Day89_GenAI_Key_Concepts_and_Foundations

September 25, 2025

Gen AI: Key Concepts and Foundations

1 Large Language Models (LLMs)

1.1 What are Large Language Models?

Large Language Models (LLMs) are advanced AI models trained on massive amounts of text data. They learn the patterns, grammar, facts, and reasoning structures of language to **predict the next word (token)** in a sequence.

We call them **"Large"** because:

- They have **billions or even trillions of parameters** (weights in the neural network).
- They are trained on **huge datasets** (books, articles, websites, code, etc.).

In simple words:

LLMs are like super-advanced autocomplete systems that can write essays, answer questions, generate code, and even carry on human-like conversations.

1.2 Why do we call them "LLMs"?

- **Large** → because of their size (data + parameters).
- **Language** → because their primary focus is understanding and generating natural language.
- **Model** → because they are machine learning models built using deep learning techniques (mainly Transformers).

1.3 Use Cases of LLMs

- **Conversational AI** → ChatGPT, Gemini, Claude.
- **Content Creation** → blog writing, story generation, marketing content.
- **Programming** → GitHub Copilot, code explanation, debugging.
- **Customer Support** → automated chatbots, FAQ assistants.
- **Education** → tutoring, summarizing study material.

- **Healthcare** → summarizing patient reports, clinical trial search.
- **Enterprise** → analyzing documents, generating reports, summarizing legal contracts.

1.4 Advantages of LLMs

- Can generate **human-like text** across domains.
- **Versatile** → one model can be used for translation, summarization, Q&A, etc.
- Enable **rapid prototyping** of AI applications (chatbots, agents, tools).
- Can integrate with external tools (search engines, databases).

1.5 Limitations of LLMs

- **Hallucinations** → Sometimes they generate **false but confident-sounding answers**.
- **Biases** → They may replicate social, cultural, or gender biases present in training data.
- **High Cost** → Training large models costs millions of dollars in compute and energy.
- **Limited Knowledge** → Models only know up to their training cut-off date unless connected to external data (RAG).
- **Context window constraints** → They cannot remember beyond their max input size.

1.6 Famous LLMs

- **GPT (OpenAI)** → GPT-3, GPT-4, GPT-4o (multimodal).
- **Gemini (Google DeepMind)** → Gemini 1.5, Gemini 2.5.
- **Claude (Anthropic)** → Claude 2, Claude 3.
- **LLaMA (Meta)** → LLaMA 2, LLaMA 3.
- **Mistral AI** → Mistral 7B, Mixtral 8x7B.
- **Cohere** → Command R series.
- **Falcon (TII UAE)** → Falcon 40B, Falcon 180B.

2 How Do LLMs Work?

1. **Autoregressive Prediction**
    - LLMs generate text one **token at a time**.

- Each token generated is fed back as input to predict the next token.

Example:
Prompt → "The cat is on the"
Model predicts → "mat" (one token at a time).

2. **Transformer Architecture**
   - LLMs use the **Transformer** neural network.
   - Key components:
     - **Self-Attention** → lets the model focus on important words in the input.
     - **Feedforward Layers** → process and transform information.
     - **Positional Encoding** → helps the model understand word order.
3. **Training Process**
   - Trained on **huge datasets** using GPUs/TPUs.
   - Objective: **Minimize the difference** between predicted tokens and actual tokens.
   - Requires **massive compute power, storage, and cost**.

3 Context Window, Temperature, Top-k, and Top-p

3.1 Context Window

- The **number of tokens (words/pieces of words)** the model can consider at once.
- Larger context window = model can handle **longer conversations/documents**.
- Example: GPT-4 → 128k tokens (about 300 pages of text).

3.2 Temperature

- Controls **randomness vs. determinism** in generation.
- Low temperature (0.2) → precise, factual, repetitive answers.
- High temperature (0.8–1.0) → creative, diverse, sometimes less factual.

3.3 Top-k Sampling

- The model looks at the **top k most likely next tokens** and randomly picks among them.
- Smaller k → less variety, more focus.
- Larger k → more creativity.

3.4 Top-p (Nucleus Sampling)

- Instead of fixed k, it chooses tokens from the **smallest set whose cumulative probability exceeds p**.
- Example: p = 0.9 → tokens chosen from the top 90% probability mass.
- More adaptive than top-k.

3.5 Combined Effect

- Developers often **combine Temperature + Top-k + Top-p** to fine-tune outputs.
- Example:
  - **Low temperature + small top-k** → good for factual Q&A.
  - **High temperature + nucleus sampling** → good for storytelling/creative writing.

4 Challenges in LLMs

4.1 Hallucinations

- LLMs sometimes **make up facts** that sound realistic.
- Example: "Einstein won the Nobel Prize in Chemistry" → incorrect.
- Fix → Use **RAG (Retrieval-Augmented Generation)** to ground answers.

4.2 Security Risks

- **Prompt Injection** → malicious inputs can trick LLMs into leaking sensitive info.
- **Data Privacy** → input data may be stored and misused if not handled carefully.
- **Biases** → harmful outputs reflecting bias in training data.

4.3 Cost

- Training → requires **massive hardware clusters**.
- Inference → serving billions of requests is expensive (energy + compute).

5 What is a Vector Database?

5.1 Definition

A **Vector Database** is a specialized database designed to store and search **vector embeddings** (numerical representations of text, images, audio, etc.).

5.2 How it Works

1. **Embedding Generation**
   - Text, image, or audio is converted into a high-dimensional **vector** (e.g., 768 or 1024 dimensions).
   - Example: "Cat" → [0.23, -0.11, 0.98, …]
2. **Storage**
   - These vectors are stored in a database.
3. **Similarity Search**
   - When a query comes in, it's converted into a vector.
   - The database finds **nearest neighbors** using Approximate Nearest Neighbor (ANN) search.
   - Example: Query = "puppy" → closest match = "dog" embeddings.

5.3 Examples of Vector Databases

- **Pinecone**
- **Weaviate**
- **Milvus**
- **Qdrant**
- **ChromaDB**

5.4 Why Important in Gen AI?

- LLMs don't **store all knowledge** inside them.
- Vector DBs allow LLMs to **retrieve knowledge dynamically** → enabling **search, RAG, personalization**.

6 What is RAG (Retrieval-Augmented Generation)?

6.1 Definition

RAG = Retrieval + Generation
It combines **retrieving external documents** with **LLM text generation**.

6.2 How it Works

1. User asks a question → "What are the side effects of aspirin?"
2. System retrieves relevant documents from a **vector database**.

3. The retrieved content is added to the LLM prompt.
4. LLM generates an answer grounded in facts.

6.3 Benefits of RAG

- **Reduces hallucinations** by grounding answers in external facts.
- **Keeps models up to date** without retraining.
- **Domain-specific knowledge** → e.g., company policies, medical reports.

6.4 Real-World Applications

- **Chatbots** that access company knowledge base.
- **Healthcare assistants** retrieving clinical data.
- **Legal research** retrieving past case law.
- **Education** summarizing textbooks with references.

7 Gen AI Application Development Steps

1. **Evaluate** → Is this really a Gen AI use case?
2. **Data Collection & Preparation** → Get domain-specific data.
3. **Choose Model Architecture** → GPT, Gemini, Claude, etc.
4. **Model Training / Fine-Tuning** → Adjust for your domain.
5. **Evaluation** → Check accuracy, reliability, bias.
6. **Optimize & Deploy** → Efficient APIs, scalable infra.
7. **Compliance & Ethics** → Ensure data privacy & fairness.
8. **Monitoring & Feedback** → Continuous improvement.

8 Gen AI Tech Stack

8.1 Frameworks

- LangChain, LlamaIndex, HuggingFace, PyTorch, TensorFlow

8.2 Vector Databases

- Pinecone, Qdrant, ChromaDB, Milvus, Weaviate

8.3 Cloud Platforms

- AWS Bedrock, Azure OpenAI Service, Google AI Studio

8.4 Famous LLMs

- GPT (OpenAI), Gemini (Google), Claude (Anthropic), LLaMA (Meta), Mistral AI

9 Key Takeaways

- LLMs are powerful **but not perfect** → they need external tools like **Vector DBs and RAG**.
- **Prompt engineering** and **parameter tuning (temperature, top-p, top-k)** control model behavior.
- **Vector databases** enable efficient similarity search for unstructured data.
- **RAG** grounds LLMs in facts, making them more reliable.
- Gen AI development is **not just about models**, but also about **data, infra, ethics, and user needs**.