COE 379L: Software Design For Responsibly Intelligent Systems

Bevo Bud GPT

Kelechi Emeruwa and Pranjal Adhikari

May 3, 2024

Table of Contents

1 Introduction

University students often need quick access to information such as university policies, academics, and other campus resources. However, manually searching through all university resources can be time consuming and inefficient. Furthermore, the information a student is seeking may not be available on any university sites. Various different forums and social sites, such as Reddit and GroupMe, exist where students can communicate with one another to answer these questions with anecdotal experience. Thus, our objective for this project is to develop a large language machine learning model where UT Austin students can interact with and retrieve answers to their questions regarding the university.

2 Data Source & Technologies

2.1 Data Source

The data source for training the model will be from the UT Austin subreddit. Within the subreddit, there are various different posts ranging from information regarding events held on campus to the classes available to take for a specific credit. All posts are available to be commented on by users, sharing their own perspective and answering questions. Furthermore, each post is tagged with a label for the type of content the post contains, such as 'Discussion', 'Announcement', and 'Question.' As our model is designed for question and answering tasks, the data collected from the subreddit will be processed to include as such.

Pushshift is a social media collection platform that has collected Reddit data since the website was founded in 2005. The data is hosted by The-Eye project and is categorized by subreddits, which ranges from the years 2005 to 2022. Furthermore, all posts within each subreddit are separated and categorized into submissions (posts) and comments which include attributes such as author, title, content, and score. The data can be downloaded and is free to use.

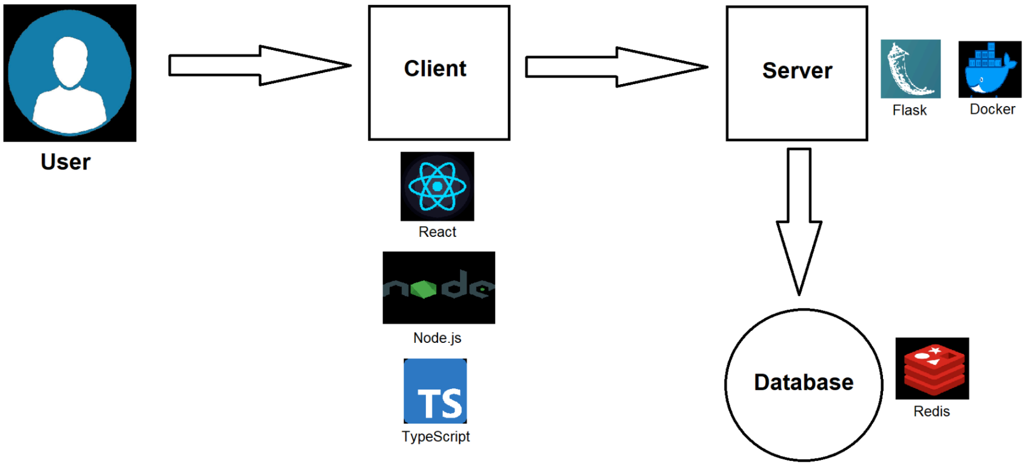
2.1.1 Data Preparation

The data gathered from the UT Austin subreddit includes 13 million total submissions with 42 million comments across all of those submissions. Before proceeding with developing the model, the data underwent several stages of processing to prepare it for the training. First, it was filtered to only include posts marked with a question mark, indicating it is with the 'Question' tag. Posts with the same title were also excluded, ensuring only unique posts remained. The comments from each of these submissions were filtered to only include comments with a score (signifying the number of upvotes) greater than 1. This ensures the answer selected for the given question within the post is a proper answer that is well received by the community. In the comment section of these posts, there also includes replies which create a thread. These replies were excluded to avoid any complications that may arise during the training process of the model.

The result submission and comments were then structured in which each comment corresponded to a submission, forming a pair for training purposes. Furthermore, the data was formatted in a conversational format in which the title of the submission is listed as a question, and the comment is listed as an answer. Lastly, the data was converted to JSON data type for flexible use. After these processing steps, there were 10,000 total unique submissions that included comments to train and develop the model.

2.2 Technologies

To design and implement the project, various technologies were used in conjunction, including Docker, Redis, Flask, Hugging Face, React, TypeScript, and Node.js. With these technologies, the model was developed and packaged for use intuitively. The design of the model can be seen in the figure below.



Project Design

The model was developed, trained, and fine-tuned using Hugging Face, which includes the Transformers library and pre-trained models designed for various natural language processing (NPL) tasks such as text classification and question answering. In our case, the question answering model BERT was used. Within the BERT model, the fine-tuning capabilities and architecture allow it to better understand the context of the words and is capable of generating human-like text. As our model is designed for user inquiry and generating an output, the BERT model fits best with its capabilities.

The trained model itself runs within a Docker container where a backend server receives requests from the user and their queries. These queries are then fed into the model for an output to the respective query. For greater accessibility, the user's queries and respective output will be stored within a database using Redis, which will be kept running also using a Docker container.

React, TypeScript, and Node.js were used to develop the user side of the web application, where users will be interacting with the model. The web application is designed as an interface, allowing users to input their UT-related questions and receive a response from the model within the web interface.

3 Methods

As mentioned before, the model was built and fine-tuned using BERT. The Transformers library was used to first import pre-trained tokenizers to convert the raw text from the submissions and comments in the subreddit data to a series of token ids for training. The AutoTokenizer class is used with *from_pretrained()* and the "distilbert/distilgpt2" model checkpoint to simplify loading the tokenizers with one command. The "distilbert/distilgpt2" model is used specifically in this project, as this distilled version of the BERT model is efficient and quick while retaining the performance of BERT. Furthermore, the GPT2 model's optimization for tasks that involve text generation (answer) given the context of a prompt (question/query) best suits our case.

With the configuration of the model listed above, training parameters were defined such as the number of epochs and seed, with values of 10 and 42, respectively. The LoraConfig (Low-Rank Adaptation) from the *PEFT* library is utilized to decrease the training time of the model. The method decomposes large matrices into smaller, low-rank matrices in the attention layers, which reduces the number of parameters that need to be fine-tuned, resulting in less resources and time used. Next, the SFTTrainer (Supervised Fine-Tuning) is used with several arguments, including the model checkpoint, tokenizer, and the LoraConfig defined above to complete the training process.

4 Results

The trained model was evaluated by calculating the loss, which measures the discrepancy between the model's predictions and the actual results of the data. In our case, the output of the model is the answer for each question scraped within the UT Austin subreddit. The loss value was calculated to be 3.93. A smaller loss value indicates better performance of the model, meaning the predicted results are closer to the actual values. Thus, it can be concluded that the model performs fairly well in a question-answering scenario. The model can further be evaluated by manually interacting with the web application with queries and verifying the results.

5 Resources

- Class website: https://coe-379l-sp24.readthedocs.io/en/latest/index.html
- UT Austin subreddit: https://www.reddit.com/r/UTAustin/
- Archived data of r/UTAustin: https://the-eye.eu/redarcs/