



دانشکده مهندسی کامپیوتر

پروژه دوم

درس پردازش زبان طبیعی

پوریا رحیمی – 99521289

مهدی قضاوی – 99522014

نیم سال دوم 1402-1403

RAG چیست؟

تکنولوژی Retrieval-Augmented Generation (RAG) یک چارچوب هوش مصنوعی است که از پایگاه دانش خارجی برای بهبود کیفیت پاسخ‌های تولید شده توسط مدل‌های زبان بزرگ (LLM) استفاده می‌کند. حالا بیایید به این سوالات پاسخ بدهیم تا بهتر با عملکرد آن آشنا شویم:

اهمیت استفاده از تکنولوژی Retrieval-Augmented-Generation (RAG) در

پاسخ‌دهی به سوالات مبتنی بر گزارش‌های سازمانی

دقت و مرتبط بودن بیشتر

RAG با ترکیب روش‌های بازیابی اطلاعات و مدل‌های تولیدی، دقت و مرتبط بودن پاسخ‌های تولیدی را افزایش می‌دهد. در زمینه گزارش‌های سازمانی، این رویکرد ترکیبی تضمین می‌کند که پاسخ‌ها نه تنها به صورت مناسب و مرتبط با زمینه باشند، بلکه از نظر واقعی نیز صحیح باشند. با بازیابی اطلاعات خاص از گزارش‌های موجود، RAG ریسک تولید اطلاعات نادرست یا غیرمرتبط را به حداقل می‌رساند که این امر برای حفظ یکپارچگی تصمیم‌گیری‌های کسب‌وکار حیاتی است.

بهره‌وری بهتر از دانش

گزارش‌های سازمانی اغلب حاوی حجم زیادی از داده‌ها و بینش‌ها هستند. مدل‌های تولیدی سنتی ممکن است در بهره‌وری موثر از این اطلاعات دچار مشکل شوند به دلیل محدودیت‌های موجود در پنجره متنی و ناتوانی در به خاطر سپردن جزئیات خاص. RAG با استفاده از مکانیزم‌های بازیابی، به اطلاعات دقیق و خاص از این گزارش‌ها دسترسی پیدا می‌کند و امکان تولید پاسخ‌هایی که به خوبی با داده‌های تاریخی و دانش پایه سازمانی آگاه شده‌اند را فراهم می‌کند.

کارآیی در دسترسی به اطلاعات

جستجوی دستی در میان گزارش‌های گسترده سازمانی می‌تواند زمان‌بر و پرخ‌خطا باشد. RAG این فرآیند را خودکار می‌کند و امکان دسترسی سریع به اطلاعات مرتبط را فراهم می‌کند و بار شناختی کارکنان را کاهش می‌دهد. این کارآیی به ویژه در سناریوهایی که اطلاعات به موقع و دقیق حیاتی هستند، مانند برنامه‌ریزی استراتژیک، گزارش‌های انطباقی و فرآیندهای تصمیم‌گیری، مفید است.

درک و پیوستگی متنی

مدل‌های RAG پیوستگی متنی را بهتر از مدل‌های بازیابی یا تولیدی مستقل حفظ می‌کنند. با ادغام اطلاعات بازیابی شده در فرآیند تولید، سیستم‌های RAG می‌توانند پاسخ‌هایی تولید کنند که منسجم و با متن در جریان هماهنگ باشند، که برای حفظ جریان منطقی در ارتباطات کسب‌وکار و بحث‌های مبتنی بر گزارش‌ها ضروری است.

اهداف پروژه پیاده‌سازی RAG برای تولید متن بر اساس گزارش‌های انسانی چه می‌باشند؟ در ادامه به چند مورد می‌پردازیم :

بازیابی دقیق اطلاعات

یکی از اهداف اصلی، توسعه سیستمی است که قادر به بازیابی دقیق بخش‌های مرتبط از گزارش‌های گسترده انسانی باشد. این شامل بهینه‌سازی الگوریتم‌های بازیابی برای شناسایی و اولویت‌بندی اطلاعات مربوطه است تا اطمینان حاصل شود که بخش تولیدی به با کیفیت‌ترین داده‌ها برای تولید پاسخ‌ها دسترسی دارد.

پاسخ‌های منسجم و متناسب با زمینه

جنبه تولیدی سیستم RAG هدفش تولید متنی است که نه تنها از نظر واقعی دقیق است بلکه منسجم و متناسب با زمینه نیز باشد. این شامل آموزش مدل تولیدی برای ادغام بدون نقص اطلاعات بازایابی شده است، به طوری که خروجی همواره با زمینه و هدف اصلی گزارش‌های انسانی هماهنگ باشد.

مقیاس‌پذیری و تطبیق‌پذیری

پروژه هدفش توسعه سیستمی مقیاس‌پذیر و تطبیق‌پذیر است که قادر به مدیریت گزارش‌هایی با طول و پیچیدگی‌های مختلف در حوزه‌های گوناگون باشد. این نیازمند آن است که سیستم RAG به اندازه کافی چندمنظوره باشد تا بتواند به انواع مختلف گزارش‌ها اعمال شود، چه اسناد مالی، گزارش‌های فنی یا برنامه‌های استراتژیک.

بهبود تعامل کاربر

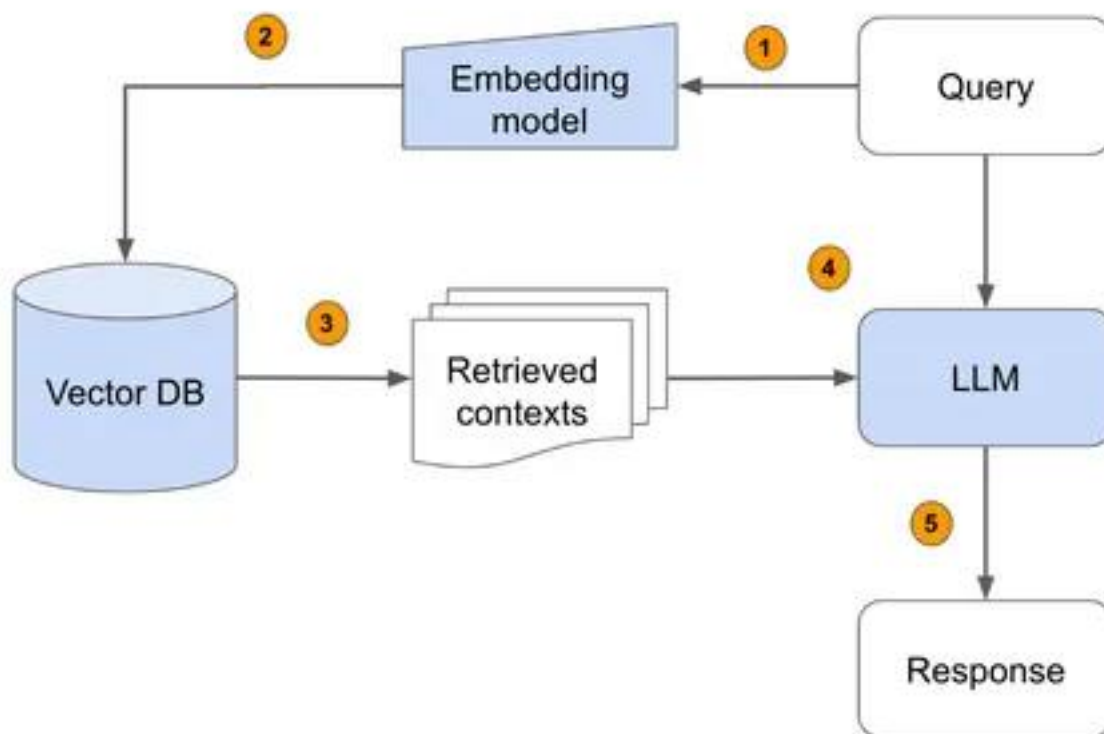
با ارائه اطلاعات دقیق و مرتبط به سرعت، سیستم RAG تعامل کاربر با پایگاه‌های دانش سازمانی را بهبود می‌بخشد. کاربران می‌توانند به صورت زبان طبیعی از سیستم سوال بپرسند و پاسخ‌های دقیق و مرتبط با زمینه دریافت کنند، که تجربه کلی کاربر و بهره‌وری را بهبود می‌بخشد.

کاهش تلاش دستی

پیاده‌سازی RAG نیاز به تلاش دستی در جستجو و ترکیب اطلاعات از گزارش‌ها را کاهش می‌دهد. این نه تنها زمان را صرفه‌جویی می‌کند بلکه به کارکنان اجازه می‌دهد بر روی وظایف

استراتژیک‌تر تمرکز کنند و سیستم RAG را برای مدیریت بازیابی و ترکیب اطلاعات روتین به کار ببرند.

با پیاده‌سازی RAG برای تولید متن بر اساس گزارش‌های سازمانی، کسب‌وکارها می‌توانند فرآیندهای مدیریت اطلاعات خود را به طور قابل توجهی بهبود دهند و منجر به تصمیم‌گیری‌های آگاهانه‌تر و عملیات‌های کارآمدتر شوند.



در پروژه‌های مربوط به تولید متن با استفاده از تکنولوژی‌های بازیابی-افزوده (RAG)، از مدل‌های زبانی پیشرفته و سیستم‌های بازیابی اطلاعات ترکیبی استفاده می‌شود. برخی از مدل‌های زبانی کلیدی شامل LLaMA3 و LaBSE هستند که در ادامه به توضیح آنها و نحوه ترکیب‌شان با سیستم‌های بازیابی اطلاعات می‌پردازیم.

LLaMA3 (Large Language Model Assisted Augmentation)

LLaMA3 یک مدل زبان بزرگ پیشرفته است که برای تولید متن با دقت و پیچیدگی بالا طراحی شده است. این مدل بهبودهای قابل توجهی در پردازش زبان طبیعی (NLP) دارد و به خصوص در تولید متنی که هم از نظر زبانی صحیح و هم از نظر مفهومی دقیق باشد، عملکرد عالی دارد.

ویژگی‌ها و کاربردها:

1. تولید متن پیشرفته: LLaMA3 قادر به تولید متن‌های پیچیده و غنی از نظر زبانی است که برای کاربردهای مختلف مانند نوشتن مقالات، پاسخ‌دهی به سوالات و تولید محتوای خلاقانه مفید است.
2. درک عمیق‌تر زبان: این مدل می‌تواند مفاهیم پیچیده را درک کند و پاسخ‌های مرتبط و دقیق تولید کند که برای استفاده در گزارش‌های سازمانی و تجاری بسیار مناسب است.
3. یکپارچگی با سیستم‌های بازیابی: LLaMA3 می‌تواند با سیستم‌های بازیابی اطلاعات ترکیب شود تا اطلاعات دقیق و مرتبط از داده‌های موجود بازیابی و در تولید متن استفاده شود.

LaBSE (Language-agnostic BERT Sentence Embedding)

LaBSE مدلی است که برای ایجاد تعبیه‌های جملاتی بدون وابستگی به زبان طراحی شده است. این مدل از تکنیک‌های BERT (Bidirectional Encoder Representations from Transformers) استفاده می‌کند تا نمایه‌هایی غنی و کاربردی از جملات در زبان‌های مختلف تولید کند.

ویژگی‌ها و کاربردها:

1. نمایه‌سازی چندزبانه: LaBSE قابلیت تولید تعبیه‌های جملاتی برای زبان‌های مختلف را دارد که این امکان را می‌دهد تا در پروژه‌های بین‌المللی و چندزبانه مورد استفاده قرار گیرد.
2. بازیابی اطلاعات دقیق: با استفاده از تعبیه‌های جملاتی، LaBSE می‌تواند به دقت اطلاعات مرتبط با سوالات را از میان اسناد و گزارش‌ها بازیابی کند.
3. تطابق جملات و مفاهیم: این مدل می‌تواند جملات و مفاهیم مرتبط را در اسناد پیدا کند که این امر برای بازیابی اطلاعات دقیق و مرتبط بسیار مفید است.

ترکیب مدل‌های زبانی با سیستم‌های بازیابی اطلاعات

سیستم‌های بازیابی اطلاعات

سیستم‌های بازیابی اطلاعات (Information Retrieval Systems) نقش حیاتی در فرآیند RAG دارند. این سیستم‌ها شامل موتورهای جستجو و الگوریتم‌های پیچیده‌ای هستند که برای یافتن و بازیابی اطلاعات دقیق و مرتبط از میان داده‌های موجود طراحی شده‌اند.

مراحل ترکیب:

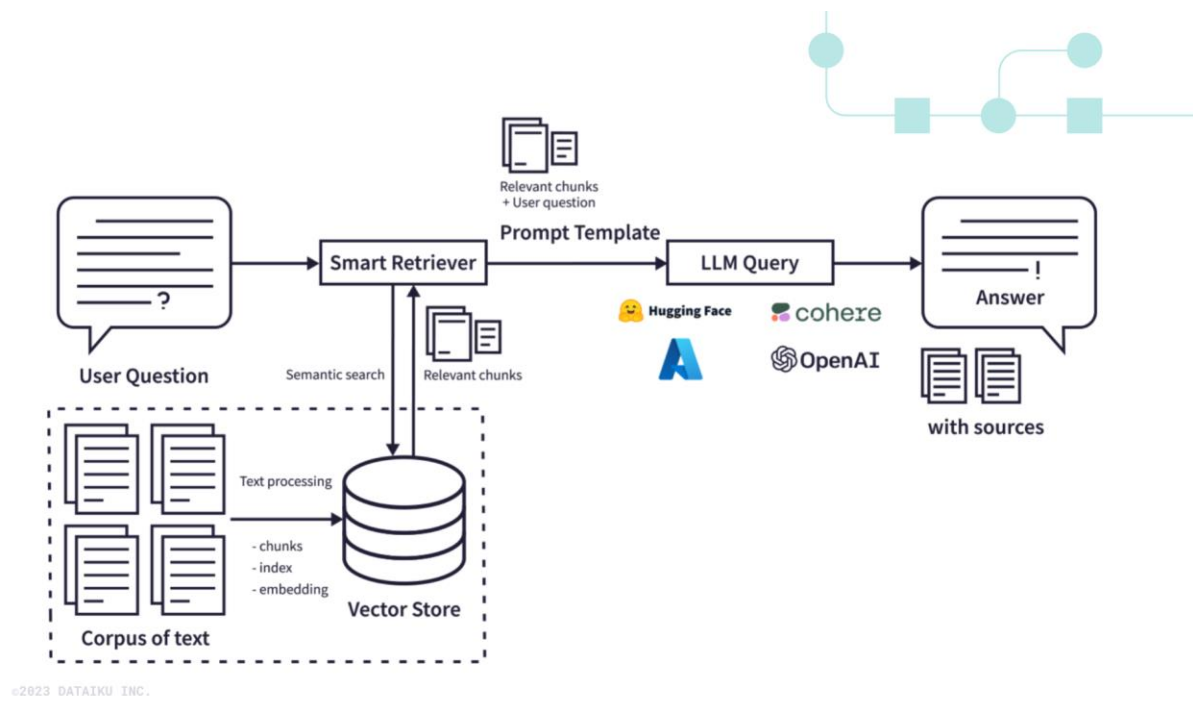
1. پیش‌پردازش داده‌ها: در این مرحله، داده‌های موجود در گزارش‌ها و اسناد سازمانی پردازش و به فرمت‌های قابل استفاده توسط سیستم‌های بازیابی و مدل‌های زبانی تبدیل می‌شوند.
2. ایندکس‌گذاری و نمایه‌سازی: اسناد و گزارش‌ها به صورت ایندکس‌گذاری شده در پایگاه داده ذخیره می‌شوند. LaBSE در این مرحله به ایجاد نمایه‌های جملاتی کمک می‌کند که اطلاعات را به صورت غنی و قابل جستجو در می‌آورد.

3. بازیابی اطلاعات: هنگام پرسش کاربر، سیستم بازیابی اطلاعات با استفاده از نمایه‌های جملاتی و الگوریتم‌های جستجو، مرتبط‌ترین بخش‌های اسناد را پیدا می‌کند.

4. تولید متن: LLaMA3 با استفاده از اطلاعات بازیابی شده توسط سیستم بازیابی، متنی دقیق و مرتبط با سوال تولید می‌کند. این متن می‌تواند شامل اطلاعات دقیق از گزارش‌ها باشد که با زبان طبیعی و منسجم تولید شده است.

5. پس‌پردازش و بهینه‌سازی: متن تولید شده توسط LLaMA3 بررسی و بهینه‌سازی می‌شود تا اطمینان حاصل شود که هم از نظر زبانی صحیح و هم از نظر محتوایی دقیق است.

در نتیجه ترکیب مدل‌های زبانی پیشرفته مانند LLaMA3 و LaBSE با سیستم‌های بازیابی اطلاعات، امکان تولید متن‌های دقیق، مرتبط و با کیفیت را فراهم می‌کند. این ترکیب به بهره‌وری بهتر از داده‌های سازمانی کمک می‌کند و فرآیند پاسخ‌دهی به سوالات و تولید گزارش‌ها را بهبود می‌بخشد. این تکنولوژی‌ها نقش مهمی در ارتقای دقت و کارایی در مدیریت اطلاعات و تصمیم‌گیری‌های کسب‌وکار ایفا می‌کنند.



گزارش کلی از روند اجرای پروژه :

پیاده‌سازی پروژه و خروجی گرفتن از مدل اجرا شده در نوت‌بوک NLP_Final_Project_2.ipynb در فایل‌های این پروژه، الصاق شده‌است. این کد به طور کلی دو کار اصلی انجام می‌دهد:

- استخراج سوالات از فایل‌های Word و Excel و ترکیب آن‌ها
- استفاده از مدل‌های پردازش زبان طبیعی برای پاسخ‌گویی به سوالات به زبان فارسی

در ابتدا برای پیاده‌سازی این کد باید کتابخانه‌های مورد نیاز برای کار با فایل‌های متنی و مدل‌های پردازش زبان طبیعی را نصب کنیم سپس پس از نصب کردن آن‌ها باید کتابخانه‌ها و توابع مورد نیاز برای باز کردن فایل‌ها را پیاده‌سازی کنیم تا بتوانیم با استفاده از داده‌هایی که از آن فایل‌ها استخراج می‌کنیم بقیه عملیات را انجام دهیم در نتیجه شروع به پیاده‌سازی توابع می‌کنیم :

این تابع سوالات را از فایل‌های Word استخراج می‌کند:

```
[6] def extract_questions_from_docx(docx_file):
    doc = DocxDocument(docx_file)
    questions = []
    for table in doc.tables:
        for row in table.rows[1:]:
            question = row.cells[1].text.strip()
            # print(question)
            if question:
                questions.append(question)
    return questions
```

این تابع سوالات را از فایل‌های Excel استخراج می‌کند:

```
[7] def extract_questions_from_xlsx(xlsx_file, column_name):
    df = pd.read_excel(xlsx_file, header=1) # Skip the first row by setting header=1
    print("Columns in the XLSX file:", df.columns.tolist()) # Debugging step to print column names
    questions = df[column_name].dropna().tolist() # Drop NaN values and convert to list
    return questions
```

در این بخش دیتاها را از فایل docx و xlsx استخراج کرده و آن ها را با هم marge می کنیم و

کل سوالات را در یک فایل CSV ذخیره

می کنیم :

```
# Combine questions
all_questions = questions_docx + questions_xlsx

# Create a DataFrame
questions_df = pd.DataFrame(all_questions, columns=['Questions'])

# Display the combined questions
print(questions_df)

# Save the DataFrame to a CSV file if needed
questions_df.to_csv('combined_questions.csv', index=False)
```

```
Questions
0    ...باسلام اینجانب دارای 300 زمین مشاع در پلاک ثبت
1    ...با سلام احترام خدمت شما من سال پیش در منطقه ۱۴
2    ...چگونه می توانم از صورتجلسات کمیسیون ماه 5 پرین
3    ...باسلام اینجانب زمین فریز شده در منطقه ۱۴ فریز
4    ...با سلام مجدد بنده از همکاران شهرداری میباشم قب
..
135  ...احداث پنجره در نورگیرهایی که مجاور حیاط همسایه
136  ...آیا املاکی که دارای رای کیسیون ماده صد میباشد
137  ... ( آیا امکان ایجاد یک واحد تجاری در واحد مسکونی
138  برای قطع درختان خشک به چه نحوی باید عمل کرد ؟
139  ...نحوه واگذاری گنر فی مابین پلاک های تجمعی چگونه
[140 rows x 1 columns]
```

این تابع محتوای فایل های Word را

می خواند:

```
[10] def read_docx(file_path):
      doc = DocxDocument(file_path)
      return "\n".join([para.text for para in doc.paragraphs])
```

آماده سازی و ایجاد مدل LaBSE برای استخراج جانمایی از متن ها:

```
# Create embeddings using LaBSE
from llama_index.embeddings.huggingface import HuggingFaceEmbedding
embed_model = HuggingFaceEmbedding(model_name="sentence-transformers/LaBSE")
```

در این بخش به تنظیم مدل ها پرداخته و مدل LLaMA-3 را لود می کنیم :

```
# Load LLaMA-3 model
from transformers import AutoTokenizer
from llama_index.llms.huggingface import HuggingFaceLLM

tokenizer = AutoTokenizer.from_pretrained("NousResearch/Meta-Llama-3-8B-Instruct")
stopping_ids = [tokenizer.eos_token_id, tokenizer.convert_tokens_to_ids("<|eot_id|>")]

llm = HuggingFaceLLM(
    model_name="NousResearch/Meta-Llama-3-8B-Instruct",
    generate_kwargs={
        "do_sample": True,
        "temperature": 0.6,
        "top_p": 0.9,
    },
    tokenizer_name="NousResearch/Meta-Llama-3-8B-Instruct",
    stopping_ids=stopping_ids,
    model_kwargs={"torch_dtype": torch.bfloat16, "load_in_4bit": True},
    # model_kwargs={"torch_dtype": 'auto'},
    device_map='auto',
)

Settings.llm = llm
```

سپس این تابع نیز برای پاسخگویی به سوالات طراحی شده پیاده سازی می شود:

```
# Function to get answers for questions
def get_answers(questions):
    answers = {}
    for question in questions:
        docs = query_engine.query(question)
        retrieved_documents = " ".join([doc.text for doc in docs.source_nodes])

        input_text = f'سوال: {question}\nمعنا: {retrieved_documents}\nپاسخ:'
        response = llm.complete(input_text)
        answers[question] = response
    return answers
```

سپس مدل را اجرا می کنیم و در این قسمت شروع به پرسش و پاسخ می کنیم و نتایج بدست آمده را در یک فایل اکسل ذخیره می کنیم تا سوال ها و پاسخ هایی که داده شده را نمایش دهیم:

```
docs = query_engine.query(questions[0])
retrieved_documents = " ".join([doc.text for doc in docs.source_nodes])

input_text = f'سوال: {questions[0]}\nمعنا: {retrieved_documents}\nپاسخ:'
response = llm.complete(input_text)
```

در کل الگوریتم‌ها و مدل‌های استفاده شده موارد زیر می باشند :

استخراج داده‌ها:

از کتابخانه‌های docx2txt و pandas برای استخراج سوالات از فایل‌های Word و Excel استفاده می‌شود.

مدل‌های برداری و جستجو:

مدل LaBSE (Language-agnostic BERT Sentence Embedding) برای ایجاد تعبیه‌های برداری استفاده می‌شود.

از VectorStoreIndex برای ذخیره و جستجوی برداری استفاده می‌شود.

مدل پردازش زبان طبیعی:

مدل Meta-Llama-3-8B-Instruct از HuggingFace برای تولید پاسخ‌ها به سوالات استفاده می‌شود.

این مدل‌ها با استفاده از داده‌های استخراج‌شده از فایل‌ها و تعبیه‌های برداری به سوالات پاسخ می‌دهند.

نتایج و بررسی‌ها:

پاسخگویی سیستم برای ۴۰ سوال از مجموعه سوالات را در فایل ModelGenerations.xlsx ذخیره کرده‌ایم. نمونه از این سوال و جواب‌ها به شرح زیر است:

متن سوال: باسلام اینجانب زمین فریز شده در منطقه ۱۴ فریز شده بالای ۵ سال با کاربری فضای سبز با مراجعه ما به شهرسازی منطقه برای قانون جدید شیوه نامه ساختمانی به ما میگویند شیوه

نامه لغو گردیده اما بنده در سایت هیچ موضوعی درباره لغو شیوه نامه ساختمانی مشاهده نکردم
تقاضا دارم راهنمایی کنید که بنده چه طور پیگیری کنم شماره پرونده ۸۹۲۵۴

پاسخ تولید شده توسط مدل: با سلام، در ابتدا باید توضیح داد که ماده 14 قانون زمین شهری مصوب 1398/04/03 به اصلاحی شده است و در واقع، حذف شده است. در واقع، این ماده قبلاً در مورد ساخت و ساز در فضاهای سبز عمومی و اراضی کشاورزی، مقرراتی برای مالکان و شهرداری تهران وضع می کرد. اما در سال 1398، این ماده اصلاح شد و در نتیجه، ساخت و ساز در فضاهای سبز عمومی و اراضی کشاورزی، مجاز نیست.

حال، در مورد پرونده شما، شماره ۸۹۲۵۴، باید گفت که شما به عنوان مالک زمین فریز شده در منطقه ۱۴ فریز شده، با کاربری فضای سبز، می خواهید بدانید که چه طور پیگیری کنید.

ابتدا، باید به شهرداری منطقه ۱۴ فریز شده، مراجعه کنید و از آنها بخواهید که در مورد وضعیت پرونده شما، توضیح دهند. در صورت نیاز، باید درخواستی کتبی و محضری برای واگذاری قانونی 70 درصد وسعت عرصه باغ به شهرداری تهران، ارائه دهید.

بررسی: همان گونه که پیداست، پاسخ مدل، تاحد بسیار قابل قبولی، دارای مفهومی مرتب با سوال کاربر بوده و اطلاعات قابل درک و مفیدی را در پاسخ به سوال کاربر بیان کرده است.

موانع و چالش ها:

بزرگ ترین چالشی که در حین پیاده سازی پروژه با آن مواجه شدیم، محدودیت دسترسی به منابع محاسباتی مورد نیاز برای استفاده از چنین مدل های سنگینی مثل LLaMa-3 بود. این مدل ها به علت تعداد وزن های زیادی که دارند، به راحتی بر روی محیط هایی از جمله Google Colab و یا حتی Kaggle با تعداد GPU بیش تر قابل لود و اجرا نیستند، به همین سبب، مجبور شدیم هنگام لود کردن مدل در فضای نوت بوک برای خروجی گرفتن و استفاده از آن، آن را به صورت ۴ بیتی و کوآنتیزه لود کنیم.

مدل‌های زبانی بزرگ، خود بر روی حجم وسیعی از داده‌های آموزشی، Train شده‌اند و حتی با وجود یک Corpora کوچک مثل همین پروژه، قادر به تولید متن‌های با مفهوم بالا هستند، هرچند اگر حجم دیتایی که مدل IR از آن، داده‌های مرتبط را استخراج می‌کند بیش‌تر و مشابه‌تر با سوالات و موضوع آن‌ها باشد، مدل تولید متن، قابلیت متن‌های مفهومی‌تر را نیز خواهد داشت.

آموخته‌ها و پیشنهادها:

اهمیت پیش‌پردازش داده‌ها: پیش‌پردازش داده‌ها به‌طور کامل و دقیق قبل از ورود به مدل‌ها ضروری بوده چرا که این کار می‌تواند به بهبود دقت و کیفیت پاسخ‌های مدل کمک کند.

افزایش دقت در سیستم‌های RAG: اطلاعات بازیابی شده در رابطه با سوال مربوطه هستند و مدل احتمالاً آنها را در طی آموزش ندیده یا اگر هم دیده به علت حجم داده‌های زیاد به خوبی به یاد نگرفته‌است.

نیاز به منابع محاسباتی: استفاده از مدل‌های پرمصرف نیازمند منابع محاسباتی قدرتمند است، بنابراین باید قبل از شروع پروژه اطمینان حاصل شود که منابع مورد نیاز در دسترس هستند.

پیشنهادهای تیم‌های آینده

مدیریت منابع محاسباتی: مدیریت دقیق منابع محاسباتی و برنامه‌ریزی استفاده از آنها به‌طور هوشمند برای بهره‌وری بیشتر از منابع، بسیار حیاتی خواهد بود.

منابع

1. Lewis, P., et al. (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." *Advances in Neural Information Processing Systems*, 33, 9459-9474

Izacard, G., & Grave, E. (2021). "Leveraging Passage Retrieval with .2
Generative Models for Open Domain Question Answering." arXiv
preprint arXiv:2007.01282

Liu, J., et al. (2021). "RAG: Knowledge-based Text Generation with .3
Retrieval-Augmented Language Models." arXiv preprint
arXiv:2005.11401

پایان