. We also support loading pretrained GPT-2 weights, tying the output projection to the input embedding matrix for vocabulary logits.

Sentiment Classification Baseline: We treat sentiment prediction as supervised classification on top of GPT-2. We use the final hidden state of the last non-padding token as a sequence representation, then apply dropout and a linear classifier trained with cross-entropy loss. We compare two adaptation settings: `last-linear-layer`, which freezes GPT-2 and trains only the classifier head, and `full-model`, which fine-tunes GPT-2 end-to-end.

Paraphrase Detection Baseline: We implement paraphrase detection by attaching a two-way classification head to GPT-2. We use the final hidden state of the last non-padding token as sequence representation and fine-tune with cross-entropy loss. Labels use YES and NO token ids. We map these to binary classes during training and map predictions back to token ids when output files are generated.

3 Experiments

- **Data**: For this milestone, we evaluate two classification settings: sentiment classification and paraphrase detection. For sentiment, we use the SST (five-way) [3] and CFIMDB (binary) splits, where each example is a single sentence paired with a sentiment label. For paraphrase detection, we use the Quora Question Pairs (QQP) splits [4], where each example is a pair of questions labeled as paraphrase or not paraphrase.
- **Evaluation Method**: We evaluate and report results for both sentiment classification and paraphrase detection using dev accuracy, and use the dev split to validate the end-to-end training and evaluation pipeline. For paraphrase detection, predictions are written as YES/NO token ids for output files, but accuracy is still computed on the labeled dev split.
- **Experimental Details.** All experiments initialize from pretrained GPT-2 weights and use the final hidden state of the last non-padding token, followed by dropout and a linear classification head trained with cross-entropy loss using the AdamW optimizer. We ran 10-epoch experiments on a Google Cloud VM with an NVIDIA L4 GPU. For sentiment classification, we compare `last-linear-layer` to `full-model`, using learning rate 1×10^{-5} and batch size 8 for full model fine-tuning. For paraphrase detection, we fine-tune the GPT-2 end-to-end with a two-way classification head using batch size 8 and learning rate 1×10^{-5}.

  Results. Below are the results for sentiment analysis and paraphrase detection experiments:

	Baseline	Model Results
LLL SST	0.462	0.467
Full Model SST	0.513	0.520
LLL CFIMDB	0.861	0.865
Full Model CFIMDB	0.976	0.980

Table 1: Sentiment analysis performance

	Baseline	Model Results
Dev	0.882	0.898
Test	0.856	0.859

Table 2: Paraphrase detection performance

We interpret the results as strong overall: For sentiment classification, full model fine-tuning outperforms last linear layer training on both SST and CFIMDB, as expected and modestly exceeds the provided baseline scores. For paraphrase detection, our dev and test accuracy was improved past provided baselines, suggesting the end-to-end pipeline is working, while leaving room for further gains through fine-tuning and the planned extensions. According to the the leaderboards at time of submission, our team ranks 22nd in paraphrase dev and 10th in paraphrase test.

4 Future work

Having completed the core GPT-2 architecture, for the rest of the project, we will be benchmarking three fine-tuning methods: LoRA, ReFT, and soft prompt tuning, against standard full-parameter fine-tuning to evaluate performance. This will allow us to conduct our comparative analysis of model adaptation strategies across our three downstream tasks.

Additionally, we will implement Direct Preference Optimization (DPO) to address the specific challenges of sonnet generation, aligning the model with style preference instead of just next-token prediction. These different extensions allow us to examine how different modifications can optimize a model for different downstream tasks, from binary classification to a creative task.

References

[1] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[3] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[4] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1804.07461.