Natural Language Processing (NLP) projects:

BloggerGPT: Blog Writing AI Bot *(Independent Study)*

Tools, techniques and technologies used: Python, LangChain, ChatGPT, Prompt Engineering, Natural Language Processing (NLP) Libraries.

- Developed an AI-powered blog writer using cutting-edge technologies and techniques, such as Python, LangChain, ChatGPT, and Prompt Engineering.
- For this, conducted in-depth research and analysis on the principles of high-quality blog writing.
- Designed and formulated specific prompts for ChatGPT at each stage of the blog creation process and orchestrated the entire chains of LangChain, to establish a cohesive and efficient BloggerGPT system.

Semantic Sentence Similarity Matcher

Tools, techniques and technologies used: Python, PyTorch, BERT, BART, T5, Sentence Embeddings, Fine-Tuning, Regression Modeling, Transformer-Based Models.

- Created the 'Semantic Sentence Similarity Matcher' by utilizing cutting-edge NLP models, including BERT, BART, and T5.
- As part of this project, tried various embeddings, fine-tuned the classification layer, and adjusted all the layers of the model to obtain the best-optimized model for sentence similarity.
- Also trained a regression model for sentence similarity to predict the probability of sentences being similar using the aforementioned transformer-based models.

Summarization Tagging

Tools, techniques and technologies used: Python, Spacy, POS Tagging, Token Matcher, Phrase Matcher, Dependency Matcher, Flask, Rule-Based Matching Techniques, Web Application Development.

- Created a research project focused on developing a rule-based system for tagging sentences within articles for summarization. This system utilizes POS tagging in Spacy.
- As part of this project, used Spacy's token matcher, phrase matcher, Dependency matcher, and other useful Spacy rule-based matching techniques.
- Based on labeled sentences from articles, identified patterns to create a Spacy rule-based system for categorizing sentences.
- Additionally, designed, developed, and deployed a user-friendly web application using Python and Flask to facilitate the testing and validation of the Spacy rules. The web app allowed for article viewing and manual sentence tag annotations.

Section-Wise Summarization Comparative Study

Tools, techniques and technologies used: Python, PyTorch, BERT, BART, Pegasus, Summarization Models, Extractive Summarization, Abstractive Summarization.

- Created a section-wise summarization project to summarize longer documents by breaking them into different sections.
- Utilized various state-of-the-art summarization models, such as HiStruct, Bert, Bart, Pegasus, etc., including extractive, abstractive, and hybrid techniques.
- Evaluated the effectiveness of the approach using F1 and Rouge scores for both section-wise summarization and baseline summarization methods to effectively measure the model's performance.

Long Document Summarization using Efficient Attentions

Tools, techniques and technologies used: Python, PyTorch, Fairseq CLI, Hepos (LongBart Model), Transformer-Based Models, GPU Computing, Azure, Lambda Labs, TACC, Long Document Summarization.

- Developed a long-document summarization model by implementing and comprehending the research paper titled 'Efficient Attentions for Long Document Summarization,' which introduces innovative techniques for efficiently summarizing long documents.

- Leveraged the Fairseq CLI tool and Hepos (LongBart model, a transformer-based model) - employing a novel, efficient encoder-decoder attention mechanism with head-wise positional strides to effectively extract salient information from the source.
- Additionally, I utilized various GPU machine providers, including Azure, Lambda Labs, TACC, and others, to train this model on the cloud for long document summarization.

Multi-document Summarization using PRIMER Model

Tools, techniques and technologies used: Python, PyTorch, PRIMER Model, Transformer-Based Models, Multi-Document Summarization, Extractive Summarization.

- Explored the pre-trained PRIMER model for a multi-document summarization task, which leverages efficient encoder-decoder transformers to streamline the processing of concatenated input documents.
- The primary objective was to evaluate the pre-trained PRIMER model's suitability for extractive summarization, with a particular emphasis on extracting crucial keywords from multiple documents by establishing connections and aggregating information across them.
- This approach underscored the significance of continuous learning and adaptability in aligning the model's capabilities with the specific requirements of the project.

Text Classification using ULMFiT

Tools, techniques and technologies used: Python, PyTorch, ULMFit, Text Classification, Transfer Learning, Fine-Tuning.

- Trained a text classification model using ULMFit in native PyTorch.
- As part of this process, first trained the model on a general corpus (i.e., Wiki2). After that, I fine-tuned the model using domain-specific data and then trained it for a specific task, which in this case was classification. During this, Applied key ULMFiT techniques, including gradual freezing, one-cycle learning rate scheduling, and early stopping.

- This approach resulted in significantly faster training times compared to large-scale models like BERT.

Web Development projects:

Web Application for Prompt Engineering and Research Support

Tools, techniques and technologies used: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template.

- Developed a web application that played a pivotal role in a research project titled "Enhancing Prompt Efficiency for Non-native English Speakers via Metadata Prompting: A Field Study."
- Designed and implemented a structured schema in SQL to efficiently manage and organize data, facilitating seamless data tracking and retrieval.
- Integrated ChatGPT into the web application to support prompt engineering tasks, including traditional prompting and meta prompting, enhancing the research's effectiveness.
- Successfully deployed the web application using Flask on a VPS (Virtual Private Server) for robust and secure access, ensuring reliable performance throughout the research project.
- Contributed to advancing the field of prompt efficiency and research support by creating a user-friendly and efficient web-based tool for engineers and researchers.

Sentence Similarity Web Application with ChatGPT Integration

Tools, techniques and technologies used: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template, LLM (ChatGPT).

- Developed a web application dedicated to Sentence Similarity Research, focusing on the structural analysis of similar sentences across multiple blogs. The project involved utilizing traditional techniques such as parsing and embedding, as well as integrating ChatGPT with prompt engineering for advanced analysis.

- Designed and implemented a structured schema within an SQL database to efficiently manage and organize research data, enabling effective tracking and retrieval of information.
- Implemented error correction features within the web application to address model-generated mistakes, enhancing the overall quality and reliability of research outcomes.
- Successfully deployed the web application using Flask on a VPS (Virtual Private Server), ensuring robust performance and secure access for researchers and collaborators.

Best Practices Annotation Web Application

Tools, techniques and technologies used: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template.

- Developed a web application designed to streamline the annotation process for research data collection and parsing, enabling efficient and accurate data labeling.
- Implemented user-friendly features within the web app to correct predicted best practices-related content data, ensuring the highest quality and accuracy of annotated data.
- Successfully deployed the web application using Flask on a VPS (Virtual Private Server), ensuring secure and reliable access for annotators and other researchers.

Summarization Tagging and Pattern Gathering Web Application

Tools, techniques and technologies used: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template.

- Developed a web application tailored for sentence labeling, aimed at facilitating human tagging of data.
- The labeled data serves as a valuable resource for ongoing research in Natural Language Processing (NLP), contributing to the advancement of NLP techniques.

- Designed and maintained an efficient SQL schema to systematically organize and manage labeled data, ensuring accessibility and ease of data retrieval.
- Successfully deployed the web application, providing a secure and accessible platform for human labelers to contribute to research efforts in summarization and pattern gathering within the NLP domain.

Computer Vision (CV) projects:

OCR Model Enhancement and Data Collection for Ad Analysis

Tools, techniques and technologies used: Python, Pytorch, SQL, OpenCV, Pre-trained OCR models.

- Created an efficient OCR (Optical Character Recognition) model by utilizing other pre-trained OCR models.
- The objective was to extract text and text size from advertisements (ADs) from various sources, including Yelp, Meta, and others, to determine the importance of specific keywords in particular ADs.
- Developed a custom Python script using tools and libraries such as BeautifulSoup (bs4), Pypuppeteer, and others to collect relevant data from advertisements.
- Created a robust model capable of processing and extracting data from a wide range of ad formats and sources.

Data Engineering projects:

StreamFlow: Automated Daily Data Pipeline

Tools, techniques and technologies used: Python, SQL, Docker, Apache Airflow, Apache Kafka.

- Designed and implemented an automated data pipeline for daily data streaming and processing using Apache Kafka and Apache Airflow within a Docker environment.
- Created Directed Acyclic Graphs (DAGs) to automate the data ingestion and processing, enabling us to send daily insights to our clients.

Social Media Data Analysis and Classification

Tools, techniques and technologies used: Python, SQL, Docker, Apache Airflow, Apache Kafka, HuggingFace.

- Designed and developed a robust data pipeline for social media sentiment analysis and zero-shot classification to gather insights into what customers are most frequently discussing about their products.
- Utilized a RESTful API to collect data from social media and employed a pre-trained zero-shot classification model to analyze the sentiments of users in relation to the client's product.
- To streamline this process, orchestrated it within a data pipeline using Apache Kafka and Apache Airflow and created a dashboard to provide visual insights.

Real-time Data Streaming and Analysis *(Independent Study)*

Tools, techniques and technologies used: Python, SQL, Docker, Apache Airflow, Apache Kafka, Tableau.

- Designed and developed a real-time data pipeline using Apache Airflow and Kafka.
- Implemented the workflows for streamlined data processing: one for data streaming and another for trend analysis.
- Created an interactive data visualization dashboard to facilitate real-time monitoring and analysis.

Batch Data Streaming and Analysis *(Independent Study)*

Tools, techniques and technologies used: Python, SQL, Docker, Apache Airflow, Apache Kafka, Tableau.

- Designed and developed a real-time data pipeline using Apache Airflow and Kafka.
- Implemented the workflows for streamlined data processing: one for data streaming and another for trend analysis.
- Created an interactive data visualization dashboard using Tableau.