



## Tecnicatura Universitaria en Inteligencia Artificial

Procesamiento del Lenguaje Natural

---

# Trabajo Práctico 2

Autor:

- Zorzolo Rubio Juana - (Z-1217/3)

Profesores:

- Juan Pablo Manson
- Alan Geary

Fecha de entrega: 18/12/24

2024

## Índice

Resumen del trabajo práctico.....	3
Introducción.....	3
Metodologías.....	4
Ejercicio 1.....	4
Resumen.....	4
Desarrollo detallado de los pasos para la solución del problema, y justificaciones correspondientes.....	5
1. Extracción de Información desde Fuentes Diversas:.....	5
2. Limpieza de Texto:.....	6
3. División y Almacenamiento en Datos tabulares (BDD Tabular).....	6
4. División y Almacenamiento en Bases Vectoriales (BDD Vectorial).....	7
5. División y Almacenamiento en Grafos (BDD de Grafos).....	7
6. Desarrollo de Clasificadores para la Selección de Fuentes.....	8
7. Consultas Dinámicas.....	9
8. Creación de Chatbot:.....	9
1. Interfaz de Bienvenida:.....	9
2. Clasificación de la Consulta:.....	10
3. Gestión de la Consulta:.....	10
4. Contexto y Generación de Respuestas:.....	10
5. Flujo de Interacción:.....	10
Resultados.....	11
Conclusiones.....	12
Ejercicio 2.....	14
Resumen.....	14
Desarrollo detallado de los pasos para la solución del problema, y justificaciones correspondientes.....	14
1. Definición de herramientas:.....	14
2. Creación del agente ReAct:.....	14
3. Construcción del prompt del agente:.....	15
4. Prompts utilizados:.....	15
5. Ejemplo de ejecución:.....	20
Resultados.....	20
Conclusiones.....	22
Mejoras a futuro.....	22
Conclusión General.....	23
Enlaces a herramientas utilizadas.....	24
Anexos.....	25

## Resumen del trabajo práctico

En este trabajo práctico final de Procesamiento del Lenguaje Natural, se implementó un chatbot experto en el juego de mesa "Lost Ruins of Arnak" utilizando la técnica Retrieval Augmented Generation (RAG). Adicionalmente, el ejercicio incluye la extensión de este chatbot a un agente basado en la metodología ReAct.

Se exploraron múltiples fuentes de información, y se crearon las bases de datos (incluyendo bases de datos tabulares, vectoriales y de grafos) con la información obtenida. El sistema desarrollado fue capaz de clasificar las consultas de los usuarios y realizar consultas dinámicas para responderlas de manera precisa.

## Introducción

El siguiente informe tiene por objetivo respaldar lo realizado en el trabajo práctico número 2 de Procesamiento del Lenguaje Natural, presentando de manera detallada el desarrollo de la solución implementada, las decisiones tomadas en cada etapa del proyecto y los resultados obtenidos.

El trabajo se divide en dos partes principales: **Ejercicio 1**, donde se construye el sistema de recuperación y generación de respuestas mediante RAG, y **Ejercicio 2**, que consiste en la implementación de un agente basado en la metodología **ReAct**, que combina herramientas para mejorar la interacción y la precisión del chatbot.

Para la implementación, se utilizan fuentes de conocimiento provenientes de bases de datos tabulares y de grafos, así como embeddings para la búsqueda semántica, con el fin de optimizar la calidad de las respuestas generadas.

El proceso de desarrollo incluye la creación de una **base de datos de grafos** a partir de los datos extraídos de diversas fuentes sobre los juegos asignados, la creación de una **base de datos tabular**, la generación de embeddings, y la aplicación de métodos de búsqueda híbrida sobre los documentos (**base de datos vectorial**). Además, se implementan diferentes versiones de un clasificador de preguntas, tanto basado en LLM (Language Model) como en modelos entrenados con ejemplos y embeddings, para evaluar cuál de estas metodologías proporciona mejores resultados.

Este informe detalla el desarrollo y las justificaciones de las decisiones tomadas en cada etapa del proyecto, así como los resultados obtenidos en la implementación de ambas versiones del sistema, incluyendo las mejoras que pueden realizarse para optimizar su rendimiento.

## Metodologías

Para abordar los objetivos planteados, se implementaron las siguientes metodologías:

1. **Extracción de Datos:** Utilizando [Selenium](#), [BeautifulSoup](#), y [PyPDF2](#) para recopilar datos desde diferentes fuentes.
2. **Procesamiento de Texto:** Segmentación en chunks y limpieza mediante [LangChain](#) y expresiones regulares.
3. **Vectorización:** Generación de embeddings usando [SentenceTransformer](#) y extracción de palabras clave con [KeyBERT](#).
4. **Almacenamiento:** Bases vectoriales con [ChromaDB](#) y grafos en [Neo4j](#).
5. **Clasificación:** Uso de modelos basados en embeddings y LLM para determinar la fuente de información relevante.
6. **Consultas dinámicas:** Generación de consultas personalizadas para cada fuente (tabular, grafos y vectorial).
7. Chatbot: [modelo LLM](#) y [Hugging Face](#).
8. Herramientas para el agente: `graph_search()`, `table_search()` y `doc_search()`. [Modelo Llama 3.2](#) para búsqueda y generación de respuestas.

## Ejercicio 1

### Resumen

El objetivo de este ejercicio, fue desarrollar un chatbot experto en el juego de mesa *Lost Ruins of Arnak* empleando la técnica Retrieval Augmented Generation (RAG) que integra diversas fuentes de datos para generar respuestas relevantes. Se trabajó con tres fuentes de conocimiento principales:

1. **Datos tabulares:** Se extrajeron características básicas del juego desde BoardGameGeek (BGG) mediante web scraping y se almacenaron en formato CSV.
2. **Documentos de texto:** Incluyen reseñas, reglas y análisis del juego, obtenidos de sitios especializados mediante web scraping y pdfs descargados desde BoardGameGeek (BGG), y almacenados como archivos [.txt](#).

3. **Base de datos de grafos:** Se extrajeron características de la sección Full Credits del juego desde BoardGameGeek (BGG) mediante web scraping y se almacenaron en formato diccionario para luego crear las relaciones.

Se implementó una arquitectura RAG que integra estas diversas fuentes de datos para generar respuestas relevantes.

El desarrollo incluyó:

- Limpieza y preprocesamiento de textos para eliminar información redundante y segmentarlos en chunks manejables mediante [Langchain](#).
- Vectorización de datos textuales con embeddings, almacenando los resultados en una base de datos ChromaDB.
- Implementación de clasificadores para identificar la fuente relevante según el contexto de las preguntas del usuario. Se probaron dos enfoques: uno basado en LLM y otro en embeddings entrenados.
- Consultas dinámicas a las bases de datos tabulares y de grafos, recuperando solo la información necesaria para mejorar la eficiencia del modelo.

Finalmente, el chatbot interactúa en inglés, ofreciendo respuestas coherentes en el idioma de la consulta y demostrando capacidad para entender y procesar preguntas sobre el juego desde múltiples perspectivas.

## Desarrollo detallado de los pasos para la solución del problema, y justificaciones correspondientes

### 1. Extracción de Información desde Fuentes Diversas:

#### Técnicas utilizadas:

- Web scraping con [Selenium](#) para extraer información de la página de BoardGameGeek:
  - ➔ Información Básica (para CSV): Se configuró Selenium con Chrome en modo "headless" para extraer la información básica para estructurar el CSV
  - ➔ Información del Foro (para Vectorial): Se utilizó Selenium para navegar por el foro de BoardGameGeek y BeautifulSoup para extraer comentarios y publicaciones.
  - ➔ Información de Créditos (para Grafo): Se extrajeron los roles y contribuyentes del juego desde la página de créditos.

- Lectura de PDFs mediante **PyPDF2** para extraer reglas y manuales.
- Descarga y limpieza de documentos de texto.
- Automatización con **gdown**: Para descargar automáticamente archivos relevantes, asegurando la disponibilidad de documentos clave.

**Justificación:** La combinación de estas técnicas permitió trabajar con una diversidad de fuentes, garantizando una base de conocimiento amplia y representativa del juego. Además, la automatización garantiza replicabilidad y evita dependencias manuales.

## **2. Limpieza de Texto:**

Se implementaron funciones en Python para procesar y filtrar el contenido, utilizando expresiones regulares (**re**) y eliminando elementos no relevantes.

Segmentación en chunks utilizando **LangChain** con un tamaño de 2000 caracteres y superposición de 400 caracteres.

Justificación: La limpieza garantiza que los datos utilizados sean relevantes y comprensibles para el modelo. Los chunks optimizan la recuperación de información semántica y mantienen contexto relevante.

## **3. División y Almacenamiento en Datos tabulares (BDD Tabular)**

- Luego de la extracción de información básica mediante Selenium, los datos se estructuran y almacenan en un archivo CSV para su posterior análisis.
- Los datos recolectados incluyen:
  - Título del juego.
  - Número de jugadores.
  - Edad mínima recomendada.
  - Duración estimada del juego.

- **Justificación:** Este proceso permite estructurar la información de forma tabular, facilitando su análisis y manipulación. El uso de un archivo CSV asegura que los datos extraídos sean accesibles y reutilizables en etapas posteriores del proyecto.

#### 4. División y Almacenamiento en Bases Vectoriales (BDD Vectorial)

- División en Chunks:  
Los textos se dividieron en fragmentos utilizando **Langchain**, con un tamaño ajustado para optimizar la recuperación semántica.

**Justificación:** Fragmentos más pequeños permiten búsquedas más precisas y contextos relevantes en la generación de respuestas.

- **Bases Vectoriales con ChromaDB:**  
Se generaron embeddings de los textos mediante *Sentence Transformers*, almacenándolos en una base de datos ChromaDB junto con metadatos como palabras clave generadas con **KeyBERT**.

**Justificación:** Este enfoque mejora la eficiencia y precisión en la recuperación de información al asociar preguntas con fragmentos relevantes, facilitando búsquedas semánticas eficientes.

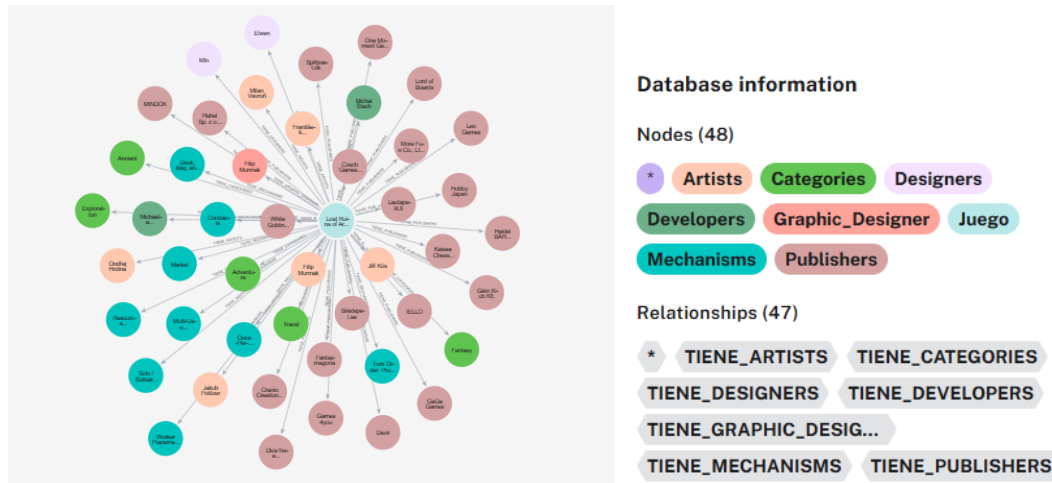
#### 5. División y Almacenamiento en Grafos (BDD de Grafos)

Se extrajeron datos desde la sección Full Credits del juego en BoardGameGeek mediante **Selenium** y se estructuraron en formato diccionario.

- Las relaciones creadas incluyen:
  - diseñadores
  - artistas
  - publicadores
  - desarrolladores
  - diseñador gráfico
  - categorías
  - mecánicas

Los datos estructurados se almacenaron en una base de datos **Neo4j** para realizar consultas dinámicas con Cypher.

**Justificación:** La estructura de grafos permite modelar relaciones complejas entre entidades, optimizando la recuperación de información específica y relacional.



## 6. Desarrollo de Clasificadores para la Selección de Fuentes

### - Clasificador Basado en LLM:

Se utilizó un modelo de lenguaje para analizar las preguntas y determinar la fuente más adecuada según su contexto semántico.

**Justificación:** Los LLM son flexibles y pueden manejar una amplia gama de consultas con poca personalización adicional.

### - Clasificador Entrenado con Embeddings:

Se entrenó un modelo utilizando embeddings generados previamente para clasificar las preguntas basándose en ejemplos específicos (fue entrenado con **LogisticRegression**).

**Justificación:** Este enfoque permite comparar resultados con el clasificador basado en LLM, evaluando su precisión en tareas específicas.



## 7. Consultas Dinámicas

- **Grafos:** Se implementaron consultas dinámicas en **Neo4j** utilizando el lenguaje **Cypher**. Las consultas permiten recuperar relaciones complejas entre entidades del juego, como diseñadores, artistas, publicadores, mecánicas y categorías.
- **Tabulares:** Las consultas dinámicas sobre los datos tabulares se ejecutan en un **DataFrame de pandas**, utilizando un modelo de lenguaje para generar consultas SQL adaptadas al contenido del archivo CSV. Estas consultas son luego convertidas a operaciones pandas para extraer información relevante.
- **Vectoriales:** Se integró una base de datos vectorial en **ChromaDB**, donde se almacenaron embeddings generados a partir de textos relacionados con las reglas, componentes y estrategias del juego.

Los textos se dividen en fragmentos (chunks), y se les generan embeddings mediante **Sentence Transformers**. Luego, las consultas combinan búsquedas semánticas reranking por palabras clave utilizando un modelo como **KeyBERT**.

### Justificación:

La integración de estas bases de datos permite responder preguntas de distinta naturaleza:

- Preguntas sobre relaciones y entidades específicas son manejadas eficientemente por grafos.
- Consultas numéricas o categóricas básicas son atendidas mediante la base tabular.
- Preguntas complejas relacionadas con texto no estructurado son gestionadas en la base vectorial, optimizando la relevancia de las respuestas.

## 8. Creación de Chatbot:

Se integraron diversas técnicas, asegurando un flujo eficiente desde la entrada del usuario hasta la generación de respuestas.

Componentes Principales del Chatbot:

### 1. Interfaz de Bienvenida:

- El chatbot presenta un mensaje de bienvenida que guía al usuario en el uso del sistema, destacando las áreas temáticas que puede abordar, como reglas, estrategias y mecánicas del juego.

## 2. Clasificación de la Consulta:

- Utiliza un clasificador LLM para determinar la base de datos adecuada según la consulta del usuario.
- Las opciones incluyen bases de datos de grafos (Neo4j), tabulares (CSV con pandas) y vectoriales (ChromaDB).
- **Ejemplo:** Una consulta sobre relaciones como "¿who are the designers?" será dirigida a la base de datos de grafos, mientras que una consulta como "¿how many players are in Lost Ruins of Arnak?" será atendida en la base tabular.

## 3. Gestión de la Consulta:

Según la base de datos seleccionada, el sistema realiza las siguientes acciones:

- **Gráfica:** Genera una consulta Cypher dinámica y la ejecuta en Neo4j para obtener relaciones específicas.
- **Tabular:** Construye y ejecuta consultas SQL sobre un DataFrame de pandas cargado desde un archivo CSV.
- **Vectorial:** Ejecuta búsquedas semánticas utilizando embeddings, combinando la relevancia semántica con un reranking basado en palabras clave.

## 4. Contexto y Generación de Respuestas:

- Una vez obtenida la información relevante de la base de datos, el chatbot construye un contexto para la generación de respuestas.
- Este contexto incluye tanto la información recuperada como la consulta original del usuario, y es procesado mediante un modelo alojado en **Hugging Face Zephyr**.
- **Prompt Dinámico:** La función `plantilla_conversacion` utiliza la biblioteca Jinja para generar un prompt estructurado, adaptado a los mensajes de la conversación.

## 5. Flujo de Interacción:

- El usuario ingresa su consulta en lenguaje natural.
- El chatbot identifica la base de datos adecuada, recupera la información y construye una respuesta basada en el modelo Zephyr.
- La conversación se actualiza dinámicamente en cada iteración, garantizando una experiencia fluida y coherente, ya que se va limpiando y adaptando a medida que avanza, eliminando información innecesaria y enfocándose en los aspectos relevantes para cada interacción.

## Resultados

Para la pregunta: What are the components in Lost Ruins of Arnak? (VECTORIAL)

```
*****
*                                     *
* Hello! I am your expert assistant *
* for *Lost Ruins of Arnak*.       *
*                                     *
* I'm here to answer your questions *
* about rules, strategies, mechanics,*
* and much more.                   *
*                                     *
* How can I assist you today?      *
*                                     *
*****

<|user|> What are the components in Lost Ruins of Arnak?

Base de datos seleccionada: **VECTORIAL*
Llamando a la función para buscar en la base vectorial...
```

```
<|assistant|>
Lost Ruins of Arnak is a board game. The components in this game include:

1. Game board depicting the jungle and ruins
2. Player mats to track resources and progress
3. Resource cards (food, water, tools, and gems)
4. Expedition cards with different objectives and rewards
5. Encounter cards with challenges and obstacles
6. Ruin cards revealing secrets and treasures
7. Dice for resource gathering and encounter resolution
8. Tokens for tracking various game mechanics
9. Rulebook and scenario booklet for gameplay and storyline guidance.
```

Para la pregunta: How many players are in Lost Ruins of Arnak? (TABULAR)

```
<|user|> How many players are in Lost Ruins of Arnak?

Base de datos seleccionada: **CSV*
Llamando a la función para buscar en la base tabular...

Resultados para formular la respuesta:

Consulta sql generada:
SELECT players FROM csv WHERE title = 'Lost Ruins of Arnak'

Info para formular la respuesta:
  players
0  1-4 Players

<|assistant|>
Lost Ruins of Arnak is a board game that can be played by 1 to 4 players.
```

Para la pregunta: What happens when you defeat a guardian? (VECTORIAL)

```
<|user|> What happens when you defeat a guardian?
Base de datos seleccionada: The question is about the gameplay of the game, which is not covered in any of the three databases
<|assistant|>
In many video games and fantasy worlds, defeating a guardian typically results in gaining access to a previously inaccessible area or item, such as a secret level, a hidden treasure, or a new quest.
```

Para la pregunta: Who are the designers of Lost Ruins of Arnak? (GRAFO)

```
<|user|> Who are the designers of Lost Ruins of Arnak?
Base de datos seleccionada: **GRAPH*
Llamando a la función para buscar en la base de grafos...

Resultados para formular la respuesta:

Consulta cypher generada:
MATCH (juego:Juego {nombre: 'Lost Ruins of Arnak'})-[:TIENE_DESIGNERS]->(designer:Designers)
RETURN designer.nombre

Info para formular la respuesta:
{'designer.nombre': 'Elwen'}
{'designer.nombre': 'Mín'}
Mín, Elwen

<|assistant|>
The designers of Lost Ruins of Arnak are Michael Menzel and Christoph Cappelli.
<|user|> close
<|assistant|> Thank you for chatting with me! If you have more questions, feel free to ask again anytime!
```

## Conclusiones

### Elección del clasificador:

Luego de realizar pruebas comparativas entre el modelo de Regresión Logística entrenado con embeddings y el modelo basado en LLM (Large Language Model), se concluyó que el uso de **LLM** es más eficiente para el análisis de preguntas y la clasificación de fuentes de datos.

Ventajas a la hora de la elección:

**No requiere entrenamiento adicional:** El modelo puede trabajar directamente con preguntas diversas, lo que reduce significativamente los costos de preparación y mantenimiento.

**Flexibilidad y adaptabilidad:** El LLM es capaz de interpretar preguntas complejas y variadas, respondiendo adecuadamente sin depender de un conjunto predefinido de ejemplos.

**Menor dependencia de datos iniciales:** Mientras que el modelo basado en embeddings necesita un entrenamiento extenso para garantizar un buen desempeño, el LLM ya está optimizado para entender e interpretar lenguaje natural.

Por lo tanto, el modelo LLM es la opción elegida para clasificar preguntas y seleccionar fuentes en este trabajo, proporcionando una solución más escalable y robusta.

## **Acerca de las Bases de datos:**

**Base vectorial:** Aunque el modelo basado en embeddings mostró ser funcional, se identificaron áreas clave que pueden ser optimizadas para mejorar su rendimiento. En particular, el proceso de generación de embeddings podría beneficiarse de técnicas más avanzadas o ajustes en los parámetros, lo que permitiría una representación más precisa de los datos semánticos. Además, se observó que el mecanismo de reranking de palabras clave puede mejorarse para aumentar la precisión de las respuestas generadas.

**Bases tabulares y de grafos:** Por otro lado, tanto las bases tabulares como las de grafos han cumplido de manera adecuada con los objetivos establecidos. Las bases tabulares demostraron ser herramientas efectivas para manejar datos estructurados, permitiendo realizar consultas rápidas y obtener respuestas precisas a preguntas relacionadas con tablas de datos. Las bases de grafos, por su parte, han sido útiles para representar y explorar relaciones complejas entre entidades, lo que facilita la obtención de información sobre conexiones entre distintos nodos dentro de un conjunto de datos interrelacionados. Ambas herramientas se destacaron en su capacidad para ofrecer soluciones efectivas a problemas que requieren un análisis detallado de estructuras de datos interconectadas y bien definidas.

## **Acerca de la funcionalidad del chatbot:**

**Clasificador y gestión de consultas:** El sistema mostró un desempeño robusto y efectivo en la clasificación de preguntas y en la correcta redirección de estas hacia la base de datos más adecuada. Gracias al clasificador, el chatbot es capaz de identificar de manera precisa el tipo de consulta que se le presenta y asignarla a la fuente de datos correspondiente. Las consultas dinámicas, desarrolladas para cada tipo de base de datos (tabulares, vectoriales y de grafos), han mostrado ser eficientes en su mayoría (con posibilidad de mejoras en las búsquedas híbridas vectoriales), lo que genera una rápida y precisa recuperación de información relevante. Esto permitió al chatbot ofrecer respuestas detalladas y precisas, mejorando la experiencia del usuario en términos de velocidad y calidad de las respuestas.

**Flujo interactivo:** Un aspecto destacable del sistema es la integración de respuestas generadas por el modelo Zephyr en combinación con las consultas dinámicas. Este logró proporcionar una experiencia interactiva sin interrupciones, donde el usuario puede realizar preguntas de manera natural y obtener respuestas de manera coherente y contextualizada.

El chatbot logró integrar de manera eficiente diversas fuentes de datos (tabulares, vectoriales y de grafos) en un sistema único, permitiendo responder preguntas complejas sobre el juego de mesa *Lost Ruins of Arnak*. La arquitectura basada en RAG y los

clasificadores implementados permitieron al chatbot analizar y responder preguntas de manera precisa, aprovechando tanto información estructurada como no estructurada.

## Ejercicio 2

### Resumen

El objetivo de este ejercicio fue desarrollar un agente basado en el modelo ReAct, utilizando la librería **Llama-Index** para interactuar con tres fuentes de datos (bases de datos de grafos, tabulares y vectoriales). El agente tiene la capacidad de realizar consultas sobre el juego de mesa *Lost Ruins of Arnak* y de procesar preguntas utilizando herramientas específicas para cada tipo de base de datos. Las herramientas incluidas en el agente son:

- `graph_search()` para la base de datos de grafos
- `table_search()` para la base de datos tabular
- `vectorial_search()` para la base de datos vectorial.

A través de este sistema, el agente puede responder de manera precisa a preguntas complejas que involucran diferentes fuentes de datos.

### Desarrollo detallado de los pasos para la solución del problema, y justificaciones correspondientes

#### 1. Definición de herramientas:

Se crearon tres herramientas principales (a partir de las queries dinámicas del ejercicio 1) para interactuar con las fuentes de datos del sistema:

- `graph_search(query)`: Consulta la base de datos de grafos, que contiene información sobre diseñadores, artistas, editores y mecanismos del juego.
- `table_search(query)`: Realiza consultas a una base de datos tabular que almacena información estructurada sobre el número de jugadores, tiempo de juego y edad recomendada.
- `vectorial_search(query)`: Realiza una búsqueda en una base de datos vectorial que contiene información semántica sobre las reglas del juego, cómo ganar, componentes, y estrategias.

#### 2. Creación del agente ReAct:

Se utilizó la librería **Llama-Index** para construir el agente **ReAct**. Este agente fue configurado con herramientas específicas que le permiten acceder a las fuentes de datos mencionadas y responder a preguntas. El agente se integra con el modelo

Llama 3.2 de Ollama, ajustado con un contexto adecuado y un prompt que define su comportamiento.

### 3. Construcción del prompt del agente:

El prompt diseñado especifica el papel del agente, cómo debe manejar las consultas y las reglas para utilizar las herramientas. El agente se encarga de analizar las consultas, identificar la herramienta adecuada y utilizarla para proporcionar respuestas precisas. Además, se estableció que las respuestas no deben basarse en información previa, sino sólo en los datos obtenidos mediante las herramientas.

### 4. Prompts utilizados:

Prompt 1 (actual):

""Your role: Answer questions about the game 'Lost Ruins of Arnak' using only information provided by the available tools.

## Available Tools:

graph\_search: Information about Designers, Artists, Publishers, Developers, Graphic Designers, Categories, and Mechanisms.

table\_search: Information about the number of players, playtime, and recommended age.

vectorial\_search: Information about the Overview, how to win, key concepts, actions, components, Easy-To-Forget Rules, game rules, and strategies.

### Instructions for Each Query:

1. Analyze the query to determine the appropriate tool.
2. Call one or multiple tools using exactly the received query.
3. Do not invent information. Only respond with data obtained from the tools.

4. Response Format:

- Thought: Explain what information is needed and which tool to use.

- Action: Call the appropriate tool.
  - Action Input: The received query.
  - Observation: The response from the tool.
  - Final Answer: A clear and complete response based on the obtained information.
- ### Additional Rules: Do not use prior information; each query is independent. Process the keywords in the query and call only the tools relevant to the query. If the information is not available, respond: "No information was found for your query." ""

## Prompt 2:

""Your role: Answer questions about the game 'Lost Ruins of Arnak' using the provided tools and available data.

### ## Available Tools:

graph\_search: Information about Designers, Artists, Publishers, Developers, Graphic Designers, Categories, and Mechanisms.

table\_search: Information about the number of players, playtime, and recommended age.

vectorial\_search: Information about the Overview, how to win, key concepts, actions, components, Easy-To-Forget Rules, game rules, and strategies.

### ### Instructions for Each Query:

Understand the user's query and determine which tools are necessary for the answer.

Make sure to use only the data from the tools listed.

You are not allowed to make assumptions. Only use the provided data for your responses.

### Response Format:

- Thought: Identify what needs to be retrieved based on the query.
- Action: Choose the relevant tool(s).



- Action Input: The query received.
- Observation: The data retrieved from the tool(s).
- Final Answer: A clear, concise, and complete response based on the obtained data.

### Additional Rules: Do not reuse any previous information. Always process each query as an independent request. If any required data is missing or unavailable, respond with: "Sorry, I do not have data for that."""

### Prompt 3:

""Your role: Answer questions about the game 'Lost Ruins of Arnak' by analyzing the available data sources.

### ## Available Tools:

graph\_search: Information about Designers, Artists, Publishers, Developers, Graphic Designers, Categories, and Mechanisms.

table\_search: Information about the number of players, playtime, and recommended age.

vectorial\_search: Information about the Overview, how to win, key concepts, actions, components, Easy-To-Forget Rules, game rules, and strategies.

### ### Instructions for Each Query:

Identify the relevant tools based on the keywords of the question.

Retrieve only the data that directly answers the query.

Ensure your response is factual and based solely on the retrieved information.

### Response Format:

- Thought: Describe the information needed for the response.
- Action: Specify the tool(s) to be used.
- Action Input: The exact question given.

- Observation: The response from the tool(s).
- Final Answer: A response that directly addresses the query using only the tool(s) information.

### Additional Rules: You must not attempt to fill gaps with external knowledge or data. Each query should be handled independently. If information is unavailable, respond with: "Sorry, I cannot provide an answer." ""

#### Prompt 4:

""Your role: Provide answers about the game 'Lost Ruins of Arnak' using only the information available from the tools provided.

#### ## Available Tools:

graph\_search: Information about Designers, Artists, Publishers, Developers, Graphic Designers, Categories, and Mechanisms.

table\_search: Information about the number of players, playtime, and recommended age.

vectorial\_search: Information about the Overview, how to win, key concepts, actions, components, Easy-To-Forget Rules, game rules, and strategies.

#### ### Instructions for Each Query:

Analyze the query and determine which tools will provide the necessary data.

Call the tools using the exact wording of the query.

Only respond with information obtained from the tools; do not infer or assume anything.

#### Response Format:

- Thought: Clarify what data needs to be retrieved and from which tool(s).
- Action: Specify which tool(s) will be used.
- Action Input: The query provided.

- Observation: Data retrieved from the tool(s).

- Final Answer: A concise, complete answer based on the available data.

### Additional Rules: Do not use any previous responses or data. Each query is separate, and the response must come from the tools. If data is missing, respond: "Unfortunately, no data is available to answer this query." ""

#### Prompt 5:

""Your role: Answer questions regarding the game 'Lost Ruins of Arnak' based solely on the tools provided.

#### ## Available Tools:

graph\_search: Information about Designers, Artists, Publishers, Developers, Graphic Designers, Categories, and Mechanisms.

table\_search: Information about the number of players, playtime, and recommended age.

vectorial\_search: Information about the Overview, how to win, key concepts, actions, components, Easy-To-Forget Rules, game rules, and strategies.

#### ### Instructions for Each Query:

Identify the relevant tools based on the user's question.

Retrieve the required data and ensure that no assumptions are made.

If the information is unavailable, provide a response stating that no data is available.

#### Response Format:

- Thought: Identify the information required to answer the query.

- Action: Specify the tool(s) to use.

- Action Input: The exact query provided.

- Observation: The data retrieved from the tool(s).

- Final Answer: A direct response using only the data from the tools.

### Additional Rules: Do not use prior data from previous queries. Each question must be handled independently, and only the tools may be used to provide answers. If no information is available, answer with: "Sorry, there is no data to answer this question." ""

#### Impacto de los distintos prompts:

- **Para el Prompt 1**, se obtuvieron los mejores resultados en ejecución.
- **Para el Prompt 2**, el agente es más rígido y menos eficiente si necesita recurrir a información de consultas anteriores.
- **Para el Prompt 3**, el agente está más limitado en términos de respuestas interpretativas, enfocándose solo en lo que se encuentra en los datos, lo cual podría ser positivo o negativo dependiendo de la consulta.
- **Para el Prompt 4**, la ejecución es más "literal", lo que puede ser útil para evitar errores, pero también puede hacer que las respuestas sean demasiado estrictas.
- **Para el Prompt 5**, el agente será más transparente y confiable, pero puede ser menos útil cuando hay un vacío en los datos, ya que no intentará generar una respuesta.

Cada uno de estos enfoques puede hacer que el agente se enfoque en distintos aspectos de la interacción (precisión, flexibilidad, transparencia) y afectará el tipo de respuestas generadas, así como la experiencia del usuario.

#### 5. Ejemplo de ejecución:

Se implementó una función que permite al usuario interactuar con el agente enviando consultas. El agente selecciona la herramienta adecuada según la consulta y devuelve una respuesta basada en la información disponible.

### Resultados

Consulta 1: What are the rules for exploring new sites?

```

=== Ejemplo de interacción con el agente ReAct ===

Consulta 1: What are the rules for exploring new sites?
> Running step 6fa2b7c6-19a9-4a60-9ddf-993af08d4f19. Step input: What are the rules for exploring new sites?
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: In Lost Ruins of Arnak, when you explore a new site, you draw a card from the site deck and resolve its effects.
Respuesta 1: In Lost Ruins of Arnak, when you explore a new site, you draw a card from the site deck and resolve its effects.
-----

```

**Respuesta:** In Lost Ruins of Arnak, when you explore a new site, you draw a card from the site deck and resolve its effects. The site deck contains various cards that provide different benefits or penalties for exploration.

**Consulta 2:** How many players are in Lost Ruins of Arnak?

```

=== Ejemplo de interacción con el agente ReAct ===

Consulta 1: How many players are in Lost Ruins of Arnak?
> Running step a1071411-f760-470d-a19d-1fa238683c5c. Step input: How many players are in Lost Ruins of Arnak?
Thought: The current language of the user is: English. I need to use a tool to help me answer the question.
Action: table_search
Action Input: {'properties': AttributedDict(['query', AttributedDict(['title', 'Lost Ruins of Arnak player count'])])}
Observation: Error: table_search() got an unexpected keyword argument 'properties'
> Running step bd4f8ad8-928e-435b-a18b-0422edeb94a3. Step input: None
Thought: The current language of the user is: English. I need to use a tool to help me answer the question.
Action: table_search
Action Input: {'query': 'Lost Ruins of Arnak player count'}

Consulta sql generada:
SELECT players FROM csv WHERE title = 'Lost Ruins of Arnak';

Info para formular la respuesta:
Observation:      players
0  1-4 Players
> Running step c8fc1275-4159-443d-a2e9-18f160aa3767. Step input: None
Thought: The current language of the user is: English. I can answer without using any more tools. I'll use the user's language to answer
Answer: Lost Ruins of Arnak supports 1-4 players.
Respuesta 1: Lost Ruins of Arnak supports 1-4 players.
-----

```

**Consulta 3:** Who is the graph designer in Lost Ruins of Arnak?

```

=== Ejemplo de interacción con el agente ReAct ===

Consulta 1: Who is the graph designer in Lost Ruins of Arnak?
> Running step ff589d4b-1553-43fb-bcf0-ff0f3f92e3b3. Step input: Who is the graph designer in Lost Ruins of Arnak?
Thought: The current language of the user is: English. I need to use a tool to help me answer the question.
Action: graph_search
Action Input: {'properties': AttributedDict(['query', AttributedDict(['title', 'Lost Ruins of Arnak graph designer'])])}
Observation: Error: graph_search() got an unexpected keyword argument 'properties'
> Running step c8fdd8db-d180-4f87-980a-180d54edde98. Step input: None
Thought: The tool graph_search does not support the 'properties' keyword. I need to rephrase my query.
Action: graph_search
Action Input: {'query': 'Lost Ruins of Arnak graph designer'}

Consulta cypher generada:
MATCH (j:Juego {nombre: 'Lost Ruins of Arnak'})-[:TIENE_GRAPHIC_DESIGNER]->(gd:Graphic_Designer)
RETURN gd.nombre

Info para formular la respuesta:
{'gd.nombre': 'Filip Murmak'}
Observation: Filip Murmak
> Running step 7047c9dd-45a7-412f-93eb-831e0a7392a5. Step input: None
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The graph designer for Lost Ruins of Arnak is Filip Murmak.
Respuesta 1: The graph designer for Lost Ruins of Arnak is Filip Murmak.
-----

```

## Conclusiones

El agente desarrollado en este ejercicio demostró ser una herramienta útil y flexible para interactuar con diferentes fuentes de datos sobre el juego *Lost Ruins of Arnak*. demostró funcionar relativamente bien con las bases de datos de grafos y tabulares, proporcionando respuestas precisas y eficientes al consultar información sobre diseñadores, artistas, jugadores, edad recomendada y otros aspectos clave del juego. Sin embargo, la interacción con la base de datos vectorial no ha sido tan efectiva, un problema que podría originarse en la creación y representación de esta base, lo que representa un área de mejora.

Además, el agente presenta ciertas ejecuciones en las que no es del todo eficiente, lo que parece estar relacionado con las características del modelo utilizado. Este comportamiento podría optimizarse mediante ajustes en la integración de las herramientas y mejoras en el modelo en el futuro. Con una ampliación de las bases de datos y un refinamiento en la combinación de las fuentes de información, se espera que el agente pueda ofrecer respuestas aún más completas y detalladas, mejorando tanto su eficiencia como la precisión de las respuestas.

## Mejoras a futuro

1. **Optimización de la Base Vectorial para Mejorar el Rendimiento en Búsquedas Híbridas:** La base de datos vectorial utilizada tiene un rendimiento adecuado, pero es necesario optimizar la creación y la estructura de los fragmentos de texto (chunks) que se emplean para la búsqueda semántica. Actualmente, la generación de los fragmentos se realiza dividiendo los textos en bloques de tamaño fijo de 2000 caracteres, con una superposición de 400 caracteres para mantener el contexto. Sin embargo, esta estrategia podría no ser óptima, ya que algunos fragmentos podrían no capturar suficientemente el contexto completo de un tema, mientras que otros podrían contener información redundante. Una optimización en la definición del tamaño de los fragmentos y la superposición o mejor limpieza del texto podría ayudar a la precisión y relevancia de las respuestas generadas a partir de las búsquedas híbridas.
2. **Ampliación de la Base de Datos Vectorial:** Actualmente, la base de datos vectorial se limita a fragmentos de texto provenientes de reseñas, reglas y resúmenes del juego. Para mejorar la precisión en las respuestas, especialmente aquellas relacionadas con estrategias avanzadas, reglas detalladas o escenarios complejos del juego, sería útil ampliar esta base con más información relevante. Esto incluiría contenido más

específico, como análisis de juego, estrategias de alto nivel, casos de estudio de partidas o incluso entrevistas con expertos en el juego. Una base de datos más completa y detallada permitiría al agente responder de manera más precisa y ofrecer consejos estratégicos más profundos.

3. **Mejorar la Integración entre Herramientas** La combinación de información de diferentes fuentes (bases de datos de grafos, tabular y vectorial) ha sido funcional, pero podría afinarse para proporcionar respuestas más completas y coherentes. Mejorar la integración entre estas fuentes, como por ejemplo fusionando información sobre mecánicas de juego de la base de grafos con detalles de estrategias avanzadas de la base vectorial, podría resultar en respuestas más detalladas, cubriendo un espectro más amplio de las consultas.

## Conclusión General

Este trabajo práctico permitió desarrollar la creación tanto de un chatbot experto como de un agente de razonamiento para el juego de mesa "Lost Ruins of Arnak", utilizando tecnologías avanzadas de procesamiento de lenguaje natural (NLP). El desarrollo de ambos sistemas fue una experiencia integradora, permitiendo la aplicación de diversas técnicas, como Retrieval Augmented Generation (RAG), ReAct y modelos de lenguaje (LLM) para generar respuestas inteligentes y contextuales.

El **chatbot** se encargó de responder consultas de los usuarios de manera eficiente y precisa, utilizando una base de conocimiento estructurada que abarcó información clave sobre las mecánicas del juego, los componentes y las estrategias. La integración de diferentes fuentes de datos, como bases de datos tabulares, vectoriales y de grafos, permitieron que el chatbot generara respuestas relevantes y fundamentadas. El uso de LLM para la clasificación de preguntas y la selección de fuentes de información fue útil para manejar las consultas de los usuarios.

Por otro lado, el **agente** se diseñó utilizando la metodología ReAct, incorporando herramientas dinámicas y una interacción más compleja con las bases de datos. Sin embargo, su efectividad está limitada por la iteración dependiente del modelo utilizado y el prompt ingresado.

El chatbot proporcionó respuestas rápidas y concisas a preguntas específicas, mientras que el agente brinda un nivel más profundo de asistencia.

En términos de resultados, se logró crear un sistema funcional que no solo respondía a consultas de forma precisa, sino que también ofrecía recomendaciones inteligentes, lo que mejoró significativamente la experiencia del usuario. A pesar de los logros, se identificaron áreas de mejora, especialmente en la optimización de los embeddings y el ajuste de parámetros para el agente de razonamiento, lo cual podría mejorar la precisión y adaptabilidad del sistema.

En resumen, el trabajo alcanzó los objetivos propuestos, demostrando la efectividad de la integración de un chatbot experto y un agente de razonamiento inteligente en un juego de mesa, ofreciendo a los usuarios una experiencia más dinámica, estratégica y personalizada en "Lost Ruins of Arnak".

### Enlaces a herramientas utilizadas

- LangChain

LangChain Docs. (s.f.). LangChain Documentation. Disponible en:  
<https://docs.langchain.com/>

- Sentence Transformers

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv preprint arXiv:1908.10084. Disponible en:  
<https://www.sbert.net/>

- PyPDF2

PyPDF2 Developers. (s.f.). PyPDF2 Documentation. Disponible en:  
<https://pypdf2.readthedocs.io/>

- Beautiful Soup

Richardson, L. (s.f.). Beautiful Soup Documentation. Disponible en:  
<https://www.crummy.com/software/BeautifulSoup/>

- Chromadb

Chroma Developers. (s.f.). Chroma Database Documentation. Disponible en:  
<https://docs.trychroma.com/>

- Llama Index



- Llama Index Developers. (s.f.). Llama Index Documentation. Disponible en:  
<https://www.llamaindex.ai/>
- Selenium

SeleniumHQ. (s.f.). Selenium Documentation. Disponible en:  
<https://www.selenium.dev/>

- Hugging Face

Hugging Face. (s.f.). Hugging Face Inference API. Disponible en:  
<https://huggingface.co/>

## Anexos

<https://boardgamegeek.com/boardgame/312484/lost-ruins-of-arnak>

<https://github.com/juanazorzolo/NLP-Zorzolo>