Assignment NO 11
Programming for Artificial Intelligence
(PAI)

Terms to Explain

- Lang-Chain
- RAG
- LLMs
- FAISS
- Vector
- VectorDB
- Generative AI
- GANs

LangChain

The LLM App Builder

LangChain is a Python (and JS) framework that helps developers build applications powered by LLMs, especially when they need to:

- Access external documents
- Interact with tools/APIs
- Maintain memory of a conversation
- Combine multiple steps or models together

Example Use Case:

You're building a chatbot for a company. LangChain can:

Let the bot answer using internal company documents. Call APIs to fetch real-time data (e.g. weather or order status). Keep track of what the user said 5 questions ago. Lang Chain helps organize this logic using chains, agents, and retrievers.

Major Components:

Using LangChain, software teams can build context-aware language model systems with the following modules.

LLM interface

LangChain provides APIs with which developers can connect and query LLMs from their code. Developers can interface with public and proprietary models like GPT

Prompt templates

Prompt templates are pre-built structures developers use to consistently and precisely format queries for AI models. Developers can create a prompt template for chatbot applications.

Agents

Developers use tools and libraries that LangChain provides to compose and customize existing chains for complex applications. An *agent* is a special chain that prompts the language model to decide the best sequence in response to a query.

Retrieval modules

LangChain enables the architecting of RAG systems with numerous tools to transform, store, search, and retrieve information that refine language model responses.

Memory

Some conversational language model applications refine their responses with information recalled from past interactions. LangChain allows developers to include memory capabilities in their systems. It supports:

- Simple memory systems.
- Complex memory structures

Callbacks

Callbacks are codes that developers place in their applications to log, monitor, and stream specific events in LangChain operations.

RAG (Retrieval-Augmented Generation)

LLM + Knowledge

Retrieval-Augmented Generation (RAG) is the process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response. Large Language Models (LLMs) are trained on vast volumes of data and use billions of parameters to generate original output for tasks like answering questions, translating languages, and completing sentences. RAG extends the already powerful capabilities of LLMs to specific domains or an organization's internal knowledge base, all without the need to retrain the model.

How works:

- User Query: "What are the benefits of Vitamin D?"
- Retriever (e.g., FAISS) finds relevant documents from a VectorDB.
- The LLM sees: “User asked: [...]. Here’s some context: [...]”
- The LLM then generates a grounded response.
- Use Case: Chatbots that answer from PDFs, textbooks, or private company data.

Importance:

LLMs are a key artificial intelligence (AI) technology powering intelligent chatbots and other natural language processing (NLP) applications. The goal is to create bots that can answer user questions in various contexts by cross-referencing authoritative knowledge sources. Unfortunately, the nature of LLM technology introduces unpredictability in LLM responses. Additionally, LLM training data is static and introduces a cut-off date on the knowledge it has.

LLMs (Large Language Models)

The Brains

LLMs like GPT-4, Claude, or LLaMA are neural networks trained on billions of sentences from the internet and books. They learn patterns of language (syntax, grammar, reasoning). They don't "know" in the human sense, but predict next words based on the input.

Capabilities:

- Summarize an article
- Answer questions
- Write stories
- Translate languages
- Generate code

LLMs power most Generative AI applications involving text.

A large language model (LLM) is a type of machine learning model designed for natural language processing tasks such as language generation. LLMs are language models with many parameters, and are trained with self-supervised learning on a vast amount of text.

The largest and most capable LLMs are generative pretrained transformers (GPTs). Modern models can be fine-tuned for specific tasks or guided by prompt engineering. These models acquire predictive power regarding syntax, semantics, and ontologies inherent in human language corpora, but they also inherit inaccuracies and biases present in the data they are trained in.

FAISS (Facebook AI Similarity Search)

Find Similar Vectors Fast

Faiss, short for Facebook AI Similarity Search, is an open-source library built for similarity search and clustering of dense vectors. Faiss can be used to build an index and perform searches with remarkable speed and memory efficiency. Additionally, it enhances search performance through its GPU implementations for various indexing methods. So generally FAISS is a library that helps you search large sets of vectors to find which ones are most similar.

Why this matters:

- Words/sentences are converted to vectors

- To find “relevant documents” for a query like "climate change," you search vector space for closeness.
- FAISS is fast and scalable, which makes it ideal for retrieval in RAG systems.

Vector

Meaning in Numbers

A vector is just a list of numbers that represents meaning. A vector in a simple term can be considered as a single-dimensional array. With respect to Python, a vector is a one-dimensional array of lists. It occupies the elements in a similar manner as that of a Python list.

For example:

- Sentence: “The cat is sleeping.”
- Vector: [0.23, -0.91, 0.04, ...] (hundreds of dimensions)

These numbers are created using embeddings models (like MiniLM, BERT, or OpenAI's embedding models).

Why useful:

- Two similar sentences have similar vectors.
- It allows semantic search, where we match based on meaning, not keywords.

VectorDB (Vector Database)

Store & Search Meaning

A vector database is any database that allows you to store, index, and query vector embeddings, or numerical representations of unstructured data, such as text, images, or audio.

Vector embeddings

Vector embeddings are useful representations of unstructured data because they map content in such a way that semantic similarity is represented by distance in n-dimensional vector space. This makes it easy to search for similarity, find relevant content in a knowledge base, or retrieve an item that best matches a complex user-generated query.

While some specialized databases only support vector embeddings, others support many other data and query types in addition to vector embeddings. Support for a wide range of data types and

query types is critical for building generative AI applications on top of rich, real-world data. As the benefits of semantic query using vector embeddings becomes clear, most databases will add vector support. In the future, we believe that every database will be a vector database. In simple words A Vector Database stores these vectors along with metadata (e.g., the source text, tags).

Examples:

Pinecone, Weaviate, Chroma, Qdrant

Generative AI

AI That Creates

Generative AI is the umbrella term for AI that creates new content. Generative artificial intelligence, also known as generative AI or gen AI for short, is a type of AI that can create new content and ideas, including conversations, stories, images, videos, and music. It can learn human language, programming languages, art, chemistry, biology, or any complex subject matter. It reuses what it knows to solve new problems.

For example:

it can learn English vocabulary and create a poem from the words it processes.

Your organization can use generative AI for various purposes, like chatbots, media creation, product development, and design.

Includes:

- Text (ChatGPT, Claude)
- Images (DALL·E, Midjourney, Stable Diffusion)
- Video (RunwayML)
- Music (AIVA, Suno)
- Code (Copilot)
- Powered by models like:
- LLMs (for text/code)
- GANs or Diffusion Models (for images/videos)

GANs (Generative Adversarial Networks)

Realistic Image Generation

A generative adversarial network (GAN) is a deep learning architecture. It trains two neural networks to compete against each other to generate more authentic new data from a given training dataset. For instance, you can generate new images from an existing image database or original music from a database of songs. A GAN is called adversarial because it trains two different networks and pits them against each other. One network generates new data by taking an input data sample and modifying it as much as possible. The other network tries to predict whether the generated data output belongs in the original dataset. In other words, the predicting network determines whether the generated data is fake or real. The system generates newer, improved versions of fake data values until the predicting network can no longer distinguish fake from original.

Generative adversarial networks create realistic images through text-based prompts or by modifying existing images. They can help create realistic and immersive visual experiences in video games and digital entertainment.

GAN can also edit images—like converting a low-resolution image to a high resolution or turning a black-and-white image to color. It can also create realistic faces, characters, and animals for animation and video.

Structure:

- Generator: Tries to create fake data that looks real.
- Discriminator: Tries to detect whether the data is real or fake.

Use Cases:

- Face generation (thispersondoesnotexist.com)
- Style transfer (turn sketches into real photos)
- Deepfakes
- Medical image generation