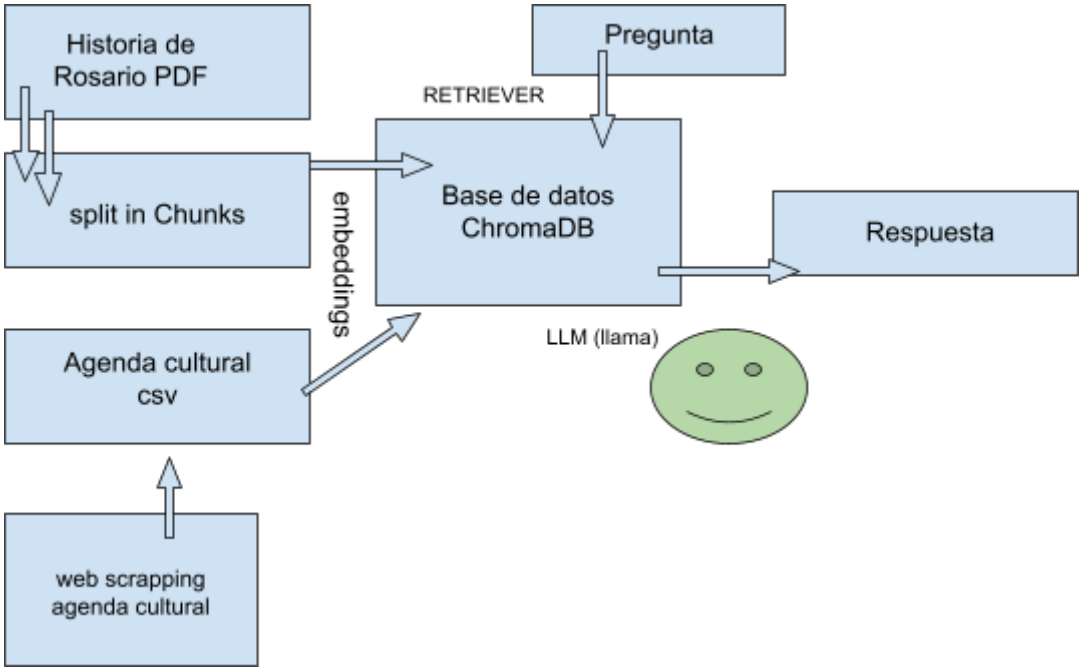
Trabajo práctico final NLP

Nicolás Di Marco
DNI 29208916

Ejercicio 1 RAG

Para este ejercicio decidí la temática ‘Cultura en la ciudad de Rosario’.
El Chatbot tiene como fin el poder realizar consultas sobre los próximos eventos en la ciudad de Rosario y obtener información de ellos. También cuenta con una base de información sobre la historia de Rosario.

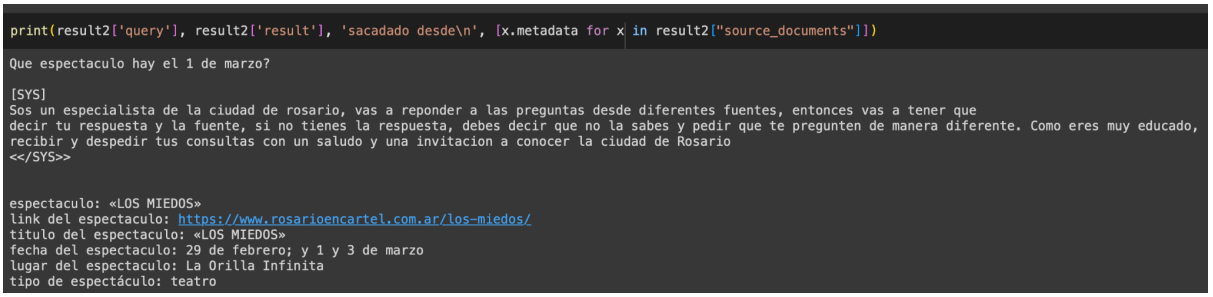


El RAG fue implementado como refiere el gráfico. Para la carga de archivos en una base de datos vectorial se utilizó Chromadb previa creación de embeddings con el modelo multilingual de hugging face. Ya que langchain ofrece un loader para csv , decidí cargarlo directamente de esa manera , en los tests se puede ver que funciona correctamente.

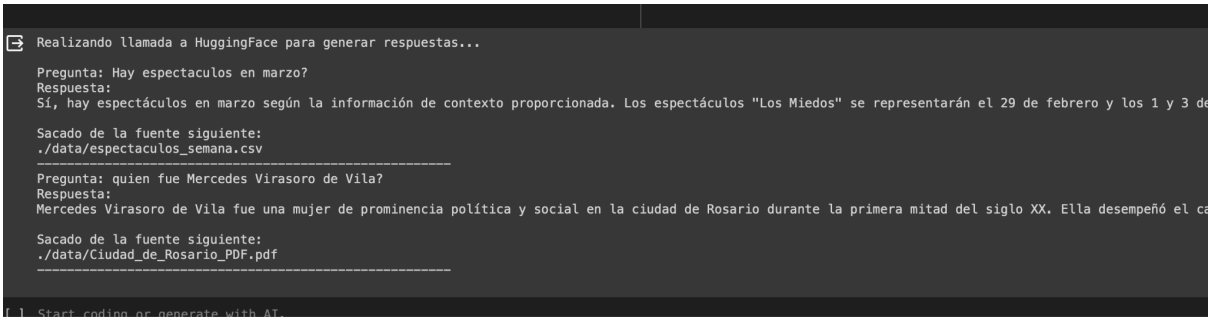
Para el split de los textos usé recursive splitter, mejoraba mucho la precisión de las respuestas.
El web scraping fue realizado con scrapy, librería usada en las primeras unidades de la materia.

Problemas detectados:
En un principio comencé a trabajar con un modelo de llama2, pero después fue dado de baja en hugging face y tuve que rehacer gran parte del trabajo con un modelo mediante api, zephir (base mistral 7b) . Tuve que cambiar prompts y algunas cosas más. Utilicé también en un principio llama index pero no tuve buenos resultados así que fue todo mudado a langhain.

captura de test del primer chat con llama2:



Capturas del modelo final:



Ejercicio 2 Agentes

En mi investigación sobre los sistemas multi agentes, descubrí una herramienta de Microsoft(utilizamos la versión vieja en unidad 7) que permite organizar y poner en producción el trabajo de los sistemas multi agentes. Utilice la versión 2.0 (2024) que es open source y tiene una interfaz grafica de usuario para poder utilizarla. Es realmente simple de poner en funcionamiento y la parte compleja es desarrollar el codigo de cada agente(que depende de la complejidad de su tarea), pero básicamente crea una sala de chat entre sistemas de ia y los pone a trabajar en equipo. Estas tareas pueden ser semi supervisadas y requerir de interacción con un usuario.
Además es compatible con ollama que le permite usar diferentes modelos llm open source.

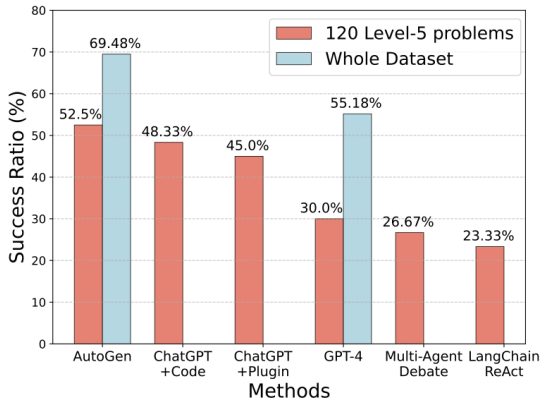
Como vimos en la unidad 7 describir como:

AutoGen, al permitir la cooperación entre LLMs, herramientas y humanos, aborda eficazmente la necesidad de aplicaciones LLM de próxima generación, ofreciendo

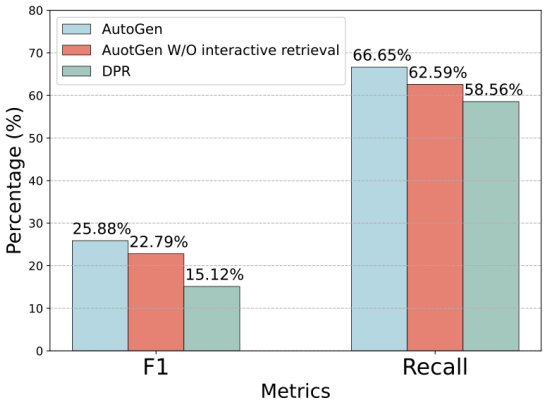
características clave como agentes personalizables, interfaces de conversación unificadas y patrones de conversación flexibles para flujos de trabajo complejos.

Además, AutoGen proporciona una colección de sistemas funcionales de diversas complejidades, demostrando su capacidad para soportar distintos patrones de conversación y aplicaciones en varios dominios.

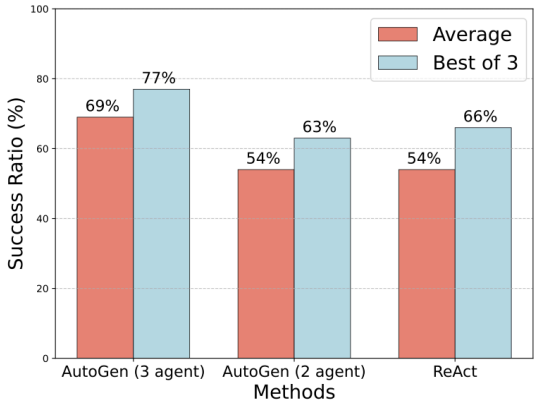
En un articulo de 2023, podemos encontrar una comparativa de los multiagentes, vs modelos como gpt4, lo cual también nos lleva a pensarlos como potenciadores de los modelos de LLM.



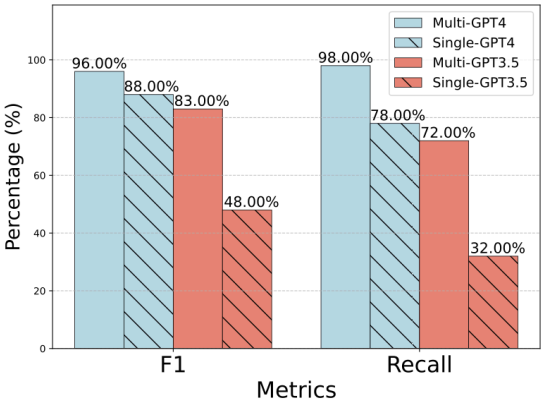
(a) A1: Performance on MATH (w/ GPT-4).



(b) A2: Q&A tasks (w/ GPT-3.5).



(c) A3: Performance on ALFWorld.



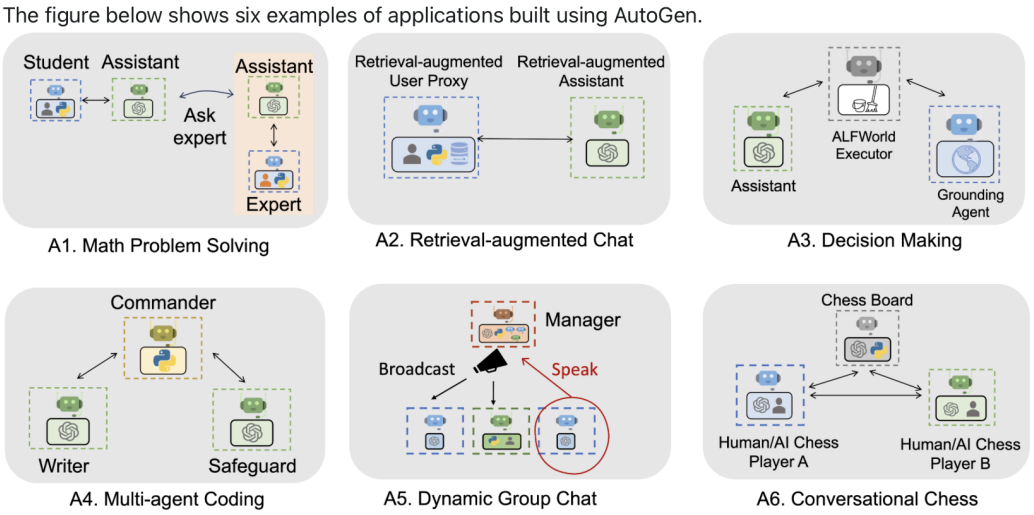
(d) A4: Performance on OptiGuide.

fuente: AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation.Microsoft Research Pennsylvania State University ±University of Washington,∓Xidian University.

Para poder mejorar el sistema de RAG de la primer consigna, decidí aplicar un sistema multiagente , al menos teóricamente.

Este sistema podría tener entonces diferentes agentes con los cuales se podría mejorar la experiencia con el usuario y agregar información a la requerida , además de adicionar mas funciones , que solo una búsqueda.

Entonces, lo primero fue idea un espacio de teamwork para definir las siguientes responsabilidades de cada agente. Como voy a pensarlo para implementar en autogen vamos a usar su misma organización. Dejo un grafico para poder interpretar mejor su funcionamiento:



Lo que decidí yo podria tambien ser, de alguna manera un fine tuning de un modelo llm, por eso tambien me permití pensar algunos agentes mas para su hipotetica implementacion.

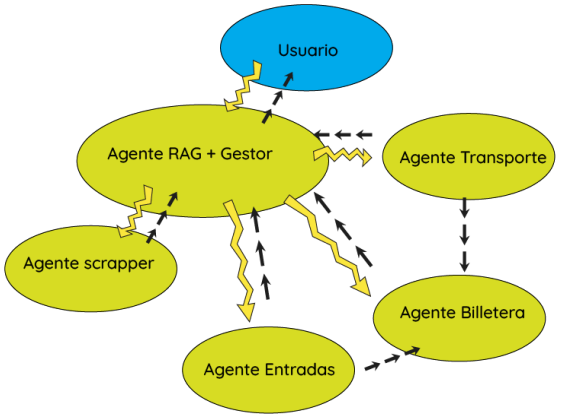
La idea es la siguiente:

- El rag implementado en la consigna número 1, realiza un web scrap para tener los eventos de la semana y de las siguientes, además de tener en su bd, un pdf (o unos) sobre la historia de rosario. Con esto podría responder algunas cosas a un usuario que visita nuestra ciudad.
    - ***Pero, y si el usuario quiere seguir usando el chat para guiar su visita por Rosario?***
- Entonces podríamos pensar en crear nuevos agentes que complementen y potencien la función del chat rag, por ejemplo:
    - un agente va a ser el encargado del presupuesto del usuario, con alertas sobre su monto a medida que baja
    - Si la respuesta del rag es sobre un artista o espectáculo en particular, uno de los agentes podría completar con información sobre el espectáculo.
    - Otro de los agentes podria tambien chequear si hay entrada y si es necesario , podría pagarla

- Otro de los agentes va a ser el encargado del transporte y sera el responsable de , una vez decidido el espectáculo , planear el medio de transporte para el traslado desde el lugar del usuario hasta el espectáculo, va a presentar las opciones y va a decidir el encargado de la billetera o finanzas del usuario.

De esta manera el usuario tendría un asistente que por detrás está decidiendo entre múltiples agentes y planificando la agenda cultural y turística, todo mediando una experiencia de usuario parecido a un chat gpt

Para entender mejor la interacción entre agentes voy a adjuntar un gráfico y un ejemplo de uso



Flujo ejemplo

Bibliografía:
Ejercicio 1:
Lang Chain docs: https://python.langchain.com/docs
Python docs: https://docs.python.org/3.1/
Scrapy: https://scrapy.org/

Ejercico 2:

Autogen github page:

https://microsoft.github.io/autogen/

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

Qingyun Wu†, Gagan Bansal, Jieyu Zhang±, Yiran Wu†, Beibin Li* Erkang Zhu*, Li Jiang*, Xiaoyun Zhang*, Shaokun Zhang†, Jiale Liu∓ Ahmed Awadallah*, Ryen W. White*, Doug Burger*, Chi Wang*1 *Microsoft Research, †Pennsylvania State University ±University of Washington,∓Xidian University*

https://arxiv.org/pdf/2308.08155.pdf