Perbandingan penggunaan qdrant dan sql

C14220261

Calvin Laguna

Codingan:

Qdrant:

https://github.com/petra-magang-ptik/menu-kantin/blob/main/petranesian-lapar/qdrant2.py

1. Buka website qdrant kemudian login atau buat akun lalu copy qdrant api dan qdrant url

   https://login.cloud.qdrant.io/u/login/identifier?state=hKFo2SBkbTZYdkFoR0hpVjZhT2pZTGN3cmgtd0Zza01jbzdHV6Fur3VuaXZlcnNhbC1sb2dpbqN0aWTZIHNjaUtOWXllNzZTYmsyV2RYRkRxQzNxNGltc2pVUU5Io2NpZNkgckkxd2NPUEhPTWRlSHVUeDR4MWtGMEtGZFE3d25lemc

2. Install requirements
   - llama-index
   - llama-index-embeddings-ollama
   - llama-index-llms-ollama
   - llama-index-retrievers-bm25
   - llama-index-vector-stores-qdrant
   - qdrant-client
   - streamlit
   - pandas
   - nest_asyncio

3. Impor Library yang penting untuk Qdrant:
   - qdrant_client: Client untuk berinteraksi dengan Qdrant, sebuah vector database.
   - re: Library untuk operasi regular expression.
   - requests: Library untuk melakukan HTTP requests.
   - json: Library untuk manipulasi data JSON.

4. Masukkan qdrant api dan url untuk mengakses collection

   ```
   QDRANT_URL = "qdrant url"
   QDRANT_API_KEY = "qdrant api"

   qdrant_client = qdrant_client.QdrantClient(url=QDRANT_URL,
   api_key=QDRANT_API_KEY)
   ```

5. Memasukan data ke dalam qdrant

https://github.com/petra-magang-ptik/menu-kantin/blob/main/petranesian-lapar/uploadqdrant.py

untuk memasukan data ke qdrant pastikan ada data csv

6. Memastikan data qdrant ada

```
collection_name = "kantin_menu"
def ensure_collection():
    collections = qdrant_client.get_collections()
    if collection_name not in [col.name for col in collections.collections]:
        qdrant_client.create_collection(
            collection_name=collection_name,
            vectors_config=VectorParams(size=768, distance="Cosine")
        )
        print(f"Collection '{collection_name}' created successfully.")
ensure_collection()
```

7.
```
vector_store = QdrantVectorStore(client=qdrant_client,
collection_name=collection_name)
storage_context = StorageContext.from_defaults(vector_store=vector_store)
```

vector_store: Tempat menyimpan vektor hasil embedding.

```
embed_model = OllamaEmbedding(base_url="http://127.0.0.1:11434",
model_name="nomic-embed-text:latest")
```

embed_model: Model OllamaEmbedding untuk mengubah teks menjadi vektor.

8.
```
system_prompt = """
Anda adalah pelayan kantin yang ramah yang dapat mengarahkan pengguna mencari makanan/minuman yang tepat.
Anda tidak perlu menyebutkan atau membuat pernyataan yang mengatakan Anda tidak dapat menampilkan gambar jika gambar berhasil ditemukan.
Tugas Anda adalah untuk menjawab dengan relevansi sesuai menu dan menyarankan gambar jika sesuai tidak perlu memberikan path dari gambar.
Jawablah semua dalam Bahasa Indonesia.
Tugas Anda adalah untuk menjadi pelayan kantin yang ramah yang dapat mengarahkan user.
Kantin yang Anda layani adalah kantin kampus Universitas Kristen Petra Surabaya.
Pada Universitas Kristen Petra terdapat 2 gedung utama yang setiap gedungnya memiliki kantin, yaitu Gedung P dan W.
"""
```

System prompt: menentukan karakter chatbot sebagai pelayan kantin yang membantu pengguna dalam mencari makanan/minuman.

9. Untuk menggunakan Gemini sebagai model

   Masukan url dan api key

```
API_KEY = "API KEY Gemini"
API_URL = f"Url Gemini”
```

   Fungsi untuk mendapatkan respons dari Gemini:

```
def get_gemini_response(user_input):
    headers = {"Content-Type": "application/json"}
    data = {
        "contents": [{"parts": [{"text": system_prompt + "\n" + user_input}]}]
    }
    response = requests.post(API_URL, headers=headers, data=json.dumps(data))
    if response.status_code == 200:
        response_json = response.json()
        return response_json["candidates"][0]["content"]["parts"][0]["text"]
    else:
        return "Error: Unable to get a response."
```

10. Mencari Menu di Qdrant

```
def find_menu(user_input, filter_price=None, stall_name=None):
    query_vector = embed_model.get_text_embedding(user_input)

    results = qdrant_client.search(
        collection_name=collection_name,
        query_vector=query_vector,
        limit=100
    )
```

SQL:

https://github.com/petra-magang-ptik/menu-kantin/blob/main/petranesian-lapar/SQL.py

1. Install requirements
   - Pandas
   - Fuzzywuzzy
   - Mysql-connector-python

2. Menyiapkan data SQL

   Masukan data sql
   https://www.freesqldatabase.com/

3. nest_asyncio.apply()
   splitter = SentenceSplitter(chunk_size=512)
   logging.basicConfig(stream=sys.stdout, level=logging.WARNING)

   SentenceSplitter: Digunakan untuk membagi teks dokumen menjadi chunk (potongan) dengan ukuran 512 karakter.

4. Mengambil data dari database

```
def fetch_data_from_db():
    conn = mysql.connector.connect(...)
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM kantin")
    rows = cursor.fetchall()
    cursor.close()
    conn.close()
    return rows
```

5. Karena harga dalam sql bersifat decimal maka harus diubah ke float

```
def convert_decimals(obj):
    if isinstance(obj, Decimal):
        return float(obj)
```

6. Fungsi pencarian gambar

```
def find_image(user_input):
    rows = fetch_data_from_db()
    image_data = []
    for row in rows:
        similarity = fuzz.partial_ratio(user_input.lower(), row['Nama_Produk'].lower())
        if similarity >= 75:
            image_path = row['Gambar']
            if isinstance(image_path, str) and os.path.exists(image_path):
                image_data.append((image_path, row['Nama_Produk']))
    return image_data if image_data else None
```

   Fungsi ini mencari gambar produk berdasarkan input menggunakan FuzzyWuzzy. Jika similarity >= 75%, gambar akan ditampilkan.

Keuntungan dan Kekurangan dari Qdrant dan SQL

Qdrant:

Keuntungan:

- Dirancang khusus untuk menyimpan dan mencari vektor dengan efisien
- Performanya lebih cepat pencariannya
- Mendukung pencarian embedding
- Mudah diintegrasikan dengan model AI

Kekurangan:

- Data tidak selalu konsisten
- Pencarian harus spesifik
- Kurang cocok untuk data terstruktur

SQL:

Keuntungan:

- Sangat cocok untuk data terstruktur dan query kompleks.
- Menjamin konsistensi
- Lebih bebas dalam pencarian

Kekurangan:

- Performanya lambat
- Tidak dioptimalkan untuk pencarian vector
- Kurang akurat karena menggunakan FuzzyWuzzy