EvaDB Project Report:
Analysis from Integrating EvaDB into Embedchain

https://github.com/alexjaegook/EvaDBInEmbedchain/blob/main/EvaDbToEmbedchain.ipynb

Jaegook Alex Kim
Georgia Institute of Technology
Atlanta, Georgia, USA
alex.jg.kim@gatech.edu

INTRODUCTION

Embedchain is an open source project [1] that loads data from a given webpage, YouTube link, or local file and creates an embedding that is stored into a vector database. From this, users can interact with the *bot* that is created by Embedchain and ask the bot questions. The bot in return will answer questions relevant to the given dataset.

For this project, I created an *elon_bot* that is supposed to represent Elon Musk. *elon_bot* is trained through two datasets: 1) 26 minute YouTube video on Elon Musk, 2) Elon Musk's Wikipedia page.

To meet the requirements for this project, I will be using EvaDB to store the data and then interacting with the bot that is created by Embedchain. Embedchain has a user-friendly function `.add("FILE")`, that will automatically store the data that will be used for the bot. I will be using EvaDB's Question Answering use case as a substitute for reading and storing the files to interact with *elon_bot*.

I will be evaluating the performance of the EvaDB integrated *elon_bot* by comparing the answers with a bot created using Embedchain. I used the same two datasets for both bots, and asked the same questions. To evaluate correctness I fact checked using credible sources.

SOURCE CODE

For readers that wish to follow along, the source code is available on my github page [2]. The python notebook will have all of my implementations.

IMPLEMENTATION DETAILS

I decided to use two EvaDB use cases that were available on the EvaDB website [3]. The two uses are "Question Answering" and "Text Summarization". I decided to use these two cases because I believed they were the best in making the best comparisons in evaluating performance between EvaDB and Embedchain.

In order to have the bot use a YouTube video as a dataset, I referred to the google colab provided in the Question Answering use case page [4]. I installed EvaDB and set up OpenAI. To convert the mp4 file into text, I used Whisper, an open source speech recognition model to transcribe the YouTube video into text [5].

I chose the YouTube video "The Story of Elon Musk" uploaded by SunnyV2 [6] because it was a long enough video that contained all the relevant information about Elon Musk until 2020, the year that the video was published. I had attempted to use other videos, but realized they were either non-informative or too long.

For my second dataset, I decided to use Elon Musk's wikipedia page [7]. I wanted an article that was informative and long, but also credible. From my prior research, wikipedia pages that have high search results tend to have more accurate information as it is looked over many times.

To store the wikipedia page, I referred to the google colab provided in the Text Summarization use case page. I downloaded

Elon Musk's wikipedia page as a pdf file, and queried through the file with EvaDB that incorporated HuggingFace's text classification model DistilBERT-base-uncased [8].

With this setup, I was able to set up two tables with EvaDB to store the data of the YouTube video and the wikipedia page. I, then, used these tables to feed data into the *elon_bot* (as shown in Figure 1)

```
import os
from embedchain import App

# Create a bot instance
elon_bot = App()

# Add the tables to elon_bot
elon_bot.add(pdf_summarizer.to_string())
elon_bot.add(text_summarizer_udf_creation)
```

Figure 1: Creating elon_bot and adding the two EvaDB tables to the bot.

RESULTS/EVALUATION

I evaluated the performance of using EvaDB to the Elon Musk bot by comparing the output to a few queries that were also put into the Embedchain based Elon Musk bot. I will be referring to *elon_bot* as the bot that uses EvaDB and *embedchain_elon_bot* as the bot that uses on Embedchain.

To fairly assess and evaluate the two bots I used the two dataset aforementioned to train both bots, and assessed the results by asking the same question. To evaluate the correctness of the answers that were outputted by each of the bots, I fact checked the responses by referring to credible sources online.

For the first benchmark, I asked the following prompt: "When did Elon cofound PayPal?". As shown in Figure 2 and Figure 3, *elon_bot* and *embedchain_elon_bot* have similar answers. However, we observe that *elon_bot* is able to be more precise as an answer that PayPal was originally X.com, which later merged to form PayPal. This fact is validated by Britannica's biography on Elon Musk [9].

The second question I asked both bots is "How many companies did Elon Musk found?". As observed from Figure 4 and Figure 5, *elon_bot* has a more comprehensive response compared to *embedchain_elon_bot*. We observe that *elon_bot* is able to list an accurate timeline of each of the companies Elon Musk founded, and a snippet of history of companies there were acquired. Furthermore, we compare the two responses and see that *embedchain_elon_bot* is unable to list all the companies that Elon Musk founded. It confidently says that Elon Musk founded "a total of four companies" which is incorrect. As we see from biography.com, Elon has founded companies such as PayPal, Neuralink, and The Boring Company [10]. All of which have not been listed by the Embedchain supported bot. Compared to the first benchmark, we see that *elon_bot* is able to organize the data received by EvaDB more accurately.

The last benchmark I decided to ask was "Where did Elon go to school?". As shown in Figure 6 and Figure 7, we observe that EvaDB is able to answer correctly and precisely. Unfortunately, Wikipedia is the only source that has the information on Elon Musk claiming that he received his degree in 1995, while other sources simply state that Elon received his degree in 1997 [9]. However, it is true that Elon received his Bachelor's from University of Pennsylvania [9, 10]. *Embedchain_elon_bot* has a short and incorrect answer saying that Elon attended "Pennsylvania University".

CONCLUSION

After evaluating the two different bots based on EvaDB and Embedchain, we observe that EvaDB is able to perform better from the three questions asked. This is probably due to EvaDB storing the two datasets in a more organized way that makes it easier and more accurate to parse and retrieve the correct answer. We are able to

conclude this as the same datasets and same questions were given to both bots. The only variable was the database being used, and we clearly see that EvaDB outperforms Embedchain.

```
# Query the bot

chatgpt_udf = """
    SELECT ChatGPT('When did Elon cofound PayPal?',text)
    FROM TEXT_SUMMARY
"""

query_to_bot = cursor.query(chatgpt_udf).df().to_string(index=False)
print("-------------- EvaDB response --------------")
elon_bot.query(query_to_bot)
```

```
-------------- EvaDB response --------------
'True. Elon Musk co-founded X.com in 1999, which later merged with Confinity to form PayPal.'
```

Figure 2: EvaDB integrated bot response to “When did Elon cofound PayPal?”

```
[83] print("-------------- Embedchain response --------------")
     embedchain_elon_bot.query("When did Elon cofound PayPal?")
```

```
-------------- Embedchain response --------------
'Elon Musk co-founded PayPal in 1999.'
```

Figure 3: Embedchain integrated bot response to “When did Elon cofound PayPal?”

```
[78] print("-------------- EvaDB response --------------")
     chatgpt_udf2 = """
         SELECT ChatGPT('How many companies did Elon Musk found?',text)
         FROM TEXT_SUMMARY;
     """

     query_to_bot2 = cursor.query(chatgpt_udf2).df().to_string(index=False)

     elon_bot.query(query_to_bot2)
```

```
-------------- EvaDB response --------------
'Some of the notable companies founded or co-founded by Elon Musk include:\n\n1. Zip2 Corporation: Founded in 1995, it provided busin
ess directories and maps for newspapers. It was later sold to Compaq for $307 million.\n\n2. X.com: Founded in 1999, it was an online
payment company that eventually became PayPal. PayPal was later acquired by eBay.\n\n3. SpaceX (Space Exploration Technologies Cor
p.): Founded in 2002, it is a private aerospace manufacturer and space transportation company.\n\n4. Tesla, Inc.: Elon Musk joined Te
sla in 2004 and became its CEO in 2008. Tesla is an electric vehicle and clean energy company.\n\n5. SolarCity (now part of Tesla): E
lon Musk played a significant role in the founding and growth of SolarCity, a solar energy services company. It was acquired by Tesla
in 2016.\n\n6. Neuralink: Founded in 2016, it is a neurotechnology company focused on developing implantable brain-machine interface
s.\n\n7. The Boring Company: Founded in 2016, it aims to construct tunnels for transportation systems to alleviate traffic congestio
n.\n\nThese are some of the major companies founded or co-founded by Elon Musk.<'
```

Figure 4: EvaDB integrated bot response to “How many companies did Elon Musk found?”

```
print("-------------- Embedchain response --------------")
embedchain_elon_bot.query("How many companies did Elon Musk found?")
```

```
-------------- Embedchain response --------------
'Elon Musk founded a total of four companies: SpaceX, Tesla (formerly Tesla Motors), SolarCity (now Tesla Energy), and OpenAI.'
```

Figure 5: Embedchain integrated bot response to “How many companies did Elon Musk found?”

```
print("-------------- EvaDB response --------------")
chatgpt_udf3 = """
    SELECT ChatGPT('Where did Elon go to school?',text)
    FROM TEXT_SUMMARY;
"""

query_to_bot3 = cursor.query(chatgpt_udf3).df().to_string(index=False)

elon_bot.query(query_to_bot3)
```

```
-------------- EvaDB response --------------
'Elon Musk attended the University of Pennsylvania (UPenn) and completed studies for a Bachelor of Arts degree in physics and a Bache
lor of Science degree in economics from the Wharton School. However, there is a discrepancy regarding the year he earned these degree
s. Musk claims he earned them in 1995, but UPenn maintains it awarded them in 1997.'
```

Figure 6: EvaDB integrated bot response to "Where did Elon go to school?"

```
print("-------------- Embedchain response --------------")
embedchain_elon_bot.query("Where did Elon go to school?")
```

```
-------------- Embedchain response --------------
'Elon Musk attended Pennsylvania University.'
```

Figure 7: Embedchain integrated bot response to "Where did Elon go to school?"

METRICS TIME/TOKENS/MONEY

From the python notebook time stamps, we can see the execution time for each of the queries for each of the questions. The execution times for the first question "when did Elon cofound PayPal?" for *elon_bot* and *embedchain_elon_bot* were 4 seconds and 2 seconds, respectively. The execution times for the second question "how many companies did Elon Musk found?" were 46 seconds and 4 seconds, respectively. The long execution time is probably due to the comprehensive list that EvaDB had to parse through to get an accurate answer. Lastly, the execution times for the third question "where did Elon go to school?" were 12 seconds and 2 seconds. There seems to be a correlation with the length of the answer and the time it takes to execute for *elon_bot*. On the other hand, it seems like *embedchain_elon_bot* consistently performs the questions in very short timeframes. This as a tradeoff results in less accurate and even incorrect answers.

I paid $5 to use the OpenAI API because it was not letting me run any of the queries when I tried using the free version. This may have been because calls I was trying to make were above the limit for the free version. I used a total of $0.18 to complete the project.

LESSONS LEARNED

This project helped me learn a variety of different libraries and integrate open source projects. I had prior experience in natural language processing, but this project helped me learn about other language models such as Whisper and DistilBERT that were available publicly by HuggingFace.

I also learnt how cool it was using databases into machine learning. It was really fun learning about how we can store data as tables and have the language models query through them to output more accurate answers when stored well (as seen in *elon_bot*). Reading the documentations of EvaDB and seeing the sample python notebooks really helped in accelerating my learning and trying to figure out what EvaDB was and how I could take advantage of it for this project.

I learnt how to use OpenAI and how we could use the API key to very easily use ChatGPT and

create a Q&A system. I had always feared using OpenAI's API because from the outside it seemed really intimidating trying to use ChatGPT into a project. However, once I started to read the documentations and the code, it was really straightforward and I am glad that I took on the challenge using OpenAI's API because I feel like I could use this newly learnt skill in my future projects.

This project had a very high learning curve, but it was very rewarding in the end seeing how I could use OpenAI's API to create an actual project. Overall, it was a great experience and just challenging enough to not get stuck for too long and see meaningful results.

CHALLENGES FACED

The biggest challenge was trying to initially find a project. I had originally planned on doing a project using Chat2DB [11] and integrating it with EvaDB to create a tool that wrote MySQL queries with audio files. However, I soon found out that the entire repository was mainly written in Chinese, making it near impossible to go through the repository and try and figure out how to integrate EvaDB into the project. This language barrier was impossible to overcome and I had to change my project last minute and think of a new idea.

I was approved of the idea of using Embedchain and comparing the bots created by EvaDB and Embedchain last minute by Dr. Joy Arulraj. Implementing the two together was difficult. I initially tried using Google Drive and YouTube links to try and upload the dataset into EvaDB. However, I was faced with errors that it could not authorize or recognize the file when it was locally downloaded on python notebook. I was able to resolve this by using DropBox links, similar to the python notebooks that showed the tutorials on how to use EvaDB.

One of the most challenging debugs was trying to add the pdf file into the *elon_bot*. I had initially tried using the function that was provided in the EvaDB tutorial, however, I was unable to retrieve the correct structure that was needed for Embedchains to create the bot with EvaDB's data frame. I experimented with different ways to utilize the function. I was unable to find a way to integrate the function. The solution I came up with was to feed the entire table into the bot. This may have been the reason why the execution times were very long as the wikipedia page table was large. This is something that I could possibly optimize in the future.

I was luckily able to finish this project because I had already looked over EvaDB's documentation prior to starting the project and had a good understanding of how to integrate any third-party into EvaDB. Overall, the project was challenging, but I was able to successfully overcome these challenges using online resources and referring to documentations.

REFERENCES

[1]https://github.com/embedchain/embedchain/blob/main/README.md

[2]https://github.com/alexjaegook/EvaDBInEmbedchain

[3]https://evadb.readthedocs.io/en/stable/source/overview/getting-started.html

[4]https://evadb.readthedocs.io/en/latest/source/usecases/question-answering.html

[5]https://huggingface.co/spaces/openai/whisper

[6]https://www.youtube.com/watch?v=vai3Vh234EE&t=322s

[7]https://en.wikipedia.org/wiki/Elon_Musk

[8]https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english

[9]https://www.britannica.com/biography/Elon-Musk

[10]https://www.biography.com/business-leaders/elon-musk

[11]https://github.com/chat2db/Chat2D