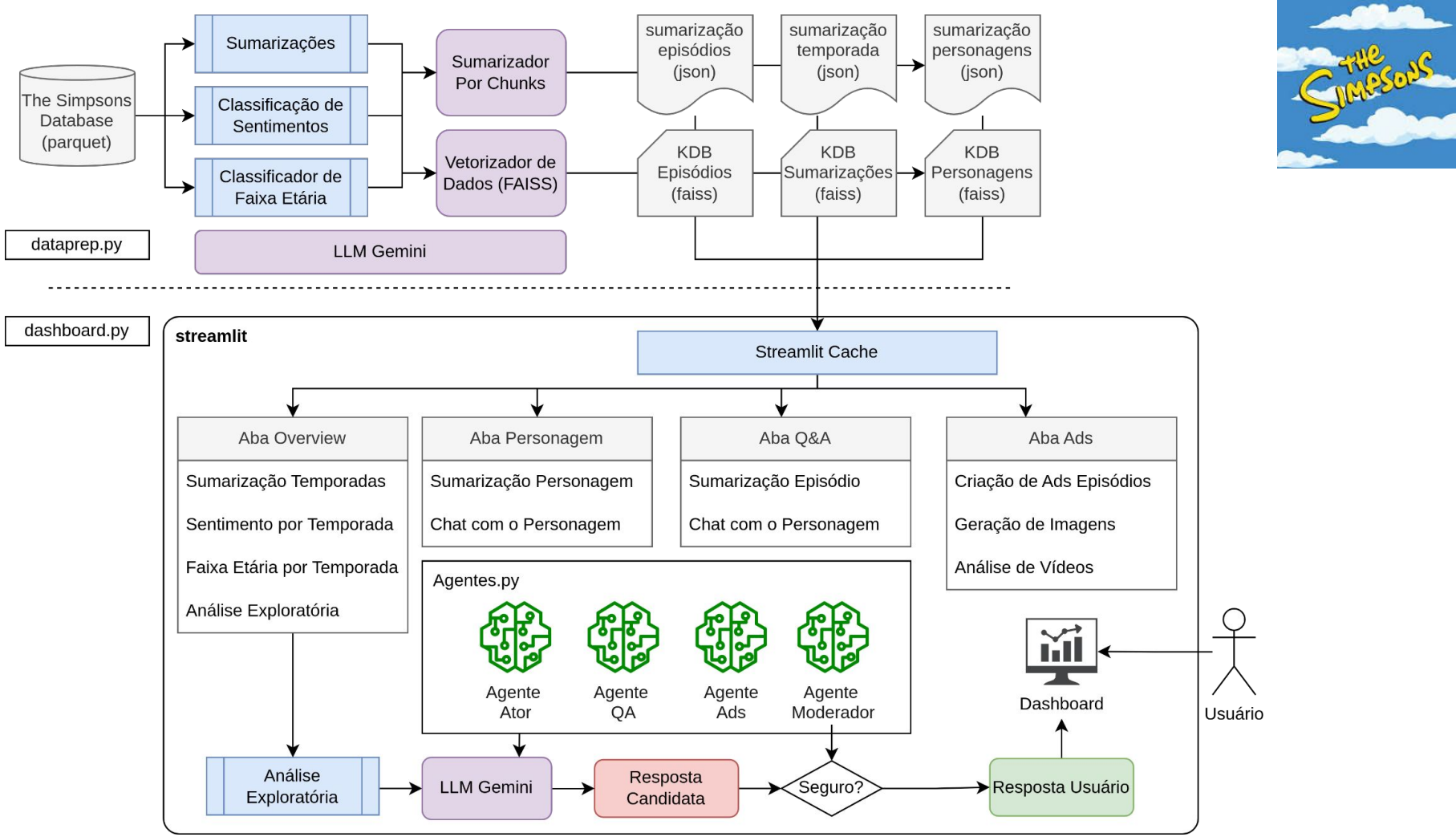
ETAPA 8

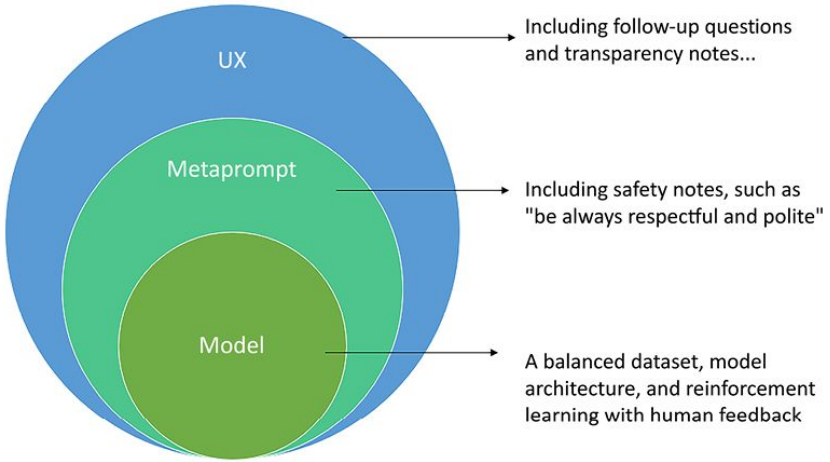
Aplicações práticas de LLM II


The Simpsons Database (parquet)
Sumarizações
Classificação de Sentimentos
Classificador de Faixa Etária
Sumarizador Por Chunks
Vetorizador de Dados (FAISS)
sumarização episódios (json)
sumarização temporada (json)
sumarização personagens (json)
KDB Episódios (faiss)
KDB Sumarizações (faiss)
KDB Personagens (faiss)
dataprep.py
LLM Gemini
dashboard.py
streamlit
Streamlit Cache
Aba Overview
Sumarização Temporadas
Sentimento por Temporada
Faixa Etária por Temporada
Análise Exploratória
Aba Personagem
Sumarização Personagem
Chat com o Personagem
Aba Q&A
Sumarização Episódio
Chat com o Personagem
Aba Ads
Criação de Ads Episódios
Geração de Imagens
Análise de Vídeos
Agentes.py
Agente Ator
Agente QA
Agente Ads
Agente Moderador
Dashboard
Usuário
Análise Exploratória
LLM Gemini
Resposta Candidata
Seguro?
Resposta Usuário

Mitigar vulnerabilidades de injeção de prompts

Responsible AI: envolve o **desenvolvimento** e uso de sistemas de **IA de forma ética e transparente**, garantindo segurança, equidade, e privacidade. Visa **minimizar riscos**, como vieses, falta de transparência e danos sociais, promovendo IA que seja confiável e **centrada em humanos**.

- **Nível de Modelo**: utilização de dados livres de vieses para evitar saídas discriminatórias e ajustes finos para garantir que suas saídas sigam princípios éticos e normas de segurança.
- **Nível de Metaprompt**: definir prompts e regras explícitas para moldar as respostas da IA, promovendo alinhamento com valores éticos e regulatórios.
- **Nível da Interface com o Usuário**: incorporar mecanismos que permitam aos usuários fornecer feedback e ajustar as saídas do modelo em tempo real, garantindo transparência e controle humano.



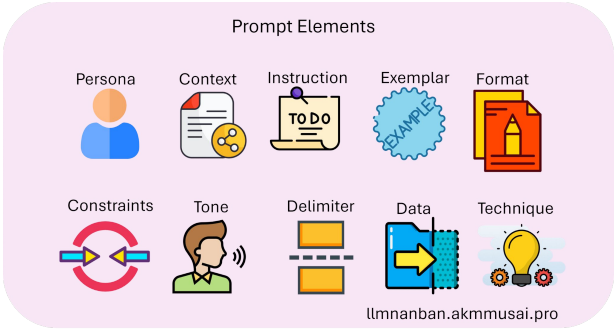
Mitigação de Riscos com Prompt

dataprep.py:

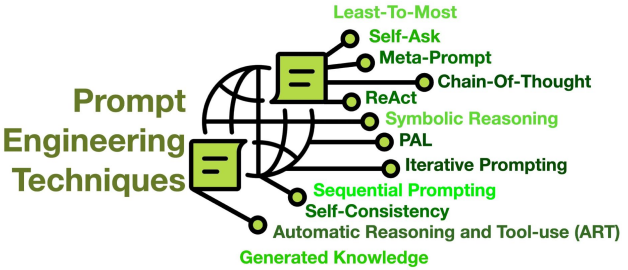
- **Classificador de Faixa Etária com BART-MNLI:** Utilizar técnica LLM de NLI para estimar a faixa etária dos episódios da série. Comparar os termos NLI com os ratings e audiência dos episódios.
- **Análise de sentimentos com Few-shot learning e MNLI:** Implementar análise de sentimentos com MNLI e few-shot learning.
- **Mitigação de riscos de sequestro de prompt:** Utilizar arquivos YAML com os prompts principais.

dashboard.py:

- **Prompt-Chaining para moderação de discurso:** Implementar um Agente Revisor para condicionar as falas ao personagem, em conjunto com o Agente Ator.
- **Prompt-Chaining para moderação de prompts:** Implementar Agente moderador de prompts, adicionando uma camada de segurança às instruções dadas pelo usuário.



12 Prompt Engineering Techniques

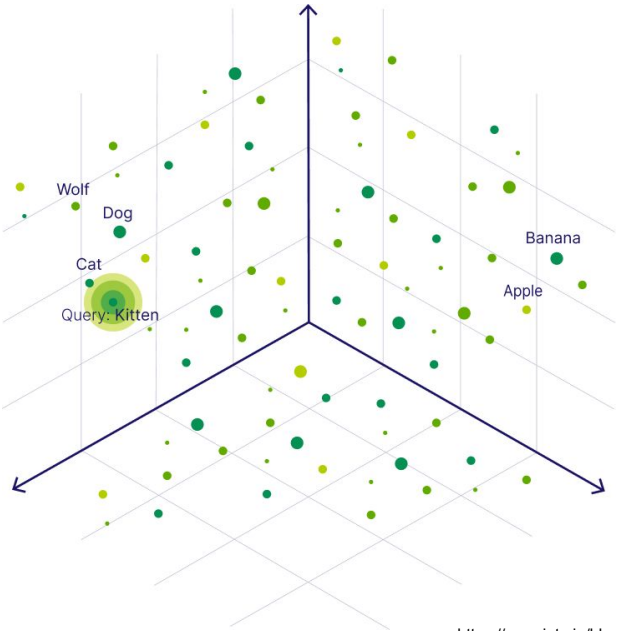


Construir um assistente para perguntas específicas com Gemini e FAISS

O que é e como funciona a FAISS?
FAISS (Facebook AI Similarity Search) é uma biblioteca otimizada para busca de similaridade em vetores. Permite o gerenciamento, indexação e busca eficiente em grandes coleções de vetores (embeddings).

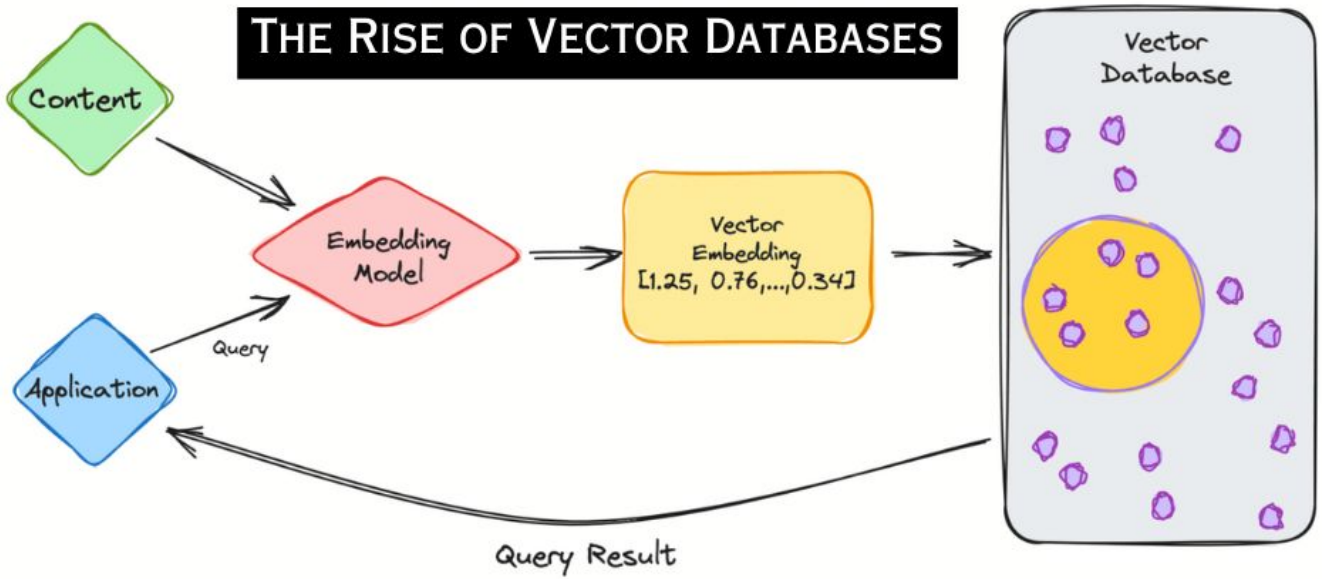
A FAISS suporta algoritmos de busca aproximada de vizinhos mais próximos, essencial para lidar com grandes volumes de dados, com suporte a métricas como distância euclidiana e similaridade de cosseno.

Indexação de vetores usando métodos diversos (quantização, hashing sensível à localidade).

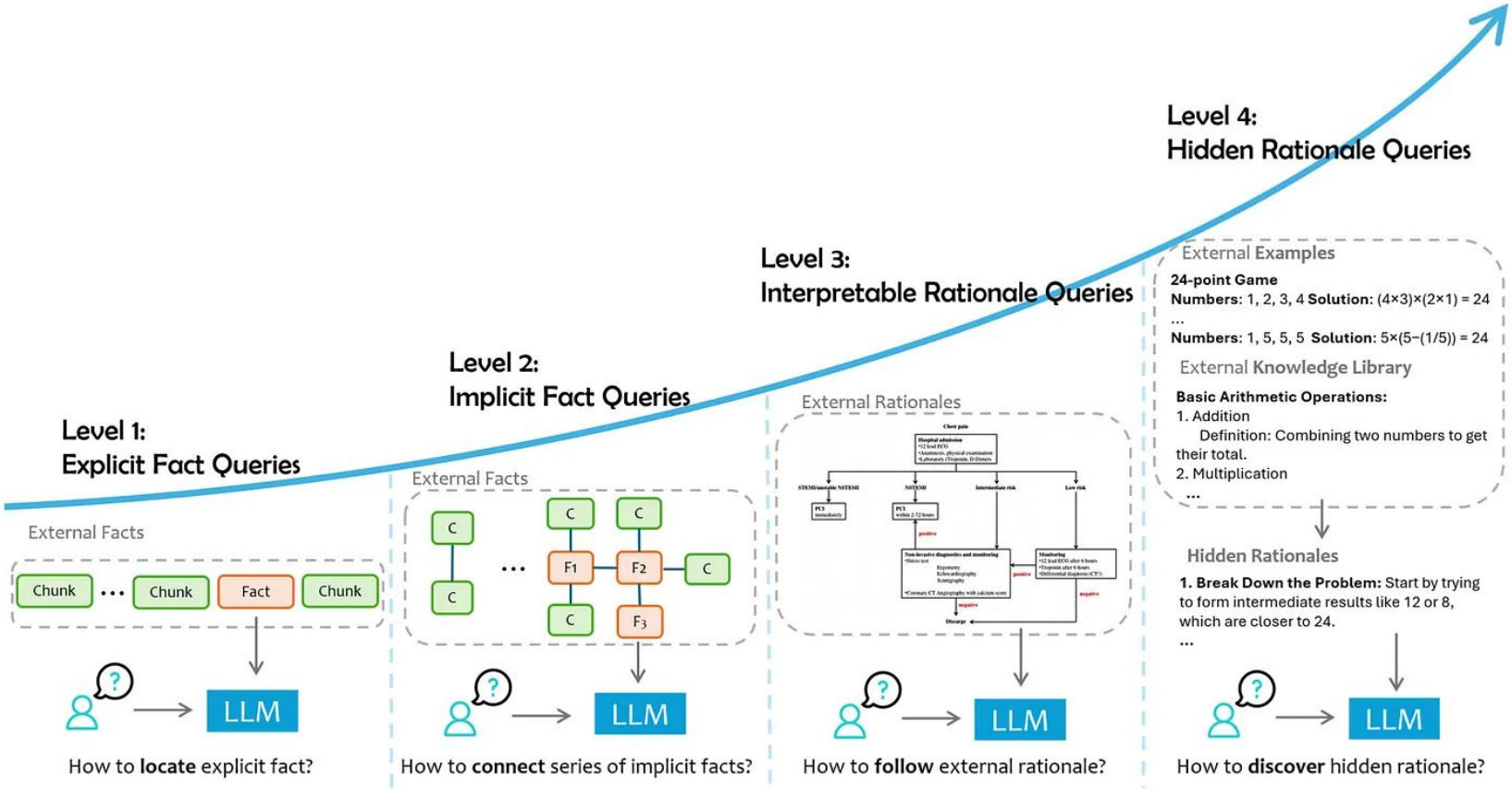


https://weaviate.io/blog/what-is-a-vector-database

Construir um assistente para perguntas específicas com Gemini e FAISS



https://dev.to/pavanbelagatti/wtf-is-a-vector-database-a-beginners-guide-16p



https://arxiv.org/pdf/2409.14924

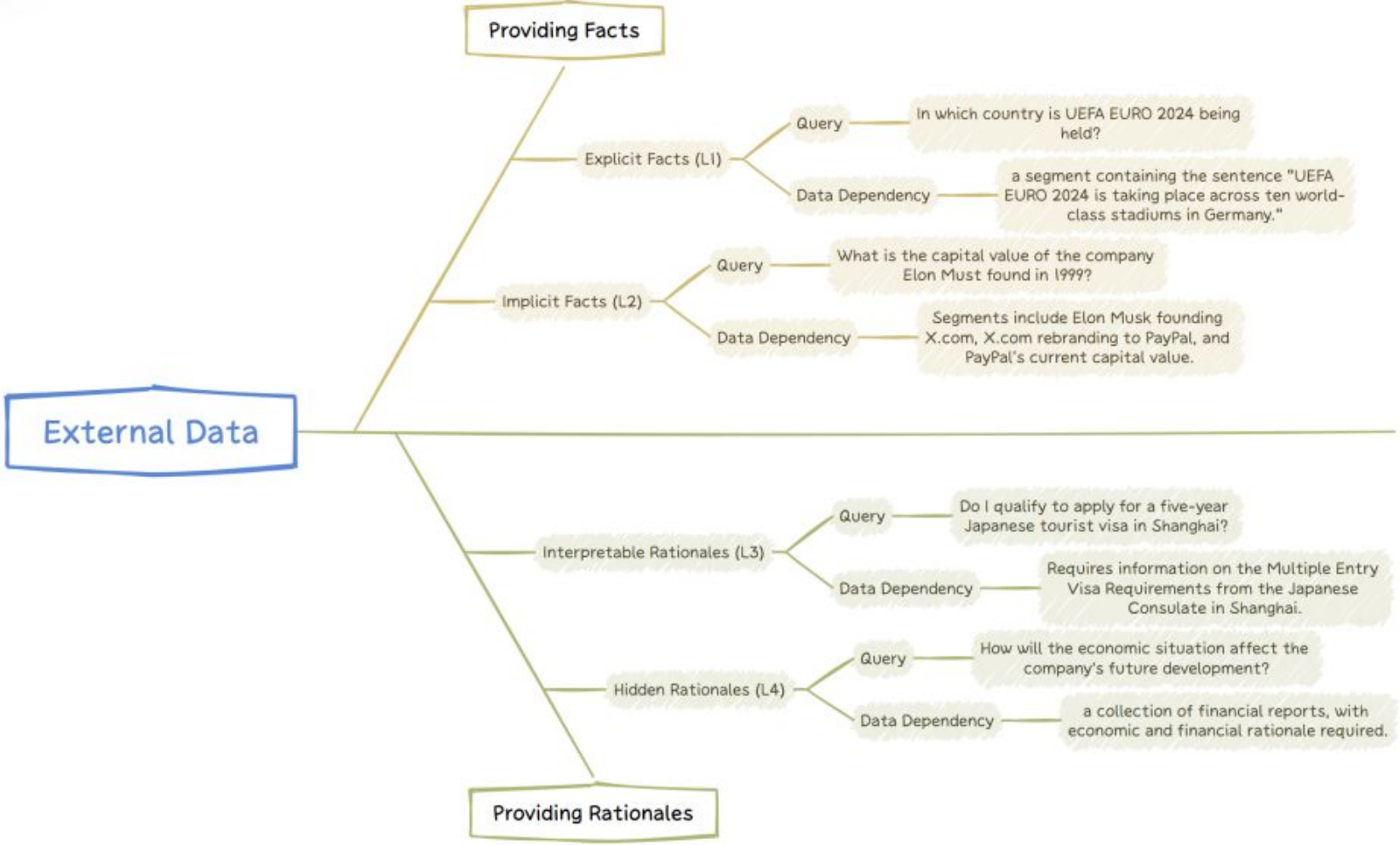


Figure 2: Summary of Query Levels in Data augmented LLM applications

Construir um assistente para perguntas específicas com Gemini e FAISS

Vantagens Técnicas:

Flexibilidade: Escolha entre diferentes tipos de índices para atender a requisitos específicos.

Escalabilidade: Projetada para lidar com bilhões de vetores, oferecendo excelente desempenho para buscas massivas.

Interoperabilidade: Disponível em C++ e Python.

Limitações Intrínsecas da FAISS:

Foco restrito em Indexação e Busca: A FAISS não extrai embeddings diretamente. É necessário usar outras ferramentas ou modelos, como LLMs, para gerar os vetores antes de indexá-los.

Complexidade no Ajuste de Parâmetros: Escolher o índice correto, ajustar os parâmetros e lidar com a compressão de vetores requer conhecimento especializado.

Dependência da Qualidade do Embedding: Se os embeddings fornecidos pelo LLM não capturam bem as relações semânticas, o desempenho da FAISS será prejudicado.

Construir um assistente para perguntas específicas com Gemini e FAISS

Otimização de Prompts para LLMs: Utilizar FAISS para buscar contextos relevantes em bases de dados antes de enviar prompts para um LLM.

- Reduz o custo computacional ao limitar o espaço de busca.
- Melhora a qualidade das respostas ao contextualizar com informações precisas.

Pesquisa Semântica: Respostas precisas baseadas em contextos relevantes.

Sistemas de Recomendação: Identificação de itens semelhantes com base em embeddings.

Assistentes Virtuais: Recuperação de informações em tempo real.

Detecção de Anomalias: Identificação de desvios em dados utilizando vizinhanças no espaço vetorial.

Construir um assistente para perguntas específicas com Gemini

dataprep.py:

- **Vetorização dos dados do show**: aplicar FAISS e/ou redis para implementar uma base de dados para utilização em aplicação tipo RAG.

dashboard.py:

- **Aplicação de KDB para Otimizar Prompts:** Interface para QA com um assistente especialista em Simpsons.
- **Assistente virtual The Simpsons:** Implementar um Agente para alteração do prompt, em estilo ReAct, capaz de buscar em diferentes bases de dados vetoriais.