

Final Project NLP(2023S1)(CHAT-PDF Clone)

Ian Wei Chong
University of Adelaide
Adelaide, Australia

a1784585@student.adelaide.edu.au

ABSTRACT

Over the last few years, 'Natural Language Processing' (NLP) has emerged as a vital field within the realm of 'Artificial Intelligence' (AI), enabling machines to process, understand and generate human language. On the other hand, have you (the reader) ever experienced a situation where you needed to grasp and understand knowledge given by a specific academic paper but lacked the will or time to full read through it? Within the past year, AI and NLP had made extensive groundbreaking progress with the creation of 'ChatGPT', allowing humans to interact intuitively to an AI who can exercise and complete different language tasks, and it is with this development that an implementation of its application programming interface (API) had been developed called 'ChatPDF'. Its goal? To assist users in gleaning information from a specific PDF as if they were learning from another person or being taught by another person. In this paper, we aim to explore an implementation of OpenAI's GPT-3.5 API to develop our own 'ChatBot', capable of processing PDFs and questions from user to produce a reasonably accurate reply based on its inputs.

ACM Reference Format:

Ian Wei Chong. 2022. Final Project NLP(2023S1)(CHAT-PDF Clone). In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 6 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 INTRODUCTION

In recent years, advancements in AI have propelled the development of innovative technologies that enhance our ability to interact with digital systems and process large amounts of information more effectively. Through certain chains of the development progress in AI, we've reached the point in time where generative AI has developed enough to sustain highly versatile and context-aware chatbots that humans can communicate with intuitively. Especially in the past year (as of writing this paper), with OpenAI's GPT-3.5 API being made more accessible to the general masses, we've seen an extreme surge in different implementations/applications of its generative pre-trained models.

One implementation of AI chatbots, is ChatPDF. An online tool capable of processing large amounts of textual information within a PDF document and allows users to perform queries to it based on the context of the given textual information. Its an amazing tool

that harnesses the power of AI and NLP to automate the entire process. Using a number of different techniques AI and NLP it essentially allows humans to be able to converse or be taught by chatbot instead of processing and reading text manually.

Which begs the question, we have a LLM in GPT-3.5's API and some PDFs with textual information. Can we develop an AI chatbot with these two things? Short answer: Yes. Throughout this paper, I aim to explain my design process in developing a simple ChatPDF clone.

2 RELATED WORKS

Before we delve into design methodology, I'd like to first run through some academic papers and other research delved into the development of NLP and AI that would fit the context of the project.

The first paper I'd like to go through "How Robust is GPT-3.5 to Predecessors? A Comprehensive Study on Language Understanding Tasks"[2]. In this paper, Chen et al. performs a comprehensive study on the robustness of GPT-3.5 in the context of NLP. It's essentially a study of several tests conducted upon GPT-3.5's NLP capabilities to determine its performance compared to other fine-tuned language models and further explores its weaknesses in degradation issues. This paper was used to give insight on the LLM GPT-3.5 to be used in the project described in this paper.

Next, going through a search for existing frameworks for ChatPDF and academic papers correlating to the topic, I stumbled upon "Applying ChatGPT and AI-powered tools to accelerate evidence reviews"[3]. This paper gives insights on how the application functions on a deeper level in the context of researching academic papers, how to use the tool for maximum efficiency and the comparison between ChatPDF as an AI-powered app compared to other AI-powered apps. It is a good paper to get deeper insights on how these tools complement evidence review, or in my opinion, it can be applied to academic papers and the like with reducing time and effort.

Moving on, upon some readings of existing literature reviews, a recurring topic regarding NLP in chatbots have seemingly stood out more than some which is the specific technique known as word embeddings. It's to my understanding that there have been several implementations of chatPDF-like applications and the commonality between them seem to be the use of word embeddings and vector space models. To this end, I've sought an academic paper in "Word Embeddings: A Survey"[1] which discusses many different techniques for implementing for building word embeddings. In the context of this paper however, I'll be using OpenAI's embeddings model for simplification yet it's still pretty useful to understand how it will work on a lower level, especially since it helped me develop a simple design architecture.

Lastly, we have "ThoughtSource: A central hub for large language model reasoning data"[4] another paper I deemed useful

Permission to make digital or hard copies of all or part of this work for personal or academic use, or to republish, is granted by ACM, provided that the copyright notice, this permission notice, and the full citation are included in the copy. This work is distributed under the Creative Commons Attribution 4.0 International License. For more information, see <https://creativecommons.org/licenses/by/4.0/>.
for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00
<https://doi.org/XXXXXXX.XXXXXXX>

in the context of ChatPDF, existing frameworks and LLM. The main idea however with this paper, is the idea of chain-of-thought (CoT) prompting to aid and enhance reasoning capabilities of LLMs. It proposes ThoughtSource as a solution to LLM's struggles to solve issues with mathematical problems or questions that require multi-hop reasoning. Its an interesting idea paper to think about improvements to the project this paper details which uses GPT-3.5 as its LLM.

3 DESIGN

Figure 1 describes a simple design/architecture for a simple chatbot incorporating OpenAI's GPT-3.5 and LangChain, a framework for developing AI applications powered by language models. Using these 2 tools I present a ChatPDF clone utilizing NLP techniques such as word embeddings and vector representation. As the application will be a command line script, I will use a notebook to process some application statistics such as token usage.

3.1 Crucial Packages

The first two crucial packages as mentioned in a previous section, are LangChain and OpenAI's GPT-3.5. Within the past year of GPT-3.5 being released to the world, numerous different AI products utilizing its API has appeared such as ChatPDF, but from this boom in AI products/services frameworks were naturally bound to be developed to develop products/services more efficiently. In this context, LangChain. To put it simply, LangChain is a framework that can be used to work together with LLMs to produce an application. The reason why LangChain was selected was because of the existing works available on the internet where people demonstrated its NLP capabilities with AI applications. It gives us a framework with tools that allow us to pre-process text, embed, plug queries into a LLM and other 'Chaining' capabilities but we're only using generic chaining in this scenario.

Next we dive deeper into another core package that I deem necessary to the project. I've elected to use meta's 'Faiss' package as my means to store vector representations. The idea is that it allows us to create a vector database where we can store our word embeddings and query it based on a user's question to a chatbot for pieces of information our LLM can infer and build a reply with.

3.2 Design Description

Referring to Figure 1 as the architecture of our chatbot, we first begin with a user and a PDF document. The user wants to be able to simplify their learning experience from a paper by asking an AI, so an AI needs to be able to process and understand the document first. 3 core components we need, word embeddings, a vector database to store embeddings and an LLM. The idea is that we use LangChain as a basis to complete most of the tasks with 'Faiss' to store vectors.

So beginning with the PDF document our application receives, we need it to undergo some pre-processing to create word embeddings. The first thing I did was to convert the PDF document into text and split them into chunks of text. The reason we have to do this is so that we don't hit the token limit of 800 per project description, we set the chunk size to relatively small number like 512 to avoid hitting it.

Another reason for chunking is because we want to be able to give the LLM only the necessary portions of data for it to produce an answer instead of the entire document besides token limits. With the chunks produced, our next step would be to embed each of the chunk giving it a numerical value that a computer would understand, a technique that captures inter-word semantics.

From there, we would need to be able to store our word embeddings for future referencing when a query comes in. This is where Faiss comes in to allows us to create a vector database easily. So now that we have a vector database full of vectors representing all the different chunks of text, we can move back to the part where a user can start asking questions about a document.

Given a question from a user, we have to first make it go through the same embedding process as the other chunks, turning it into a vector that we can query the database with to find relevant chunks pertaining to the question. Once we've done that and retrieve the relevant chunks of text, we pass it to the LLM with both the original question and the relevant chunks of data.

With the core of it done, we now continue to make the process user-friendly in a command-line interface as a python script. The input being: -

```
- python ChatPaper.py [xxx.pdf]
```

Remember to replace xxx.pdf with your own PDF and that it has to be in the same directory as where you are running your script. The script was made using conda using Python 3.10.

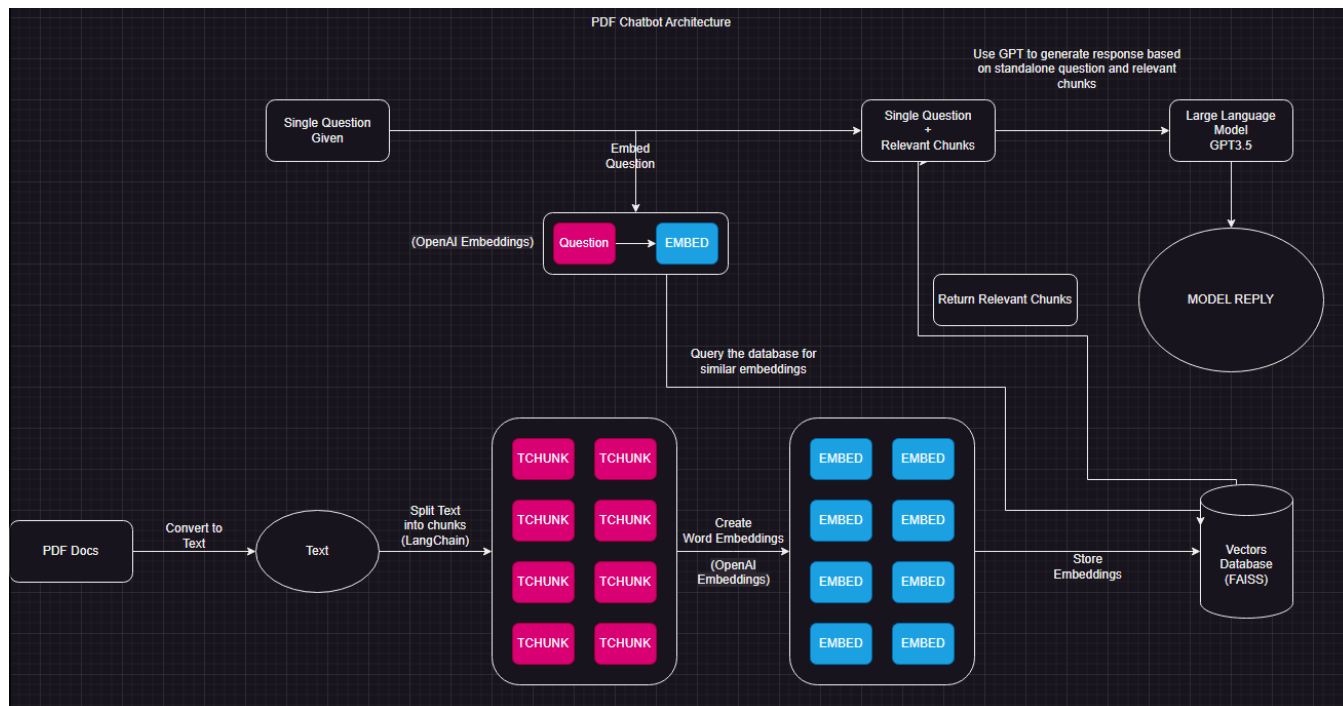


Figure 1: Chatbot Architecture/Design

4 RESULTS AND EXAMPLES

With our model now complete we can take a look at some tests using some papers. The first paper being "Language Models are Few-Shot Learners" and the second being "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". I've named them example1 and example2 respectively when running my chatbot script.

```

Administrator: Anaconda Prompt
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py example1.pdf
please wait
Enter question:what kind of neural network architecture
is used in this paper?
Tokens Used: 658
  Prompt Tokens: 647
  Completion Tokens: 11
Successful Requests: 1
Total Cost (USD): $0.01316
answer:
  This paper does not discuss any particular neural netw
ork architecture.
Enter question:|
  
```

Figure 2: "Language Models are Few-Shot Learners" and its given question "what kind of neural network architecture is used in this paper?"

```

Administrator: Anaconda Prompt
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje
python ChatPaper.py example1.pdf
please wait
Enter question:what datasets have been used for evaluat
ion
Tokens Used: 506
  Prompt Tokens: 495
  Completion Tokens: 11
Successful Requests: 1
Total Cost (USD): $0.01012
answer:
  Natural Questions, WebQuestions, TriviaQA.
Enter question:|
  
```

Figure 3: "Language Models are Few-Shot Learners" and its given question "what datasets have been used for evaluation"

```

Administrator: Anaconda Pro x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje
>python ChatPaper.py example2.pdf
please wait
Enter question:what are the main discoveries of this p
aper?
Tokens Used: 649
    Prompt Tokens: 616
    Completion Tokens: 33
Successful Requests: 1
Total Cost (USD): $0.012979999999999999
answer:
This paper does not present any discoveries. It is an
appendix for a paper titled "BERT: Pre-training of De
ep Bidirectional Transformers for Language Understandi
ng".
Enter question:|

```

Figure 4: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" and its given question "what are the main discoveries of this paper?"

```

Administrator: Anaconda Pro x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje
>python ChatPaper.py example2.pdf
please wait
Enter question:what is the key insight of the proposed
method?
Tokens Used: 703
    Prompt Tokens: 654
    Completion Tokens: 49
Successful Requests: 1
Total Cost (USD): $0.014060000000000001
answer:
The key insight of the proposed method is that a deep
bidirectional model is more powerful than either a le
ft-to-right model or the shallow concatenation of a le
ft-to-right and a right-to-left model.
Enter question:|

```

Figure 5: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" and its given question "what is the key insight of the proposed method?"

Shown above are some results and examples of the chatbot in action working as intended.

5 DISCUSSION AND INVESTIGATIONS

As shown in the previous section the chatbot built according to the design seems relatively stable, but a few interesting factors and tests comes to mind after these basic the basic tests were passed.

The first investigation I conducted was "If chunks of text were used for word embeddings and only relevant vectors were retrieved from the database, what happens if you ask an overarching question like summarizing the paper?" Its an interesting thought because if it fails that means the model only works if you ask specific questions.

```

Administrator: Anaconda Pro x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje
python ChatPaper.py example1.pdf
please wait
Enter question:summarize paper
Tokens Used: 615
    Prompt Tokens: 539
    Completion Tokens: 76
Successful Requests: 1
Total Cost (USD): $0.012300000000000002
answer:
This paper analyzes the potential contamination of a
dataset by comparing the clean subset with the origina
l dataset. It reports the normalized difference in the
means and the p-value to measure the contamination. I
t then discusses how models that consistently produce
texts that are more impressive than human articles, co
uld potentially lower human performance on the task. A
dditional non-news samples can be found in Appendix F.
Enter question:|

```

Figure 6: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" and its summary

But to my first attempt at breaking the model, it seemed like the chatbot could still withstand the attempt and thus returned a summary of the paper.

The next attempt at breaking the chatbot, I tried to attack it through its memory, if it could remember previously asked questions and reply.

```

Administrator: Anaconda Pron x + - □ x
please wait
Enter question:what are the main discoveries of this pa
per?
answer:
This paper does not present any discoveries. It is an
appendix for a paper that presents the BERT model for l
anguage understanding. The paper it is an appendix for
presents the BERT model and additional implementation d
etails, experiments, and ablation studies.
Enter question:what question did i ask previously?
answer:
The question asked previously was: Given a question an
d a passage from Wikipedia containing the answer, the t
ask is to predict the answer text span in the passage.
Enter question:|

```

Figure 7: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" and testing remembrance

Lo and behold, errors occurred, judging by the given answer I suspect the chatbot got confused by a particular chunk and assumed I was referring to it due to the matching query. So it could be said that a possible improvement might be necessary just before the query portion to test whether the question given by users require inference or the answer could be gleaned from its conversational chain. I should mention that Figure 7 was from before I added token

count code which seemed to cause a different output than this so I've elected not to swap figure 7 as it demonstrates a particular behaviour.

Another thing I've tried was to ask questions sequentially, to see if answers were affected or not.

```

(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py example1.pdf
please wait
Enter question:what kind of neural network architecture
is used in this paper?
Tokens Used: 658
    Prompt Tokens: 647
    Completion Tokens: 11
Successful Requests: 1
Total Cost (USD): $0.01316
answer:
    This paper does not discuss any particular neural netw
ork architecture.
Enter question:what datasets have been used for evaluat
ion
Tokens Used: 506
    Prompt Tokens: 495
    Completion Tokens: 11
Successful Requests: 1
Total Cost (USD): $0.01012
answer:
    Natural Questions, WebQuestions, and TriviaQA
Enter question:|
  
```

Figure 8: "Language Models are Few-Shot Learners" and test- ing questions coming after that

So as compared to Figure 2 and 3 we can see the chatbot still provided the same answers but it can also be said more testing is necessary with more testing to provide a solid answer. Another interesting note is that the chatbot can handle sequential questions.

Now we move towards the discussion, throughout this project I found it interesting how simplified the development has been through the existence of LangChain, building chatbots with mem- ory has become relatively simple in the grand scheme of things as it even manages the API calls. The idea that one package was able to provide different modules to developing AI apps from Modell/O, Agents, Memory and Callbacks practically give you all the tools to develop a chatbots.

In regards to the questions given by the project scope, I believe question 1 to be executed and completed fully when the chatbot was able to accept a document and query, process and reply, but more testing is required to test whether question 2 and 3 were completed. Question 2 I realize was very similar to the failed remembrance due to mis-matching which I believe can be fixed with by utilizing LangChain's 'callback' module to act as a form of filter before proceeding with queries, 'inferencing' the logs.

For question 3, LangChain has a 'Memory' module too which persists application state between runs of a chain. I believe that using this in conjunction with 'Callback' would produce a method to localize references.

6 FUTURE WORK

In regards to future work, I believe one of the major flaws of this simple implementation would be the use of only one LLM. With LangChain's capability of 'chaining', we can actually use multiple LLMs to produce a more 'Utility' based chain of models to more specifically cater to research papers. For example, if we were to use two LLMs to query we can then use a LangChain package "AnalyzeDocumentChain" to further investigate and analyze the results from both LLMs to produce a higher quality answer.

7 CONCLUSION

In conclusion, I've developed a simple ChatPDF clone that works built upon a LangChain framework and GPT-3.5 as its LLM. Its not the most perfect version of a PDF chatbot with many possible upgrades to it such as the one mentioned in the Future Work section to incorporate more LLMs and implement chaining, but its a good start.

A system built on NLP techniques of word embeddings a vector database and LLM that accepts a question and a PDF document and returns a reply from and LLM.

REFERENCES

- [1] Felipe Almeida and Geraldo Xexéo. 2023. Word Embeddings: A Survey. arXiv:1901.09069 [cs.CL]
- [2] Xuanting Chen, Junjie Ye, Can Zu, Nuo Xu, Rui Zheng, Minlong Peng, Jie Zhou, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. How Robust is GPT-3.5 to Predecessors? A Comprehensive Study on Language Understanding Tasks. arXiv:2303.00293 [cs.CL]
- [3] Kien Nguyen-Trung, Alexander K Saeri, and Stefan Kaufman. 2023. Applying ChatGPT and AI-powered tools to accelerate evidence reviews. <https://doi.org/10.31219/osf.io/pcrqf>
- [4] Simon Ott, Konstantin Hebenstreit, Valentin Liévin, Christoffer Egeberg Hother, Milad Moradi, Maximilian Mayrhauser, Robert Praas, Ole Winther, and Matthias Samwald. 2023. ThoughtSource: A central hub for large language model reasoning data. arXiv:2301.11596 [cs.CL]

A FINAL PROJECT TEST QUESTIONS

```

(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py 2305.11206.pdf
please wait
Enter question:What is the hypothesis about alignment i
n this paper?
Tokens Used: 639
    Prompt Tokens: 608
    Completion Tokens: 31
Successful Requests: 1
Total Cost (USD): $0.012780000000000001
answer:
    The hypothesis is that alignment is largely about lear
ning style, and that it is possible to tune a pretraine
d language model with a rather small set of examples.
Enter question:|
  
```

Figure 9: "Question Type 1: Q1 of QT1"

```

Anaconda Prompt - python C x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py 2305.11206.pdf
please wait
Enter question:What is the experiment setup of this pap
er?
Tokens Used: 727
    Prompt Tokens: 600
    Completion Tokens: 127
Successful Requests: 1
Total Cost (USD): $0.01454
answer:
The experiment setup of this paper is to compare LIMA
to other models by generating a single response for eac
h test prompt and having crowdworkers or GPT-4 label wh
ich output they prefer. The authors also fine-tune a 7B
parameter LLAM model on various datasets, controlling
for the same hyperparameters. They then sample 5 respon
ses for each test set prompt and evaluate response qual
ity by asking ChatGPT (GPT-3.5 Turbo) to grade the help
fulness of a response on a 1-6 likert scale. In additio
n to manually authored examples, they sample 50 trainin
g examples from Super-Natural Instructions.
Enter question:|

```

Figure 10: "Question Type 1: Q2 of QT1"

```

Anaconda Prompt - python C x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py 2305.11206.pdf
please wait
Enter question:What is the main discovery of this paper
?
Tokens Used: 644
    Prompt Tokens: 602
    Completion Tokens: 42
Successful Requests: 1
Total Cost (USD): $0.01288
answer:
The main discovery of this paper is that Alpaca 65B an
d DaVinci003, despite being trained on 52 times more da
ta than LIMA, tend to produce less preferable output th
an LIMA.
Enter question:How to explain the phenomenon observed i
n this paper?
Tokens Used: 587
    Prompt Tokens: 535
    Completion Tokens: 52
Successful Requests: 1
Total Cost (USD): $0.01174
answer:
The paper observed that scaling up input diversity and
output quality have measurable positive effects on ali
gnment, while scaling up quantity alone might not. This
suggests that pretraining and its relative importance
over large-scale instruction tuning and reinforcement l
earning approaches is a key factor in alignment.
Enter question:|

```

Figure 11: "Question Type 2: Q1 and Q2 Sequential of QT2"

```

Anaconda Prompt - python C x + - □ x
(finalprojectNLP) C:\Users\Ian\Desktop\NLP\final proje>
python ChatPaper.py 2305.13048.pdf
please wait
Enter question:find the key reference for the following
paper
Tokens Used: 667
    Prompt Tokens: 648
    Completion Tokens: 19
Successful Requests: 1
Total Cost (USD): $0.013340000000000001
answer:
The key reference for this paper is ArXiv , abs/1709.0
4057.
Enter question:|

```

Figure 12: "Question Type 3: Q1 of QT3"