JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

SCIENTIFIC RESEARCH PAPER RECOMMENDATION USING GPT-3 PROMPT ENGINEERING

[1]Sarabu Joshna, [2]B.Ramya Sree
[1,2]Assistant Professor
[1,2]Computer Sscience and Engineering
[1,2]Bharat Institute of Engineering and Technology, Hyderabad, India.

Abstract : Writing a research paper is a way to present your original findings to other researchers or to evaluate their work. As a result, they play a crucial role in how modern research develops, where one researcher's work builds on that of another. Papers must strive to enlighten, not impress, in order to accomplish their mission. They must be easily readable, which means they must be precise, clear, and brief. If they are informative as opposed to cryptic or self-centered, they are more likely to be mentioned by other researchers. The idea of personalized research paper suggestions was very popular, and since then, it has undergone numerous improvements to become more successful and efficient. The power of recommendation algorithms has changed how viewers interact with research papers and articles, resulting in a more engaging and personalized experience. Utilizing cutting-edge techniques that enable people to find the material they like can help you stay ahead of the curve as the entertainment business develops.

IndexTerms- Collaborative Filtering, Autoencoder, GPT-3, Cryptic.

I. INTRODUCTION

An awareness of user behavior and tastes, as well as the capacity to evaluate massive volumes of data, are necessary for developing an efficient research paper recommender project. We may strengthen the bond between viewers and the material by making individualized recommendations, which raises user satisfaction and retention rates. It is no secret that tailored recommendations will dominate the entertainment market, and a platform that uses the strength of these algorithms is revolutionizing how viewers interact with research papers and articles.

Data Source: The arXiv

The arXiv dataset is freely available via Google Cloud Storage buckets. Each paper has an entry in this file with the following information: Identification, Submitter:Who wrote the research paper?, Names of the paper's authors, Comments on the paper's title: Information about the amount of pages and the figures journal-ref[1] provides information on the journal in which the paper was published. Identifies a digital object.abstract: The categories in the paper's abstract are: Versions of the ArXiv system's categories and tags are: a history of versions.

II. RELATED WORK

Data Preprocessing and Feature Extraction Using Collaborative Filtering

Collaborative filtering produces recommendations by examining similarity between people and things at the identical time, reducing some of the shortcomings of content-based filtering. Serendipitous proposals are now conceivable; that is, collaborative filtering algorithms can recommend something for customer A by considering the choices of a user B whom shares those interests. Furthermore, rather than having to construct features by hand, their embeddings can be trained automatically. As a result, for research paper recommendations, collaboration can be employed to acquire the best document for completing study.

III.EXPERIMENTS AND RESULTS

We used the arXiv data, for collaborative filtering. Next, we divided the data into train and test using the functionality of the train-test split. Here, we were careful to consider which data point should be used as the train data and which as the test data. If a person with ID 2800, for instance, evaluated more than one paper, we used the user's most recent timestamp as the test set and the rest as the train set. Using train-test split while running recommendation algorithms is extremely popular. However, one must exercise caution when dividing the data because we might train by using the most recent user rating and test by using the previous rating [2].

We discovered a very intriguing idea about implicit and explicit feedback while developing this model. Explicit feedback is nothing more than the collection of precise quantitative data, such as the quantity of likes or other explicit scale evaluations. Implicit data, on the other hand, is unofficial information gathered from user contact.

The explicit feedback dataset was used to run the model. Therefore, we have to conduct OneHotEncoding on features like ratings in order to transform the dataset from explicit feedback to implicit feedback [3].

After applying the one hot encoding it is observed that all the ratings contain positive values. because the reason behind this the data which we have taken is the papers which have reviewed by the user. So, there is a need to introduce a class called negative class.

So, to achieve the negative class introduction we used the Python 3 so that for every 3positive class atleast one negative class is achieved. In order to use the collaborative our data is preprocessed so it can be given to collaborative filtering.

IV.RECOMMENDATION SYSTEM USING PROMPT ENGINEERING AND GPT-3:

Large models of language are computerized systems that have been taught to learn and comprehend human language using vast volumes of textual data. These models can do a number of language-based tasks, including translation, summarization, and question-answering since they are built to process and generate genuine language.

Prompt Engineering and NLP-based large language models go hand in hand. Rather than being delivered implicitly, it is a concept in artificial intelligence where the task description is included clearly in the input and expressed as a query or command. Prompt engineering often involves transforming one or more tasks into a prompt-based dataset, training a language model with "prompt-based learning" or merely "prompt learning," and then analyzing the results.

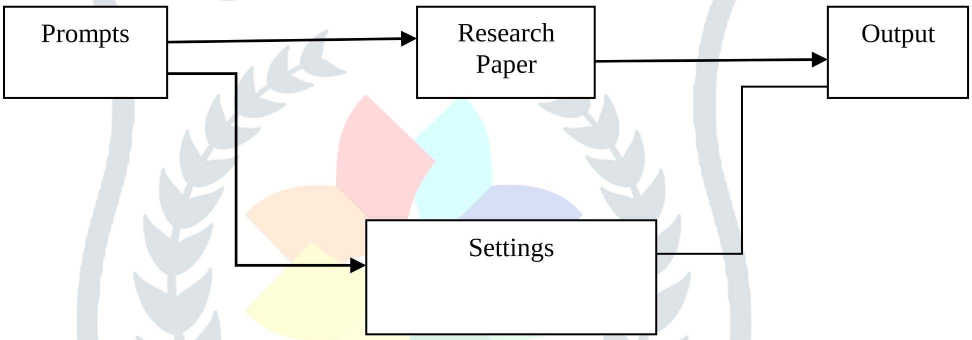

Fig: Pictorial representation of prompt recommendation.

We developed and tested a number of prompts designed to elicit particular information from the GPT-3 model and gave it the context and limitations it needed to provide recommendations. Due to the excellent capabilities of the DaVinci model, we initially intended to employ it. However, after experimenting with various prompts, we found that we were not happy with the ratings that the model generated [6]. It frequently gave the user's previously watched research papers' average ratings. After doing more research, we discovered that the dialog-based GPT 3.5 turbo model produced the greatest results.

Additionally, the turbo model had its share of drawbacks, just like most things in life. No matter how carefully we customized the prompt or provided detailed instructions, because it is a dialog-based approach, it refused to generate a single numeric response. This model's ratings were inflated with phrases like:

"Based on your past viewings and ratings, I predict you will like A Hybrid Variation Autoencoder for collaborative filtering and give it a rating of 4.0 out of 5".

We decided to send the output created by GPT 3.5 Turbo to DaVinci to extract the rating from the phrase since we wanted a single numeric output to compare it to the actual ratings as well as the results of the VAE model [5]. We also tried to utilize Ada because it was a simpler challenge, but it was unable to generate enough outcomes.

```
# Code snippet of the 2 models we finally ended up using:
response = openai.ChatCompletion.create
        model="gpt-3.5-turbo",
    messages=[
    #{"role": "system", "content": "You are a research paper rating expert."},
     {"role": "system", "content": "You are a research paper rating expert. You ignore the subjectivity of the matter and only return a single number as output, no words."},
     {"role": "user", "content": "I have rated the research paper I have viewed on a scale of 1 to 5, where 1 indicates that I did not like the research paper at all and 5 means that I satisfied with the research paper.}"},
    {"role": "assistant", "content": """"My viewing history: """ + row[' research paper_ratings_y'] },
    {"role": "assistant", "content": """ research paper I have not viewed: """ + row['title']},
    #{"role": "assistant", "content": """Desired output format '{research paper Name: rating}' """},
```

```
        {"role": "user", "content": "Taking into consideration my viewing history and how I have rated research papers of the corresponding genre, plot and cast, how much would I like the research paper I havenot viewed on a scale of 1 to 5? Do not give a range, only give one number. I want to get an idea of whether to view it or not."}
        ]
    #max_tokens=3,)
  a = response.choices[0].message.content

  response1 = openai.Completion.create(
    model="text-davinci-003",
    prompt=['Extract the rating mentioned in: '+ a + 'Return a single floating point number. Desired output format: number'],
    temperature=0,
    max_tokens=10,
    top_p=1,
    frequency_penalty=0,
    presence_penalty=0)
  df_test_1.loc[index, 'prompt_response'] = response1.choices[0].text
```

Here's a rundown of the prompts we tried:

We began with the prompts listed below. We weren't thinking about how GPT would rank the research paper depending on the user's past at first. We just reasoned that if the genuine rating was larger than 3.5 and GPT also stated that the user should view the research paper, we would be able to determine whether or not the GPT 3 had made the accurate prediction [7].

DaVinci:

"I've seen and rated the following films out of 5 stars: **[Research paper 1: rating, Research paper 2: rating, Research paper 3: rating, Research paper 4: rating, Research paper 5: rating, Research paper 6: rating, Research paper 7: rating, Research paper 8: rating, Research paper 9: rating, Research paper 10: rating]".**

I have not viewed the research paper, and I would like your recommendation on whether or not I should view it based on my ratings of these 10 research papers. However, we were unable to obtain the predicted rating by responding to this question; instead, we received qualitative rather than quantitative comments. To validate its performance, we thought it would be great to receive the exact rating from GPT. Hence, we modified the prompt and attempted the ones below.

V. CONCLUSION AND FUTUREWORK

Ranking plays a vital role in searching the relevant research paper as its helping the researcher in getting the valid data without consuming much time. We have recommended the research paper using chat GPT prompt engineering this can be extended further by involving many technologies which may give accurate results. Machine learning is an emerging technology now a days where searching made easy for every area. Hence, this research can be extended further using different methodologies using machine learning and deep learning techniques.

VI. ACKNOWLEDGMENT

We would like to acknowledge the support, encouragement and facilities provided to us by BIET management during our research work and realizing this research paper smoothly.

REFERENCES

[1]https://www.doi.org/

[2]https://medium.com/@aiandinsights/personalize-your-content-recommendations-with-chatgpt-dbc7f9d59877/12-387-150487359549-56.pdf

[3]https://devpost.com/software/breadbot-a-chatbot-researchpaper-recommendation-system#:~:text=use%20breadbot%3a%20a%20chatgpt%2dbased,with%20a%20personalized%20research paper%20suggestion.

[4]https://www.wsj.com/articles/can-chatgpt-recommend-research papers-a-film-buff-put-it-to-the-test-5e7e6516.

[5]https://blog.reachsumit.com/posts/2023/05/chatgpt-for-recsys/.

[6]https://github.com/harrisonpim/chatgpt-film-recommendations.

[7]https://medium.com/@krees/product-design-with-chatgpt-unaware-recommendation-engine-ce493da1595e.