

Fine-tuning GPT-2 for Short Query Intent Classification

Jin Young Lee

June 9, 2025

Abstract

This report presents our implementation and experimental analysis of a GPT-2-based language model, fine-tuned across four distinct natural language tasks: sentiment analysis, paraphrase detection, sonnet generation, and intent classification for short user queries. Our study aims to bridge generative pretraining with downstream task-specific performance. We report baseline results, identify model limitations, and explore cloze-style formulation for classification and creative generation via autoregressive decoding.

1 Introduction

Recent advancements in natural language processing (NLP) demonstrate the effectiveness of transformer-based architectures. GPT-2, a decoder-only model, exhibits strong capabilities in both language understanding and generation. This project involves implementing core GPT-2 components, fine-tuning for classification and generation, and evaluating the model’s performance across multiple tasks.

2 Task Descriptions

2.1 Sentiment Analysis

We fine-tune GPT-2 on two datasets: Stanford Sentiment Treebank (SST) and CFIMDB. For SST, a 5-class classification task, we utilize phrase-level sentiment annotations. For CFIMDB, we address binary sentiment classification on long movie reviews.

2.2 Paraphrase Detection

The Quora Question Pairs dataset is used to determine semantic similarity between sentence pairs. We reformulate the classification task as cloze-style

generation, prompting the model with a paraphrase question and expecting a “yes” or “no” response.

2.3 Sonnet Generation

We train the model to generate Shakespearean sonnets. The task involves predicting the next token given prior lines, preserving rhyme and structure inherent to sonnets.

2.4 Short Query Intent Classification

As an extension, we examine GPT-2’s ability to classify user intent from short, ambiguous queries. Users often provide minimal input (e.g., “Weather?”, “Pizza nearby”), posing a challenge for traditional NLP models due to limited context. We fine-tune our GPT-2 model using the MASSIVE dataset (SetFit/amazon_massive_intent_en-US), evaluating its intent classification performance across queries of varying lengths. We focus on short queries to test the hypothesis that GPT-2’s deep contextual embeddings enable it to disambiguate intent even under severe information sparsity.

3 Model Implementation

3.1 GPT-2 Architecture

Our model is based on a 12-layer GPT-2 with masked multi-head attention and feed-forward layers. We implement key components:

- `CausalSelfAttention` for masked self-attention.
- `GPT2Layer` stacking attention and MLP blocks.
- `GPT2Model` combining token and positional embeddings.

3.2 Implementation Details

The GPT-2 implementation consists of several key components:

3.2.1 Embedding Layer

The model uses two types of embeddings:

- Word embeddings: Maps input tokens to dense vectors
- Position embeddings: Adds positional information to each token

These embeddings are combined and passed through a dropout layer.

3.2.2 GPT-2 Layer

Each GPT-2 layer contains:

- Multi-head self-attention with causal masking
- Layer normalization before attention and feed-forward
- Feed-forward network with GELU activation
- Residual connections around each sub-layer

3.2.3 Model Architecture

The complete model:

- Processes input through embedding layers
- Passes through 12 transformer layers
- Applies final layer normalization
- Returns both sequence outputs and last token representation

3.3 Training Setup

We use HuggingFace tokenizers and initialize with pre-trained weights. Optimization is performed using the Adam optimizer with bias correction and decoupled weight decay. Fine-tuning varies between last-layer tuning and full-model tuning.

4 Sentiment Classification Results

Using the classifier head on the last token’s representation:

- SST last-layer dev accuracy: 46.2%
- SST full-model dev accuracy: 51.3%
- CFIMDB last-layer dev accuracy: 86.1%
- CFIMDB full-model dev accuracy: 97.6%

The significant improvement in full-model fine-tuning for CFIMDB suggests that the deeper context captured in longer sequences benefits from broader parameter updates.

5 Paraphrase Detection via Cloze-Style Prompting

We construct cloze-formulated prompts like: Is ‘‘sentence A’’ a paraphrase of ‘‘sentence B’’?

We decode the next token and compare it with token ids for “yes” or “no”. Performance is measured via accuracy on Quora’s dev/test sets.

5.1 Observations

Cloze formulation leverages GPT-2’s autoregressive nature. Preliminary experiments show that decoder-only models can handle classification by token generation, although error rates arise from ambiguity or tokenization noise.

6 Sonnet Generation

GPT-2 is fine-tuned on 143 sonnets and evaluated on 12 held-out examples, conditioned on their first three lines. Evaluation metric is CHRF.

6.1 Qualitative Analysis

Generated outputs often preserve meter and rhyme patterns. However, semantic coherence and novelty vary. Sample outputs demonstrate syntactic fluency, but shallow semantic depth.

7 Short Query Intent Classification Results

We segment test samples from the MASSIVE dataset by length to compare performance on short versus long queries. The model shows promising F1-scores even on short utterances, though performance slightly declines as length decreases. Qualitative analysis reveals that ambiguity is more pronounced in imperative or single-word inputs.

8 Conclusion and Future Work

Our project validates GPT-2’s flexibility across structured (classification) and unstructured (generation) tasks. The additional short-query intent classification task underscores GPT-2’s ability to handle minimal context using deep pretraining. Future directions include parameter-efficient fine-tuning (LoRA), incorporating rhyme constraints in generation, and second-order optimization (e.g., Shampoo) for faster convergence.

Acknowledgments

We thank the CS224N staff for providing the starter code and detailed documentation.

A Appendix: Implementation Details

Sanity checks passed for attention and optimizer modules. All experiments conducted on a single NVIDIA A100 GPU. Hyperparameters: learning rate $1e-4$, batch size 16, epochs 5–10. For intent classification, we used standard accuracy and F1 metrics segmented by query length.