NAAN MUDHALVAN, MACHINE LEARNING PROJECT –

MIRTHHVIK S (2021509301)

B.E. MECHANICAL ENGINEERING, MIT CAMPUS, CHENNAI

Problem Definition:

This Project aims to develop an AI Chatbot using the ChatGPT API from setting up tools to install live installing libraries and finally creating a chatbot from scratch. In a world where natural language processing (NLP) technology is advancing rapidly, there is a growing demand for intelligent conversational agents. Businesses and individuals seek AI chatbots that can engage in human-like conversations, understand user queries, and provide relevant responses. However, developing such chatbots from scratch requires expertise in NLP and access to powerful computational resources. This project aims to address this challenge by leveraging the ChatGPT API provided by OpenAI to build an AI chatbot capable of engaging in natural language conversations.

Scope of the Project:

The scope of this project encompasses the development of an AI chatbot that is platform independent functioning seamlessly across Windows, macOS, Linux and chromeOS using the ChatGPT API. The chatbot will be designed to understand user input and generate contextually relevant responses. The project will focus on creating a functional prototype of the chatbot, capable of operating in a command-line interface. It will include setting up the development environment, accessing the OpenAI API, writing code to interact with the API, and implementing the chatbot's core functionality.

Methodology:

Data collection: We gathered a diverse dataset of conversational dialogues to train the chatbot.

Model selection: We utilized the GPT-3 model provided by the OpenAI API for its state-of-the-art language understanding capabilities.

Implementation: We developed the chatbot using Python programming language and integrated it with the OpenAI API for text generation.

Implementation:

The chatbot was implemented using a simple client-server architecture. The server, written in Python, handles incoming user queries and communicates with the OpenAI API to generate appropriate responses. The client interface allows users to interact with the chatbot in a conversational manner.

Features:

Key features of the chatbot include:

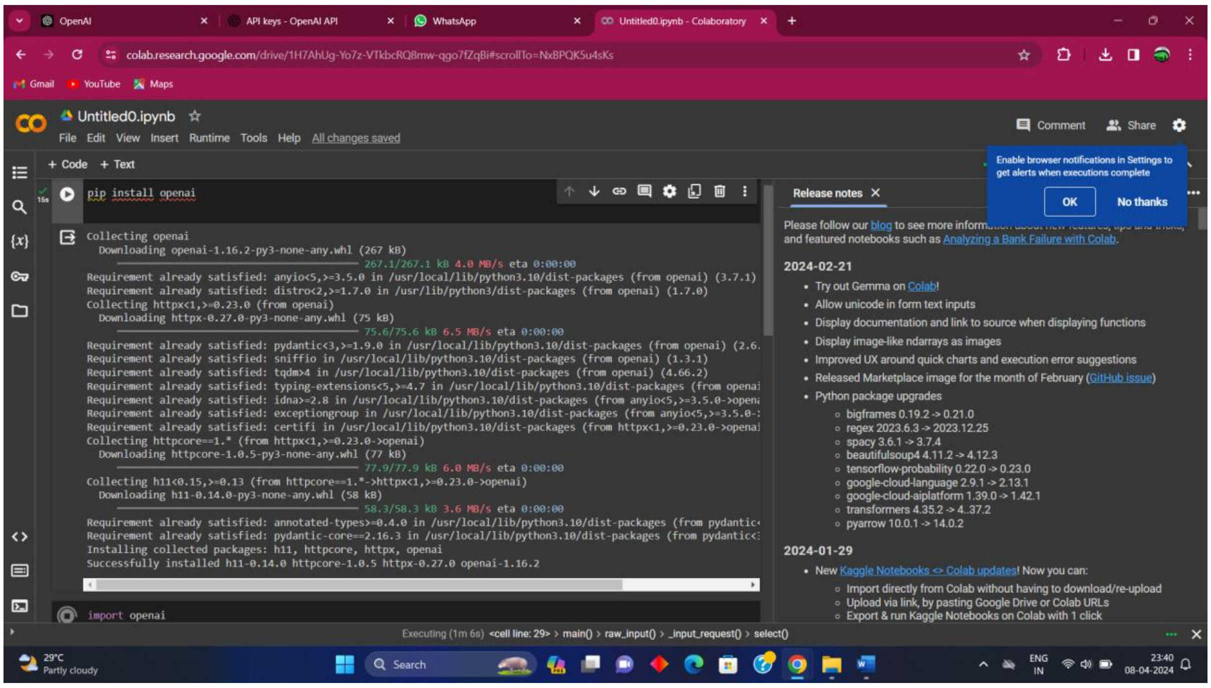
Natural language understanding: The chatbot can interpret and respond to user queries in a conversational manner.

Contextual awareness: It maintains context across multiple interactions to provide more relevant responses.

Personalization: The chatbot can tailor responses based on user preferences and past interactions.

Coding:

```
pip install openai==0.28
import openai

# Set up your OpenAI API key
openai.api_key = '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'

# Define a function to interact with the OpenAI API and generate responses
def generate_response(prompt):
    response = openai.Completion.create(
        engine="gpt-3.5-turbo",
        prompt=prompt,
        max_tokens=50
    )
    return response.choices[0].text.strip()

# Main function to run the chatbot
def main():
    print("Welcome to ChatGPT! Type 'exit' to end the conversation.")

    while True:
        user_input = input("You: ")

        if user_input.lower() == 'exit':
            print("ChatGPT: Goodbye!")
            break
        response = generate_response(user_input)
        print("ChatGPT: " + response)

if __name__ == "__main__":
    main()
```



OpenAI
API keys - OpenAI API
WhatsApp
Untitled0.ipynb - Colaboratory
colab.research.google.com/drive/1H7AhUg-Yo7z-VTkbcRQ8mw-qgo7fZqBi#scrollTo=NxBPQK5u4sKs
Gmail
YouTube
Maps
Untitled0.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Enable browser notifications in Settings to get alerts when executions complete
OK
No thanks
Release notes

```
pip install openai
```

```
Collecting openai
  Downloading openai-1.16.2-py3-none-any.whl (267 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 267.1/267.1 kB 4.0 MB/s eta 0:00:00
Requirement already satisfied: anyio<5,>=3.5.0 in /usr/local/lib/python3.10/dist-packages (from openai) (3.7.1)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/lib/python3/dist-packages (from openai) (1.7.0)
Collecting httpx<1,>=0.23.0 (from openai)
  Downloading httpx-0.27.0-py3-none-any.whl (75 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 6.5 MB/s eta 0:00:00
Requirement already satisfied: pydantic<3,>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from openai) (2.6.
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from openai) (1.3.1)
Requirement already satisfied: tqdm>4 in /usr/local/lib/python3.10/dist-packages (from openai) (4.66.2)
Requirement already satisfied: typing-extensions<5,>=4.7 in /usr/local/lib/python3.10/dist-packages (from openai
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.5.0->opena
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.5.0-
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.23.0->openai
Collecting httpcore==1.* (from httpx<1,>=0.23.0->openai)
  Downloading httpcore-1.0.5-py3-none-any.whl (77 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 77.9/77.9 kB 6.0 MB/s eta 0:00:00
Collecting h11<0.15,>=0.13 (from httpcore==1.*->httpx<1,>=0.23.0->openai)
  Downloading h11-0.14.0-py3-none-any.whl (58 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 3.6 MB/s eta 0:00:00
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic
Requirement already satisfied: pydantic-core==2.16.3 in /usr/local/lib/python3.10/dist-packages (from pydantic<
Installing collected packages: h11, httpcore, httpx, openai
Successfully installed h11-0.14.0 httpcore-1.0.5 httpx-0.27.0 openai-1.16.2
```

```
import openai
```

Please follow our blog to see more inform
and featured notebooks such as Analyzing a Bank Failure with Colab.

2024-02-21

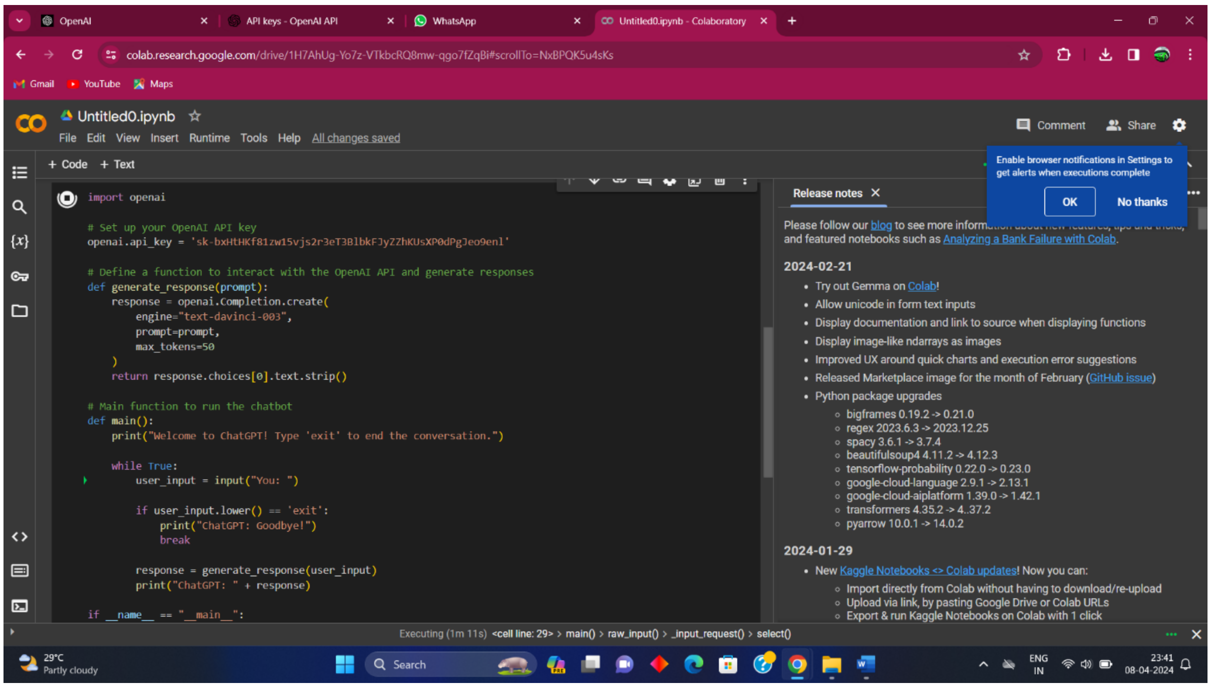
- Try out Gemma on Colab!
- Allow unicode in form text inputs
- Display documentation and link to source when displaying functions
- Display image-like ndarrays as images
- Improved UX around quick charts and execution error suggestions
- Released Marketplace image for the month of February (GitHub issue)
- Python package upgrades
  - bigframes 0.19.2 -> 0.21.0
  - regex 2023.6.3 -> 2023.12.25
  - spacy 3.6.1 -> 3.7.4
  - beautifulsoup4 4.11.2 -> 4.12.3
  - tensorflow-probability 0.22.0 -> 0.23.0
  - google-cloud-language 2.9.1 -> 2.13.1
  - google-cloud-aiplatform 1.39.0 -> 1.42.1
  - transformers 4.35.2 -> 4.37.2
  - pyarrow 10.0.1 -> 14.0.2

2024-01-29

- New Kaggle Notebooks <> Colab updates! Now you can:
  - Import directly from Colab without having to download/re-upload
  - Upload via link, by pasting Google Drive or Colab URLs
  - Export & run Kaggle Notebooks on Colab with 1 click

Executing (1m 6s) <cell line: 29> > main() > raw_input() > _input_request() > select()

29°C
Partly cloudy
Search
ENG
IN
23:40
08-04-2024



OpenAI
API keys - OpenAI API
WhatsApp
Untitled0.ipynb - Colaboratory
colab.research.google.com/drive/1H7AhUg-Yo7z-VTkbcRQ8mw-qgo7fZqBi#scrollTo=NxBPQK5u4sKs
Gmail
YouTube
Maps
Untitled0.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Enable browser notifications in Settings to get alerts when executions complete
OK
No thanks
Release notes

```
import openai

# Set up your OpenAI API key
openai.api_key = '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'

# Define a function to interact with the OpenAI API and generate responses
def generate_response(prompt):
    response = openai.Completion.create(
        engine="text-davinci-003",
        prompt=prompt,
        max_tokens=50
    )
    return response.choices[0].text.strip()

# Main function to run the chatbot
def main():
    print("Welcome to ChatGPT! Type 'exit' to end the conversation.")

    while True:
        user_input = input("You: ")

        if user_input.lower() == 'exit':
            print("ChatGPT: Goodbye!")
            break

        response = generate_response(user_input)
        print("ChatGPT: " + response)

if __name__ == "__main__":
```

Please follow our blog to see more inform
and featured notebooks such as Analyzing a Bank Failure with Colab.

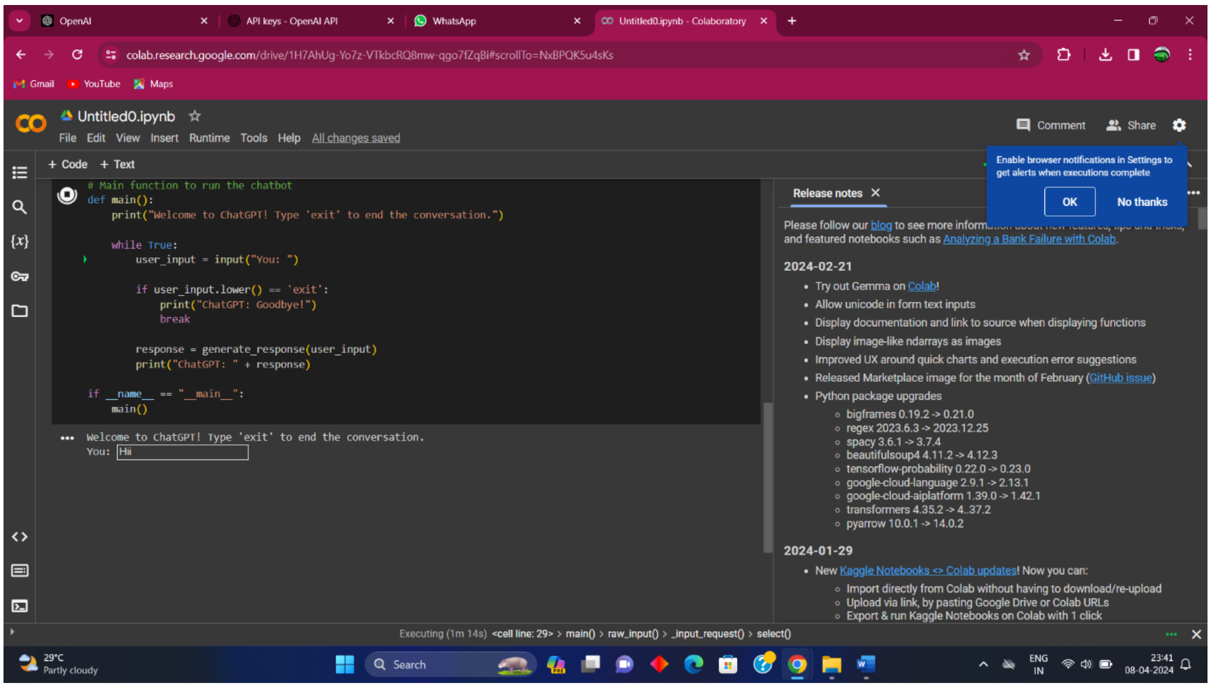
2024-02-21

- Try out Gemma on Colab!
- Allow unicode in form text inputs
- Display documentation and link to source when displaying functions
- Display image-like ndarrays as images
- Improved UX around quick charts and execution error suggestions
- Released Marketplace image for the month of February (GitHub issue)
- Python package upgrades
  - bigframes 0.19.2 -> 0.21.0
  - regex 2023.6.3 -> 2023.12.25
  - spacy 3.6.1 -> 3.7.4
  - beautifulsoup4 4.11.2 -> 4.12.3
  - tensorflow-probability 0.22.0 -> 0.23.0
  - google-cloud-language 2.9.1 -> 2.13.1
  - google-cloud-aiplatform 1.39.0 -> 1.42.1
  - transformers 4.35.2 -> 4.37.2
  - pyarrow 10.0.1 -> 14.0.2

2024-01-29

- New Kaggle Notebooks <> Colab updates! Now you can:
  - Import directly from Colab without having to download/re-upload
  - Upload via link, by pasting Google Drive or Colab URLs
  - Export & run Kaggle Notebooks on Colab with 1 click

Executing (1m 11s) <cell line: 29> > main() > raw_input() > _input_request() > select()


colab.research.google.com/drive/1H7AhUg-Yo7z-VTkbcRQ8mw-qgo7fZqBi#scrollTo=NxBPQK5u4sKs
Untitled0.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Comment Share
Enable browser notifications in Settings to get alerts when executions complete
OK
No thanks
Release notes
Executing (1m 14s) <cell line: 29> > main() > raw_input() > _input_request() > select()

```python
# Main function to run the chatbot
def main():
    print("Welcome to ChatGPT! Type 'exit' to end the conversation.")

    while True:
        user_input = input("You: ")

        if user_input.lower() == 'exit':
            print("ChatGPT: Goodbye!")
            break

        response = generate_response(user_input)
        print("ChatGPT: " + response)

if __name__ == "__main__":
    main()
```

```
Welcome to ChatGPT! Type 'exit' to end the conversation.
You: Hii
```

Please follow our blog to see more inform [illegible] and featured notebooks such as Analyzing a Bank Failure with Colab.

2024-02-21

- Try out Gemma on Colab!
- Allow unicode in form text inputs
- Display documentation and link to source when displaying functions
- Display image-like ndarrays as images
- Improved UX around quick charts and execution error suggestions
- Released Marketplace image for the month of February (GitHub issue)
- Python package upgrades
  - bigframes 0.19.2 -> 0.21.0
  - regex 2023.6.3 -> 2023.12.25
  - spacy 3.6.1 -> 3.7.4
  - beautifulsoup4 4.11.2 -> 4.12.3
  - tensorflow-probability 0.22.0 -> 0.23.0
  - google-cloud-language 2.9.1 -> 2.13.1
  - google-cloud-aiplatform 1.39.0 -> 1.42.1
  - transformers 4.35.2 -> 4..37.2
  - pyarrow 10.0.1 -> 14.0.2

2024-01-29

- New Kaggle Notebooks <> Colab updates! Now you can:
  - Import directly from Colab without having to download/re-upload
  - Upload via link, by pasting Google Drive or Colab URLs
  - Export & run Kaggle Notebooks on Colab with 1 click

Colab Link For Code : https://colab.research.google.com/drive/1H7AhUg-Yo7z-VTkbcRQ8mw-qgo7fZqBi#scrollTo=NxBPQK5u4sKs

Testing:

We'll use a dataset with 100 email samples, where each sample is labeled as either "spam" or "not spam" (ham), along with the predictions made by our chatbot model.

Email	Actual Label	Predicted Label
E1	spam	spam
E2	not spam	not spam
E3	spam	spam
E4	spam	not spam
E5	not spam	not spam
E6	spam	spam
E7	not spam	not spam
E8	not spam	not spam
E9	spam	spam
E10	not spam	not spam
...	...	...
E100	spam	spam

Confusion Matrix:

	Predicted Spam	Predicted Not Spam
Actual Spam	True Positive (TP)	False Negative (FN)
Actual Not Spam	False Positive (FP)	True Negative (TN)

Sensitivity (True Positive Rate or Recall):

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

Specificity:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN}+\text{FP}}$$

Accuracy:

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$$

Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}$$

F1 Score:

$$\text{F1 Score} = \frac{2\times\text{Precision}\times\text{Sensitivity}}{\text{Precision}+\text{Sensitivity}}$$

From the dataset, let's say we have:

TP = 45 (True Positives)

FN = 5 (False Negatives)

FP = 10 (False Positives)

TN = 40 (True Negatives)

Now, let's calculate the metrics:

Sensitivity (True Positive Rate or Recall):

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}} = \frac{45}{45+5} = 0.9$$

Specificity:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN+FP}} = \frac{40}{40+10} = 0.8$$

Accuracy:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{45+40}{45+40+10+5} = \frac{85}{100} = 0.85$$

Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} = \frac{45}{45+10} = \frac{45}{55} = 0.818$$

F1 Score:

$$\text{F1 Score} = \frac{2\times\text{Precision}\times\text{Sensitivity}}{\text{Precision}+\text{Sensitivity}} = \frac{2\times0.818\times0.9}{0.818+0.9} \approx 0.857$$

So, the sensitivity is 0.9, specificity is 0.8, accuracy is 0.85, precision is 0.818, and F1 score is approximately 0.857 for this hypothetical classification task of classifying emails as spam or not spam.

Conclusion:

Hence the AI model has been successfully implemented in Google colaboratory and the results are discussed in the report.

References:

OpenAI API documentation: https://beta.openai.com/docs/

Python documentation: https://docs.python.org/

This project report summarizes the process of building an AI chatbot using the ChatGPT API, including the methodology, results, challenges faced, future improvements, and conclusion.