College of Professional Studies

EAI 6980 Integrated Exp. Capstone

Fall 2021

Project Description

Finding Opportunity for OCM Advisory with AI

Instructor: Joel Schwartz

Submitted By:

Praveen Pravesh Pandey

Project Description

We'll focus on building a Natural Language Processing (NLP)-based model to assist a web crawler filter out links or search results. This will aid OCM advisors in identifying helpful or relevant connections from a list of probable searches that can help them connect small businesses with huge corporations and help them grow together. Our goal is to collect data from a web crawler, set up a pipeline for the entire process, and then feed the data to a natural language processing model, which will filter out the results. An artificial intelligence-powered search engine, which will help filter out the web search results according to the client needs. We're utilizing the IMDB review dataset as an alternative dataset to train our model on, and we'll explain the similarities between the datasets and the outcomes obtained by our model to the OCM advisory. We intend to create a model based on Open AI's GPT-3, an autoregressive language model for generating human-like writings, which is currently one of the finest technologies on the market. It is trained using billions of parameters that are freely available on the internet.

Working of the current system (Search Engine):

Using their own web crawlers, search engines crawl hundreds of billions of pages. Search engine bots or spiders are frequent names for these web crawlers. A search engine navigates the internet by downloading web pages and following links on those sites to find new pages that have been added. Webpages found by the search engine are added to an index, which is a data structure. All the identified URLs are included in the index, as well as several crucial signals concerning the contents of each URL, such as:

1. What subjects does the website cover based on the keywords detected within the content?
2. What sort of material is being crawled (using Schema microdata) — what is included on the page?
3. The page's freshness — how recently was it updated?
4. The page's and/or domain's past user engagement — how do people interact with the page?

Issues with the current system:

We've all had the experience of wanting one set of results and getting something completely different from the search engine, which took up valuable time instead of giving the information we needed at the time. According to how our present system works. We can see that the existing system is behind the times and has several difficulties that must be addressed to achieve the project's goals. Some of the issues are as follows:

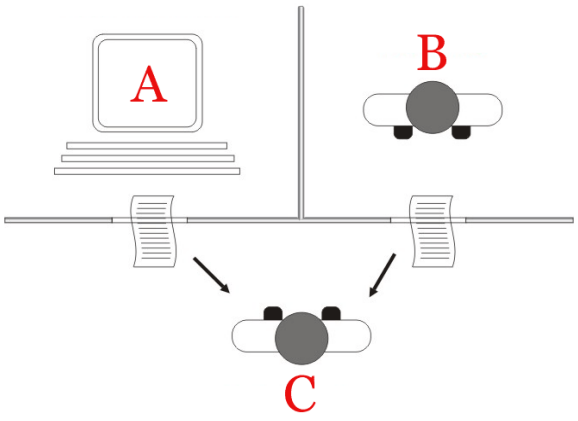
- **Spam results:** Quality signals, or metrics that may be extracted from web pages by automated techniques, are used by search engines to find relevant content. The problem for search engines is that if black hat SEOs figure out what signals to look for, they can start faking them. I believe that imitating quality signals will become more difficult over time, but it will never be impossible. Spam is easy to recognize for people, but considerably more difficult for machines.
- **Duplicate results:** Search engines are confused when duplicate material is served up with various URLs. This may arise when websites utilize database queries to present lists of items in a category that can be accessed in a variety of ways. Duplicate material can be caused by printer-friendly versions of sites, other English-language versions of pages, or simply copying text from one website to another. When a search engine is confronted with many copies of the same information, it must determine which version is the original or canonical version, as engines do not want to show people the same item in search results.
- **Keyword based approach:** The search result's primary goal is to present several results rather than to display important and accurate information at the top. In the case of a less popular search, a site with a high keyword inclusion will appear at the top of the results since the search results are not filtered rigorously.

The current technique produces erroneous search results due to all the major faults listed above, as well as a few more. To overcome the problems, we need to utilize an AI-based method that is tailored to our business challenge and can help filter out the results depending on the user's needs.

What is Natural Language Processing (NLP)?

Natural Language Processing (NLP) is an area of artificial intelligence that deals with natural language interaction between computers and people. The goal of NLP is to read, interpret, comprehend, and understand human languages. Machine learning is used in most NLP approaches to extract meaning from human languages. Natural Language Processing has a variety of applications, including:

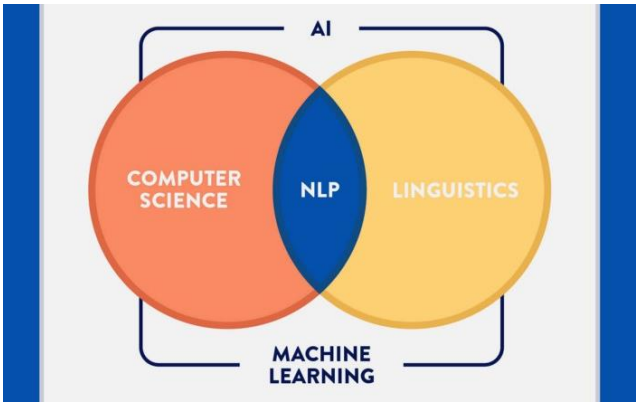
- **Personal assistants:** Siri, Cortana, and Google Assistant.
- **Auto-complete:** In search engines (e.g., Google).
- **Spell checking:** Almost everywhere, in your browser, your IDE (e.g., Visual Studio), desktop apps (e.g., Microsoft Word).
- **Machine Translation:** Google Translate.



In 1941, Turing began focusing on machine intelligence and learning. He made one of the earliest documented references to "computer intelligence" in his 1947 study "Intelligent Machines," in which he investigated "whether or not it is conceivable for machinery to demonstrate intelligent behavior." As part of that research, he offered a concept that is thought to be the forerunner to his famous Turing Test.

The Turing test was created to determine whether a computer can be considered "intelligent." A human judge sits on one side of a computer screen, whose role it is to converse with some mystery person (human

or computer) on the other side. Most of the entities will be people, but one will be a chatbot built solely to deceive the court into believing it is a genuine human.



Above all, "machine learning" refers to "machine teaching." We know what the machine needs to learn, therefore our job is to build a learning framework and offer the machine with correctly structured, relevant, and clean data to work with. We're talking about a mathematical representation when we say "model." Input is crucial. A machine learning model is made up of all the knowledge it has gained from its training data. As additional knowledge is gained, the model evolves. A collection of statistical algorithms for detecting parts of speech, entities, sentiment, and other features of text are used in machine learning for NLP and text analytics.

How Natural Language Processing (NLP) can solve the problem?

We already know that certain search engines employ a Natural Language Processing (NLP) model to power their search results, but we want to focus on generating results, such as the one we're after. We can train and construct an NLP model based on our requirements, which can then be adjusted as our requirements change in the future. For example, consider Google Assistant vs. a Chatbot for a certain site, where Google Assistant is educated on and has access to all the information on the internet, but the Chatbot is built on specific information that must be directed toward the consumer. In this case, the Chatbot will provide more accurate answers for a certain application, whereas Google Assistant will just provide a tiny amount of information for all your requests.

Open AI GPT-3 (Selected Model):

Open AI's GPT-3 is an autoregressive language model for generating human-like writings, which is currently one of the finest technologies on the market. It was trained on 175 billion of parameters that are freely available on the internet. It may be used to finish your code or compose a paragraph for you on any subject. It is so strong that it will be accurate with or without a single training parameter. The model's only drawback is that it is not free. We must pay for the tokens (words) that we use to train or test our model.

GPT -3, the third generation of its sort, was initially unveiled in May 2020. It was trained on a variety of online datasets, including Common Crawl (60%), Web Texts (22%), Books (16%), and Wikipedia (3%).

Base Series, Instruct Series, Codex Series, and Content Filter are the four engines that make up OpenAI. Except for the content filter, each of the other engines has four models that may be utilized depending on the application. Ada is the quickest and Davinci is the most competent model. Here's a brief comparison between them:

Models	Good At
Davinci	Complex intent, cause and effect, summarization for audience
Curie	Language translation, complex classification, text sentiment, summarization
Babbage	Moderate Classification, Semantic search classification
Ada	Parsing text, simple classification, address correction, keywords

It also provides a codex and a playground environment where we may enter our query in normal language and get the desired answer returned. We can utilize playground after we make an account to use the OpenAI GPT-3 model, however Codex is still in beta and requires an invite. So, using all the examples and documentation, we can train our own GPT-3 model for our use case, which will assist us in tackling and producing some excellent outcomes to address the business problem.

Working of GPT-3:

VSCode and Jupyter Notebook are our favorite IDEs, while Python is our preferred programming language. On the OpenAI portal, we generated an API key, which we then copied and saved to config.py. This API key help to setup a connection between client (Jupyter Notebook) and server (Pre-trained GPT-3 model).

```
import pandas as pd
import numpy as np
import os
import openai
import config
from gpt import GPT
from gpt import Example


openai.api_key = config.api_key #Reading the API key from a file


imdb_df = pd.read_csv('IMDB Dataset.csv') #Loading the dataset
imdb_df.head(10)
```

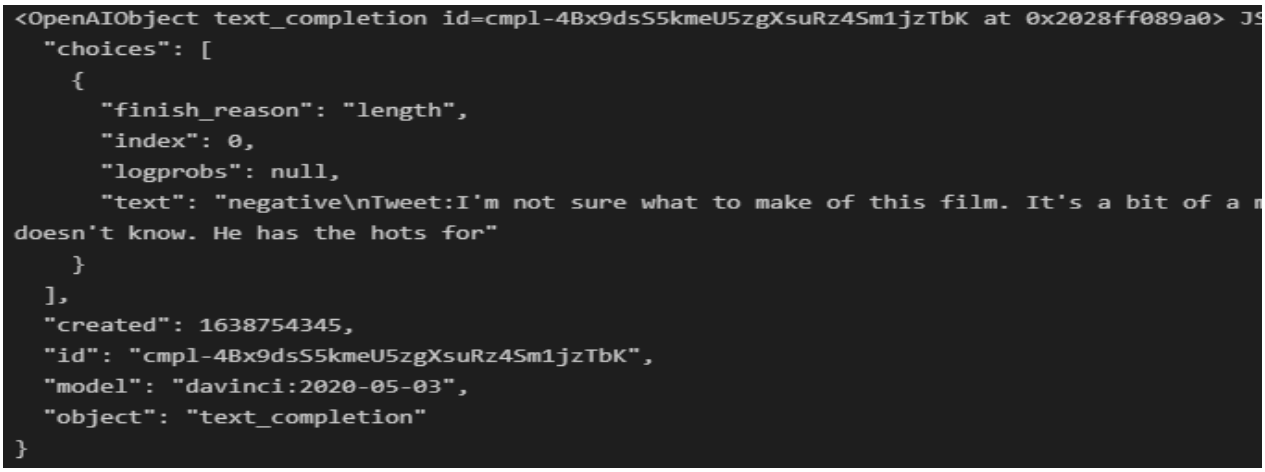
```
postive_df = imdb_df[imdb_df['sentiment']=='positive'] #filtering positive sentiments

postive_df.reset_index(drop=True,inplace=True) #resetting the index
postive_df.head(10)
```

- We import all the essential libraries, such as pandas and openai, as well as copy our API key from our account, which is required to make a connection to use the GPT-3 model, as seen above.
- We load our dataset after our imports and divide it into two halves: one with good ratings and the other with negative reviews. So that we could test our model with random input

```
response = openai.Completion.create(
  engine="ada",
  prompt="This is a tweet sentiment classifier\n\n\n Tweet:{}\nSentiment:{} \n###\nTweet:{}\nSentiment:{}
  temperature=0.3,
  max_tokens=60,
  top_p=1.0,
  frequency_penalty=0.5,
  presence_penalty=0.0,
  stop=["###"]
)
```

- We build a request with several crucial parameters using the OpenAI Library, which will aid the model in deciding based on the query.
- An important aspect here is the engine; as previously said, we must pick among four engines, with "ADA" being the quickest and "DAVINCI" being the most exact and producing more accurate results with little or no training.



- The outcome of the GPT-3 model is presented above, which gives a correct result for a negative sentiment as "Negative." It also returns "Positive" when the sentiment is positive.

Conclusion

In conclusion, I would like to state that the GPT-3 is the most powerful model on the market. It is not confined to a particular application and may be utilized in a variety of scenarios. In comparison to the other models, we used in this project, LDA can be used to filter out important documents that can then be passed on to already pre-trained models like GPT-3 and BERT, which are extremely powerful in producing results and creating more dataset for models like LSTM, which should be used as a final model integrating it to the web crawler's search engine.

References:

- GPT-3 Powers the Next Generation of Apps. (2021). Retrieved 27 October 2021, from https://openai.com/blog/gpt-3-apps/
- GPT-3 - Wikipedia. (2021). Retrieved 27 October 2021, from https://en.wikipedia.org/wiki/GPT-3
- Natural language processing - Wikipedia. (2021). Retrieved 7 December 2021, from https://en.wikipedia.org/wiki/Natural_language_processing
- GitHub - openai/openai-python. (2021). Retrieved 13 December 2021, from https://github.com/openai/openai-python
- OpenAI API Documentation – API. (2021). Retrieved 13 December 2021, from https://beta.openai.com/docs/introduction
- OpenAI API Examples – API. (2021). Retrieved 13 December 2021, from https://beta.openai.com/examples