300 Interview/Viva Questions and Answers for LLM Engineer Roles

Table of Contents

🧩 Stage 1: Foundation (0–1 years)

A. Python & Software Engineering

1. **What are Python's key data structures?**
   Lists, tuples, sets, and dictionaries. Lists are mutable sequences; tuples are immutable; sets avoid duplicates; dictionaries store key-value pairs.
2. **Difference between `is` and `==` in Python?**
   `is` checks identity (same object in memory), while `==` checks equality of values.
3. **What are Python decorators?**
   Functions that wrap other functions to modify behavior. Used in logging, authentication, etc.
4. **How does list comprehension work in Python?**
   It allows compact expression of loops: `[x for x in iterable if condition]`.
5. **What is a lambda function?**
   A small anonymous function using the `lambda` keyword. Example: `lambda x: x + 1`.
6. **What is the difference between `__init__` and `__new__`?**
   `__init__` initializes an object after creation, `__new__` creates the object.
7. **What is a Python generator?**
   Functions that yield values one at a time using `yield`, ideal for memory efficiency.
8. **How does exception handling work in Python?**
   Use `try-except-finally`. `except` catches exceptions; `finally` runs regardless.
9. **What are Python's scopes?**
   LEGB: Local, Enclosing, Global, Built-in.
10. **Difference between mutable and immutable types?**
    Mutable can be changed (lists), immutable cannot (tuples, strings).
11. **What is pip and venv?**
    `pip` is Python's package manager; `venv` creates isolated environments.
12. **What are Python's magic methods?**
    Special methods with `__` (e.g., `__str__`, `__len__`) for class behaviors.
13. **What is the GIL in Python?**
    Global Interpreter Lock: prevents concurrent execution of bytecode in CPython.
14. **Difference between multiprocessing and multithreading?**
    Multiprocessing uses separate processes; threading shares memory.
15. **How to manage dependencies in a Python project?**
    Use `requirements.txt` or tools like `pipenv`, `poetry`.
16. **What is a context manager?**
    Manages resources with `with` keyword, e.g., file operations.
17. **What is duck typing?**
    Python checks behavior (methods/attributes) rather than type.
18. **Explain Python's garbage collection.**
    Uses reference counting and cyclic garbage collector.

19. **How to handle large files in Python?**
    Use generators, `with open(...) as f:` and read in chunks.
20. **What are Python type hints?**
    Syntax like `def func(a: int) -> str` to annotate types.
21. **What is a Python package vs module?**
    Module: `.py` file; Package: folder with `__init__.py`.
22. **What is the difference between `args` and `kwargs`?**
    `*args` for positional, `**kwargs` for keyword variable-length arguments.
23. **What is PEP8?**
    Python style guide for writing readable code.
24. **Explain list vs generator comprehensions.**
    List comprehensions return list, generator comprehensions yield items lazily.
25. **What is `__slots__`?**
    Restricts dynamic attribute creation; saves memory.
26. **What are Python metaclasses?**
    Classes of classes; define class behavior.
27. **Explain shallow vs deep copy.**
    Shallow: references inner objects; Deep: recursively copies all levels.
28. **What is a static method vs class method?**
    Static: no access to class/object; Class method: receives `cls`.
29. **Explain async and await in Python.**
    Enables asynchronous programming using `async def` and `await`.
30. **What is a memory view object in Python?**
    Efficient buffer access to binary data without copying.

🧠 B. Machine Learning & NLP Fundamentals

31. **What is tokenization in NLP?**
    Breaking down text into tokens—words, subwords, or characters—for processing.
32. **Difference between stemming and lemmatization?**
    Stemming truncates words to their root forms; lemmatization finds dictionary-based base forms.
33. **What is TF-IDF?**
    A statistical measure of word importance, combining frequency in a document and inverse frequency across the corpus.
34. **Define precision and recall.**
    Precision = TP / (TP + FP), Recall = TP / (TP + FN).
35. **What is a confusion matrix?**
    A table summarizing prediction results: TP, TN, FP, FN.
36. **What is overfitting?**
    When a model performs well on training data but poorly on unseen data.

37. **How do you prevent overfitting?**
    Techniques include regularization, dropout, data augmentation, and early stopping.
38. **Explain cross-validation.**
    A technique to assess model performance by splitting data into multiple train-test subsets.
39. **What is a loss function?**
    A function that quantifies the difference between predicted and actual values.
40. **Name some common loss functions.**
    MSE, Cross-Entropy, Hinge Loss, MAE.
41. **What is backpropagation?**
    Algorithm for training neural networks by updating weights based on gradients.
42. **What is the vanishing gradient problem?**
    In deep networks, gradients shrink and hinder learning in early layers.
43. **What is ReLU?**
    Activation function: `f(x) = max(0, x)`; helps alleviate vanishing gradient.
44. **What is an embedding in NLP?**
    A dense vector representing a word or sentence, capturing semantic meaning.
45. **Name common word embedding techniques.**
    Word2Vec, GloVe, FastText, BERT embeddings.
46. **What is transfer learning?**
    Using a pre-trained model and fine-tuning it for a specific task.
47. **What is a transformer?**
    A model architecture using self-attention to process sequences.
48. **Explain attention mechanism.**
    It computes a weighted sum of input representations, focusing on relevant parts.
49. **Difference between BERT and GPT?**
    BERT is bidirectional and used for understanding; GPT is autoregressive and used for generation.
50. **What is positional encoding?**
    Adds order information to embeddings in transformer models.

🚀 C. LLM APIs, Prompting, and Fine-Tuning (51–100)

51. **What is self-attention in transformers?**
    A mechanism that allows each token to attend to all other tokens, helping capture contextual relationships.
52. **What is the role of the softmax function in attention?**
    Converts attention scores into a probability distribution to weigh input tokens.
53. **What is the difference between encoder and decoder in transformers?**
    Encoder processes input sequences; decoder generates outputs based on encoder and previously generated tokens.

54. **What are masked language models?**
    Models like BERT that predict masked tokens in a sentence to learn bidirectional context.
55. **What is causal (autoregressive) language modeling?**
    Models like GPT that predict the next word given previous context.
56. **How do you use HuggingFace Transformers?**
    Load models with `from_pretrained`, tokenize with `AutoTokenizer`, run inference with `AutoModelFor...`.
57. **What are LLM APIs used for?**
    Accessing powerful language models (e.g., GPT-4) via endpoints to generate, summarize, or classify text.
58. **What is prompt engineering?**
    Designing effective inputs to elicit desired outputs from LLMs.
59. **What is zero-shot prompting?**
    Giving the model a task without examples—relies on model's general language understanding.
60. **What is few-shot prompting?**
    Including a few examples in the prompt to demonstrate task format.
61. **What is chain-of-thought prompting?**
    Asking LLMs to show reasoning steps, improving performance on complex tasks.
62. **What is temperature in generation APIs?**
    Controls randomness: 0 = deterministic, 1 = high creativity.
63. **What is top-k sampling?**
    Selects randomly from top k most probable tokens to introduce variability.
64. **What is top-p (nucleus) sampling?**
    Samples from smallest possible token set whose cumulative probability exceeds p.
65. **What is token limit in LLMs?**
    The maximum number of input/output tokens the model can process. GPT-4-8k/32k, etc.
66. **What is a system prompt?**
    A special instruction given to guide model behavior across conversations.
67. **How do you evaluate prompt effectiveness?**
    Use metrics like BLEU/ROUGE or manual scoring; iterate based on output relevance.
68. **What is prompt injection?**
    Security exploit where user input manipulates model behavior.
69. **What is grounding in LLM outputs?**
    Ensuring responses are backed by source documents or context to reduce hallucination.
70. **How to connect LLMs to external tools?**
    Via agents, tool calling, or function-calling interfaces in APIs like OpenAI functions.
71. **What is fine-tuning?**
    Adjusting pre-trained model weights on task-specific data.
72. **What is SFT (Supervised Fine-Tuning)?**
    Trains the model on input-output pairs to perform specific tasks.

73. **What is RLHF?**
    Reinforcement Learning with Human Feedback — used to align model outputs with human preferences.
74. **What is LoRA?**
    Low-Rank Adaptation adds trainable matrices to existing model weights for efficient fine-tuning.
75. **What is PEFT?**
    Parameter-Efficient Fine-Tuning techniques like LoRA, adapters, prompt tuning.
76. **What is QLoRA?**
    Combines LoRA with quantized models (e.g., 4-bit) to reduce memory use during training.
77. **What are adapters?**
    Lightweight modules inserted in model layers to adapt behavior with fewer parameters.
78. **What is HuggingFace PEFT library?**
    Library supporting efficient fine-tuning methods for transformers.
79. **What is DeepSpeed?**
    A library from Microsoft for large-scale training with parallelism and memory optimizations.
80. **What is FSDP?**
    Fully Sharded Data Parallel — sharding model weights/gradients across GPUs to enable large model training.
81. **What are evaluation metrics for LLMs?**
    BLEU, ROUGE, BERTScore, and GPT-based evaluators for text similarity and relevance.
82. **What is hallucination in LLMs?**
    When a model generates incorrect or fabricated information.
83. **What is a retrieval-augmented generation (RAG) system?**
    Combines document retrieval with language generation to answer questions based on external knowledge.
84. **What is FAISS?**
    A library for efficient similarity search and clustering of dense vectors.
85. **What is Pinecone?**
    A managed vector database for similarity search and RAG systems.
86. **What is Qdrant?**
    Open-source vector search engine, used with embedding models.
87. **How to use OpenAI embeddings for RAG?**
    Embed documents using OpenAI API, store in FAISS or similar, query top-k and pass to LLM.
88. **What are LangChain and LlamaIndex?**
    Frameworks for building LLM-powered applications with tools, memory, agents, and retrieval.
89. **What is a context window?**
    The number of tokens the model can see in a single interaction.

90. **What are hallucination mitigation strategies?**
    Use RAG, factual consistency checks, system prompts, or output post-processing.
91. **What is quantization in LLMs?**
    Reducing model precision (e.g., float32 → int8) to speed up inference and reduce memory.
92. **What is GPTQ?**
    Quantized Post-Training approach to reduce model size and retain accuracy.
93. **What is INT8 quantization?**
    Representing weights using 8-bit integers instead of 32-bit floats.
94. **What is ONNX?**
    Open Neural Network Exchange — a format to export models for cross-platform inference.
95. **What is TensorRT?**
    NVIDIA's SDK to optimize inference on GPUs.
96. **What is TGI (Text Generation Inference)?**
    HuggingFace server to deploy and serve LLMs with low latency.
97. **What is vLLM?**
    Optimized transformer inference engine using PagedAttention for faster batch decoding.
98. **What is FastAPI used for?**
    Building RESTful APIs; useful for serving models.
99. **How to deploy LLMs in production?**
    Package with Docker, expose via FastAPI, monitor with Prometheus or W&B.
100. **What is MLOps in the LLM context?**
    Practices for managing LLM lifecycle: training, serving, monitoring, and updating.

🔧 D. Advanced Deployment, LLMOps, and RAG Systems (101–150)

101. **What is inference latency?**
    The time it takes for a model to produce an output after receiving an input.
102. **How can you reduce inference latency?**
    Use quantization, batch processing, efficient tokenization, and optimized serving libraries like vLLM or TGI.
103. **What is batch inference?**
    Running multiple inputs simultaneously to improve throughput and GPU utilization.
104. **What is model checkpointing?**
    Saving model states during training to allow resuming or rollback.
105. **What is TorchScript?**
    A way to serialize PyTorch models for production deployment.
106. **What is model sharding?**
    Splitting model weights across multiple devices to handle large models.

107. **What is a model registry?**
A centralized store for model versions, metadata, and tracking (e.g., MLflow, W&B).
108. **What is containerization?**
Packaging code and dependencies into a portable environment using Docker or similar tools.
109. **What is CI/CD in MLOps?**
Continuous Integration and Delivery — automating testing and deployment pipelines.
110. **What are health checks in model serving?**
API endpoints that confirm if the model is live, responsive, and functioning correctly.
111. **How do you monitor model performance post-deployment?**
Use tools like Prometheus, Grafana, and Weights & Biases to track latency, throughput, accuracy.
112. **What is concept drift?**
Change in the data distribution over time, affecting model performance.
113. **How to detect concept drift?**
Monitor prediction confidence, model accuracy trends, and statistical tests.
114. **What is model versioning?**
Managing multiple versions of models for rollback and experimentation.
115. **What is a shadow deployment?**
Running new models in parallel with production to compare outputs without affecting users.
116. **What is a blue-green deployment?**
A technique to minimize downtime and risks by switching between two identical environments.
117. **What is load balancing in model serving?**
Distributing requests across multiple servers or instances to prevent overload.
118. **What are cold starts in LLMs?**
Latency spikes when inactive models are restarted or reloaded.
119. **What is warm start?**
Initializing model inference with preloaded weights to avoid startup delays.
120. **How to persist chat history across sessions?**
Store messages in a database or vector store; use session IDs to associate context.
121. **What is a vector store?**
A database optimized for similarity search on dense embeddings (e.g., FAISS, Pinecone).
122. **What is cosine similarity?**
A metric to compute similarity between two vectors; widely used in RAG systems.
123. **How do you generate embeddings for documents?**
Use OpenAI, HuggingFace, or Cohere models to encode text into dense vectors.
124. **How to chunk documents for RAG?**
Break content into meaningful segments (e.g., by sentence or paragraph) to maintain context.
125. **What is hybrid retrieval?**
Combining keyword-based (BM25) and vector-based retrieval for more robust search.

126. **What is reranking in RAG?**
Reordering retrieved chunks based on relevance using cross-encoders or LLMs.
127. **What is the difference between dense and sparse retrieval?**
Dense uses embeddings; sparse relies on term frequency and inverted indexes.
128. **What is LangChain Retriever?**
An abstraction to interface with various retrieval methods in LangChain.
129. **What is context window overflow?**
When the combined prompt and documents exceed the token limit of the LLM.
130. **How to handle context window limitations?**
Use sliding windows, summarization, chunk reranking, or use longer context models.
131. **What is token truncation?**
Cutting input text to fit within token limits; risks loss of key context.
132. **How to build a multi-document QA system?**
Retrieve top passages from multiple documents, format them into a prompt, and query an LLM.
133. **What is chunk overlap?**
Slightly overlapping segments when chunking documents to maintain continuity.
134. **How to evaluate retrieval performance?**
Use recall@k, precision@k, MRR, and relevance judgments.
135. **What is document reranking with BERT?**
Re-ranking retrieved results using BERT-based models for higher semantic accuracy.
136. **What are hallucinations in RAG systems?**
Generated content not supported by retrieved context.
137. **How to reduce hallucinations in RAG?**
Improve chunk quality, reranking, context relevance, or use factuality filters.
138. **What is prompt stuffing?**
Overloading the prompt with too much context, leading to degraded performance.
139. **What is few-shot RAG?**
Combines few-shot examples and retrieved documents for stronger grounding.
140. **What are long-context LLMs?**
Models like Claude 2 or GPT-4-32k that support extended token inputs.
141. **What are the risks of retrieval-based systems?**
Mismatched context, hallucinations, stale documents, or irrelevant chunks.
142. **What is document ranking?**
Ordering documents based on their relevance to the query.
143. **What is multi-hop retrieval?**
Chaining multiple retrieval steps to answer complex queries.
144. **What is cross-encoder vs bi-encoder?**
Cross-encoder scores query-doc pairs jointly; bi-encoder embeds separately and compares via dot product.
145. **How do you evaluate a chatbot?**
Use automated metrics (BLEU, ROUGE), human feedback, task success rate, and safety benchmarks.

146. **What are hallucination detection techniques?**
Consistency checks, fact-checking models, retrieval alignment, and user feedback.
147. **What are grounding techniques?**
Linking outputs back to retrieved context; citation markers, reranking, or references.
148. **What is document summarization?**
Condensing a document into a shorter version using extractive or abstractive methods.
149. **What is latent space in embeddings?**
High-dimensional space where semantically similar texts are located close together.
150. **What is top-k retrieval?**
Selecting the k most similar documents based on embedding similarity.

🧪 E. Evaluation, Safety, Tooling, and Advanced LLM Applications (151–200)

151. **What is BLEU score?**
A metric to evaluate generated text against a reference by comparing n-gram overlaps.
152. **What is ROUGE score?**
Measures recall-oriented overlap between generated and reference texts, useful for summarization.
153. **What is BERTScore?**
Uses BERT embeddings to compute semantic similarity between candidate and reference text.
154. **What are GPT-based evaluators?**
LLMs (e.g., GPT-4) used to assess coherence, accuracy, and fluency of generated text.
155. **What is human-in-the-loop (HITL) evaluation?**
Involves humans rating outputs for quality, grounding, and safety.
156. **How to conduct A/B testing for LLM outputs?**
Show users two model responses, gather preferences, and measure engagement or success.
157. **What is model alignment?**
Ensuring model outputs follow human values, instructions, and avoid harmful content.
158. **What is toxicity detection?**
Identifying harmful, offensive, or biased language in model outputs using classifiers or heuristics.
159. **What is red teaming in LLMs?**
Systematic probing of models to expose safety vulnerabilities and generate adversarial prompts.
160. **What is output filtering?**
Post-processing model responses to block unsafe or undesired content.
161. **How to reduce bias in LLMs?**
Use debiasing datasets, diverse training data, and post-hoc adjustment techniques.
162. **What is prompt grounding?**
Anchoring generation to a trusted source document or retrieval context.

163. **What is hallucination feedback loop?**
When a model’s past hallucinated outputs are reused and reinforced as context.
164. **What is the purpose of system messages in chat models?**
Provide high-level behavior guidelines to steer the assistant's tone and style.
165. **What is an LLM tool agent?**
An orchestrated agent that uses tools like calculators, web search, or file I/O in conjunction with LLM.
166. **What is a memory in LLM apps?**
Mechanism to retain information between user turns or sessions.
167. **What is a conversational agent framework?**
A structured platform for building LLM-powered chat systems with memory, tools, and context handling (e.g., LangChain).
168. **What are OpenAI function calls?**
Structured calls from LLMs to predefined tools or functions, improving interaction reliability.
169. **What is LangServe?**
A serving utility from LangChain for deploying chains and agents as APIs.
170. **What is LangGraph?**
A graph-based framework to build multi-agent workflows using LLMs.
171. **What are vector-capable document loaders?**
Tools in LangChain or LlamaIndex to load and convert documents into retrievable chunks.
172. **What is document parsing?**
Extracting clean, structured text from PDFs, DOCX, HTML, etc., for NLP/RAG workflows.
173. **How to build a multi-turn chatbot?**
Track chat history, use memory/context storage, and design prompts to maintain persona.
174. **What is long-term memory in chatbots?**
Persistence of important user preferences or prior facts across sessions.
175. **What is function-calling JSON format?**
Structured schema for LLMs to return JSON to call specific backend functions.
176. **What is a tool-using agent loop?**
A cycle where LLM generates an action, invokes a tool, observes the result, and iterates.
177. **What is a planner-executor pattern?**
A planning module generates high-level goals; executor LLM/tool carries out steps.
178. **What is an autonomous agent?**
A system that makes decisions and takes actions with minimal human oversight, often with LLM+tools.
179. **What is evaluation for autonomous agents?**
Includes task success, number of tool calls, safety checks, and runtime duration.
180. **What are benchmarks for LLMs?**
HELM, MMLU, BIG-bench, TruthfulQA, ARC — measure reasoning, safety, and factuality.

181. **What is the difference between OpenAI Evals and human evals?**
OpenAI Evals automate performance tracking using templates; human evals rely on real user judgments.
182. **What is instruction tuning?**
Training models to follow natural language instructions more reliably.
183. **What is dataset curation?**
Collecting and cleaning task-specific data with annotations for training and evaluation.
184. **What is data labeling noise?**
Inconsistent or incorrect human labels that reduce model quality.
185. **How to automate LLM evaluation?**
Use GPT-4 to score outputs, apply regex checks, or embed-based similarity metrics.
186. **What is prompt compression?**
Reducing prompt length via summarization or abstraction to fit context windows.
187. **What is hybrid summarization?**
Combines extractive and abstractive techniques to generate informative summaries.
188. **What is a jailbreak in LLMs?**
A prompt or trick designed to bypass safety filters or restrictions.
189. **How to defend against jailbreaks?**
Fine-tuning, input validation, layered safety filters, and adversarial training.
190. **What is user intent classification?**
Predicting the user's goal or category to route queries or enhance understanding.
191. **What is fallback prompting?**
A backup prompt used when a model fails or produces an irrelevant answer.
192. **What is an LLM safety harness?**
A protective layer that monitors, filters, and validates LLM outputs before delivery.
193. **What is adversarial prompting?**
Intentionally crafted inputs to confuse, break, or mislead the model.
194. **What is semantic caching?**
Storing embeddings and outputs to reuse answers for similar queries.
195. **What are cost control techniques for LLM apps?**
Use token-efficient models, rate-limiting, caching, and content filtering.
196. **What are LLMOps pipelines?**
Orchestration workflows for model tuning, deployment, monitoring, and feedback collection.
197. **What is API rate limiting?**
Controlling how many requests users can make to an API in a time window.
198. **What are open-weight vs closed-weight models?**
Open-weight: source and weights available (e.g., LLaMA); closed-weight: proprietary (e.g., GPT-4).
199. **What is model distillation?**
Training a smaller model to replicate a larger model’s behavior for efficiency.
200. **What is the future of LLM engineering?**
Real-time multimodal agents, continual learning, scalable infrastructure, and personalized copilots.

🧬 F. Multimodal LLMs, Federated Learning, and Interpretability (201–300)

🔁 Multimodal LLMs (Images/Video + Text)

201. **What are multimodal models?**
    Models that process and generate across more than one data modality, e.g., text + image.
202. **Name some examples of multimodal LLMs.**
    GPT-4-Vision, Flamingo, LLaVA, Kosmos-1, Gemini.
203. **What is CLIP?**
    Contrastive Language-Image Pretraining — links images and text by aligning their embeddings.
204. **What is BLIP-2?**
    Bootstrapped Language-Image Pretraining — an architecture for image captioning and VQA.
205. **What is visual grounding in LLMs?**
    Linking textual descriptions to image regions or visual concepts.
206. **How do multimodal LLMs encode images?**
    Use vision encoders (e.g., ViT, ResNet) to convert images to embeddings.
207. **What is an image-text embedding space?**
    A shared vector space where similar text/image pairs are close together.
208. **How do you fine-tune multimodal LLMs?**
    Combine image-caption pairs or use instruction tuning with image inputs.
209. **What is visual question answering (VQA)?**
    The task of answering natural language questions about images.
210. **What is image captioning?**
    Generating a natural language description of an image.
211. **What is vision-language pretraining (VLP)?**
    Joint training of visual and textual inputs to improve multimodal understanding.
212. **How does GPT-4-Vision process images?**
    Uses an internal vision encoder to embed image data and feed it into the transformer.
213. **What are challenges in multimodal training?**
    Data alignment, modality imbalance, large compute requirements, image resolution handling.
214. **What are applications of multimodal LLMs?**
    OCR, VQA, image search, autonomous agents, multimodal assistants.
215. **What is multimodal context fusion?**
    Combining visual, auditory, and textual signals for integrated reasoning.

🌐 Federated Learning & Decentralized AI

216. **What is federated learning?**
A technique to train models across decentralized devices while keeping data local.
217. **What is FL client-server architecture?**
Clients train local models; the server aggregates updates to form a global model.
218. **What is FedAvg?**
Federated Averaging algorithm used to aggregate model weights across clients.
219. **What are challenges in FL?**
Data heterogeneity, unreliable clients, communication cost, model drift.
220. **How is privacy preserved in FL?**
Data never leaves local device; use of differential privacy or secure aggregation.
221. **What is secure aggregation?**
Ensures server can only see aggregated updates, not individual contributions.
222. **What is differential privacy in FL?**
Adds noise to updates to protect individual data points.
223. **What is split learning?**
Splits model across client/server to reduce computation on edge devices.
224. **What is homomorphic encryption in FL?**
Enables computation on encrypted data, ensuring privacy.
225. **Use cases for FL in LLMs?**
On-device personalization, edge AI, healthcare, and finance applications.
226. **How does personalization work in FL?**
Each client fine-tunes a shared base model to local data.
227. **What is cross-device vs cross-silo FL?**
Cross-device: many small devices (e.g., phones); cross-silo: fewer institutional clients.
228. **What is model poisoning in FL?**
A malicious client sends manipulated updates to degrade model performance.
229. **How to defend FL from adversarial attacks?**
Use robust aggregation, anomaly detection, and secure protocols.
230. **What is continual learning in FL?**
Adapting models over time with evolving client data.

🔎 Interpretability & Explainability Tools

231. **What is model interpretability?**
The ability to understand how a model makes its decisions.
232. **What is AttentionViz?**
A tool for visualizing attention weights in transformer models.
233. **What is Captum?**
A PyTorch library for interpretability using methods like saliency, integrated gradients.
234. **What are saliency maps?**
Visual overlays showing input areas most responsible for the model's output.
235. **What are SHAP values?**
Explain output by assigning contribution scores to each feature.

236. **What is Integrated Gradients?**
A technique that accumulates gradients along the input path to explain predictions.
237. **What is LIME?**
Local Interpretable Model-Agnostic Explanations — approximates predictions with simple local models.
238. **What is attention rollout?**
Aggregates attention across layers to track how input tokens influence output.
239. **What is a probe in model analysis?**
A simple classifier trained on model internals to test for encoded knowledge.
240. **What are counterfactual explanations?**
Describe how an input must change to alter the model's prediction.
241. **What is feature attribution?**
Assigning importance scores to input features.
242. **What is a neuron activation visualization?**
Plots or highlights the firing strength of model neurons for given inputs.
243. **How to debug LLM prompts?**
Log token-level probabilities, use reduced prompts, test systematically.
244. **What is model introspection?**
Techniques to explore internal representations or embeddings.
245. **What is embedding projection?**
Visualizes high-dimensional embeddings using t-SNE or PCA.
246. **How to interpret classification logits?**
Raw scores before softmax — can indicate model confidence or uncertainty.
247. **What is layer-wise relevance propagation (LRP)?**
Attribution method that traces output relevance backward through layers.
248. **How to evaluate interpretability tools?**
Use faithfulness, consistency, and completeness as metrics.
249. **Why interpretability is important for LLMs?**
Ensures trust, safety, accountability, and bias detection.
250. **What is transparency in AI systems?**
The ability to audit, understand, and reproduce model decisions.

🤖 G. Multi-Agent Systems, Knowledge Integration, GenAI Startups & Future Trends (251–300)

🤝 Multi-Agent & Autonomous Systems

251. **What is a multi-agent LLM system?**
A system where multiple LLMs (or LLM + tools) collaborate to solve complex tasks.
252. **What is an agent in AI?**
A component that observes, reasons, and acts within an environment to achieve a goal.
253. **What is an executor in multi-agent LLM frameworks?**
The agent or component responsible for carrying out planned tasks or steps.

254. **What is a planner in LLM agents?**
A module that decomposes high-level goals into sub-tasks.
255. **What is a coordinator agent?**
An agent responsible for assigning or delegating tasks among other agents.
256. **What is ReAct framework?**
Combines reasoning and acting: LLM outputs thought and action alternately.
257. **What are tool-using LLM agents?**
Agents that invoke external tools (APIs, calculators, search engines) via structured calls.
258. **What is LangGraph used for?**
Building complex multi-agent workflows with branching, retries, and conditional logic.
259. **What are autonomous AI agents?**
Systems like AutoGPT and AgentGPT that perform tasks with minimal human guidance.
260. **What are memory types in agents?**
Short-term (chat history), long-term (external knowledge), episodic (task events).
261. **What is an agent loop?**
The cycle of observation → reasoning → action → observation, repeated until goal is met.
262. **What is task decomposition?**
Splitting a complex task into smaller, manageable subtasks.
263. **What is inter-agent communication?**
The process of agents exchanging messages or outputs to collaborate effectively.
264. **What is the OpenAgents project?**
An open framework for building LLM-based multi-agent systems.
265. **What are evaluation metrics for agents?**
Task completion rate, steps taken, reward score, and human ratings.

🧠 Knowledge Integration & Retrieval

266. **What is a knowledge graph?**
A structured representation of entities and relationships used to augment reasoning.
267. **What is knowledge distillation?**
A smaller model learns to mimic the behavior of a larger, more complex model.
268. **How do LLMs use external knowledge?**
Through retrieval-augmented generation (RAG), APIs, or linked databases.
269. **What is a knowledge base chatbot?**
An assistant grounded in static or dynamic documentation (e.g., internal wikis).
270. **What is semantic search?**
Retrieval based on meaning and context rather than keyword matching.
271. **What is knowledge-aware generation?**
Tailoring responses to facts retrieved from structured/unstructured sources.
272. **What are ontology-based systems?**
Systems guided by a formal specification of concepts and relationships in a domain.
273. **What is triple extraction?**
Extracting subject-predicate-object triples from text to populate a knowledge graph.

274. **What are open-domain QA systems?**
Question-answering models capable of responding to queries across multiple topics.
275. **What is few-shot retrieval?**
Retrieval enhanced by including few-shot examples in the search or generation context.

💼 LLM Applications & GenAI Products

276. **What are copilots?**
Context-aware assistants embedded in development, writing, or design environments.
277. **What is a domain-specific LLM?**
An LLM fine-tuned or trained for a specific industry or task (e.g., legal, medical).
278. **What are some popular LLM APIs?**
OpenAI, Anthropic Claude, Google Gemini, Mistral, Cohere.
279. **What is text-to-SQL generation?**
Automatically converting natural language questions to SQL queries.
280. **What is structured data extraction?**
Parsing documents or chat into JSON, tables, or forms.
281. **What is code generation with LLMs?**
Using prompts to generate boilerplate, functions, or explain existing code.
282. **What is AI content moderation?**
Using LLMs or classifiers to detect and filter toxic, unsafe, or off-topic content.
283. **What is automated summarization?**
Reducing long text into concise summaries using extractive or abstractive methods.
284. **What are LLM-driven workflows?**
Automating multi-step processes by chaining LLM outputs and tool invocations.
285. **What are retrieval-enhanced copilots?**
Tools like GitHub Copilot + RAG context from codebases or documentation.

🚀 Future Trends & Advanced Ideas

286. **What is continual pretraining?**
Ongoing model training with new data to maintain relevance and reduce drift.
287. **What is multimodal agent architecture?**
An agent capable of understanding and generating across text, image, and audio.
288. **What is self-evaluation in LLMs?**
The model critiques or ranks its own output to refine generation.
289. **What is synthetic data generation?**
Using LLMs to create training or test data for low-resource domains.
290. **What is open-ended task planning?**
Allowing agents to formulate goals and steps dynamically based on environment.
291. **What are personalized LLMs?**
Models tailored to individual users' preferences, goals, or prior behavior.
292. **What is edge deployment of LLMs?**
Running models on mobile or embedded devices with low latency requirements.

293. **What is agentic evaluation?**
Judging performance of LLM agents over extended tasks and interactions.
294. **What is LLM watermarking?**
Embedding invisible patterns into output to trace origin or verify authenticity.
295. **What is synthetic QA pair generation?**
Automatically generating question-answer pairs to augment datasets.
296. **What is the role of GenAI in education?**
Intelligent tutors, feedback tools, and curriculum design through LLMs.
297. **What are AI copilots for enterprise?**
Internal assistants trained on org-specific data for search, support, and automation.
298. **What is auto-labeling?**
Using LLMs to annotate datasets for training downstream models.
299. **What is AI-assisted research?**
LLMs help summarize papers, generate hypotheses, or draft content.
300. **What is the outlook for LLM engineers?**
High demand across sectors; future skills include multi-agent design, safety alignment, and custom LLM pipelines.