# Natural Language Processing (NLP) projects:

## BloggerGPT: Blog Writing AI Bot

**Technologies used**: Python, LangChain, ChatGPT, Prompt Engineering, etc.

- Conducted cutting-edge experiments with Large Language Model (LLM) tools, Such as ChatGPT, to explore their capabilities and potential applications.
- Successfully developed an AI-powered blog writer using ChatGPT and LangChain, enabling the generation of blog content based on provided subjects, ideas, and keywords.
- Demonstrated proficiency in integrating advanced AI technologies to automate content creation processes.
- Achieved a seamless synergy between ChatGPT and LangChain, allowing the AI bot to produce high-quality blog posts, meeting specific content requirements.
- Enhanced efficiency and productivity in content generation by harnessing the power of AI, and streamlining the writing process for various subjects and topics.

## BERT in Sentence Similarity Research: A Comprehensive Review

**Technologies used**: Python, PyTorch, LLM (BERT), etc.

- Spearheaded the creation of a comprehensive survey study dedicated to the exploration of sentence similarity, with a primary focus on leveraging labeled data and BERT models for analysis.
- Demonstrated a strong commitment to thoroughly exploring and elucidating semantic textual similarity, underlining its significance within the survey study.
- Effectively highlighted the advantages of fine-tuning pre-trained models as a central theme of the survey, offering valuable insights into model optimization.
- Enriched the survey study with additional context and in-depth explanations, ensuring that readers receive a comprehensive overview of the research findings and methodologies.

- Strategically planned and executed an exploration of unsupervised domain adaptation techniques within the survey, further enhancing its depth and relevance.
- Demonstrated dedication to knowledge sharing by preparing code resources in both script and notebook formats, enhancing accessibility and user-friendliness for fellow researchers and practitioners.

## Summarization Tagging Experiment

**Technologies used**: Python, Flask, SQL, Data Scraping, Data Cleaning, etc.

- Successfully conducted a research project focused on developing a rule-based system for categorizing sentences within articles as part of a summarization tagging experiment.
- Collaborated effectively with a manual tagger to annotate sentences from diverse articles within the chosen domain, leveraging human expertise in pattern recognition and tagging.
- Translated insights gained from human tagging into the creation of an automated Spacy rule-based system for categorizing sentences in other test articles.
- 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

**Technologies used**: Python, Pre-trained models, PyTorch, Data Scraping, Data Cleaning, etc.

- Successfully executed an experiment focusing on section-wise summarization, comparing its effectiveness to conventional summarization methods for breaking down lengthy articles into individual sections and summarizing them.
- Evaluated a wide range of pre-trained summarization models, including extractive, abstractive, and hybrid techniques.

- Systematically tracked and documented F1 and Rouge scores for both section-wise summarization and baseline summarization methods, facilitating comprehensive performance analysis.
- Generated essential datasets to support experimentation for both baseline and section-wise summarization.
- Completed summarization tasks for abstractive models and a subset of extractive models.
- Initiated the resolution process for models lacking requisite checkpoints.
- Expanded performance evaluation by recording precision and recall Rouge scores for various models and approaches.
- Created a comprehensive summary report for a sample article, featuring a comparative analysis of precision, recall, and F1 Rouge scores for all models and approaches, along with highlighting the extracted summary portion.
- Conducted supplementary experiments to enhance understanding of both summarization approaches, including exploring different input combinations and parameters to ensure a fair and thorough performance comparison between section-wise and baseline summarization.

## Research Paper: "Efficient Attentions for Long Document Summarization"

**Technologies used**: Python, GPU, Fairseq, BART-large model, Data preprocessing, etc.

- Embarked on a research project dedicated to implementing and comprehending the "Efficient Attentions for Long Document Summarization" model, which introduces innovative techniques for efficient long document summarization.
- Leveraged the Fairseq CLI tool to conduct initial experiments involving language modeling and summarization tasks using government report data.
- Demonstrated proficiency in data preprocessing by employing the BART-large model and training it for a few epochs, gaining valuable insights into data preparation for Fairseq model training.
- Successfully established data processing pipelines and model architectures in accordance with the research paper's guidelines.

- Proactively identified and addressed challenges encountered during training with the provided parameters, actively seeking solutions to ensure project progress.
- Undertook the task of implementing the "Longbart" model based on the research paper, addressing issues such as extended training times on a single GPU and script errors related to device assertions.
- Continuously fine-tuned the model's implementation, persistently troubleshooting to overcome challenges and ensure successful training and inference.
- Explored various GPU machine providers, including Azure, Lambda Labs, TACC, and others, to run these models on the cloud during experimentation, optimizing computational resources for research purposes.
- Striving for an in-depth understanding of the model's inner workings, parameters, and hyperparameters to harness its efficiency and effectiveness for long document summarization, aligning with the research paper's objectives.

## Conversion of "Efficient Transformer Model for Longer Sequences" from Fairseq to Plain PyTorch

**Technologies used**: Python, PyTorch, Fairseq, BART-large model, Data preprocessing, etc.

- Successfully converted the Fairseq implementation of the "Efficient Transformer Model for Longer Sequences" paper into a standalone PyTorch module.
- Developed encoder and decoder modules, eliminating Fairseq dependencies while preserving model integrity.
- Addressed specific parameter adaptations related to the Fairseq implementation for a seamless transition.
- Ensured data compatibility by studying Fairseq's preprocessing data format.
- Achieved a self-sufficient PyTorch implementation with operational encoder and decoder components.

## Experimentation with PRIMER Model for Multi-document Summarization

**Technologies used**: Python, PyTorch, etc.

- Conducted extensive experimental work on the pre-trained PRIMER model, purpose-built for multi-document summarization tasks.
- Utilized a diverse multi-news dataset and conducted experiments with various search queries to compile multiple documents, typically the top 5 Google search results, for summarization.
- Employed the pre-trained PRIMER model to assess its potential for extractive summarization, with a focus on capturing essential keywords from multiple documents.
- Adapted the project's direction by exploring alternative research papers and summarization methods tailored to the specific project requirements.
- Demonstrated a proactive approach to refine strategies and explore various summarization techniques, highlighting a willingness to adapt and optimize approaches based on project objectives.
- Focused on delivering effective multi-document summarization solutions tailored to the specific content and use case needs, ultimately enhancing content accessibility and relevance.
- Emphasized the importance of continuous learning and adaptability by aligning model capabilities with the project's specific requirements.

## Research and Implementation of Transformer Models

**Technologies used**: Python, PyTorch, etc.

- Engaged in an in-depth study of transformer models and their applications in natural language processing.
- Studied the foundational "Attention Is All You Need" paper, which introduced the transformer architecture, and gained insights into its inner workings.
- Reviewed code implementations to grasp the practical aspects of transformer models.

## Implementation of ULMFiT in PyTorch

**Technologies used**: Python, PyTorch, Data processing, Data preparation, Transfer Learning, etc.

- Implemented ULMFiT in PyTorch, utilizing Wiki2 data for language modeling.
- Achieved a remarkable test accuracy of approximately 90%.
- Demonstrated significantly faster training times compared to large-scale models like BERT.
- Applied key ULMFiT techniques, including gradual freezing, one-cycle learning rate scheduling, and early stopping.
- Contributed to NLP research by showcasing the power of transfer learning with ULMFiT.

# Web Development projects:

## Web Application for Prompt Engineering and Research Support

**Technologies used**: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template, etc.

- 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

**Technologies used**: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template, LLM, etc.

- 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.
- Contributed to the field of sentence similarity research by creating a user-friendly and powerful web-based tool that combines traditional methods with advanced AI techniques for more accurate and efficient analysis.

## Best Practices Annotation Web Application

**Technologies used**: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template, etc.

- 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 researchers.
- Contributed to improved research efficiency and data quality by creating a robust and accessible tool for data annotation and correction, aligning with best practices in data collection and analysis.

## Summarization Tagging and Pattern Gathering Web Application

**Technologies used**: Python, Flask, SQL, HTML, CSS, JavaScript, Jinja Template, etc.

- 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

**Technologies used**: Python, Pytorch, SQL, OpenCV, Pre-trained OCR models, etc.

- Significantly improved text detection and recognition accuracy by selecting and implementing advanced pre-trained OCR models.
- Evaluated multiple OCR models using images from the Yelp and meta ad library, achieving substantial enhancements in text extraction capabilities.
- Developed a custom Python script utilizing tools and libraries such as BeautifulSoup (bs4), Pypuppeteer, and others to collect relevant data from advertisements.
- Designed and implemented an automated data extraction process for advertisements, extracting key information for analysis and insights.
- Created a robust Python script capable of processing and extracting data from a wide variety of ad formats and sources, ensuring data consistency and accuracy.
- Demonstrated expertise in OCR model selection, fine-tuning, and practical solution development, coupled with data collection capabilities, to support informed decision-making in advertising strategies.

# Data Engineering projects:

## StreamFlow: Automated Daily Data Pipeline

**Technologies used**: Python, SQL, Docker, Apache Airflow, Apache Kafka, etc.

- Designed and implemented an automated data pipeline for daily data streaming and processing.
- Configured Apache Kafka in Docker for efficient data handling.
- Developed DAGs (Directed Acyclic Graphs) for automated data ingestion and processing using Apache Airflow.
- Achieved scalability and fault tolerance to handle large data volumes.
- Enhanced data-driven decision-making by providing timely insights from external sources through automated daily processing.

## Social Media Data Analysis and Classification

**Technologies used**: Python, SQL, Docker, Apache Airflow, Apache Kafka, Data Visualization, HuggingFace Models, etc.

- Designed and executed a robust system for sentiment analysis and zero-shot classification, covering diverse scenarios such as multi-label and multi-class classification.
- Implemented sentiment analysis at both document and sentence levels to provide nuanced insights.
- Successfully deployed zero-shot classification methods, achieving high accuracy in categorizing social media content.
- Demonstrated expertise in natural language processing (NLP) by segmenting text into sentences using Spacy and applying multi-label and multi-class classification techniques.
- Enhanced project outcomes with impactful data visualization, aiding in effective data interpretation and communication.

## Real-time Data Streaming and Analysis

**Technologies used**: Python, SQL, Docker, Apache Airflow, Apache Kafka, Data Visualization, Data Analysis, etc.

- Designed and developed a real-time data streaming system for dynamic data sources.
- 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.
- Enabled continuous monitoring of trends, offering valuable insights for analysis and research.

## Batch Data Streaming and Analysis

**Technologies used**: Python, SQL, Docker, Apache Airflow, Apache Kafka, Data Visualization, Data Analysis, etc.

- Designed and developed a real-time data streaming system for dynamic data sources.
- Developed a batch data processing project for efficient data handling.
- Implemented workflows to streamline data processing: one for real-time data streaming and another for trend identification.
- Created an interactive data visualization dashboard for in-depth historical trend analysis.
- Provided valuable insights into past trends and popular topics for research and monitoring purposes.