**1. Why do we need chunking in NLP and RAG?**

Chunking is needed because LLMs have a token limit, and searching long documents efficiently requires breaking them into smaller parts. Without chunking:

* The model might miss key information.
* Retrieval could bring back irrelevant or incomplete context.
* Large documents may exceed token limits, making them hard to process.

**2. Why use a vector database instead of SQL for AI-powered search?**

A vector database (like Pinecone, Weaviate, or FAISS) is designed for **semantic search**, meaning it retrieves **similar** data based on meaning, not exact matches.

**Key Differences:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **SQL** | **Vector DB** |
| Data Type | Structured (tables, rows) | Unstructured (embeddings) |
| Search Type | Exact match | Similarity search |
| Speed | Slow for high-dimensional data | Fast for high-dimensional data |
| Use Case | Traditional queries | AI-powered retrieval |

**3. What is prompt engineering, and why is it important?**

Prompt engineering is designing effective instructions for LLMs to get the best results. It's important because **how** you ask influences **what** you get.

**Examples of Techniques:**

* **Zero-shot prompting** – Directly ask the model a question.
* **Few-shot prompting** – Give examples before the query.
* **Chain-of-thought prompting** – Encourage step-by-step reasoning.
* **Role-based prompting** – Ask the model to act as an expert.

**4. What are inference parameters in LLMs, and why use temperature = 1?**

Inference parameters control text generation behavior.

**Key Parameters:**

* **Temperature** (0–1+) → Controls randomness.
  + **Low (0-0.5):** More predictable.
  + **High (0.8-1.5):** More creative.
  + **1.0 (Balanced):** Mix of creativity and coherence.
* **Top-k sampling** → Picks from the top K likely words.
* **Top-p (nucleus sampling)** → Picks from the top **p%** of probable words.

Using **temperature = 1** gives a mix of **creativity** and **coherence** in responses.

**5. How to develop a Multimodal RAG system for text & images?**

A **Multimodal RAG** retrieves both text and image data.

**Architecture:**

1. **Text embeddings** → Using models like **OpenAI’s text-embedding-ada-002** or **BERT**.
2. **Image embeddings** → Using models like **CLIP** or **BLIP**.
3. **Fusion model** → Combines text + image data.
4. **Vector database** → Stores both embeddings.

**Tech Stack:**

* **LangChain** for RAG pipeline.
* **FAISS/Pinecone** for storage.
* **CLIP** for image retrieval.

**6. Is OCR the best approach for extracting text in a multimodal RAG system?**

**Limitations of OCR:**

* Struggles with **handwritten text** and **poor-quality images**.
* **Misses layout context** (tables, figures).
* **No semantic understanding** (only extracts text).

**Alternatives:**

* **LayoutLM** → Extracts text + layout info.
* **Donut** → Directly generates structured output without OCR.
* **Tesseract OCR + LLM** → Hybrid approach for better accuracy.

**7. How to generate embeddings from transactional data efficiently?**

**Steps:**

1. Convert the table into meaningful **text-based representations**.
   * Example: "User A bought Item X for $50 on Jan 1"
2. Use **transformers (like BERT, T5, or Sentence-BERT)** to create embeddings.
3. Store embeddings in a **vector database**.

**Scaling for millions of rows:**

* Use **FAISS/HNSW indexing** for fast retrieval.
* Process data in **batches** instead of all at once.
* Use **distributed computing (Spark, Ray, Dask)** for parallel processing.

**8. How to summarize a 100-page PDF efficiently?**

**Approach:**

1. **Split the PDF into chunks** (e.g., by paragraphs).
2. **Use an embedding model** (e.g., OpenAI’s text embeddings) to find key sections.
3. **Summarize each chunk** using models like **T5, GPT-4, or BART**.
4. **Combine & refine** summaries into a final version.

**Tools:**

* **LangChain** (for chunking & retrieval).
* **Hugging Face models** (for summarization).
* **LLMs (GPT-4, Claude)** for final refinements.

**9. Difference between GPT-3.5 and GPT-4?**

**Main Differences:**

|  |  |  |
| --- | --- | --- |
| Feature | GPT-3.5 | GPT-4 |
| Accuracy | Good | Better |
| Reasoning | Decent | Stronger |
| Multimodal | No | Yes (handles text + images) |
| Context Window | Shorter | Longer |

**Why GPT-4 is better?**

* **More accurate answers.**
* **Improved logic and reasoning.**
* **Can process images (GPT-4V).**