End to end project in NLP (with GitHub Deployment action):

Text Summarization

Jawaharlal Nehru University

School of Computer and Systems Sciences

Submitted by :

Shreya Pal

Enrollment number:

22/10/JC/032

Submitted to:

Dr. Piyush Pratap Singh

Contents

- Introduction - Definition and Purpose
- Approaches Text Summarization
- Project Overview
- Requisites
- Libraries and Model Used
- Methods and Techniques in Text Summarization
- Evaluation Metrics for Text Summarization
- Components
- Code, Prediction Pipeline and Result in User App
- Application of Text Summarization in NLP
- Challenges in Text Summarization
- Conclusion

Introduction

Definition:

Text summarization refers to the process of condensing a given piece of text to a shorter version while retaining its key information and main ideas. The goal is to create a concise and coherent summary that captures the essential meaning of the original text.

Purpose of this Project:

Text summarization aims to distill the essential information from large volumes of text, enhancing accessibility and efficiency. By automating the summarization process through Natural Language Processing (NLP), this project enables quick extraction of key insights, aiding users in grasping the main ideas without the need to sift through extensive content, thereby saving time and facilitating effective information consumption.

Approaches to Text Summarization

Extractive Summarization

Extractive summarization selects key sentences from the original text based on criteria like word frequency or relevance. Leveraging information retrieval and machine learning techniques, it ranks and extracts the most important content. This method efficiently condenses large amounts of information, creating concise summaries while avoiding the generation of entirely new content.

Abstractive Summarization

Abstractive summarization crafts summaries with varied language and structure, generating a fresh representation of information. It necessitates a deep understanding of content and the ability to mimic human language. Employing advanced NLP techniques like natural language generation, abstractive summarization models utilize deep learning, including recurrent neural networks (RNNs) and transformers, to produce nuanced and contextually rich summaries.

Project Overview

In this text summarization project, I'm adopting a hybrid methodology that combines elements of both extractive and abstractive summarization techniques. This approach aims to extract key information directly from the source text while also incorporating the ability to generate concise and coherent summaries by paraphrasing and rephrasing the extracted content. This midway approach seeks to enhance the overall effectiveness and informativeness of the summarization process. These are the steps I followed:

- Project Template Creation
- Project Setup and Requirements Installation
- Logging, Utils and Exception Module
- Project Workflows
- All Components Notebook Experiment
- All Components Modular Code Implementation
- Training Pipeline
- Prediction Pipeline and User App Creation

Requisites

- OOPs Concepts of Python Programming
- Natural Language Processing (NLP) - Basic Hugging Face
- GitHub Actions

Libraries and Model Used

- **transformers:** Hugging Face's Transformers library provides pre-trained models and a variety of natural language processing utilities for tasks such as text generation, translation, and sentiment analysis.
- **transformers[sentencepiece]:** An extension of the Transformers library with support for SentencePiece, a library for tokenizing text into subword units.
- **rouge_score:**rouge_score is a library for computing ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores, commonly used in the evaluation of text summarization and machine translation.
- **nltk:** NLTK (Natural Language Toolkit) is a library for working with human language data, providing tools for tasks such as tokenization, stemming, and part-of-speech tagging.
- **tqdm:** tqdm is a library for adding progress bars to Python code, making it easy to visualize the progress of iterative processes.
- **datasets:** Hugging Face's Datasets library provides a collection of datasets for natural language processing tasks, making it easy to download and work with various data sources.
- **python-box==6.0.2:** python-box is a library for working with Python dictionaries as if they were attribute-accessible objects, providing a more convenient syntax for dictionary manipulation.
- **uvicorn==0.18.3:** Uvicorn is an ASGI (Asynchronous Server Gateway Interface) server that can run web applications built on top of asynchronous frameworks, such as FastAPI.
- **sacrebleu==2.3.2:**sacreBLEU is a library for evaluating machine-translated text using the BLEU (Bilingual Evaluation Understudy) metric.
- **matplotlib:** Matplotlib is a data visualization library in Python, providing tools for creating static, animated, and interactive plots.
- **torch:** PyTorch is an open-source machine learning library, widely used for deep learning tasks, including neural network design and training.
- **notebook:** The notebook package is associated with Jupyter Notebooks, providing tools for interactive computing and data visualization.
- **ensure==1.0.2:** ensure is a small utility library for ensuring that certain conditions are met, providing a concise syntax for conditional checks.
- **fastapi==0.78.0:** FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.7+ based on standard Python type hints.
- **Jinja2==3.1.2:**Jinja2 is a templating engine for Python, used for dynamically generating HTML, XML, or other markup in web applications.
- **pyYAML:**pyYAML is a YAML parser and emitter for Python, allowing for the easy reading and writing of YAML files.
- **py7zr:** py7zr is a Python library for working with 7z archives, providing compression and decompression functionality.
- **pandas:** Pandas is a powerful data manipulation and analysis library, offering data structures like DataFrames for working with structured data.

Methods and Techniques in Text Summarization

Statistical Methods

These methods use statistical algorithms to identify key information and generate summaries based on statistical patterns.

Machine Learning Algorithms

Machine learning algorithms, such as supervised and unsupervised models, are used to extract important content and generate summaries.

Deep Learning Models

Deep learning models, including neural networks, have shown promising results in understanding and summarizing complex textual data.

Evaluation Metrics for Text Summarization

ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

ROUGE measures the quality of a summary by comparing it to reference summaries based on recall and overlap of n-grams.

BLEU (Bilingual Evaluation Understudy)

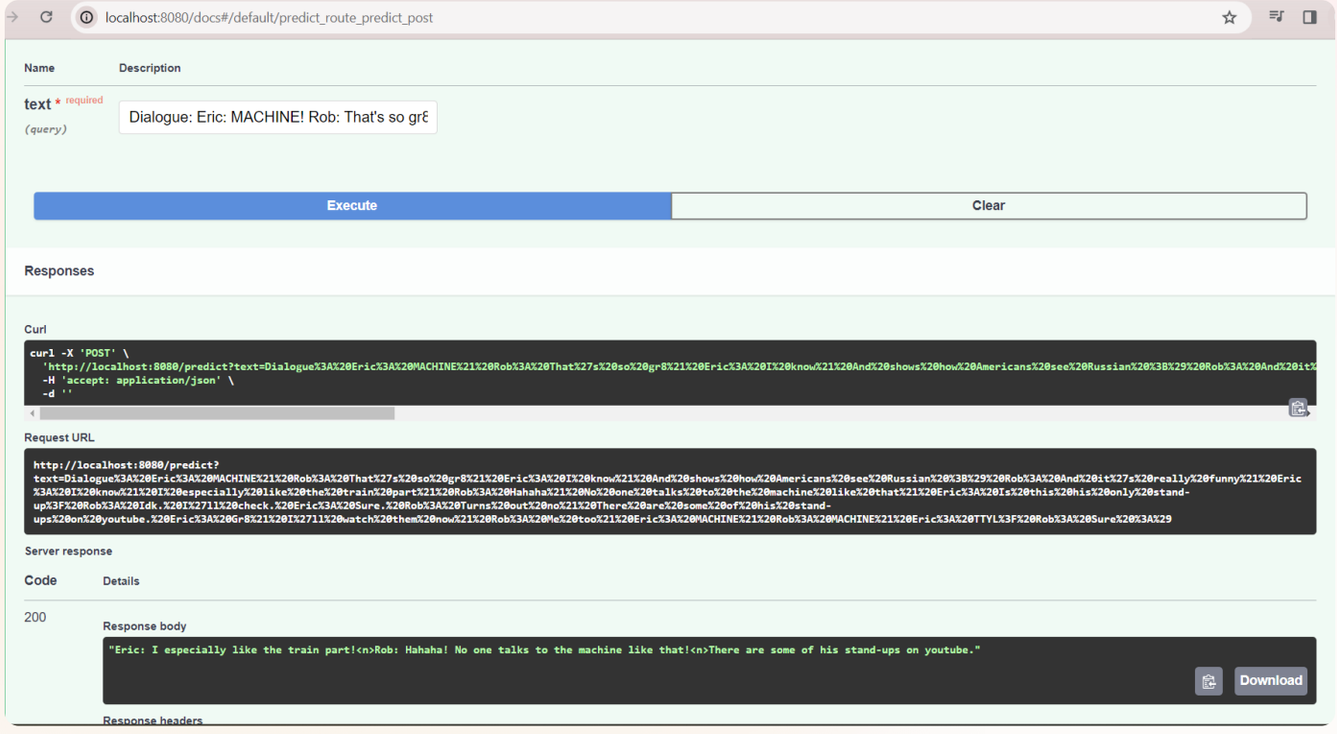
BLEU evaluates the quality of a summary by comparing it to one or more reference summaries based on n-gram precision.

Components

- **Data Ingestion** - This data ingestion component defines a `DataIngestion` class with methods to download a file from a specified URL and extract its contents into a local directory, utilizing logging and configuration parameters to manage the process, such as the source URL, local file paths, and extraction directory.
- **Data Validation -** This data validation component checks if all required files specified in the `ALL_REQUIRED_FILES` attribute of the configuration exist in a specified directory and updates a status file with the validation result, returning a boolean indicating whether the validation was successful or not.
- **Data Transformation -** This data transformation component uses the Hugging Face Transformers and Datasets libraries to convert a dialogue-summary dataset, loaded from disk, into a processed format suitable for training a summarization model, involving tokenization and formatting of input and target sequences.
- **Data Evaluation -** This data evaluation component assesses the performance of a sequence-to-sequence model (in this case, Pegasus) on a test dataset using the ROUGE metric, generating summaries for input articles and calculating ROUGE scores, which are then saved to a CSV file for analysis.
- **Model Trainer -** This model trainer component utilizes the Hugging Face Transformers library to fine-tune a pre-trained sequence-to-sequence model (Pegasus) on a specified dataset, adjusting hyperparameters based on the provided configuration, and saves the trained model and tokenizer for later use in summarization tasks.

Code, Prediction Pipeline and Result in User App

- **Project link :** [GitHub - Shreyapal27/Text_Summarizer_Project]
- **Prediction Pipeline:** This prediction pipeline uses a pre-trained sequence-to-sequence model (such as Pegasus) to generate a summary for a given input text, utilizing the Hugging Face Transformers library and a configured tokenizer, and then prints both the original input text and the generated model summary.
- **Result in User App:**



Applications of Text Summarization in NLP

Document Summarization

Summarizing lengthy documents like research papers and reports saves time and provides quick access to key information.

News Summarization

Automatically generating concise summaries of news articles helps readers grasp the main points quickly.

Social Media Summarization

Summarizing social media posts and comments helps users stay updated and comprehensively understand discussions.

Challenges in Text Summarization

1 Maintaining the Semantic Meaning

Ensuring the summary retains the core meaning of the original text is a challenge due to the complexity of language.

2 Handling Different Languages and Domains

Each language and domain presents unique challenges in summarization due to variations in grammar, vocabulary, and context.

3 Dealing with Long and Complex Documents

Summarizing lengthy and intricate documents without losing important information poses significant challenges.

Conclusion

1 Summary of Key Points:

I tried to explore text summarization in NLP, covering its approaches, challenges, methods, evaluation metrics, and applications.

2 Future Directions in Text Summarization Research:

Ongoing research aims to improve the accuracy and efficiency of text summarization algorithms for various domains and languages. And I myself would like to deploy this project on cloud so that I can train my model on larger data and improve accuracy.

Thank You