Question #1 *Topic 1*

A Generative AI Engineer has created a RAG application to look up answers to questions about a series of fantasy novels that are being asked on the author's web forum. The fantasy novel texts are chunked and embedded into a vector store with metadata (page number, chapter number, book title), retrieved with the user's query, and provided to an LLM for response generation. The Generative AI Engineer used their intuition to pick the chunking strategy and associated configurations but now wants to more methodically choose the best values.
Which TWO strategies should the Generative AI Engineer take to optimize their chunking strategy and parameters? (Choose two.)

A. Change embedding models and compare performance.

B. Add a classifier for user queries that predicts which book will best contain the answer. Use this to filter retrieval.

C. Choose an appropriate evaluation metric (such as recall or NDCG) and experiment with changes in the chunking strategy, such as splitting chunks by paragraphs or chapters. Choose the strategy that gives the best performance metric.

D. Pass known questions and best answers to an LLM and instruct the LLM to provide the best token count. Use a summary statistic (mean, median, etc.) of the best token counts to choose chunk size.

E. Create an LLM-as-a-judge metric to evaluate how well previous questions are answered by the most appropriate chunk. Optimize the chunking parameters based upon the values of the metric.

Suggested Answer: *CE*

trendy01 1 month ago
D has limitations depending on the number of tokens in the LLM, which may be disadvantageous in constructing a chunk strategy that sufficiently reflects the context of the question. C and E are more comprehensive and effective optimization strategies because they experimentally evaluate various chunk division strategies and directly measure the suitability of questions and answers.
upvoted 1 times

awron_durat 1 month ago
Answers CE make sense to me! Either using a metric and comparing different methods or LLM as a judge makes sense to me.
upvoted 1 times

Rajmlops 1 month, 3 weeks ago
right answer
upvoted 1 times

Question #2 *Topic 1*

A Generative AI Engineer is designing a RAG application for answering user questions on technical regulations as they learn a new sport. What are the steps needed to build this RAG application and deploy it?

A. Ingest documents from a source –> Index the documents and saves to Vector Search –> User submits queries against an LLM –> LLM retrieves relevant documents –> Evaluate model –> LLM generates a response –> Deploy it using Model Serving

B. Ingest documents from a source –> Index the documents and save to Vector Search –> User submits queries against an LLM –> LLM retrieves relevant documents –> LLM generates a response -> Evaluate model –> Deploy it using Model Serving

C. Ingest documents from a source –> Index the documents and save to Vector Search –> Evaluate model –> Deploy it using Model Serving

D. User submits queries against an LLM –> Ingest documents from a source –> Index the documents and save to Vector Search –> LLM retrieves relevant documents –> LLM generates a response –> Evaluate model –> Deploy it using Model Serving

Suggested Answer: *B*

awron_durat 1 month ago

First, the RAG application including the LLM has to be built, and then you can evaluate and deploy, so B.

upvoted 2 times

Question #3 *Topic 1*

A Generative AI Engineer just deployed an LLM application at a digital marketing company that assists with answering customer service inquiries.
Which metric should they monitor for their customer service LLM application in production?

A. Number of customer inquiries processed per unit of time

B. Energy usage per query

C. Final perplexity scores for the training of the model

D. HuggingFace Leaderboard values for the base LLM

Suggested Answer: *A*

awron_durat 1 month ago

I said A inquiries/time because B and C are more for optimization or training and D would be for the development phase when originally building the application, not production.

upvoted 1 times

Question #4 *Topic 1*

A Generative AI Engineer is building a Generative AI system that suggests the best matched employee team member to newly scoped projects. The team member is selected from a very large team. The match should be based upon project date availability and how well their employee profile matches the project scope. Both the employee profile and project scope are unstructured text.
How should the Generative AI Engineer architect their system?

A. Create a tool for finding available team members given project dates. Embed all project scopes into a vector store, perform a retrieval using team member profiles to find the best team member.

B. Create a tool for finding team member availability given project dates, and another tool that uses an LLM to extract keywords from project scopes. Iterate through available team members' profiles and perform keyword matching to find the best available team member.

C. Create a tool to find available team members given project dates. Create a second tool that can calculate a similarity score for a combination of team member profile and the project scope. Iterate through the team members and rank by best score to select a team member.

D. Create a tool for finding available team members given project dates. Embed team profiles into a vector store and use the project scope and filtering to perform retrieval to find the available best matched team members.

Suggested Answer: *D*

trendy01 1 month ago
Answer D,
From a Generative AI engineering perspective, D is the most effective and practical approach for optimizing the match between project scope and team profile.
upvoted 1 times

trendy01 1 month ago
C, This method ranks team members by checking their availability and calculating a similarity score between their profile and project scope, allowing you to recommend the most suitable team members by taking both factors into account. The score-based approach allows you to quantitatively assess the suitability of each team member, contributing to increasing the reliability of recommendations.
upvoted 1 times

awron_durat 1 month ago
A and D were both good answers at first but it makes more sense to embed team profiles than project scopes because of the way you'd want to do searches on the system.
upvoted 1 times

Question #5 *Topic 1*

A Generative AI Engineer is designing an LLM-powered live sports commentary platform. The platform provides real-time updates and LLM-generated analyses for any users who would like to have live summaries, rather than reading a series of potentially outdated news articles. Which tool below will give the platform access to real-time data for generating game analyses based on the latest game scores?

A. DatabricksIQ

B. Foundation Model APIs

C. Feature Serving

D. AutoML

Suggested Answer: *C*

trendy01 1 month ago

Answer C,
B. Foundation Model APIs
Description: An API that accesses the LLM model, allowing you to use the model's features, but is not suitable for collecting real-time sports data directly.
C.Feature Serving
Description: Feature Serving is a function that provides features that a machine learning model can use in real time. This allows real-time sports data (e.g. scores, statistics) to be fed into the model, making it ideal for generating analytics based on this data.
upvoted 1 times

Harry_D 1 month, 1 week ago

I am changing my vote. I think C. Feature Serving is the correct answer. In this post from microsoft, it talks about what is feature serving.
https://learn.microsoft.com/en-us/azure/databricks/machine-learning/feature-store/feature-function-serving

With Databricks Feature Serving, you can serve structured data for retrieval augmented generation (RAG) applications, as well as features that are required for other applications, such as models served outside of Databricks or any other application that requires features based on data in Unity Catalog.

The code provided in https://learn.microsoft.com/en-us/azure/databricks/machine-learning/feature-store/feature-serving-tutorial gives an example of how to use feature serving.
upvoted 1 times

awron_durat 1 month ago

I agree! DatabricksIQ and AutoML are more focused on model development and optimization, not real-time data provisioning. Foundation Model APIs provide general pre-trained models for tasks, but don't handle live data integration for near real-time sports commentary.
upvoted 1 times

Harry_D 1 month, 2 weeks ago

I vote for B. Foundation Model API.
upvoted 1 times

Question #6 *Topic 1*

A Generative AI Engineer has a provisioned throughput model serving endpoint as part of a RAG application and would like to monitor the serving endpoint's incoming requests and outgoing responses. The current approach is to include a micro-service in between the endpoint and the user interface to write logs to a remote server.
Which Databricks feature should they use instead which will perform the same task?

A. Vector Search

B. Lakeview

C. DBSQL

D. Inference Tables

Suggested Answer: *D*

trendy01 1 month ago
D, Inference Tables are used to store and manage prediction results from machine learning models. This provides the ability to record and monitor forecast data.

This is because it is useful for recording request and response data and managing the results. Therefore, it can serve as an effective tool for monitoring incoming requests and outgoing responses in real time.
upvoted 1 times

awron_durat 1 month ago
Must be D. Vector Search makes no sense, DBSQL doesn't relate to log tasks, and Lakeview says "Lakeview Dashboards provide users with a comprehensive toolset for creating, analyzing, and disseminating data-driven insights." from this website:
upvoted 1 times

Question #8 *Topic 1*

A Generative AI Engineer is creating an LLM-based application. The documents for its retriever have been chunked to a maximum of 512 tokens each. The Generative AI Engineer knows that cost and latency are more important than quality for this application. They have several context length levels to choose from.
Which will fulfill their need?

A. context length 514; smallest model is 0.44GB and embedding dimension 768

B. context length 2048: smallest model is 11GB and embedding dimension 2560

C. context length 32768: smallest model is 14GB and embedding dimension 4096

D. context length 512: smallest model is 0.13GB and embedding dimension 384

Suggested Answer: *D*

awron_durat 1 month ago

Selected Answer: D

D is correct.

upvoted 1 times

srihdar 1 month, 1 week ago

D is the answer

Since cost and latency are more important than quality for this application, the smallest model with the shortest context length (512 tokens) will be the most efficient in terms of resource usage and speed. This option provides lower latency and cost compared to the larger models, making it a suitable choice for the engineer's requirements.

upvoted 1 times

Question #10 *Topic 1*

A Generative AI Engineer is responsible for developing a chatbot to enable their company's internal HelpDesk Call Center team to more quickly find related tickets and provide resolution. While creating the GenAI application work breakdown tasks for this project, they realize they need to start planning which data sources (either Unity Catalog volume or Delta table) they could choose for this application. They have collected several candidate data sources for consideration: call_rep_history: a Delta table with primary keys representative_id, call_id. This table is maintained to calculate representatives' call resolution from fields call_duration and call start_time. transcript Volume: a Unity Catalog Volume of all recordings as a *.wav files, but also a text transcript as *.txt files. call_cust_history: a Delta table with primary keys customer_id, cal1_id. This table is maintained to calculate how much internal customers use the HelpDesk to make sure that the charge back model is consistent with actual service use. call_detail: a Delta table that includes a snapshot of all call details updated hourly. It includes root_cause and resolution fields, but those fields may be empty for calls that are still active. maintenance_schedule – a Delta table that includes a listing of both HelpDesk application outages as well as planned upcoming maintenance downtimes.
They need sources that could add context to best identify ticket root cause and resolution.
Which TWO sources do that? (Choose two.)

A. call_cust_history

B. maintenance_schedule

C. call_rep_history

D. call_detail

E. transcript Volume

Suggested Answer: *DE*

trendy01 1 month ago

D.call_detail
This delta table contains the details of the call and has root_cause and resolution fields. This provides important information to understand and solve the problem.

E. Transcript Volume
This Unity catalog volume includes text transcriptions of conversations with customers. Provide specific details related to your customer's issue to help determine the root cause.

upvoted 1 times

HemaKG 1 month, 1 week ago

I got answer as B & E. Request to review into the same.

upvoted 1 times

Question #12 *Topic 1*

When developing an LLM application, it's crucial to ensure that the data used for training the model complies with licensing requirements to avoid legal risks.
Which action is NOT appropriate to avoid legal risks?

A. Reach out to the data curators directly before you have started using the trained model to let them know.

B. Use any available data you personally created which is completely original and you can decide what license to use.

C. Only use data explicitly labeled with an open license and ensure the license terms are followed.

D. Reach out to the data curators directly after you have started using the trained model to let them know.

Suggested Answer: *B*

Community vote distribution

D (100%)

HardLearner 1 week, 5 days ago

Selected Answer: D

Definitely D, it is not appropriate to use model either in dev or prod before checking the licensing

upvoted 1 times

568f95c 1 month, 2 weeks ago

Selected Answer: D

You can use personal data because it has not licensing if it is yours, You should not reach out after using data from external source, to find out if you now can use it.

upvoted 2 times

mojj 1 month, 2 weeks ago

The answer must be D

upvoted 3 times

Question #13 *Topic 1*

A Generative AI Engineer is testing a simple prompt template in LangChain using the code below, but is getting an error.

```
from langchain.chains import LLMChain
from langchain_community.llms import OpenAI
from langchain_core.prompts import PromptTemplate

prompt_template = "Tell me a {adjective} joke"

prompt = PromptTemplate(
     input_variables=["adjective"],
     template=prompt_template
)

llm = LLMChain(prompt=prompt)
llm.generate([{"adjective": "funny"}])
```

Assuming the API key was properly defined, what change does the Generative AI Engineer need to make to fix their chain?

A.
```
prompt_template = "Tell me a {adjective} joke"

prompt = PromptTemplate(
     input_variables=["adjective"],
     template=prompt_template
)

llm = LLMChain(prompt=prompt)
llm.generate("funny")
```

B.
```
prompt_template = "Tell me a {adjective} joke"

prompt = PromptTemplate(
     input_variables=["adjective"],
     template=prompt_template
)

llm = LLMChain(prompt=prompt.format("funny"))
llm.generate()
```

C.
```
prompt_template = "Tell me a {adjective} joke"

prompt = PromptTemplate(
     input_variables=["adjective"],
     template=prompt_template
     llm=OpenAI()
)

llm = LLMChain(prompt=prompt)
llm.generate([{"adjective": "funny"}])
```

D.
```
prompt_template = "Tell me a {adjective} joke"

prompt = PromptTemplate(
     input_variables=["adjective"],
     template=prompt_template
)

llm = LLMChain(llm=OpenAI(), prompt=prompt)
llm.generate([{"adjective": "funny"}])
```

Suggested Answer: *C*

Community vote distribution

D (100%)

fr2022 2 weeks, 1 day ago

Selected Answer: D

from langchain.chains import LLMChain

from langchain_community.llms import OpenAI

from langchain_core.prompts import PromptTemplate

```
prompt_template = "Tell me a {adjective} joke"
prompt = PromptTemplate(
input_variables=["adjective"], template=prompt_template
)
llm = LLMChain(llm=OpenAI(), prompt=prompt)
```

upvoted 1 times

4af18fc 1 month, 2 weeks ago

Selected Answer: D

Option D is correct as we need to pass the OpenAi() constructor to the LLMChain()

upvoted 3 times

Question #16 *Topic 1*

A Generative Al Engineer interfaces with an LLM with prompt/response behavior that has been trained on customer calls inquiring about product availability. The LLM is designed to output “In Stock” if the product is available or only the term “Out of Stock” if not.
Which prompt will work to allow the engineer to respond to call classification labels correctly?

A. Respond with “In Stock” if the customer asks for a product.

B. You will be given a customer call transcript where the customer asks about product availability. The outputs are either “In Stock” or “Out of Stock”. Format the output in JSON, for example: {“call_id”: “123”, “label”: “In Stock”}.

C. Respond with “Out of Stock” if the customer asks for a product.

D. You will be given a customer call transcript where the customer inquires about product availability. Respond with “In Stock” if the product is available or “Out of Stock” if not.

Suggested Answer: *B*

Community vote distribution

D (100%)

trendy01 1 month ago

Selected Answer: B

B is providing a **specific format (JSON format)** for engineers to classify inventory status for customer inquiries via LLM. This requires the model to clearly output inventory status and output this structured in JSON format, making it suitable for post-processing the data or using it in automated systems.

Compared to other options, B provides specific guidelines to accurately manage classification labels.

A and C each require an “In Stock” or “Out of Stock” response, but do not provide clear guidance on specific labeling formats or processes.
D requires a response about inventory status, but is insufficient because it does not provide a specific data format or structured output.

upvoted 1 times

maciejmirski 1 month ago

Selected Answer: D

D is correct

upvoted 1 times

HemaKG 1 month, 1 week ago

D. You will be given a customer call transcript where the customer inquiries about product availability. Respond with “In Stock” if the product is available or “Out of Stock” if not.
Dear service provider, Please do not leave it with CORRECT ANSWER and Community Vote Distribution. Please provide silid explanation.

upvoted 1 times

4af18fc 1 month, 2 weeks ago

Selected Answer: D

Option B doesn't tell when to use "In Stock" and "Out of Stock".
D should be the correct answer

upvoted 1 times

Question #19 *Topic 1*

A Generative AI Engineer has been asked to build an LLM-based question-answering application. The application should take into account new documents that are frequently published. The engineer wants to build this application with the least cost and least development effort and have it operate at the lowest cost possible.
Which combination of chaining components and configuration meets these requirements?

A. For the application a prompt, a retriever, and an LLM are required. The retriever output is inserted into the prompt which is given to the LLM to generate answers.

B. The LLM needs to be frequently with the new documents in order to provide most up-to-date answers.

C. For the question-answering application, prompt engineering and an LLM are required to generate answers.

D. For the application a prompt, an agent and a fine-tuned LLM are required. The agent is used by the LLM to retrieve relevant content that is inserted into the prompt which is given to the LLM to generate answers.

Suggested Answer: *A*

Community vote distribution

A (100%)

trendy01 1 month ago

Selected Answer: A

A works by using a retriever to retrieve information from a new document and then inserting it into the prompt to pass it on to the LLM. This structure is effective in providing up-to-date information, reflecting frequently updated documentation, while minimizing cost and development effort.

B describes the update frequency of LLM, but lacks a description of the actual application architecture.

C only mentions prompt engineering and LLM, and doesn't explain how to handle updates for new documents.

D describes an agent-using approach, but lacks specifics on how an agent-using structure would achieve the same effective information retrieval and insertion as A.

upvoted 1 times

Question #23 *Topic 1*

A Generative AI Engineer is developing a chatbot designed to assist users with insurance-related queries. The chatbot is built on a large language model (LLM) and is conversational. However, to maintain the chatbot's focus and to comply with company policy, it must not provide responses to questions about politics. Instead, when presented with political inquiries, the chatbot should respond with a standard message: "Sorry, I cannot answer that. I am a chatbot that can only answer questions around insurance."
Which framework type should be implemented to solve this?

A. Safety Guardrail

B. Security Guardrail

C. Contextual Guardrail

D. Compliance Guardrail

Suggested Answer: *D*

trendy01 1 month ago

Selected Answer: D

D. Compliance Guardrail is a suitable choice to solve this problem.

Compliance guardrails are a way to ensure that your chatbot adheres to certain policies, where company policy requires it to block answers to political questions and return a standard message.

upvoted 1 times

mojj 1 month, 2 weeks ago

A. Safety Guardrail

Reasoning:
A Safety Guardrail is designed to ensure that a conversational AI system like a chatbot stays within the intended boundaries of its domain, preventing it from generating unsafe or irrelevant responses. In this case, it ensures the chatbot avoids responding to political questions, which is essential for maintaining focus on insurance-related queries and complying with company policies.

upvoted 3 times

Question #24 *Topic 1*

A Generative AI Engineer I using the code below to test setting up a vector store:

```
from databricks.vector_search.client import VectorSearchClient

vsc = VectorSearchClient()

vsc.create_endpoint(
     name= "vector_search_test",
     endpoint_type= "STANDARD"
)
```

Assuming they intend to use Databricks managed embeddings with the default embedding model, what should be the next logical function call?

A. vsc.get_index()

B. vsc.create_delta_sync_index()

C. vsc.create_direct_access_index()

D. vsc.similarity_search()

Suggested Answer: *B*

Community vote distribution

B (100%)

trendy01 1 month ago

Selected Answer: B

If the question assumes "if you want to use the default embedding model managed by Databricks", then B. vsc.create_delta_sync_index() might be a more appropriate answer.

upvoted 1 times

Question #26 *Topic 1*

A Generative AI Engineer wants to build an LLM-based solution to help a restaurant improve its online customer experience with bookings by automatically handling common customer inquiries. The goal of the solution is to minimize escalations to human intervention and phone calls while maintaining a personalized interaction. To design the solution, the Generative AI Engineer needs to define the input data to the LLM and the task it should perform.
Which input/output pair will support their goal?

A. Input: Online chat logs; Output: Group the chat logs by users, followed by summarizing each user's interactions

B. Input: Online chat logs; Output: Buttons that represent choices for booking details

C. Input: Customer reviews; Output: Classify review sentiment

D. Input: Online chat logs; Output: Cancellation options

Suggested Answer: *B*

Community vote distribution

B (100%)

trendy01 1 month ago

Selected Answer: B

B, This option can be useful for automatically handling customer inquiries in a structured way that allows you to provide immediate responses when customers ask questions about their reservations.

upvoted 1 times