Fine-tuning GPT-2 for Medical Question Answering

This document provides a step-by-step guide on fine-tuning a GPT-2 language model using the Medical Question Answering Dataset (MedQuAD) for the purpose of generating responses to medical queries.

Project Goal and Objectives

The ultimate goal is to enable the fine-tuned GPT-2 model to provide relevant answers to medical questions. This is achieved through the following objectives:

- **Data Preparation:** Process, analyze, and extract relevant features from the MedQuAD dataset.
- **Model Preparation:** Load a pre-trained GPT2 tokenizer and a pre-trained GPT2 language model.
- **Fine-tuning:** Train the GPT2 model on the medical dataset to specialize its knowledge in the medical domain.
- **Evaluation:** Test the fine-tuned model with user prompts and compare its performance with an untuned GPT2 model.

Understanding the Data and the Model

Feature	Description
Dataset	Medical Question Answering Dataset (MedQuAD)
Dataset Format	Question-answer pairs with additional medical context
Data Attributes	Question, Answer, Focus, Concept Unique Identifier (CUI), Semantic Type, Semantic Group
Model	GPT-2
Model Architecture	Decoder-only Transformer
Model Training Data	Massive text dataset (not specified)
Expected Outcome	Generate relevant responses to medical queries after fine-tuning

Key Steps in Fine-tuning the GPT-2 Model

The document outlines a structured process for fine-tuning the model:

1. **Environment Setup:**
    - Install necessary libraries: pyarrow, datasets, accelerate, transformers, os, re, NumPy, pandas, matplotlib.pyplot, sklearn.model_selection, torch.
2. **Data Preprocessing and Exploratory Data Analysis (EDA):**
    - **Handle Missing Values:** Identify and replace missing values in the dataset using appropriate techniques like dropping rows or imputing with "Unknown".
    - **Remove Duplicates:** Eliminate duplicate question-answer pairs to ensure data quality.
    - **Analyze Data:** Perform exploratory analysis on the 'Focus' column to understand data distribution and identify the top 100 categories.
3. **Dataset Splitting:**

- Divide the MedQuAD dataset into training and validation sets based on the top 100 categories in the 'Focus' column, ensuring a stratified sampling approach.

4. **Text Preprocessing and Saving:**
    - Combine question and answer text into a specific format: <question> + question text + <answer> + answer text + <end>.
    - Concatenate the processed text data for training and validation sets and save them into separate text files.
5. **Tokenization:**
    - Load a pre-trained GPT2Tokenizer.
    - Use the tokenizer to encode the training and validation text data.
    - Save the tokenized data for subsequent model training.
6. **Data Collation:**
    - Create a DataCollator object using DataCollatorForLanguageModeling to enable batch processing, padding, and potential data augmentation during training.
7. **Model Loading and Fine-tuning:**
    - Load a pre-trained GPT2LMHeadModel.
    - Define training arguments, including output directory, number of epochs, batch size, learning rate, etc..
    - Fine-tune the GPT-2 model using the prepared training data, tokenizer, and specified training arguments.
    - Save the fine-tuned model and tokenizer for later use.
8. **Model Testing and Comparison:**
    - Develop a function to generate responses using the fine-tuned GPT-2 model, accepting a prompt as input.
    - Test the model with sample medical questions and observe the generated responses.
    - Load an untuned GPT2LMHeadModel for comparison.
    - Evaluate the performance difference between the fine-tuned and untuned models using pre-defined test prompts.

This approach showcases a practical application of fine-tuning large language models for specialized tasks like medical question answering.