نام گروه

Tensor Titans

اعضای گروه

غزل عسکری

یاسمین صرافی

سپیده سلیمانیان

امیرحسین رجبی

محمدرضا ویلانی

گزارشکار کار با Metis

مقدمه:

هدف این تمرین، اتصال یکی از فریمورک‌های LangChain یا Llama Index به API متیس بود تا بتوانیم وظایف ترجمه را با استفاده از این سرویس انجام دهیم. در این راستا ابتدا مستندات API متیس را بررسی کردیم تا نحوه تعامل با آن را بفهمیم و سپس از میان دو فریمورک نام‌برده، LangChain را انتخاب کردیم. در این گزارش، ابتدا یک کد ساده بدون استفاده از LangChain نوشته شد و سپس با استفاده از این فریمورک، کدی بهینه‌تر و با قابلیت‌های بیشتر ایجاد کردیم. در ادامه دلایل انتخاب LangChain و توضیحات بیشتری درباره مراحل پیاده‌سازی ارائه می‌شود.

بررسی مستندات و انتخاب فریمورک:

ابتدا مستندات API متیس را بررسی کردیم. متیس یک سرویس API ارائه می‌دهد که امکان ایجاد جلسه‌های گفتگو و ارسال پیام‌های مختلف را به کاربران می‌دهد. از آنجا که یکی از وظایف اصلی این پروژه ترجمه متن ها بود، تصمیم گرفتیم از قابلیت‌های API متیس برای ترجمه جملات از انگلیسی به فارسی استفاده کنیم. پس از بررسی مستندات، متوجه شدیم که می‌توان با ارسال درخواست‌های POST به API متیس، یک مکالمه (conversation) را آغاز کرد و سپس پیام‌ها را برای پردازش به این جلسه ارسال نمود.

برای این کار، دو فریمورک مختلف بررسی شد: LangChain و Llama Index. از میان این دو، LangChain به دلیل توانایی بیشتر در مدیریت زنجیره‌های پردازشی و قالب‌بندی خودکار پیام‌ها انتخاب شد. LangChain برای پروژه‌هایی که به پردازش متوالی و مدیریت داده‌های متنی نیاز دارند، مناسب‌تر است. همچنین، LangChain قابلیت‌هایی برای اتصال به APIهای مختلف از جمله APIهای ارائه‌دهنده مدل‌های زبانی را دارد که آن را برای این تمرین مناسب‌تر کرد.

دلیل استفاده از LangChain:

LangChain مزایای متعددی نسبت به نوشتن کدهای دستی برای اتصال به API دارد:

انتزاع بالاتر: LangChain به ما این امکان را می‌دهد که سطح انتزاعی بالاتری برای پردازش‌ها ایجاد کنیم. این یعنی به جای تمرکز روی جزئیات نحوه ارسال و دریافت پیام‌ها، می‌توانیم به سادگی زنجیره‌های پردازشی را تعریف کنیم و آنها را به API متصل کنیم.

قابلیت توسعه‌پذیری: اگر در آینده بخواهیم وظایف بیشتری را به سیستم اضافه کنیم، مثل خلاصه‌سازی متن، ایجاد چت‌بات یا تجزیه و تحلیل داده‌های متنی، با استفاده از

LangChain به راحتی می‌توانیم این وظایف را به زنجیره‌های موجود اضافه کنیم. این کار با استفاده از کد دستی، نیاز به تغییرات و مدیریت بیشتری دارد.

قالب‌بندی خودکار پیام‌ها: یکی از قابلیت‌های کلیدی LangChain قالب‌بندی خودکار پیام‌هاست. ما می‌توانیم با تعریف قالب‌هایی (template) برای ترجمه یا سایر وظایف، به سادگی پیام‌ها را بدون نیاز به مدیریت دستی به API ارسال کنیم. در پروژه ما، برای ترجمه، قالبی تعریف کردیم که مشخص می‌کرد دستیار متیس باید جملات انگلیسی را به فارسی ترجمه کند. سپس پیام‌های ورودی بر اساس این قالب به‌صورت خودکار قالب‌بندی شدند و به API ارسال شدند.

مدیریت بهتر سیستم و کاربر: در LangChain، پیام‌های سیستم (system message) و کاربر (human message) به خوبی تفکیک شده‌اند. این تفکیک به ما کمک می‌کند که کنترل بیشتری بر نحوه پردازش اطلاعات و تنظیمات مرتبط با هر پیام داشته باشیم. در این تمرین، یک پیام سیستمی تعریف شد که به متیس اعلام می‌کرد باید نقش یک مترجم را ایفا کند و پیام‌های بعدی به عنوان پیام‌های کاربر برای ترجمه ارسال شدند.

مراحل کار:

۱. پیاده‌سازی کد بدون استفاده از LangChain:

در این بخش از کد، به‌صورت دستی با API متیس ارتباط برقرار کردیم. فرآیند به شرح زیر است:

ایجاد جلسه مکالمه: ابتدا یک جلسه گفتگو (conversation session) با متیس ایجاد می‌شود. در این مرحله، اطلاعاتی شامل شناسه ربات (bot_id) و پیام اولیه به API ارسال می‌شود. API پاسخ داده و شناسه جلسه (session_id) را برمی‌گرداند.

ارسال پیام برای ترجمه: پس از ایجاد جلسه، پیام کاربر (متن مورد ترجمه) به جلسه ارسال می‌شود. برای این کار، درخواست دیگری به API ارسال می‌شود که شامل پیام موردنظر است. متیس پیام را دریافت و پردازش می‌کند و نتیجه را به عنوان پاسخ به ما برمی‌گرداند.

این روش به خوبی کار می‌کند، اما نیاز به نوشتن کد دستی برای مدیریت درخواست‌ها و پاسخ‌ها دارد که در پروژه‌های بزرگ‌تر می‌تواند پیچیدگی و زمان بیشتری ببرد.

۲. پیاده‌سازی کد با استفاده از LangChain:
در این مرحله، از LangChain برای پیاده‌سازی همان فرآیند استفاده کردیم. مراحل کار به این شکل بود:

تعریف قالب برای ترجمه: در ابتدا، یک قالب (template) تعریف کردیم که به LangChain اعلام می‌کرد نقش دستیار این است که متن را از انگلیسی به فارسی ترجمه کند. این قالب شامل دو بخش بود:

یک پیام سیستمی که مشخص می‌کرد LangChain باید نقش یک مترجم را ایفا کند.
یک پیام کاربر که متن مورد ترجمه را شامل می‌شد.
قالب‌بندی خودکار پیام‌ها: پس از تعریف قالب، LangChain به‌صورت خودکار پیام‌های سیستم و کاربر را قالب‌بندی کرد. این پیام‌ها سپس به API متیس ارسال شدند تا وظیفه ترجمه انجام شود.

دریافت و پردازش پاسخ: پاسخ متیس از API دریافت شده و توسط LangChain پردازش شد و به کاربر نمایش داده شد. LangChain مدیریت تمامی مراحل از ارسال پیام تا دریافت پاسخ را به‌صورت خودکار انجام داد، که باعث ساده‌تر و بهینه‌تر شدن کد شد.

تفاوت‌ها و نتایج:

در کد بدون LangChain، مدیریت هر مرحله از تعامل با API به‌صورت دستی انجام می‌شد. این شامل ارسال درخواست‌ها، دریافت پاسخ‌ها و قالب‌بندی داده‌ها بود. با اینکه این روش برای وظایف ساده مثل ترجمه جملات قابل انجام است، اما برای پروژه‌های بزرگ‌تر و پیچیده‌تر به‌سرعت پیچیده می‌شود.

در مقابل، با استفاده از LangChain، تمامی این فرآیندها به‌صورت خودکار و انتزاعی مدیریت شدند. LangChain نه تنها کار ما را در این پروژه ساده‌تر کرد، بلکه این امکان را به ما داد که در آینده به راحتی زنجیره‌های پردازشی پیچیده‌تری ایجاد کنیم و وظایف بیشتری به سیستم اضافه کنیم.

نتیجه‌گیری:

استفاده از LangChain باعث شد که کد ما ساده‌تر، تمیزتر و مقیاس‌پذیرتر شود. در پروژه‌های پیچیده یا در مواردی که نیاز به افزودن وظایف جدید وجود دارد، استفاده از LangChain می‌تواند فرآیند توسعه را به شکل قابل توجهی ساده‌تر کند. همچنین، با توجه به اینکه LangChain قالب‌بندی و مدیریت پیام‌ها را به‌صورت خودکار انجام می‌دهد، تمرکز توسعه‌دهنده بر روی منطق اصلی برنامه باقی می‌ماند.