Web Scraping and GPT Model Training Assignment Documentation

Contents

Code Files

- **`topHospital.py`**: Contains the code for top hospital URL collections, specifically the `hospital_list()` function.
- **`playScrap.py`**: Contains the code for scraping the website and saving data in `scraped_json.json` file
- **`model.ipynb`:** Contains the code for data preprocessing and model training & saving

Data Files

- **`scraped_data.json`:** Contains the scraped data from the top hospital websites.

Trained Model

The trained Private GPT model is saved in ./gpt_finetuned folder

Data Collection

The first code file, `topHospital.py`, contained the `hospital_list()` function. This function scraped the top hospital links from the Newsweek website. It extracted the links from a table on the webpage and stored them in a list. The function returned the list of hospital links as the output.

To collect data from the top 50 hospitals' websites, a web scraping script was implemented. The script utilized the `requests` library to send HTTP requests to the website and the `BeautifulSoup` library to parse the HTML content.

- Note: We collected data specifically within the `<p>` tags. To reduce time, we omitted other tags.
- Note: to save time while scraping and training the depth(sub links) in which our scraper went was set to 0. But it can be changed to any number.

Data Preprocessing

The data preprocessing was performed within the `model.ipynb` file. The cleaning steps included removing unnecessary whitespace, escape characters, and other unwanted characters. Regular expressions were used to perform these cleaning operations. The cleaned data was then stored in a format suitable for model training.

Model Training

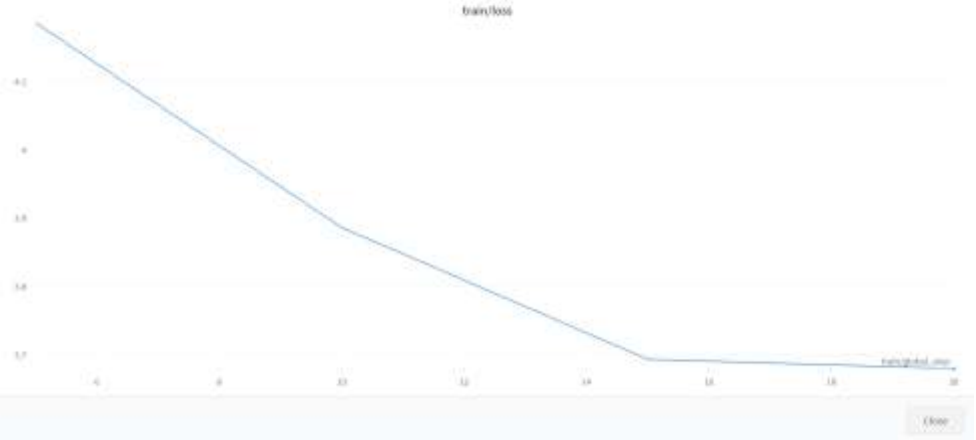
The `model.ipynb` file focused on training a Private GPT model using the cleaned data obtained from the data cleaning step. The script utilized several libraries, including `transformers` library to train the model. The following steps were involved in the model training process:

Data Preparation:

- The GPT2 tokenizer from the `transformers` library was used to tokenize the text data.

Model Configuration and Training:

- The pretrained GPT2 model from huggingface was used.
- The model was moved to the GPU for training using the `.to('cuda')` method.
- logging the training progress was logged in Wandb, an online platform for tracking and visualizing machine learning experiments. The `WandbCallback()` was added as a callback to log the training progress.



- After training, the trained model checkpoint was saved.

Conclusion

The project successfully achieved the objectives of web scraping the top hospital websites and training a Private GPT model using the scraped data.