A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

Generative AI - > Use Ai to generate content/something. To generate text using chatgpt, to generate images using Dall-E, Generate code – Github Copilot, audio, video etc

Note: All this content are getting generated newly and not extracted and displayed from some DB.

**Key Backgrounds in Gen AI{**

1. **Artificial Intelligence:**

It focuses on creating intelligence system capable of performing task that require human intelligence. For eg,. Detect credit card debts. Incorporating human like intelligence into machines.

1. **Machine Learning:**

To incorporate human like intelligence into machine, we need the help of machine learning. It is a subset of AI, which focuses on development of algorithms and models that enable computers to learn and make predictions or decisions without explicit programming. Train machines 🡪 it learns something 🡪 they can make decision or prediction on their own.

It requires

1. lots of training data like different color, size, image etc.
2. Computation power
3. Algorithms
4. **Deep Learning:**

Is a form of machine learning. It is based on the concept of Neurons. It is subset of Machine learning, that focuses on teaching computers and make decisions by processing data through neural networks inspired by the human brain.

A diagram of a human brain

Description automatically generated

A diagram of machine learning

Description automatically generated

**Conventional AI:**

Where we train the machine with tons of data and based on that, the machine will learn and identify the given object. It can also make predictions, classification, NLM, Computer vision etc.,

A cartoon robot next to an arrow

Description automatically generated

A diagram of a computer system

Description automatically generated

Now with generative AI, the machine will be taught with tons and tons of test data and based on that it will generate image, text, audio, video etc., In the below example, first we teach the machine to learn what an apple looks like but then when we asked to generate it creates a new image which is not from the test data supply but created on its own based on learning.

A cartoon robot with an arrow

Description automatically generated

A close-up of a white background

Description automatically generated

Contextual understanding – it is nothing but when I ask a question to chatgpt about what is the capital of tamil nadu? Followed by what is the temperature there? Then it should be intelligent enough to think that I am still talking about Chennai without explicitly mentioning the name in the subsequent question.

A screenshot of a computer

Description automatically generated

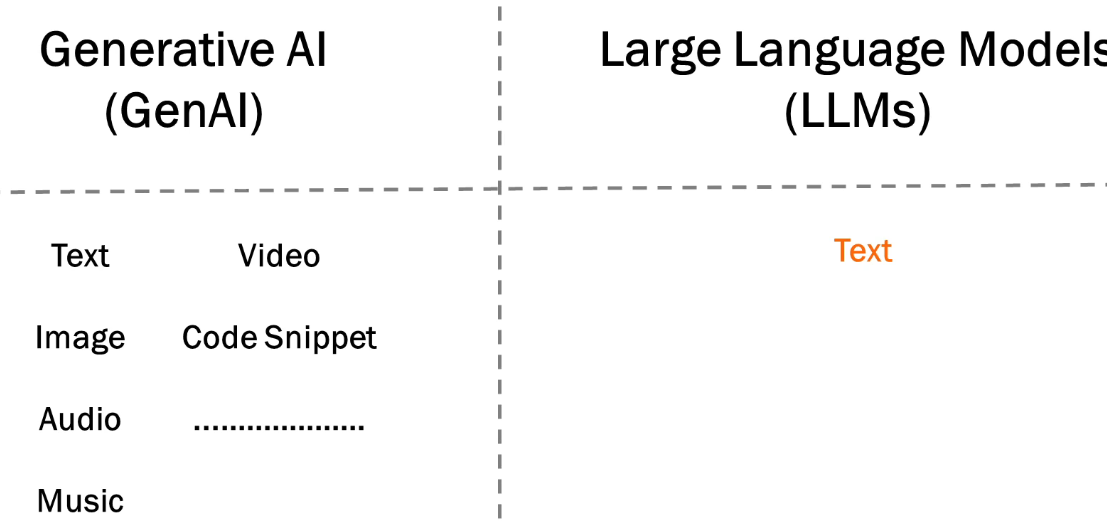
**Key Terminologies:**

LLM – Large Language Model

In simplest form it is nothing but text. They can understand the text, they can create new words, sentences etc., It is designed for understanding and generate human like text.

Eg., Auto suggesting words while typing in whatsapp, text that gets typed in chatgpt

**Difference between Gen AI and LLM**



LLM uses Neural network called Transformers.

A white background with black text

Description automatically generated

**Prompt Engineering:**

A white background with orange text

Description automatically generated with medium confidence

**Embeddings:**

A close up of text

Description automatically generated

**Fine Tuning:**

Training a pre-trained model on a specific domain to get the better result.

How to fine tune

1. Self-Supervised – give your foundation model a big pile of data that is specific to your domain.
2. Supervised – labelled training data i.e., in input and output model. For eg., input - how to find the broken bones, output – use X-Ray.
3. Reinforcement – feedback-based learning.

A diagram of a diagram of a server

Description automatically generated

ChatGPT answers the user query based on the way it got trained. It will give you the result only for the query on which it was trained i.e., documents that are publicly available for the end user. If you ask what the revenue of Apple is it can answer on the other hand if you ask what the salary of your company CEO is then it won’t unless it is available publicly.

So, in our case we are going to create a chatbot using python. First, we are going to make it trained on a document that we create as shown in the architecture, it has been converted into smaller chunk of data and then it got converted into embedding (number format of given text) using openAI API. The result will be store in vector store DB. So, whenever a user asks a query, it will do a semantic search (which will find the closest match for the given text in numbers) then it will be ranked ascendingly. The closest match will then be handled by LLM and displayed as a text to the end user.

Things we need to create our own chatbot.

1. Pycharm IDE to program in python
2. OpenAI API key

**How to get OpenAI key:**

Google openai api 🡪 signup 🡪 access platform.openai.com/api-keys 🡪 APIKeys on the left pane

Install the following libraries in pycharm after creating a new project.

1. Pip install streamlit pypdf2 langchain faiss-cpu

Where streamlit – utility to create UI interface.

Pypdf2 – to read pdf.

Langchain – an opensource interface to use openai

Faiss-cpu – vector store to store embeddings.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated



C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\langchain\vectorstores\\_\_init\_\_.py:35: LangChainDeprecationWarning: Importing vector stores from langchain is deprecated. Importing from langchain will no longer be supported as of langchain==0.2.0. Please import from langchain-community instead:`from langchain\_community.vectorstores import FAISS`.

To install langchain-community run `pip install -U langchain-community`.

warnings.warn(

C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\langchain\_core\\_api\deprecation.py:119: LangChainDeprecation

Warning: The class `OpenAIEmbeddings` was deprecated in LangChain 0.0.9 and will be removed in 0.2.0. An updated version of the class exists in the langchain-openai package and should be used instead. To use it run `pip install -U langchain-openai` and import as `from langchain\_openai import OpenAIEmbeddings`.

warn\_deprecated(

2024-04-30 19:11:58.360 Uncaught app exception

Traceback (most recent call last):

File "C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\langchain\_community\embeddings\openai.py", line 327,

in validate\_environment

import openai

ModuleNotFoundError: No module named 'openai'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

File "C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\streamlit\runtime\scriptrunner\script\_runner.py", li

ne 584, in \_run\_script

exec(code, module.\_\_dict\_\_)

File "C:\Users\Ragavendran.v\PycharmProjects\GenAIChatBot\main.py", line 37, in <module>

embeddings = OpenAIEmbeddings(openai\_api\_key = OPENAI\_API\_KEY)

File "C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\langchain\_core\\_api\deprecation.py", line 183, in wa

rn\_if\_direct\_instance

return wrapped(self, \*args, \*\*kwargs)

File "C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\pydantic\v1\main.py", line 339, in \_\_init\_\_

values, fields\_set, validation\_error = validate\_model(\_\_pydantic\_self\_\_.\_\_class\_\_, data)

File "C:\Users\Ragavendran.v\AppData\Roaming\Python\Python38\site-packages\pydantic\v1\main.py", line 1100, in validate\_model

ImportError: Could not import openai python package. Please install it with `pip install openai`.

Govt – 80k – mgr medical university

1.5 – mgr unvi – 10, 12, tc, community certificate, photo - 4,