GPT-2 Inference Visualization - SFC Project

Matej Hornik

Brno University of Technology
Faculty of Information Technology
Božetěchova 1/2, 612 00 Brno – Královo Pole
xhorni20@fit.vut.cz

November 29, 2024

Abstract

This report details the development of a GPT-2 Visualization Tool for class Soft Computing at BUT FIT. The project provides a Python-based implementation of a GPT-2 language model with an interactive dashboard for exploring internal mechanisms such as attention matrices and embeddings. The report discusses the model architecture, visualization techniques, and application setup.

1 Introduction

Transformer models, particularly GPT-based architectures, have revolutionized natural language processing (NLP). This project focuses on implementing a GPT-2 model and developing interactive visualizations to provide insights into its operation. The main goals include generating text from custom prompts and exploring the model's attention and embedding mechanisms using a user-friendly interface.

2 GPT-2 Architecture: Deep Dive

2.1 Transformer Architecture Fundamentals

The GPT-2 model is built upon the Transformer architecture, introduced in the seminal paper "Attention Is All You Need" by Vaswani et al. Unlike previous sequence-to-sequence models that relied on recurrent or convolutional neural networks, Transformers use a self-attention mechanism to process input sequences.

2.2 Key Architectural Components

The GPT-2 model consists of several critical components that enable its powerful language understanding and generation capabilities:

- **Token Embedding:** Converts discrete tokens (words or subwords) into dense vector representations. Each token is mapped to a high-dimensional vector space, allowing semantic relationships to be captured.

- **Positional Embedding:** Provides information about the position of tokens in the sequence. Since Transformers process entire sequences simultaneously, positional embeddings are crucial for maintaining the order of tokens.
- **Multi-Head Self-Attention:** The core mechanism that allows the model to weigh the importance of different tokens when processing each token. Multiple attention heads enable the model to focus on different aspects of the input simultaneously.
- **Feed-Forward Neural Networks:** Applied after the attention mechanism, these networks further transform the representations, introducing non-linearity and additional computational capacity.
- **Layer Normalization:** Stabilizes the learning process by normalizing the inputs across layers, which helps prevent vanishing or exploding gradients.

2.3 Autoregressive Language Modeling

GPT-2 is an autoregressive model, meaning it generates text sequentially, with each new token conditioned on the previous tokens. The causal (masked) self-attention ensures that predictions for a given token depend only on the known tokens that precede it.

2.4 Mathematical Formulation of Key Operations

2.4.1 Self-Attention Mechanism

The self-attention mechanism can be mathematically expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Where:

- Q: Query matrix
- K: Key matrix
- V: Value matrix
- d_k: Dimension of the key vectors

2.4.2 Multi-Head Attention

Multi-head attention allows parallel processing of different representation subspaces:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2}$$

2.4.3 Activation Function

GPT-2 uses the GELU (Gaussian Error Linear Unit) activation function, which provides a smooth, non-linear transformation:

$$\text{GELU}(x) = x \cdot \Phi(x) \tag{3}$$

Where $\Phi(x)$ represents the cumulative distribution function of the standard normal distribution.

This section comprehensively explains GPT-2's architecture, matching the visualization and mathematical representations in Streamlit application. It covers the key components implemented, such as token embeddings, positional embeddings, multi-head attention, and the GELU activation function.

3 Technical Implementation

The core of this project is a custom implementation of GPT-2, provided in `model.py`, with the following features:

- **GPT-2 Architecture:** Implements token embeddings, positional embeddings, and transformer layers.
- **Attention Mechanism:** Visualizes attention matrices across multiple heads.
- **Embedding Analysis:** Displays token and positional embeddings for input sequences.
- **Activation Functions:** Explores the GELU activation function.

3.1 Model Components

The primary components include:

- **CausalSelfAttention:** Implements multi-head self-attention with a causal mask to ensure autoregressive behavior.
- **MLP:** A feed-forward network with GELU activation and dropout.
- **LayerNorm:** Custom layer normalization used in original GPT-2 implementation by OpenAI.

3.2 Visualization Features

Visualizations are implemented using Streamlit and Plotly:

- **Attention Matrix:** Displays attention weights across tokens.
- **Embedding Analysis:** Interactive plots of token and positional embeddings.
- **GELU Function:** Plots the GELU activation function.

4 Setup and Usage

4.1 Installation

To install the project, copy the zip file to directory `sfc_project` and extract the files. Then install necessary packages using python and then with `Poetry`:

```
sudo apt install python3-pip
sudo apt install python3-poetry
cd sfc_project
poetry install
```

4.2 Running the Application

Launch the Streamlit app using the following command:

```
poetry run streamlit run sample.py
```

Open an browser and go to *http://localhost:8501/* to see the visualizations of gpt2 inference.

4.3 Usage

- Enter a prompt in the sidebar.
- Click **Generate** to create text and explore visualizations.

5 Dependencies

The project requires the following packages:

- Python 3.11 or higher
- Torch 2.5.1
- Transformers 4.6.1
- Streamlit 1.40.2
- Plotly 5.24.1

6 Conclusion

This project demonstrates the implementation of a GPT-2 model with interactive visualizations using streamlit. It provides a deeper understanding of its inner workings. With this tool, one can easily understand how modern llm works and the attention mechanism in them.

Appendix

Project Structure

- `model.py`: GPT-2 implementation
- `sample.py`: Streamlit interface
- `helpers.py`: Visualization utilities
- `poetry.toml`: Dependency management

References

1. OpenAI GPT-2 Implementation: `https://github.com/openai/gpt-2`
2. Hugging Face Transformers: `https://github.com/huggingface/transformers`
3. Streamlit Documentation: `https://docs.streamlit.io/`