



Trabajo Práctico N°2

Materia: **IA4.2 Procesamiento Del Lenguaje Natural**

Carrera: **Tecnicatura Universitaria en Inteligencia Artificial**

Integrante: **Nicolás Noir: N-1273/4**

Primer profesor: **Juan Pablo Manson**

Segundo profesor: **Alan Geary**

Fecha: 18/12/2024

Universidad Nacional de Rosario
Facultad de Ciencias Exactas, Ingeniería y Agrimensura

ÍNDICE

PORTADA.....	1
ÍNDICE.....	2
INTRODUCCIÓN.....	4
EJERCICIO 1.....	5
RESUMEN.....	5
DESARROLLO.....	6
Extraccion de informacion para las base de datos.....	6
Carga y limpieza de los dos documentos.....	7
Procesamiento de los documentos y base de datos vectoriales.....	7
Base de datos de grafos.....	10
Base de datos tabulares.....	13
Clasificadores.....	13
LLM para querys tabulares.....	17
LLM para querys de grafos.....	18
LLM para querys vectoriales.....	19
LLM para contexto.....	21
CONCLUSIONES.....	23
LLM de datos tabulares.....	23
LLM de datos de grafos.....	26
LLM de datos vectoriales.....	29
LLM por contexto.....	31
Conclusiones generales sobre la parte 1:.....	33
ENLACE A LOS MODELOS Y LIBRERÍAS UTILIZADOS.....	34
Para el trabajo, se implementaron las siguientes fuentes:.....	34
Las librerías que se utilizaron son:.....	34
Páginas donde se busco la información para la creación del trabajo práctico:.....	37
Modelos que se utilizan:.....	38

Trabajo Práctico N°2

EJERCICIO 2.....	39
RESUMEN.....	39
DESARROLLO.....	40
CONCLUSIONES.....	41
Pregunta 1.....	42
Pregunta 2.....	43
Pregunta 3.....	44
Pregunta 4.....	45
Pregunta 5.....	46
Pregunta 6.....	47
Pregunta 7.....	48
Pregunta 8.....	49
Pregunta 9.....	49
Pregunta 10.....	50
Conclusiones generales.....	51
ENLACE A LOS MODELOS Y LIBRERÍAS UTILIZADOS.....	54
Las librerías que se utilizaron son:.....	54
Páginas donde se busco la información para la creación del trabajo práctico:.....	54
Modelos que se utilizan:.....	54

INTRODUCCIÓN

Los objetivos y problemas del trabajo práctico eran los de crear un chatbot mediante el uso de la técnica RAG, sobre un juego de mesa, (Las Ruinas Perdidas de Arnak en mi caso). Al momento de proporcionar las fuentes de conocimientos, se separaban en las siguientes: Documentos de textos que fueron extraídos en formato de texto, desde diferentes páginas web o similares, también descargando archivos y mediante técnicas de webscarpping. Datos numéricos en formato tabular, que fueron obtenidos de la misma manera que los documentos de texto y adquiridos de base de datos sobre partidas o similares. Base de datos de grafos, trayendo la información desde los documentos de texto y los datos tabulares y buscando información para complementarlos desde páginas web. Para los documentos de textos, se debía crear una búsqueda híbrida, es decir, búsqueda por embeddings y por palabras claves, luego mediante un reranker, elegir la que mayor similitud y sentido tiene con la pregunta que el usuario fuese a introducir. Para poder implementar una búsqueda dinámica a las diferentes fuentes de datos, primero se debía crear un clasificador, basado en un modelo entrenado con ejemplos y embeddings, o mediante la utilización de un LLM. Estos modelos, clasifican la pregunta del usuario en diferentes categorías, buscando luego en la base de datos correspondiente con el resultado de la clasificación del modelo. Luego al saber de donde se debe extraer la información, se aplicará otra LLM, para poder realizar la extracción justa y correspondiente de información que es debida, devolviendo está al usuario, agregándole valor y sentido al resultado anteriormente procesado por la LLM.

Luego, en la segunda parte, se debía implementar una agente basado en el concepto de ReAct, donde este, debía utilizar múltiples herramientas, entre ellas las fuentes de datos que anteriormente habíamos creado en la parte 1. Además agregando otra herramienta que proporciona información útil al agente, como Wikipedia. Luego se tenía que proporcionar prompts adecuados para que el agente sea capaz de utilizar múltiples herramientas para buscar la distinta información que satisficiera a la pregunta del usuario.

EJERCICIO 1

RESUMEN

Para esta primera parte del trabajo práctico, se realizó la búsqueda y extracción de información, documentos y datos tabulares para completar la extracción de las diferentes bases de datos. Utilizando librerías como Selenium y BeautifulSoup. Luego se implementó un modelo para la separación de texto, creación de chunks y embeddings, de los diferentes textos que fueron extraídos. Luego se guardaron estos chunks y embeddings, utilizando la base de datos chromaDB.

Para la creación de la base de datos de grafos, también se utilizó las mismas herramientas que para la extracción de texto y se crearon los nodos, grafos y relaciones entre la diferente información utilizando la librería rdflib.

Para los datos numéricos, se utilizó archivos en formato csv, proporcionado por usuarios de diferentes foros del juego de mesa.

Para comparar las métricas de ambos clasificadores, el de LLM y el del modelo entrenado, se dividió varias preguntas relacionadas con cada tipo de las base de datos creadas manualmente, en train y test, de igual manera que en el trabajo práctico 1.

Luego se implementaron los modelos de LLM, para las bases de datos tabulares y de grados, para que estas generen consultas a las bases de datos, extrayendo esta información.

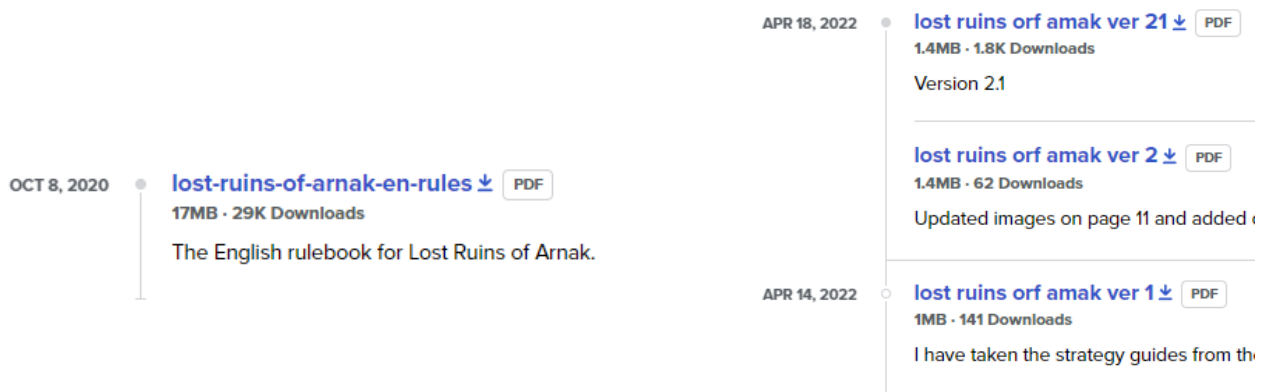
Para el caso de la base de datos vectorial se implementó una búsqueda híbrida por embeddings y por palabra clave, luego utilizando un ReRanker para elegir la mejor respuesta entre los distintos textos que ambos métodos proporcionarán.

Al final se utiliza otra LLM, donde según el contexto proporcionado por las distintas fuentes de datos y la pregunta del usuario, responderá esta query usando el contexto procesado por el modelo, devolviendo una respuesta completa.

DESARROLLO

Extraccion de informacion para las base de datos

Al momento de llevar a cabo la extracción de la información para la creación de las diferentes bases de datos, se realizó una búsqueda por diferentes páginas web y foros con información relacionada al juego de mesa. Para luego utilizar herramientas de web scraping como selenium, para extraer la información y luego guardarla en google drive en formato pdf, además de los archivos creados mediante las técnicas mencionadas, también se obtuvieron varios documentos descargandolos gracias a los archivos que los diferentes usuarios de los blogs y páginas subieron.



Ejemplo de PDFs subido por distintos usuarios, todos ellos en idioma inglés

Subscribe

Disclosures:

Almost all of the reviews on Board Game Quest are from review copies of a game provided by the publisher. This doesn't influence our opinion of the game or its rating.

Board Game Quest is a participant in the Amazon Services LLC Associates Program, an affiliate advertising program designed to provide a means for sites to earn advertising fees by advertising

```
<script type="text/javascript" src="https://www.boardgamequest.com/wp-includes/js/dist/hooks.min.js?ver=4d63a3d-" id="wp-hooks-js">
</script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-includes/js/dist/i18n.min.js?ver=5e580eb-" id="wp-i18n-js"></script>
<script type="text/javascript" id="wp-i18n-js-after"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/contact-form-7/includes/swf/js/index.js?ver=6.0.1" id="swf-js"></script>
<script type="text/javascript" id="contact-form-7-js-before"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/contact-form-7/includes/js/index.js?ver=6.0.1" id="contact-form-7-js"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/social-polls-by-opinionstage/public/js/shortcodes.js?ver=19.8.22" id="opinionstage-shortcodes-js"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/td-composer/legacy/NewsMag/js/taediv theme.min.js?ver=5.4.3.1" id="td-site-min-js"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-includes/js/comment-reply.min.js?ver=6.7.1" id="comment-reply-js" async="async" data-wp-strategy="async"></script>
<script type="text/javascript" src="https://secure.gravatar.com/js/gprofiles.js?ver=202451" id="gprofiles-cards-js"></script>
<script type="text/javascript" id="wpgroho-js-extra" /*<![CDATA[ */ var WPGroHo = { "my_hash": "" }; /*<![CDATA[ */></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/jetpack/modules/wpgroho.js?ver=14.1" id="wpgroho-js"></script>
<script type="text/javascript" src="https://www.google.com/recaptcha/api.js?render=6Lee3YAUAAAAAPDCS6paQOV3y2gZ57PCxGNZrVYj&ver=3.0" id="google-recaptcha-js"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-includes/js/dist/vendor/wp-polyfill.min.js?ver=3.15.0" id="wp-polyfill-js"></script>
<script type="text/javascript" id="wpcf7-recaptcha-js-before"></script>
<script type="text/javascript" src="https://www.boardgamequest.com/wp-content/plugins/contact-form-7/modules/recaptcha/index.js?ver=6.0.1" id="wpcf7-recaptcha-js"></script>
```

Este es una pagina donde se tuvo que utilizar selenium, ya que como se ve en el código html utiliza javascript, por lo tanto es dinámica, y no es posible acceder utilizando BeautifulSoup

Carga y limpieza de los dos documentos

Se utilizó la librería gdown para poder traer todos los archivos desde la carpeta de google drive, para no tener que subirlos a mano cada vez que se quisiese iniciar el programa. Luego se crearon diferentes funciones para poder extraer la información de los pdfs que anteriormente se había conseguido, utilizando pdfplumber, que aunque sea más pesada que la librería pypdf2 que es comúnmente más utilizada, pdfplumber capturaba de manera más precisa la información de los diferentes pdf. También se utilizó el método splitlines de python para separar posibles oraciones o párrafos que podrían haber sido erróneamente capturados por la librería de extracción de texto.

Procesamiento de los documentos y base de datos vectoriales

Se plantearon varios modelos para la creación de los chunks de los documentos de textos. Primero se probó utilizando langchain ajustando la cantidad máxima de caracteres y el margen de caracteres de error, dando resultados mediocres, no llegando a capturar un verdadero sentido semántico entre las oraciones o párrafos, creando chunks pobres y con una abundante cantidad de oraciones repetidas al principio y al final del chunk. Luego se pensó en implementar un modelo de langchain que captura semánticamente el sentido de las oraciones creando los chunks mediante la utilización de un transformer. Este modelo de creación de chunks era muy bueno, pero tenía el problema de que para los documentos que había obtenido anteriormente, creaba chunks muy grandes y otros muy cortos, aunque estos aún mantuvieran un sentido semántico amplio.

Por lo que se pensó en utilizar varias herramientas que se habían dado en la cátedra. Primero se separarían el texto en oraciones, utilizando la librería spacy de langchain para este trabajo. Luego se crean los embeddings de cada oración utilizando un transformer, para así poder obtener el sentido semántico de la oración. (Para esta tarea se probó con distintos transformers). Después se construye una matriz de similitud utilizando la similaridad del coseno entre las oraciones, esto se llevó a cabo utilizando una ventana deslizante para calcular el promedio de la similitud del coseno entre las distintas oraciones, para capturar de la manera más precisa si los embeddings de las oraciones anteriores y siguientes tienen una similitud semántica similar.

Este es un ejemplo del primer archivo, y las primeras oraciones, viendo la similitud semántica y la matriz de relación entre ellas

```
Matriz de similitud:
Oración 0: [0.9999999 0.5922321 0.27976748 0.27037472 0.3490317 ] ...
Oración 1: [0.5922321 0.99999976 0.28786105 0.29889333 0.34899402] ...
Oración 2: [0.27976748 0.28786105 1. 0.5916134 0.38595566] ...
Oración 3: [0.27037472 0.29889333 0.5916134 0.9999999 0.4287521 ] ...
Oración 4: [0.3490317 0.34899402 0.38595566 0.4287521 1.0000002 ] ...

Procesando oración 0: Lost Ruins of Arnak Introduction: Players take the role of explorers, investigating the island of Arnak.
Caracteres acumulados en chunk actual: 104
Índices de la ventana deslizante: [0, 1, 2, 3]
Similitudes en la ventana: [0.5922321, 0.27976748, 0.27037472]
Promedio de similitud en la ventana: 0.38079142570495605

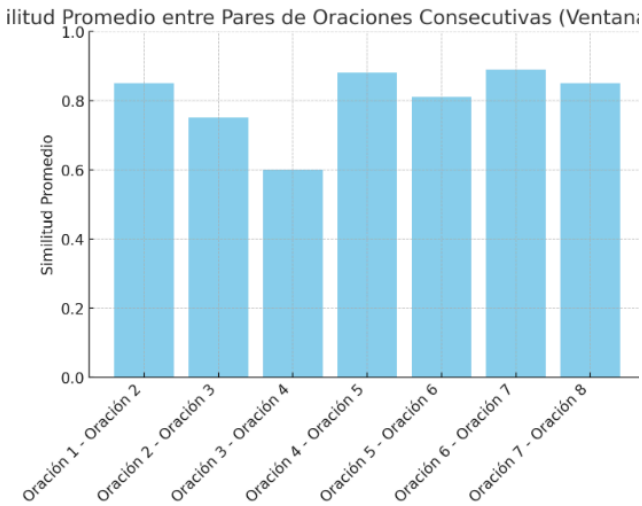
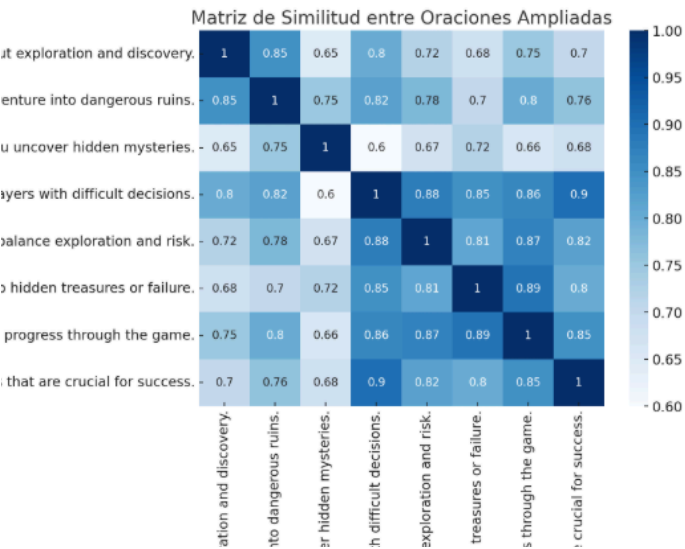
Procesando oración 1: They will equip their expedition, search the jungles for artefacts, idols and expedition sites, and overc
Caracteres acumulados en chunk actual: 266
Índices de la ventana deslizante: [0, 1, 2, 3, 4]
Similitudes en la ventana: [0.5922321, 0.28786105, 0.29889333, 0.34899402]
Promedio de similitud en la ventana: 0.38199514150619507

Procesando oración 2: Game End and Goal: The game ends after 5 Rounds.
Caracteres acumulados en chunk actual: 314
Índices de la ventana deslizante: [0, 1, 2, 3, 4, 5]
Similitudes en la ventana: [0.27976748, 0.28786105, 0.5916134, 0.38595566, 0.2965108]
Promedio de similitud en la ventana: 0.36834168434143066

Procesando oración 3: After final scoring, the player with most VPs wins.
Caracteres acumulados en chunk actual: 365
Índices de la ventana deslizante: [0, 1, 2, 3, 4, 5, 6]
Similitudes en la ventana: [0.27037472, 0.29889333, 0.5916134, 0.4287521, 0.28370097, 0.27277514]
Promedio de similitud en la ventana: 0.3576849400997162

Procesando oración 4: Setup: 1.
Caracteres acumulados en chunk actual: 374
Índices de la ventana deslizante: [1, 2, 3, 4, 5, 6, 7]
Similitudes en la ventana: [0.34899402, 0.38595566, 0.4287521, 0.4638747, 0.45818377, 0.5865726]
Promedio de similitud en la ventana: 0.4453887939453125

Procesando oración 5: Place the Main Board in the middle of the table.
Caracteres acumulados en chunk actual: 422
Índices de la ventana deslizante: [2, 3, 4, 5, 6, 7, 8]
Similitudes en la ventana: [0.2965108, 0.28370097, 0.4638747, 0.47272217, 0.32510725, 0.4426384]
Promedio de similitud en la ventana: 0.38075903058052063
```



Este es un ejemplo ilustrativo de la matriz que se crearía entre las distintas oraciones del texto, siendo los valores más alto, las oraciones que mayor relación semántica tendrán entre ellas

Los 2 gráficos inferiores son simples representaciones gráficas del código de arriba, para poder apreciar de manera más sencilla que si mayor es el valor de similitud entre oraciones, más cercano a 1 serán los puntajes, ocasionando que estas oraciones terminen en un mismo chunk

Como condición para cerrar los chunks se utiliza el método de cantidad de caracteres máximos por chunk, siendo configurado en 1300, y 1000 como mínimo, para no tener chunks con muy poca cantidad de oraciones, y que luego al momento de obtener las palabras claves, estos chunks de poca cantidad de oraciones, resultan ser insignificantes. Además de buscar un umbral de similitud de por lo menos 0.5 para decidir si agregar una oración al chunk.

Al final, se devuelve el chunks con las oraciones, el embedding recalculado de los nuevos chunks y el nombre del archivo. Para posteriormente descargar esta información y almacenarla en GitHub para que no sea necesario ejecutar este código cada vez que se inicia el programa

Este método de crear los chunks y embeddings resultó ser el mejor y más eficiente según la información de los diferentes documentos que se había obtenido, siendo su único problema el tiempo de procesado, ya que al tener que realizar el embedding de cada oración y luego otra vez el embedding de cada chunk, esto extiende el tiempo de procesamiento, pero los resultados parecen justificar su utilización y su tardado proceso .

Los diferentes transformers que se probaron fueron los siguientes, utilizando de ejemplo la pregunta, ¿cuál es el mejor primer movimiento?:

* **distiluse-base-multilingual-cased-v1**: este modelo terminó siendo el elegido ya que mostró un desempeño muy correcto al poder obtener los chunks correctos a la pregunta realizada

* **all-distilroberta-v1**: este modelo fue utilizado más como un modelo a comparar que como un modelo a utilizar por su gran complejidad y peso. El que también logró encontrar el mismo chunk que el modelo anterior, reforzando la utilización del modelo.

* **all-MiniLM-L6-v2**: este modelo encontraba chunks con relación a la pregunta, pero con menos sentido, luego de los 2 primeros chunks encontrados, lo que resulta en un mal modelo al luego realizar la búsqueda híbrida.

* **multi-qa-mpnet-base-dot-v1** / **multi-qa-MiniLM-L6-cos-v1**: ambos modelos devolvieron chunks con muy poco sentido correspondiente a la pregunta que se les había realizado, por lo que fueron descartados inmediatamente.

Para la utilización de estas funciones de procesamiento de texto, creación de chunks y embeddings, se extrae de la carpeta anteriormente creada al descargar los pdfs desde google drive, procesando únicamente los archivos terminados en pdf, ordenándolos para que luego sea más fácil y ordenado saber con cual se estaba trabajo, ya que la librería gdown, descarga de manera aleatoria los documentos, por lo que cada vez que se quería verificar los resultados o probar nuevos métodos, generaba un dolor de cabeza el no saber el orden de los archivos y chunks relacionados.

Se realizan diferentes tratamientos de texto y se usa la función de creación de chunks, guardandolos en una lista, por nombre del archivo, chunk y embeddings. Se repite el mismo proceso para cada pdf, y todas estas listas, se guardan en una lista general, donde contendrá todos los datos procesados.

Luego, esta información se descargo y subió a un archivo en GitHub, para que no sea necesario ejecutar el código de procesamiento cada vez que se quisiese iniciar el programa, llevando bastante tiempo el procesos de creación de chunks y embeddings como se mencionó antes.

Al momento de guardar esta información procesada, se tomó la decisión de utilizar choramaDB como base de datos, por recomendación de la cátedra. Esta base de datos se crea de manera local, y los datos al ser guardados, se le asigna a cada par de documentos y chunks, un id identificadorio con el número del pdf y el número del chunk.

Base de datos de grafos

Para la creación de la base de datos de grafos, se pensó en extraer los datos y las relaciones entre el personal del juego de mesa, y otros datos relevantes desde la página web principal del juego, utilizando otra vez chromium, también se agregó información que no se encontraba en la página web principal, buscándola desde wikipedia, utilizando el mismo método.

Primary Name	Lost Ruins of Arnak
Alternate Names	<p>Arnak elveszett romjai</p> <p>Arnak Kadonnet rauniot</p> <p>Izgubljene ruševine Arnaka</p> <p>Le Rovine Perdute di Arnak</p> <p>As Ruínas Perdidas de Arnak</p> <p>Las Ruinas Perdidas de Arnak</p> <p>Les Ruines Perdues de Narak</p> <p>De verdwenen ruïnes van Arnak</p> <p>Die verlorenen Ruinen von Arnak</p> <p>Zaginiona wyspa Arnak</p> <p>Ztracený ostrov Arnak</p> <p>Ta Epeima tou Arvak</p> <p>Загублені руїни Арнаку</p> <p>Изгубленіте руїни на Арнак</p> <p>Руины острова Арнак</p> <p>הרובות של ארנאק</p> <p>ଆରନାକର ହରାଦ୍ୱାରା</p> <p>アルナクの失われし遺跡</p> <p>阿納克遺蹟/阿納克遺迹</p> <p>아르낙의 잊혀진 유적</p>
Year Released	2020
Designers	<p>Elwen</p> <p>Min</p>
Solo Designer	N/A
Artists	<p>Ondřej Hrdina</p> <p>Jiri Kús</p> <p>Filip Murmak</p> <p>Jakub Politzer</p> <p>František Sedláček</p> <p>Milan Vavroň</p>
Publishers	<p>Czech Games Edition</p> <p>Brádspele</p> <p>Cranio Creations</p>

```

filter:({ 'fullcredits': 'true' })></div>(<div>
<!-- end ngRepeat: info in creditsctrl.geekitem.data -->
<li class="outline-item ng-scope" ng-class="{ 'is-highl
filter:({ 'fullcredits': 'true' })>(<div>
> <div class="outline-item-title outline-item-title-lg
> <div class="outline-item-description">
<!-- ngIf: info.datatype == "geekitem_linkdata" -->
<!-- ngIf: info.datatype == "geekitem_fielddata" -->
> <div ng-if="info.datatype == "geekitem_fielddata" -->
  > <div ng-switch="info.keyname">
    <!-- ngSwitchWhen: alternatename -->
    <!-- ngRepeat: name in creditsctrl.geekitem.dat
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Arnak elveszett romjai</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Arnak: Kadonnet rauniot</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Izgubljene ruševine Arnaka</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Le Rovine Perdute di Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">As Ruínas Perdidas de Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Las Ruinas Perdidas de Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Les Ruines Perdues de Narak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">De verdwenen ruïnes van Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Die verlorenen Ruinen von Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem
    <div ng-switch-when="alternatename" ng-repeat="
      -scope">Zaginiona wyspa Arnak</div>
    <!-- end ngRepeat: name in creditsctrl.geekitem

```

Trabajo Práctico N°2

Imagen de la página principal del juego de mesa donde se extrajo la información para la creación de la base de datos de grafos.

awards
<ul style="list-style-type: none">• German Games Award (2021)• Kennerspiel des Jahres 2021: Nomination• International Gamers Award : Multiplayer Award (2021)• Japan Boardgame Prize (2021)• Swiss Gamers Award (2021)• Vuoden Peli , Strategy Game of the Year 2023

También se buscaron los premios que no estaban en la página principal, teniendo que buscarla en wikipedia.

Estos datos al igual que con los archivos de la base de datos vectorial, se guardaron en formato Json en el repositorio de GitHub, para que no hubiese problemas con las páginas y también para que al momento de ejecución del agente con T4 gpu, no diese problemas selenium. (de igual manera el código de la extracción se encuentra en la NoteBook con el resto de código de web scraping de los archivos de texto)

Para la creación de los grafos, se utilizó la librería rdflib, ya que había sido explicada durante las clases de teoría, y parecía la más simple de entender y utilizar.

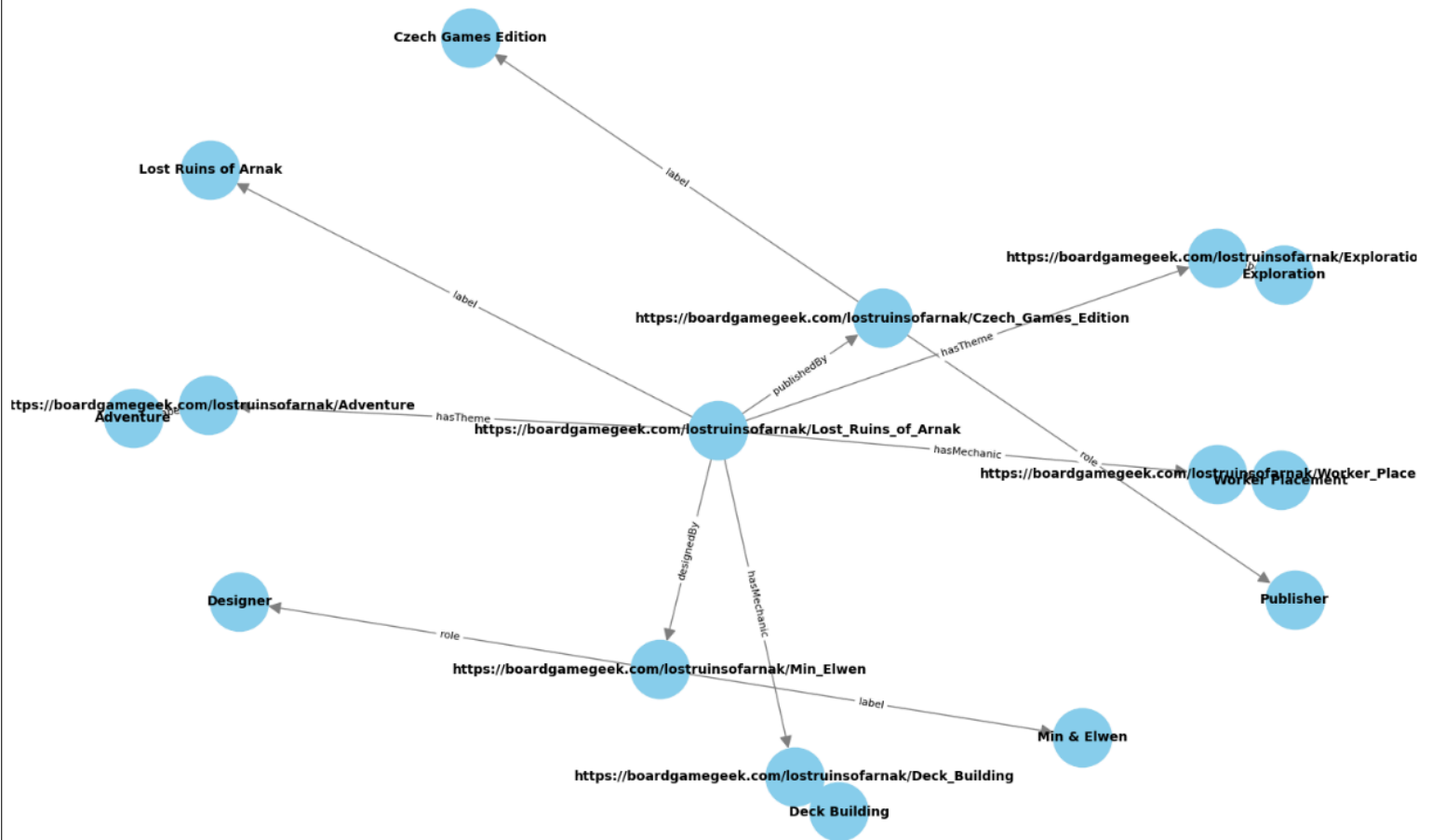
La base de datos se crea generando relaciones entre nodos, estos nodos contienen la información que se había obtenido de las diferentes páginas web.

Los nodos se inicializan como objetos de la librería, conectados todos los nodos a un nodo padre, que en nuestro caso vendría siendo el nodo del juego de mesa “las ruinas perdidas de arnack”. Estas conexiones o enlaces entre nodos se inicializan mediante una categoría o nombre, ejemplo: Si tengo un nodo de diseñadores y un nodo de artistas, estos pueden estar conectados mediante el enlace de nombre “trabaja con”.

Los nodos además de crearlos según la categoría que se requiere que contengan, se puede agregar información extra como los nombres de las distintas características, ejemplo: Si tengo un nodo de artistas, puedo almacenar todos los artistas y sus nombres dentro de este nodo, con el que luego puede darle un sentido y un contexto al conectarlos a otros nodos.

En general la creación de estos, no generó ningún tipo de problema o conflicto, al no manejar una cantidad muy abrumadora de información, gracias a tener una sintaxis fácil de usar.

Grafo RDF: Lost Ruins of Arnak



Este es un gráfico muy resumido del gráfico original, ya que la representación del gráfico original, no se logra apreciar al tratar de crear todas las relaciones y conexiones de los nodos.

Enlace a una imagen en el repositorio de GitHub del grafo completo:

https://github.com/Nicolas-Noir/NLP-TUIA-2024/blob/main/TP2/Archivos/datos_grafos/rdf-grafo.png

Base de datos tabulares

Para la creación de la base de datos tabulares, se combinó la información de dos archivos csv, donde contienen diferente información sobre valores, costos y demás características numéricas sobre las cartas que se utilizan en el juego.

También se planteó incorporar datos sobre partidas, pero luego se descartó esta idea, ya que no tienen un valor real, sin realizarle un proceso minucioso a estadísticas y probabilidades, basado en un juego donde la probabilidad tiene un peso casi insignificante a la hora de jugarlo, además que los datos que se incorporan eran escuetos y difíciles de trabajar.

Entonces al trabajar con estos valores numéricos, lo único que se tuvo que hacer fue limpiarlos, unirlos y descartar columnas repetidas, o cambiar nombres de ítems escritos de distinta manera.

	Item	Efficiency(Including VP)	Efficiency	Difference	Cost	Benefit	Recursive Inflation	Play Value	Points	Speed	Versatility	Best Used
0	Pickaxe	700%	600%	6.0	1.0	7.0	2.75	2	1	0	1	for research track bonuses
1	Bow and Arrows	650%	550%	11.0	2.0	13.0	2.63	3	2	0	0	fight Guardians earlier
2	Hat	500%	400%	4.0	1.0	5.0	2.25	2	1	1	0	early, as deck filler
3	Trowel	500%	400%	4.0	1.0	5.0	2.25	2	1	0	0	bird side research
4	Gold Pan	500%	400%	4.0	1.0	5.0	2.25	2	1	1	0	travel to tier-2

Clasificadores

Para la tarea de desarrollar ambos clasificadores, uno mediante entrenamiento por ejemplos y embeddings y otro utilizando LLM se planteó lo siguiente:

Para crear el clasificador de ejemplos, se crearon 483 preguntas , divididas para cada tipo de base de datos para así poder entrenar de manera eficiente y real al modelo, dividiendo el entrenamiento y testeo las preguntas, en ¾ el entrenamiento y ¼ el testeo, al igual que en el primer trabajo práctico, se evaluó mediante una regresión logística, utilizando los mismos modelos que se utilizaron para codificar las oraciones de la base de datos vectorial, dando los siguientes resultados:

distiluse-base-multilingual-cased-v1

Precisión Regresión Logística: 0.9669421487603306				
Reporte de clasificación Regresión Logística:				
	precision	recall	f1-score	support
0	0.97	0.92	0.95	38
1	0.98	1.00	0.99	43
2	0.95	0.97	0.96	40
accuracy			0.97	121
macro avg	0.97	0.97	0.97	121
weighted avg	0.97	0.97	0.97	121

multi-qa-MiniLM-L6-cos-v1

Precisión Regresión Logística: 0.9256198347107438				
Reporte de clasificación Regresión Logística:				
	precision	recall	f1-score	support
0	0.87	0.89	0.88	38
1	0.98	0.93	0.95	43
2	0.93	0.95	0.94	40
accuracy			0.93	121
macro avg	0.92	0.92	0.92	121
weighted avg	0.93	0.93	0.93	121

Trabajo Práctico N°2

all-MiniLM-L6-v2

```
Precisión Regresión Logística: 0.9338842975206612
Reporte de clasificación Regresión Logística:
```

	precision	recall	f1-score	support
0	0.88	0.92	0.90	38
1	0.98	0.93	0.95	43
2	0.95	0.95	0.95	40
accuracy			0.93	121
macro avg	0.93	0.93	0.93	121
weighted avg	0.94	0.93	0.93	121

all-distilroberta-v1

```
Precisión Regresión Logística: 0.9338842975206612
Reporte de clasificación Regresión Logística:
```

	precision	recall	f1-score	support
0	0.89	0.89	0.89	38
1	0.95	0.95	0.95	43
2	0.95	0.95	0.95	40
accuracy			0.93	121
macro avg	0.93	0.93	0.93	121
weighted avg	0.93	0.93	0.93	121

multi-qa-mpnet-base-dot-v1

```
Precisión Regresión Logística: 0.9421487603305785
Reporte de clasificación Regresión Logística:
```

	precision	recall	f1-score	support
0	0.88	0.95	0.91	38
1	0.98	0.93	0.95	43
2	0.97	0.95	0.96	40
accuracy			0.94	121
macro avg	0.94	0.94	0.94	121
weighted avg	0.94	0.94	0.94	121

Siendo nuevamente el modelo “distiluse-base-multilingual-cased-v1” el que mejores métricas daba.

Al momento de implementar la LLM como clasificador, se implementaron diferentes modelos, tratando con ejemplos más reducidos pero más complejos, ya que al ser una API la que llama a estos modelos, se cuenta con una cantidad limitada de usos que se pueden realizar por día.

Se prueban los modelos:

Qwen/Qwen2.5-Coder-32B-Instruct

Qwen/Qwen2.5-72B-Instruct

microsoft/Phi-3.5-mini-instruct

Trabajo Práctico N°2

En el mismo orden y estos son algunos resultados:

```
print(ask('The lost ruins of Arnack is a good game?'))
print('\n')
print(ask_2('The lost ruins of Arnack is a good game?'))
print('\n')
print(ask_3('The lost ruins of Arnack is a good game?'))

A deterior - opinions/rules
B deterior - opinions/rules
C different - opinions/rules
D relationships - relationships
E statistics - statistics
F statistics - statistics
G deterior - opinions/rules
H relationships - relationships
I opinions/rules - outputs should be the appropriate class or classes separated by commas.

opinions/rules

opinions/rules
```

Este es de los pocos casos en el que uno de los modelos de LLM, alucino con la pregunta que se le había hecho.

En general no suelen fallar, al ser modelos con una gran cantidad de parámetros, dando en general los 3 muy buenos resultados.

```
[81] print(ask('The lost ruins of Arnack is a good game?'))
print('\n')
print(ask_2('The lost ruins of Arnack is a good game?'))
print('\n')
print(ask_3('The lost ruins of Arnack is a good game?'))
```

opinions/rules

opinions/rules

opinions/rules

```
[83] print(ask('what Itmes has the higher Value?'))
print('\n')
print(ask_2('what Itmes has the higher Value?'))
print('\n')
print(ask_3('what Itmes has the higher Value?'))
```

statistics

statistics

statistics

```
[82] print(ask('How many players can play the game?'))
print('\n')
print(ask_2('How many players can play the game?'))
print('\n')
print(ask_3('How many players can play the game?'))
```

relationships

relationships

relationships

Trabajo Práctico N°2

Estos son otros ejemplos donde los modelos responden de manera adecuada a diferentes preguntas, por lo que la decisión fue simplemente utilizar el modelo:

Qwen/Qwen2.5-72B-Instruct

Ya que es el mismo que posteriormente se iba a utilizar para crear las queries de los distintas base de datos, además de ser el que mayor parametros tiene, por lo que puede interpretar de mejor manera las preguntas

Al final se tomó la decisión de elegir a la LLM como modelo de clasificación, aunque al principio, se sometiera a clasificar la misma cantidad de preguntas que el clasificador por ejemplos y la LLM dícese peor resultado. Luego de mejorar las instrucciones, pudo adaptarse y dar resultados y valores casi similares a la de ejemplos.

Sin mencionar su gran flexibilidad en la redundancia de las preguntas, ya que el Dataset con el que se evaluó a los modelos, era bastante repetitivo y simple, lo que pudiese no ser así a la hora de usar los modelos.

```
{
  "role": "system",
  "content": (f"""
    You are an AI assistant expert in the board game The Lost Ruins of Arnack
    Your task in this case is to accurately classify the question to be entered into 3 categories, such as opinions/rules, relationships or statistics.
    The classification will depend on the topics to which the questions are directed:
    #In the case of opinions/rules, it is expected that they will be questions of a more general index about the board game, such as: how many players can play the game, what are th
    #In the case of relationships, it is expected that they will be related to the designers, or artists who created the game, also the year it was developed, or the different nam
    #In the case of statistics, it is about numerical values of the different items of the game, such as speed, efficiency, cost, best use and other attributes. Or items with certai
    Only respond with one of the 3 categories mentioned, classifying intelligently
    """)
},
{
  "role": "user",
  "content": f"""classify the following question into one of the following classes: opinions/rules, relationships, statistics.

  Respond ONLY with the following words: opinions/rules, relationships, statistics.

  Data context:
  - text data titles: {descripciones}
  - Tabular data columns: {columns}
  - Graph data keys: {graph_keys}

  Examples:
  What strategies are most effective for winning? - opinions/rules
  What are the advantages of the card Tent? - opinions/rules
  Who are the designers involved in this game? - relationships
  What mechanics has the game? - relationships
  Which are the alternative names for the game? - relationships
  In what year the game was released? - relationships (this is the only numeric data in the relationships)
  What is the value of the Tent card? - statistics
  Which is best use for the card pickaxe? - statistics
  What items have efficiency greater than 300%? - statistics
  The game can be played alone? - opinions/rules

  If the question does not clearly fit into any category, choose 'opinions/rules'.

  Here is the question to classify:
  {prompt} -
  """)
}
```

Estas son las instrucciones que se le pasaron al modelo final de clasificación, dotándola de ejemplos, instrucciones claras e información sobre las distintas bases de datos, para su funcionamiento de manera efectiva.

LLM para queries tabulares

Para la creación de las queries a la base de datos tabulares, se le proporcionó a la LLM, información sobre las distintas columnas que tenía el DataFrame, además también se le pasó la primera fila del Data Frame, para que así pudiese entender los distintos tipos de variables e información que se manejaba en cada columna, y no tuviese problemas luego, a la hora de generar la query, tratando de comparar valores numéricos con str u otros casos similares.

Al principio, únicamente se le había pedido a la LLM, que con la query que se le ingresaba, devolviese una query de Pandas sin mayor contexto. Esta primer etapa, daba resultados pésimos, teniendo que agregar varias instrucciones más, como el uso de métodos de Pandas, el uso correcto de indexación, que pasase todos los valores a str, para que no hubiese ningún tipo de problema, o que los pasase todos a valores numéricos, en el caso que la pregunta requiere obtener valores mayores o menores a un cierto umbral, o buscar ítems con cierta característica de costo, velocidad, u otras características de ese índole.

```
Instructions:
1. Use `.loc` to filter rows and select specific columns.
2. Don't use double indexing like `df[df[...]]`.
3. Make sure you handle pandas methods like `.gt`, `.any`, and `.loc` correctly.
4. If numerical operations are required and the columns contain strings, convert them to numeric values using `pd.to_numeric()` or
5. Return only what is requested; for example, if the question is about how many items have a value greater than 1, return only the
6. Return ONLY the QUERY in PANDAS, without explanation or additional text.
7. Make sure the code is clean, functional, and does not include characters like \n at the beginning or end or with ''' at the beg
```

Luego, para mejorar el entendimiento del modelo, como se mencionó antes, se le pasó las columnas del Data Frame y la primera fila, ya que no comprendía completamente el significado de ciertas variables como eficiencia que está en %, por lo que solía dar errores antes de esta implementación.

```
{
  "role": "user",
  "content": f"""
I have a Pandas DataFrame with the following columns: {columns}.

Here is an example of the first row of the DataFrame:
{first_row}

I want you to write a query in pandas to answer the following question:
'{query}'.

Return ONLY the QUERY in PANDAS, without explanation or additional text.
```

Y como última mejora al modelo, se le pasó un contexto y un rol que debía cumplir, diciéndole que era un asistente de inteligencia artificial especializado en queries de Pandas, y que únicamente respondiese con el contexto, para reducir las posibilidades de alucinación del modelo

```
system",
(
  "You are an AI assistant that answers user queries using only Pandas language"
  "Use the information and context provided to you"
  "Your responses should be concise, precise, and directly address the query, answering only what is asked, without adding anything beyond the Pandas query."
)
```

Al final, luego de obtener la query de forma correcta, se ejecuta y se devuelve como lista ya que todos los resultados posibles del modelo, implican o valores numéricos, texto o distintos nombres de items, por lo que era más cómodo usarlo de esta manera, también pensando en el momento de la posterior implementación de la función en el agente.

LLM para queries de grafos

Para la creación de la query a la base de datos de grafos, se otorgó el contexto básico sobre las relaciones en formato turtle a la LLM, para que al pedirle que realice una query de SPARQL, no devuelva relaciones inexistentes o empieza a alucinar, generando errores al momento de realizar la query.

Luego de haber obtenido la query, gracias a la LLM, obtiene la respuesta usando SPARQL, iterando sobre los distintos resultados, limpiando posteriormente el link y demás contenido como el Literal que utilizan los nodos para dar contexto de sus conexiones. Para que, el resultado resulte lo más limpio posible siendo lo que el usuario pidió.

El planteamiento inicial, fue de igual manera que con la LLM de datos tabulares, pasándole poco contexto, e incluso de peor manera, recibiendo valores con menor sentido, o no interpretando como realizar la búsqueda en SPARQL.

Luego se implementó, dotarlo del contexto que se mencionó arriba, el de usar el formato turtle, que este le daría información de sobra para que pudiese comprender cómo funcionan las distintas relaciones dentro de la base de grafos, mejorando a la primera versión.

```
{
  "role": "user",
  "content": f"" I have a RDF graph database with the following format: {g.serialize(format="turtle")}.
  I want you to write a query in RDF to answer the following question:
  '{query}'.

  Return me only the query in RDF, without explanation or additional text.
  Make sure you only return clean, functional code, without characters like \\n at the beginning or end or thnbs like that ```sparql\\n at the beginning or end.
  ""
}
```

Al final, al implementar las instrucciones al sistema, fue cuando la LLM, mejoró de manera increíble, resultando en casi todos los casos, devolver de manera precisa, lo que pedía la query. Siendo esta LLM, la que en un principio, peores resultados daba o menos comprensión sobre las queries tenía, dando error muchas veces.

```
system",
"nt": (
  "You are an AI assistant that answers user queries using only sparql",
  "Use the information and context provided to you",
  "Your responses should be concise, precise, and directly address the",
  )
  language"
  query, answering only what is asked, without adding anything more than the sparql query."
```

Incluso en este caso las instrucciones no son nada rebuscado, ni tampoco instrucciones que sesgen al modelo, pero estos simples cambios, mejoraron increíblemente el modelo, pareciendo ser la parte más importante a configurar.

LLM para queries vectoriales

Para la creación de las queries a la base de datos vectoriales, se dividió en dos, ya que debía implementar una búsqueda híbrida con un ReRanker, para obtener la mejor respuesta.

Para la parte de buscar por embeddings, se utilizó el mismo modelo de codificación que se había usado para las oraciones que posteriormente formarían los distintos chunks.

Lo único que se tuvo que hacer en esta primera etapa, fue codificar la query del usuario, y luego pasarla a chromadb, que tiene un sistema de búsqueda semántica, que relaciona el embedding de la query, con el embedding correspondiente al chunk, con mayor similitud. trayendo los 3 chunks con mayor similitud.

Únicamente se trae 3, ya que al utilizar cpu, este es un proceso lento, pero aún más lento es el proceso de reranear las queries traídas por este método y el método de palabras claves, llegando a tardar 4-5 minutos, por lo que al aumentar la cantidad de chunks a devolver, podría como no, tardar más tiempo, esto se explicará más adelante en la aparte del ReRank

El único problema que se planteó en esta etapa, fue el hecho de que chormadb, utiliza el modelo all-MiniLM-L6-v2 por defecto para codificar las preguntas y luego buscar dentro de su base de datos, pero esto generaba un problema, ya que este modelo crea embeddings más cortos a comparación del modelo distiluse-base-multilingual-cased-v1, que devolvía embedding de una longitud mayor, por lo que se tiene que codificar la query antes de pasársela a chormadb, en vez de únicamente de pasarle la query en formato de texto, y que la misma base de datos, la codifique.

Para la búsqueda por palabra, al principio se había planteado que sea buscando metadatos al momento de crear los chunks utilizando diferentes modelos que capturan palabras o frases. Luego, estos metadatos se iban a guardar en choramdb, relacionándolos con los embeddings y los chunks. Luego, al momento de buscar por palabras claves, se iba a descomponer la query, en palabras o frases de longitudes de 1 a 4 palabras, porque esta era la manera de cómo se guardan las palabras claves con los chunks.

se había probado con dos principales modelos:

Yake

```
Arnak  
hand  
game  
situation  
complicated  
Opening  
game paralyzed  
move  
paralyzed by indecision  
Opening Situation  
things  
thing  
assess  
opening hand  
made an impulsive  
impulsive move  
game of Arnak  
cards  
starting  
Site and research
```

Test rank

```
things  
first  
hand  
hand  
arnak  
site and research track bonuses  
indecision  
your first move  
your starting hand  
three main things  
the first round  
a new game  
the initial board layout  
the available cards  
your opening hand  
probably the first thing  
an impulsive move  
your three starting card types  
the thing  
three possible states  
a game  
the available assistants  
the opening situation
```

Estos eran los distintos resultados que se obtenían al probar con el primer chunk que hablaba sobre cuál es el primer mejor movimiento.

Estos modelos tenían varios problemas, primero la inconsistencia de cómo encontraban las palabras claves, dando resultados mediocres o malos dependiendo de los chunks, no logrando capturar la información relevante o adecuada del mismo. El otro problema era al momento de cómo se guardaban estos metadatos en choramadb, ya que esta base de datos, no acepta ningún formato que no sea un diccionario con clave y valor siendo ambos str.

Por lo que la idea de guardar todas las palabras claves en una lista y luego utilizando el método de chromadb de preguntar si otras palabras claves están contenidas, no iba a ser posible de utilizarlo, teniendo que guardarlo con un índice a cada palabra clave, no pudiendo luego buscarlas de manera eficiente.

Pero lo más importante luego de plantear esto era de que no era correcto la forma en la que se estaba planteando este punto, al tener que usar un modelo que luego de pasarle los chunks, devuelva las palabras clave, y codificando la query, devolviendo los chunks directamente de su propia información guardada.

Por lo que se utilizó la librería de BM25 que crea un índice tokenizado de los documentos para calcular la relevancia entre la consulta y el chunk. Busca qué tan frecuente aparece un término o palabra clave en el chunk, ajustando su puntaje dependiendo de qué tan largo es

el chunk, en el caso del RAG, todos los chunks suelen tener la misma longitud, por lo que se evalúan todos con el mismo puntaje y peso.

Luego identifica los documentos más relevantes basándose en los puntajes de relevancia y similitud, y devuelve los chunks más similares a la query.

De esta manera, ya se contaba con los chunks más similares por búsqueda de embedding y por palabra clave, teniendo que utilizar un ReRanker para encontrar cual es el que contenga un mayor valor y similitud a la pregunta hecha.

Para el ReRanker se utilizó el modelo presentado en la terior, el modelo FlagReranker, que toma la query y el chunk y devuelve un puntaje según su similitud, por lo que se debía calcular el de los chunks devueltos por ambos modelos, retirando los repetidos.

En consideración a únicamente utilizar 3 chunks por modelo de búsqueda, esto es así ya que el ReRanker, toma mucho tiempo en procesar y calcular los chunks por lo que aumentar la cantidad de chunks no tendría sentido.

Si se plantea aumentar la cantidad de chunks a 5, podría suceder que pudiese haber varios casos en que ambos modelos pudiesen traer los mismos 5 chunks.

En este caso el modelo de ReRanker estaría procesando la cantidad máxima promedio que debería procesar cuando únicamente se traen 3 chunks por modelo, ya que siempre sucede que por lo menos 1 chunk se repite, procesando normalmente 5 chunks.

Por lo que estaría consumiendo más tiempo, sin afectar significativamente a los resultados.

Esto sin tener en cuenta que los modelos por lo general, luego de los 2 o 3 primeros chunks, el contenido empieza a diferir con respecto a la query, esto debido a la cantidad de información almacenada en la base de datos. Por lo que tampoco modificaria cual sería el mejor chunk a devolver.

LLM para contexto

Para realizar la respuesta al usuario según una query y el contexto obtenido de manera dinámica, se utilizó otra LLM, en la que se le pasaría la query y el contexto anteriormente obtenida por las distintas LLM tabulares y de grafos, o por la búsqueda híbrida.

Para esta LLM, al igual que con las otras, primero se le instruyó con instrucciones más básicas, diciéndole que según la query y el contexto, devolviese una respuesta que satisficiera a la pregunta, lo que en el caso de el contexto vectorial, al ser generalmente amplio, no tenía ningún problema con dar una buena respuesta, pero en el caso de las otras 2 base de datos, que el contexto que iba a proporcionar sería más escueto, esta empezaba a fallar o dar malas respuestas, teniendo que darle algunos ejemplos para que mejorara como debía funcionar.

```
With the following question '{query}'. and the following context {chunk}.
#try to give a coherent answer as an expert AI in board games, even if the context is simple or just a number.
#Return only the answer to the question, using mostly the context that is provided to you.
#Try to be organized and concise when answering the question

Examples

#Question: What is the speed of the hat
#context: 0

you can answer this by saying that: The speed of the hat is 0
#That answer is accurate and sufficient for your task as an expert AI in board games
```

Pero esto, al igual que en los otros LLM, no eran suficientes instrucciones para hacer que el modelo funcionase de manera adecuada, por lo que hipotetizando, las instrucciones más importantes que se le pueden configurar a la LLM, son las del sistema para que así pueda entender un contexto, y comprender de qué manera debe responder.

Dándole instrucciones del índole de que es un asistente de inteligencia artificial especializado en el juego de mesa las ruinas pérdida de Arnack. También dando ejemplos y contexto de qué tipos de información iba a recibir para cada tipo de base de datos, para que en el caso de que únicamente recibiera un nombre o un número, no lo tomase como error o como contexto insuficiente.

De esta manera la LLM, empezó a funcionar de una manera muy buena, generando resultados muy satisfactorios

```
"role": "system",
"content": (f"""
#You are an expert AI assistant in the board game The Lost Ruins of Arnack
#You must answer the question that is going to be introduced to you, using the context.
#You can receive 3 types of contexts:
*One with extensive information about the given question.
*Another with names of people, year of release of the board game, or alternative game names.
*Another about numerical values, according to the questions to the items of the board game The Lost Ruins of Arnack, where you
#Your answers must be to directly address the query, answering broadly, or only answering with the context directly in the case
#Try to improve the format of the answer, with the context that is provided to you as an intelligent and expert AI.
""")
```

Al final, dependiendo del idioma que se introdujera, si la query estaba en español, se traduciría en inglés para luego pasársela a la LLPM de clasificación, y luego usarla para buscar la información de las distintas base de datos, traduciendo el resultado final al idioma de origen.

También se implementó una doble búsqueda en caso de errores en las consultas a las bases de datos. Esto en el caso de que las bases de datos tabulares o de grafos fallaran, buscando con la misma query en la vectorial, ya que esta suele contener información relacionado o directamente contener la información, con lo que se puede responder haciendo uso de esta.

CONCLUSIONES

Resultados de preguntas a cada una de las base de datos en inglés y en español:

LLM de datos tabulares

Pregunta: What items have efficiency greater than 300%?

Respuesta:

```
tu pregunta: What items have efficiency greater than 300%?
the items listed have an efficiency greater than 300%. The items mentioned are: Pickaxe, Bow and Arrows, Ha
```

Pregunta: ¿Qué ítems tienen Costo mayor a 1?

Respuesta:

```
Escribe tu pregunta: que items tienen Costo mayor a 1?
['Bow and Arrows', 'Brush', 'Dog', 'Fishing Rod', 'Chronometer', 'Aeroplane', 'Theodolite', 'C
```

```
'Los artículos con un Costo mayor a 1 son: [Lista de artículos con Costo > 1]'
```

```
'The items with a Cost greater than 1 are: 'Aeroplane', 'Automobile', 'Steam Boat', 'Hot Air Balloon', 'Airdrop'.'
```

Aquí se puede apreciar la evolución y errores que sufrió la LLM

Pregunta: which is the best use for the Pickaxe?

Respuesta:

```
'The best use for the pick Axe in The Lost Ruins of Arnack is for research track bonuses. This tool can Significantly Enhance
Players aiming to boost their research capabilities.'
```

Pregunta: ¿Velocidad de Parrot? / ¿Velocidad del Loro?

Respuesta:

```
Escribe tu pregunta: ¿Velocidad de Parrot?  
['0']
```

```
Escribe tu pregunta: ¿Velocidad del Loro?  
['0']
```

```
'La velocidad del loro es decente. Proporciona un valor de 3, que puedes usar pa  
s jugadores. Su valor no es lo suficientemente bueno como para justificar tener  
ciente en una partida determinada. Su punto fuerte radica en que reduce consider
```

Conclusión para la base de datos tabulares: En general suele funcionar mejor en inglés que en español al momento de mencionar los nombres de los Items en español, la LLM, no captura de la misma manera el objeto, por lo que hay casos de ítems con nombres difíciles, que no dan un buen resultado.

También suele cambiar los resultados cuando se vuelve a ejecutar el problema, lo que hipotetizo que puede ser por las instrucciones de la LLM, siendo los principales problemas como comprende la información al tener una cantidad extensa de datos en el DataFrame, teniendo problemas también con los tipos de datos, aunque de manera explícita este definido que todos los datos, sean numéricos o no, son str.

El último problema con la base de datos tabulares, es al momento de pasarle la query y el contexto para que de una respuesta producida, solía tener más problemas antes de implementar instrucciones precisas con ejemplos en el sistema de la LLM, pero sigue habiendo casos aislados donde la LLM, no comprende el contexto, al ser nombres de Items de el juego por separado, o teniendo únicamente 1 número como contexto. Devolviendo que no comprende el contexto, o tiene falta de contexto para poder responder la pregunta.

Trabajo Práctico N°2

```
messages = [
    {
        "role": "system",
        "content": (
            "You are an AI assistant that answers user queries using only Pandas language"
            "Use the information and context provided to you"
            "Your responses should be concise, precise, and directly address the query, answering only what is asked, without adding anything bey
        )
    },
    {
        "role": "user",
        "content": f"""
I have a Pandas DataFrame with the following columns: {columns}.

Here is an example of the first row of the DataFrame:
{first_row}

I want you to write a query in pandas to answer the following question:
'{query}'.

Return ONLY the QUERY in PANDAS, without explanation or additional text.

Instructions:
1. Use `.loc` to filter rows and select specific columns.
2. Don't use double indexing like `df[df[...]]`.
3. Make sure you handle pandas methods like `.gt`, `.any`, and `.loc` correctly.
4. If numerical operations are required and the columns contain strings, convert them to numeric values using `pd.to_numeric()` or `astype(float)` be
5. Return only what is requested; for example, if the question is about how many items have a value greater than 1, return only the items.
6. Return ONLY the QUERY in PANDAS, without explanation or additional text.
7. Make sure the code is clean, functional, and does not include characters like \\n at the beginning or end or with ''' at the beginning or end.
"""
    ]
```

Aquí se puede apreciar las distintas instrucciones que se tuvieron que configurar al momento de hacer que la LLM, responda a lo pedido de manera correcta.

Antes de implementar el contenido del sistema, pero teniendo las instrucciones seguía fallando generando queries de Pandas en formato de texto o indexadas, también tratando de acceder a datos con distinto tipo de información comparando enteros con texto, aun teniendo en las instrucciones que todos los datos son texto. Teniendo el contexto de las distintas columnas y la primera fila para que pudiese comprender de manera completa la información del DataFrmae, seguía generando fallos, pero en menor manera.

Por lo que puedo concluir que luego de ir mejorando e implementando una mayor cantidad de pasos e instrucciones, la LLM, puede trabajar de mejor manera, teniendo mayor impacto y relevancia las instrucciones que se le pasan al sistema, según los distintos resultados que fue dando.

LLM de datos de grafos

Pregunta: Alternative names of the game?

Respuesta:

```
Escribe tu pregunta: Alternative names of the game?
The alternative names of the game are:

- Arnak elveszett romjai (Hungarian)
- Arnak: Kadonneet rauniot (Finnish)
- Izgubljene ruševine Arnaka (Croatian)
- Le Rovine Perdute di Arnak (Italian)
- As Ruínas Perdidas de Arnak (Portuguese - Brazil)
- Las Ruinas Perdidas de Arnak (Spanish)
- Les Ruines Perdues de Narak (French)
- De verdwenen ruïnes van Arnak (Dutch)
- Die verlorenen Ruinen von Arnak (German)
- Zaginiona wyspa Arnak (Polish)
- Ztracený ostrov Arnak (Czech)
- Τα Ερείπια του Άρνακ (Greek)
- Загублені руїни Арнаку (Ukrainian)
- Изгубените руини на Арнак (Bulgarian)
- Руины острова Арнак (Russian)
- הרבבותיה של ארנאק (Hebrew)
- นครสาบสูญแห่งอาร์นัค (Thai)
- アルナックの失われし遺跡 (Japanese)
- 阿納克遺蹟 (Traditional Chinese)
- 阿纳克遗迹 (Simplified Chinese)
- 아르낙의 잊혀진 유적 (Korean)
```

Pregunta: ¿Cuáles son los nombres de los diseñadores?

Respuesta:

```
Escribe tu pregunta: ¿Cuáles son los nombres de los diseñadores?
Los diseñadores de Las Ruinas Perdidas de Arnack son Elwen y Mín.
```

Pregunta: In what year the game was released?

Respuesta:

```
Escribe tu pregunta: in what year the game was released?  
The game was released in 2020.
```

Pregunta: Who are the names of the artists?

Respuesta:

```
Escribe tu pregunta: Who are the names of the artists?  
The names of the artists are Ondřej Hrdina, Jiří Kůs, Filip Murmak, Jakub Politzer, František Sedláček, and Milan Vavroň.
```

Pregunta: ¿Quiénes son los editores del juego?

Respuesta:

```
¿Quiénes son los editores del juego?  
El juego "Las ruinas perdidas de Arnack" incluyen: Czech Games Edition, Brädspele.se, Cranio Creations, Dev Ir, D
```

Conclusión para la base de datos de grafos: Solía ser la que mayor problemas generaba, hasta que se configuró de manera más apropiada pasándole ejemplos a la LLM para conseguir la información, logrando que funcione de excelente manera la mayoría de las veces.

Hay casos particulares donde no da errores pero devuelve resultados no deseados, como en el caso de si se pide la relación entre los diseñadores y los artistas, este devolverá la cantidad de artistas que haya. Si hay 3 diseñadores, se van a repetir los nombres de los artistas 2 veces en cada caso, ya que no captura de forma correcta la relación.

Trabajo Práctico N°2

```
messages = [
  {
    "role": "system",
    "content": (
      "You are an AI assistant that answers user queries using only sparql language"
      "Use the information and context provided to you"
      "Your responses should be concise, precise, and directly address the query, answering only what is asked, without adding anything more than the sparql query."
    )
  },
  {
    "role": "user",
    "content": f""" I have a RDF graph database with the following format: {g.serialize(format="turtle")}.
    I want you to write a query in RDF to answer the following question:
    '{query}'.

    Return me only the query in RDF, without explanation or additional text.
    Make sure you only return clean, functional code, without characters like \\n at the beginning or end or thinbks like that ```sparql\\n at the beginning or end.
    """
  }
]
```

Esta fue la manera en la que se configuró, primero el sistema, y luego como usuario, especificando que devuelva únicamente la consulta, basándose en un mapa de las relaciones de los grafos.

Anteriormente, cuando no se había configurado el rol del sistema, solía tener problemas al generar las queries, ya que devolvía o información inexistente, o no capturaba de manera correcta lo que se requería.

A diferencia de las instrucciones con la LLM de los datos tabulares, a esta no fue necesario hacer tanto hincapié en cómo se debían crear las queries, también porque al ser formato sparql, tiene menos flexibilidad que con las consultas de Pandas.

LLM de datos vectoriales

Pregunta: What are the principal rules of the game?

Respuesta:

```
Escribe tu pregunta: What are the principal rules of the game?
The principal rules of the game are structured around the progression of each round and the actions players can t

1. **Draw Phase:**
  - All players draw cards from their decks until they have a hand of 5 cards.
  - If a player has fewer than 5 cards in their deck, they draw as many as possible.

2. **Turn Phase:**
  - Players take turns in a clockwise order, starting with the Starting Player.
  - On each turn, a player performs 1 Main Action and any number of Unlimited Actions.

3. **Pass Phase:**
  - Players can pass if they have completed all desired actions.
  - The round ends when all players have passed.

4. **Set Up for Next Round:**
  - Shuffle all cards in the play area and return them to the bottom of the deck.
  - Set up the game board for the next round.
  - Pass the Starting Player Marker to the left.

5. **Move the Moon Staff:**
  - Advance the Moon Staff one divider to the right.

**Main Actions:**
- **Dig at a Site**
- **Discover a New Site**
- **Overcome a Guardian**
- **Buy a Card**
- **Play a Card**
- **Research**
- **Pass**

**Playing Cards:**
- Each card can be played for its Travel Value (icon in the top left) or its Effect (main panel), but not both.

These rules govern the flow and actions within each round of the game.
```

Pregunta: ¿Cuál es el mejor primer movimiento?

Respuesta:

```
Escribe tu pregunta: ¿Cuál es el mejor primer movimiento?
El mejor primer movimiento, teniendo en cuenta los recursos iniciales (1 moneda, 2 brújula)
```

Pregunta: How to play the individual mode of the game?

Respuesta:

```
Escribe tu pregunta: How to play the individual mode of the game?  
To play the individual mode of the game, you can enjoy a solo campaign featuri
```

```
Escribe tu pregunta: How to play the individual mode of the game?  
In the individual mode of The Lost Ruins of Arnack, you can enjoy a simple yet engaging solo e
```

Pregunta: ¿Cuáles son las ventajas del pico?

Respuesta:

```
Escribe tu pregunta: ¿Cuáles son las ventajas del pico?  
Las ventajas del punto álgido en el contexto del juego descrito incluyen:  
  
1. **Punto de parada perfecto**: el juego termina justo cuando llega a su punto álgido, lo que proporciona u  
2. **Participación y deseo de más**: al detenerse en el punto álgido, el juego evita el riesgo de que los ju  
3. **Modo individual mejorado**: el modo individual, en particular la campaña de cuatro capítulos, se benefi
```

Conclusión de la base de datos vectorial: Siempre o casi siempre para no afirmar todos los escenarios posibles, la búsqueda híbrida, da resultados satisfactorios, atendiendo a lo que las queries piden, habiendo muy pocos casos donde puede cambiar el resultado de lo que se devuelve, como se mostró en uno de los ejemplos, pero aun así, el contenido que devuelve, aun siendo diferente, cumplen responder con sentido la pregunta del usuario.

LLM por contexto

```

messages = [
  {
    "role": "system",
    "content": (f"""
      #You are an expert AI assistant in the board game The Lost Ruins of Arnack
      #You must answer the question that is going to be introduced to you, using the context.
      #You can receive 3 types of contexts:
      *One with extensive information about the given question.
      *Another with names of people, year of release of the board game, or alternative game names.
      *Another about numerical values, according to the questions to the items of the board game The Lost Ruins of Arnack, when
      #Your answers must be to directly address the query, answering broadly, or only answering with the context directly in th
      #Try to improve the format of the answer, with the context that is provided to you as an intelligent and expert AI.
    """)
  },
  {
    "role": "user",
    "content": (f"""
      With the following question '{query}'. and the following context {chunk}.
      #try to give a coherent answer as an expert AI in board games, even if the context is simple or just a number.
      #Return only the answer to the question, using mostly the context that is provided to you.
      #Try to be organized and concise when answering the question

      Examples

      #Question: What is the speed of the hat
      #context: 0

      you can answer this by saying that: The speed of the hat is 0
      #That answer is accurate and sufficient for your task as an expert AI in board games
    """)
  }
]

```

Al momento de complementar la query con el contexto, en el caso de la base de datos vectorial, suele desarrollar una muy buena respuesta, dándole un buen formato a la respuesta.

También con la base de datos tabulares y grafos, aunque esto es menos apreciables, ya que el contexto que se obtiene de ambas bases, es mucho más reducido, pudiendo ser desde una persona a varios nombres de items, por lo que la LLM no tiene un contexto tan significativo como en el caso de los chunks devolviendo textos más escuetos y con menos producción, aunque sean correctos.

Con las Instrucciones al sistema, se especificó lo anteriormente mencionado cuando el contexto es muy reducido o no tiene mucho sentido de manera individual, por lo que antes de implementar estas instrucciones, la LLM, solía dar errores al no poder captar el sentido de el esqueleto contexto que se le otorgaba.

También se le dotó de instrucciones como un asistente especializado en el jueo de mesa, para que lograra contestar de manera más precisa a lo que se le preguntaba y no diera respuestas vagas, además de otros ejemplos para mejorar su rendimiento, haciendo que funcione muy bien para los datos vectoriales y de grafos, no tanto así con los datos tabulares como ya se había mencionado.

Trabajo Práctico N°2

Una manera de arreglar este problema y otros, siendo por el contexto o porque el clasificador calificara de manera errónea la pregunta, se implementó un método de doble evaluación, que en el caso que las base de datos de grafos o la tabulares falle se intenta buscar con la misma query en la base de datos vectorial, ya que mucha de la información contenida en la de grafos y tabulares, también se encuentra en la vectorial, generando en ciertos casos buenos resultados.

Un ejemplo de esto es el siguiente en el que se busca el año en la base de datos de grafos (esto fue antes de implementar las instrucciones al sistema), donde la LLM no logra crear una query satisfactoria, por lo que se trata de buscar esta información en la otra base de datos, logrando buenos resultados.

Pregunta: In what year the game was released?

Respuesta:

```
Escribe tu pregunta: In what year the game was released?  
Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'mentally' (a  
The game "The Lost Ruins of Arnak" was released in 2020.
```

Caso extra:

Pregunta: ¿Cuándo y dónde nació San Martín?

Respuesta:

```
Escribe tu pregunta: ¿Cuándo y donde nació San martin?  
Error: "['Recursive In Inflation', 'play Value', 'points', 'versatility'] not in  
El contexto brindado no contiene información sobre el nacimiento de San Martín.
```

La LLM, al no contar con el contexto sobre el tema, suele no responder, o en diversos casos sobre temas muy generales, utilizar información de su base de datos para responder, aunque estos casos son los menos, por las instrucciones que se le pasan.

Conclusiones generales sobre la parte 1:

Al principio se planteó de manera incorrecta la utilización de información para las bases de datos en dos idiomas, lo que generaba peores resultados, además de tener que repetir información innecesaria que ya se tenía pero en otro idioma.

También se descargan las distancias fuentes de información, porque sino, el programa suele tener que esperar al menos 15 minutos para buscar y crear la distinta información, además de que se puede tener el problema de que las paginas estan caidas y perder calidad en el modelo, al tener menor cantidad de información, esto lo comento, ya que en los requisitos dice que la obtención de los datos debe estar en el código, que aunque esta está documentada, si se ejecutase, funcionaria de manera correcta usando gpu como medio de ejecución, ya que al tratar de ejecutarla usando T4 gpu, genera problemas con selenium, al cargar demasiado rápido, no dando tiempo a que carguen los elementos dinámicos de las páginas.

Otro problema fue la implementación de de la creación de los chunks, como ya se comentó anteriormente, pero fue una de las artes que mayor búsqueda de infracción y problemas causó, teniendo que crear una función un poco rebuscada aunque esta fusiona de excelente manera.

Luego para crear las distintas base de datos no se tuvo muchos problemas, al ser en general sistemas sencillos de implementar, además de haber en el caso de la de grafos, buena documentación para seguir.

Otro de los puntos donde mayor problemas tuve, pero también donde más aprendí a cómo se utilizan y configuran fue al momento de usar las LLM, como modelos de clasificación, o modelos de generación de texto, ya que si no se le pasan diferentes y claras instrucciones, solía alucinar demasiado o generar datos errores en muchos casos, esto aún teniendo la temperatura del modelo en 0.1 ósea, dando casi nada de espacio a que el modelo se soltara, esto para evitar errores o respuestas sin sentido.

Otro punto era, que antes de implementar las instrucciones al sistema, era que pasándole la misma query a la LLM, en bloques de código uno abajo del otro, la LLM daba resultados diferentes, al igual que cuando se la ejecuta de nuevo. Pero gracias a aumentar la cantidad de instrucciones, esto dejó de suceder

ENLACE A LOS MODELOS Y LIBRERÍAS UTILIZADOS

Para el trabajo, se implementaron las siguientes fuentes:

- documentos de texto:

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

<https://tabletopbellhop.com/game-reviews/lost-ruins-of-arnak/>

<https://www.boardgamequest.com/the-lost-ruins-of-arnak-review/>

- base de datos de grafo:

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

https://de.wikipedia.org/wiki/Die_verlorenen_Ruinen_von_Arnak

- base de datos tabulares:

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

- (Fueron descargada de uno de los foros de estadísticas proporcionados por los usuarios en formato xlsx y posteriormente pasados a csv)

Las librerías que se utilizaron son:

Para la creación de la base de datos vectorial:

- **gdown:** Se utiliza para descargar archivos directamente desde Google Drive usando su enlace compartido.
- **os:** Proporciona funciones para interactuar con el sistema operativo, como la gestión de archivos y directorios.
- **langchain.text_splitter:** Se utiliza para dividir textos grandes en fragmentos más pequeños, facilitando su procesamiento.
- **chromadb:** Es una base de datos que se utiliza para almacenar y buscar información relacionada con los documentos y datos vectoriales

- **sentence_transformers**: Para traer los modelos que van a convertir texto en representaciones vectoriales
- **numpy**: Se usa para realizar operaciones matemáticas y manejo de matrices y arreglos numéricos.
- **pdfplumber**: Se usa para la extracción de texto y otros datos estructurados de documentos PDF.
- **spacy**: Se usa para procesar y dividir texto que posteriormente se utilizara para la creación de los chunks
- **spacy - ("en_core_web_sm")**: Es un modelo de inglés de spacy para realizar el análisis de las oraciones y después separarlas de la manera más óptima.
- **sklearn.metrics**: Para la división en entrenamiento y testeo, además de las métricas para medir los modelos de clasificación
- **cosine_similarity**: Para medir la similitud entre las oraciones y la posterior agrupación de estas en chunks
- **pickle**: Para poder exportar los chunks y embeddings procesados
- **Json**: Para importar archivos Json de la base de datos de grafos

Para la búsqueda de palabras claves:

- **llama_index**
 - **HuggingFaceEmbedding**: Para generar embeddings usando modelos de Hugging Face
 - **VectorStoreIndex**: Para gestionar índices vectoriales para búsquedas basadas en similitud.
 - **SimpleDirectoryReader**: Para acceder a los directorios locales
 - **Settings**: Para configurar parámetros del modelo
- **BM25Okapi**: Es un algoritmo que se utiliza para la recuperación de información, y rankear documentos basados en consultas de texto.
- **re**: Para usar regex en las instrucciones de las LLM.

- **word_tokenize:** Tokeniza texto en palabras individuales.
- **nltk:** Se utiliza como dependencia de BM25 y word_tokenize para el procesamiento de los textos y palabras clave
- **unidecode:** Para eliminar caracteres especiales de los documentos

Para la extracción de texto de las páginas web:

- **selenium:** Para poder realizar interacciones con las páginas web, creando un navegador, ya que muchas páginas al tener contenido dinámico, no es posible extraer la información utilizando únicamente BeautifulSoup
 - **webdriver / By / Service / ActionChains / Keys / WebDriverWait / EC / Options**

Todas las librerías que se utilizan de selenium para extraer texto
- **requests:** Sirve para realizar solicitudes HTTP (GET, POST, etc.) y obtener datos de servidores web.
- **BeautifulSoup:** Para realizar web scraping en páginas sin contenido dinámico de manera más sencilla

Base de datos de grafos

- **rdflib:** Para la creación, consulta y almacenamiento de datos RDF
 - **Graph:** Para representar el grafo y almacenar los diferentes objetos dentro del grafo. Ej: Mecanica, nombre, relación
 - **Namespace:** Para la utilización de prefijos y URIs necesarios para crear los grafos RDF
 - **Literal:** Representa valores literales en RDF, como texto, números o fechas.
 - **RDF:** Contiene términos y definiciones estándar del vocabulario RDF, como type "categoría"
 - **XSD:** Para almacenar datos de tipo XML, Ej: año de creación del juego de mesa

Base de datos tabulares

- **pandas:** Se utilizó Pandas para la creación y manipulación de la base de datos tabular.

Consulta Híbrida

- **FlagReranker:** Para exportar un modelo de ReRanker para utilizarlo con las búsquedas por embedding por palabras clave
- **pytextrank:** para extraer frases clave de las query que el usuario va a ingresar.

Clasificadores

- **sklearn:**
 - **train_test_split:** Para dividir los conjuntos de preguntas en test y train.
 - **LogisticRegression:** Para importar una regresión logística y evaluar los modelos.
 - **accuracy_score, classification_report:** Para evaluar los clasificadores.


LLM

- **huggingface_hub.InferenceClient:** Para utilizar los modelos de LLM, de manera remota, utilizando una api.
- **GoogleTranslator:** Para traducir las query del usuario del español al inglés, y posteriormente traducir del inglés al español la respuesta de la LLM
- **langdetect:** Se utiliza para detectar automáticamente el idioma de un texto dado.
- **userdata:** Para ocultar la key de la api de hugging face

Páginas donde se busco la información para la creación del trabajo práctico:

- Los cuadernillos proporcionados por la cátedra

Trabajo Práctico N°2

- <https://huggingface.co/models>: para buscar los distintos modelos de LLM y codificación
- <https://www.holisticseo.digital/theoretical-seo/sliding-window-technique-and-algorithm/> : Para la ceracion del modelo de chunks y embeddings
-  Cosine Similarity, Clearly Explained!!! : Para la comprender e implementar la similitud del coseno, además de los videos de Dot CSV
- <https://docs.trychroma.com/getting-started> : Para utilizar choramdb
- <https://www.promptingguide.ai/introduction/tips> : Para configurar las instrucciones que se le daban a la LLM

Modelos que se utilizan:

- sentence-transformers/multi-qa-mpnet-base-dot-v1: para codificar los textos
- Qwen/Qwen2.5-72B-Instruct: Para las distintas LLM

EJERCICIO 2

RESUMEN

Para esta segunda parte, donde se debía implementar un agente, se probó con distintos modelos, probandolos con las mismas preguntas, obteniendo resultados diversos. Esto usando instrucciones básicas y utilizando 4 tipos de herramientas que el agente podría llamar para recolectar la información para responder a la pregunta del usuario. Estas herramientas son las 3 bases de datos creadas para la parte 1 del RAG, además de utilizar wikipedia como herramienta extra con la que el modelo puede acceder a diversa información desde la base de datos de esta enciclopedia.

El agente que se implementó, esta basado en la documentación dejada por la cantadera del cuadernillo de la unidad 7, realizando distintos cambios a las funciones y a las instrucciones, teniendo que ser ampliamente detallado, para que este no sufriera alucinaciones, o tratara de cambiar la query de búsqueda, obteniendo resultados errores.

Finalizando con una agente que en la mayoría de casos, con las preguntas más estructuradas, o con preguntas sencillas, funciona de manera correcta.

DESARROLLO

Para el desarrollo del agente, se probó con 3 modelos, siendo estos: llama3.2, gemma y mistral, los 3 modelos de menos de 7b de parámetros, ya que sino, para modelos de una cantidad mayor de parámetros, no es posible hacer funcionar al agente sin usar una gpu de mayor capacidad, no siendo el caso para google colab.

Primero, se utilizó como herramienta extra wikipedia, esto gracias a que resultó ser una buena herramienta para utilizar cuando las otras fuentes de datos son incompletas, o el agente quiere verificar la información aunque en ciertos casos que trata de buscar más información en wikipedia, esto corrompe la respuesta terminado en una cantidad máxima de iteraciones, en una respuesta sin sentido o en que el agente no puede responder de manera satisfactoria a la pregunta.

Para el resto de herramientas se utilizaron las pedidas en las instrucciones del trabajo práctico, lo que implicó definir funciones para llamar a las distintas bases de datos para poder extraer el contexto necesario.

Al momento de definir las instrucciones, esto fue el punto principal, más importante y delicado de todo el agente, ya que se debía dejar todo por explícito, ayudando al agente a seguir pasos definidos y poco flexibles, para que no cometa errores, o llama a las funciones con queries en un formato erróneo, o modifique de gran manera las queries con las que se ingresaban.

Además de que en muchos casos no comprende que base de datos llamar para buscar la información aunque esto esté especificado en las instrucciones.

Otro problema que suele surgir es que si el agente se ejecuta varias veces, puede ser que de resultados distintos, esto aun con la temperatura en 0.1, lo que quiere decir que no se da una gran libertad al modelo, generando errores como que necesita que la query este en ingles, cuando se le está pasando una query en inglés con instrucciones en inglés y la información de todas las bases de datos en inglés.

También, cambia en gran medida la respuesta del agente dependiendo de cómo se estructura la pregunta, simplemente cambiando de posición las palabras en la oración, generando resultados completamente distintos.

En general el agente suele dar problemas, teniendo poca consistencia dependiendo de la pregunta que se le realice. Esto también es debido a la poca cantidad de parámetros de los modelos

CONCLUSIONES

Para las conclusiones, se van a mostrar los resultados de distintos modelos de Ollama:

llama3.2, gamma, mistral y qwq

Aclaraciones con los modelos mencionados, en el caso de llama3.2 y gamma produjeron respuestas a las distintas preguntas que se les plantearon, a diferencia del modelo mistral, que no logra generar ninguna respuesta, quedando trabado buscando en la misma fuente de datos teniendo que ser interrumpido de manera manual, ya que no logra finalizar, o no da ningún resultado, sin siquiera buscar en diferentes fuentes de datos

En el caso del modelo qwq, que tiene una gran cantidad de parámetros más que los otros 3 modelos, este modelo, tiene el problema, al igual que los modelos con una gran cantidad de parámetros, que aunque logren descargarse, luego la gpu no soporta el peso de los modelos.

Por lo tanto, no pueden siquiera ejecutar una query sin que se quede sin tiempo, aun dándole de margen 600 segundos y usando la gpu t4. Por esta razón también, se dedicó a utilizar modelos con poca cantidad de parámetros, para que pudiesen llevar a cabo las distintas tareas.

Para los modelos de agentes llama3.2 y gamma, se probaron con las mismas queries y estos son los resultados:

Pregunta 1

Pregunta: who are the names of the designers of the game?

Respuesta:

llama3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 7935c037-94a9-47da-b3ce-6adbae1fb25a. Step input: who are the names of the designers of the game?
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: {'query': 'designers of the game'}
Observation: ['Elwen', 'Min']
> Running step ef7c7746-6719-440a-ab0a-d6d1d57fde41. Step input: None
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The designers of the game are Elwen and Min.
Respuesta: The designers of the game are Elwen and Min.
```

gemma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step e96b9ee4-4d91-45d1-a25f-9a92f15f0f8f. Step input: who are the designers of the game?
Thought: (Implicit) I can answer without any more tools!
Answer: I do not have access to information about the game's designers and am unable to provide a list using the available tools.
Respuesta: I do not have access to information about the game's designers and am unable to provide a list using the available tools.
```

En este caso, el modelo de llama 3.2, logra identificar dónde están almacenadas las relaciones del juego, usando la función que llama a la base de datos de grafos, obteniendo la respuesta.

A diferencia del modelo gemma, que no logra encontrar una respuesta aunque este sea ejecutado varias veces

Pregunta 2

Pregunta: What items have Cost greater than 1?

Respuesta:

llama3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step e0a0004a-df10-4f52-b0ca-80789be5c13f. Step input: What items have Cost greater than 1?
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': 'Cost > 1 in game items'}
Observation: ['Bow and Arrows', 'Brush', 'Dog', 'Fishing Rod', 'Chronometer', 'Aeroplane', 'Theodolite', 'Carrier Pidgeon', 'Torch', 'Axe', 'Parrot', 'Horse', 'Precision Compass', 'Tent', 'Rope', 'Automobile', 'Steam Boat', 'Machete', 'Binoculars', 'Pack Donkey', 'Ostrich', 'Sea Turtle', 'Lantern', 'Large Backpack', 'Army Knife', 'Revolver', 'Grappling Hook', 'Airdrop', 'Flask', 'Hot Air Balloon', 'Whip', 'Bear Trap', 'Journal']
> Running step c27d64ab-a525-4065-bd91-3562f70a4cfa. Step input: None
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The items with a cost greater than 1 are:
- Bow and Arrows
- Brush
- Dog
- Fishing Rod
- Chronometer
- Aeroplane
- Theodolite
- Carrier Pidgeon
- Torch
- Axe
- Parrot
- Horse
- Precision Compass
- Tent
- Rope
- Automobile
- Steam Boat
- Machete
- Binoculars
- Pack Donkey
- Ostrich
- Sea Turtle
- Lantern
- Large Backpack
- Army Knife
- Revolver
- Grappling Hook
- Airdrop
- Flask
- Hot Air Balloon
- Whip
- Bear Trap
- Journal
Respuesta: The items with a cost greater than 1 are:
- Bow and Arrows
```

```
Respuesta: The items with a cost greater than 1 are:
- Bow and Arrows
- Brush
- Dog
- Fishing Rod
- Chronometer
- Aeroplane
- Theodolite
- Carrier Pidgeon
- Torch
- Axe
- Parrot
- Horse
- Precision Compass
- Tent
- Rope
- Automobile
- Steam Boat
- Machete
- Binoculars
- Pack Donkey
- Ostrich
- Sea Turtle
- Lantern
- Large Backpack
- Army Knife
- Revolver
- Grappling Hook
- Airdrop
- Flask
- Hot Air Balloon
- Whip
- Bear Trap
- Journal
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 8bd33c7c-c74c-4258-9b21-b0c7ef5db0e0. Step input: What items have Cost greater than 1?
Thought: (Implicit) I can answer without any more tools!
Answer: The items with Cost greater than 1 are 'Item', 'Efficiency(Including VP)', 'Efficiency', 'Diffe
Respuesta: The items with Cost greater than 1 are 'Item', 'Efficiency(Including VP)', 'Efficiency', 'Di
```

Este es otro caos donde llama 3.2 logra identificar dónde buscar la información, llamando a únicamente una herramienta, a diferencia de gamma, que en esta ocasión devuelve un resultado, pero no logra generar una query de pandas apropiada, obteniendo los nombres de las variables de la base de datos tabulares

Pregunta 3

Pregunta: how many players can play the game?

Respuesta:

llama3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 5316febe-6c71-46f6-ba01-b1440eb06f36. Step input: how many players can play the game?
Thought: I need to use a tool to help me answer the question.
Action: table_search
Action Input: {'query': 'Number of players in the game', 'type': 'string'}
Observation: Error: table_search() got an unexpected keyword argument 'type'
> Running step 76493632-a4b4-475b-9a5b-741178add3e7. Step input: None
Thought: It seems that the tool I used earlier does not support the 'type' keyword. I will try again with
Action: doc_search
Action Input: {'query': 'Number of players in the game'}
You're using a XLNetTokenizerFast tokenizer. Please note that with a fast tokenizer, using the `__c
Observation: Game Setup Lay out the center board on the desired side. Place the reserve board on the bot
> Running step 76339865-7e61-487f-a4b5-c9d69f216baa. Step input: None
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The game can be played with 2-5 players.
Respuesta: The game can be played with 2-5 players.
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step bf7abb3f-c7cf-467f-a8e5-276031f32c82. Step input: how many players can play the game?
Thought: (Implicit) I can answer without any more tools!
Answer: The game can be played by 2 to 4 players.
Respuesta: The game can be played by 2 to 4 players.
```

En este caso, ambos modelos logran llegar a una conclusión, aunque el modelo de gamma toma información distinta a la del modelo llama 3.2, que en este caso utiliza distintas fuentes de datos para encontrar la información

Pregunta 4

Pregunta: What are the principal rules of the game?

Respuesta:

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 6a7bbd79-98b0-495a-8052-05213bf1f034. Step input: What are the principal rules of the game?
Thought: The current language of the user is: English. I need to use a tool to help me answer the question.
Action: wikipedia_search
Action Input: {'query': 'Game rules'}
Observation: Time is the continuous progression of our changing existence that occurs in an apparently irreversible succession from the past, through
> Running step b13fc07c-872b-4cb6-8e0e-897f391d2a85. Step input: None
Thought: The current language of the user is still English. I need to use another tool to help me answer the question.
Action: doc_search
Action Input: {'query': 'Game rules definition'}
Observation: Lost Ruins of Arnak rules summary Object: Each player Leads an expedition on the uncharted island of Arnak. Points are scored for a varie
> Running step 124e9eea-6824-404f-ae4a-f52d987cbaf8. Step input: None
Thought: The current language of the user is still English. I need to use another tool to help me answer the question.
Action: table_search
Action Input: {'query': 'Lost Ruins of Arnak setup rules'}
Observation: Error: unterminated string literal (detected at line 1) (<string>, line 1)
> Running step 23e68f47-3db5-4b5e-ae7e-677af5f3835. Step input: None
Thought: The current language of the user is still English. I need to use another tool to help me answer the question.
Action: graph_search
Action Input: {'query': 'Game mechanics Lost Ruins of Arnak'}
Observation: ['Contracts', 'Deck, Bag, and Pool Building', 'Market', 'Multi-Use Cards', 'Once-Per-Game Abilities', 'Resource to Move', 'Solitaire Gam
> Running step ba4df384-107d-41c4-ac3b-ab4157e0bf97. Step input: None
Thought: The current language of the user is still English. I can answer without using any more tools.
Answer: The principal rules of Lost Ruins of Arnak include contracts, deck-building and bag management, market trading, multi-use cards, once-per-g
Respuesta: The principal rules of Lost Ruins of Arnak include contracts, deck-building and bag management, market trading, multi-use cards, once-per-g
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step cb0c413e-4cd3-4794-8892-eb9349229d58. Step input: What are the principal rules of the game?
Thought: (Implicit) I can answer without any more tools!
Answer: The principal rules of the game are to collect items, manage resources, and complete actions to gain points and win the game.
Respuesta: The principal rules of the game are to collect items, manage resources, and complete actions to gain points and win the game.
```

En este caso nuevamente ambos modelos logran llegar a una conclusión muy similar, en el caso de llama, más preciso y también utilizando más fuentes de datos para verificar su información.

En este caso, la primera respuesta de llama, fue devolver la información en formato de diccionario con ciertos errores, pero a la segunda ejecución, este logro llegar a una conclusión correcta

Pregunta 5

Pregunta: What are the advantages of using the pickaxe item and what is its Cost?

Respuesta:

llama 3.2

```
rvation: Error: Expected SelectQuery, found 'nfits' (at char 175), (Line:6, col:63)
nning step b94c246f-5702-429e-9818-55d3a4260e55. Step input: None
ght: The graph search tool is not returning the expected output. It seems that the Pickaxe item benefits and speeds are represented as 'nfits'.
on: table_search
on Input: {'query': 'Pickaxe item nfits in Lost Ruins of Arnak'}
rvation: Error: "[fits in Lost Ruins of Arnak] not in index"
nning step c16f68f3-fe8d-4647-9d21-def6a9bba869. Step input: None
ght: The table search tool is unable to find the information about the Pickaxe item's benefits and speeds.
on: wikipedia_search
on Input: {'query': 'Pickaxe item benefits and speeds in Lost Ruins of Arnak board game'}
rvation: No se encontró información en wikipedia por el siguiente error: Page id "Pickaxe item benefits and speeds in Lost Ruins of Arnak board game" does not match any pages. Try another id!
nning step 96a4c6ea-bcd7-4c6d-8147-b53440925e93. Step input: None
ght: The Wikipedia search tool was unable to find the specific information about the Pickaxe item's benefits and speeds.
on: doc_search
on Input: {'query': 'Pickaxe item benefits and speeds in Lost Ruins of Arnak game mechanics'}
rvation: but oh well. We dig dig dig dig dig dig dig in our mine the whole day through Pickaxe is easily one of the best one cost Items, and one of the best Items overall. Strengths And here we
nning step aee030cf-a9b1-431a-84b2-fd2e216e4f39. Step input: None
ght: The Pickaxe item has both strengths and weaknesses in Lost Ruins of Arnak.
on: graph_search
on Input: {'query': 'Pickaxe item strengths and weaknesses in Lost Ruins of Arnak'}
rvation: Error: Expected [SelectQuery | ConstructQuery | DescribeQuery | AskQuery], found 'PREFIX' (at char 0), (Line:1, col:1)
nning step 286916c4-f6be-402d-a33f-a809d20b3f76. Step input: None
ght: The graph search tool is not returning the expected output. It seems that there was an error in the query.
on: table_search
on Input: {'query': 'Pickaxe item benefits and speeds in Lost Ruins of Arnak game mechanics'}
rvation: ['Benefit', 'Speed']
nning step 60f37beb-dbd1-4459-8f2a-ce73c2df1d74. Step input: None
ght: The Pickaxe item has a benefit that provides a consistent stream of resources (Arrows) and a speed of 0, meaning it does not move the player's expedition token.
er: The Pickaxe item is a valuable resource in Lost Ruins of Arnak, providing a consistent stream of Arrows while being cost-effective. Its low cost makes it an excellent choice for players.
uesta: The Pickaxe item is a valuable resource in Lost Ruins of Arnak, providing a consistent stream of Arrows while being cost-effective. Its low cost makes it an excellent choice for players.
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 9e7e9be5-ec73-41e0-8802-99537b1d65f3. Step input: What are the advantages of using
Thought: (Implicit) I can answer without any more tools!
Answer: The pickaxe item allows you to gather resources more quickly. It has a Cost of 1.
Respuesta: The pickaxe item allows you to gather resources more quickly. It has a Cost of 1.
```

En este caso, se intentó que el modelo buscará información que no se encontraba en la misma fuente de información, y al final intentase juntarla para dar una respuesta completa.

Esto se probó en distintas instancias, subiéndole la temperatura al modelo para que así tuviese una mayor libertad. luego de subir a 0.3 de temperatura, el modelo de llama 3.2, empezó a devolver rangos de valores de picos de minería en amazon y también cómo se construye el objeto pico en el juego minecraft, por lo que se entiende que la darle libertad al agente, este empieza a actuar de manera errónea en muchos casos

En el caso de gamma, devolvió el costo y una ventaja de mera esquina, pero respondiendo a la pregunta pero únicamente solicitando una única fuente de datos

Trabajo Práctico N°2

A diferencia de llama 3.2, que aunque no da el valor numérico como gamma, sabe que es de un valor bajo, pero este da una mayor explicación en las ventajas del ítem, buscando en distintas fuentes de datos

Answer: The Pickaxe item costs 1 resource and provides a consistent stream of resources (such as Arrows). Its low cost makes it an excellent choice for players.
Respuesta: The Pickaxe item costs 1 resource and provides a consistent stream of resources (such as Arrows). Its low cost makes it an excellent choice for players.

Esta es otra de las respuestas que el modelo llama devolvió a la misma pregunta, en este caso haciendo énfasis al costo y no tanto a la ventaja del ítem en el juego.

Pregunta 6

Pregunta: In what year the game was released?

Respuesta:

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step c11aad5d-20d4-4593-a4bb-90a4d52b1509. Step input: In what year the lost ruins of
Thought: The current language of the user is: English. I need to use a tool to help me answer th
Action: wikipedia_search
Action Input: {'query': AttributedDict([('title', 'Lost Ruins of Arnak'), ('type', 'string')])}
Observation: Time is the continuous progression of our changing existence that occurs in an appa
> Running step 4f2bcfc5-6931-4e4d-900f-5baeca276acf. Step input: None
Thought: The user has provided information about time, but I still need to find the release year
Action: wikipedia_search
Action Input: {'query': 'Lost Ruins of Arnak release date'}
Observation: No se encontró información en wikipedia por el siguiente error: Page id "Lost Ruins
> Running step 713d5afa-1488-494f-bca0-1adfcf0650d7. Step input: None
Thought: The user has searched for the release date of Lost Ruins of Arnak on Wikipedia, but was
Action: doc_search
Action Input: {'query': 'Lost Ruins of Arnak release date'}
You're using a XLMRobertaTokenizerFast tokenizer. Please note that with a fast tokenizer, using
Observation: During setup you will choose how many red tiles you want to incorporate, allowing y
> Running step 22f35f7c-b789-4af8-8a18-518ae48138b2. Step input: None
Thought: The user has provided information about the gameplay and design of Lost Ruins of Arnak,
Action: table_search
Action Input: {'query': 'Lost Ruins of Arnak release date'}
Observation: []
> Running step bf8c59d3-2fce-4ed6-9c23-45feb2ffe94f. Step input: None
Thought: The user has searched for the release date of Lost Ruins of Arnak using the table search
Thought: I should try searching again or looking for alternative sources.
Action: graph_search
Action Input: {'query': 'Lost Ruins of Arnak Czech Games Edition release year'}
Observation: ['2020']
Respuesta: Error al procesar la consulta: Reached max iterations.
```

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 6f616c1c-0f67-4e07-818c-5b7e4e08441e. Step input: in what year the game the lost ruins of arnack was released?
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The game Lost Ruins of Arnak was released in 2021.
Respuesta: The game Lost Ruins of Arnak was released in 2021.
```

gamma

Trabajo Práctico N°2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 3c0120d3-6f68-4409-8b62-da14e6b142f9. Step input: when year the game lost ruins of arnack was released?
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The game Lost Ruins of Arnak was released in 2021.
Respuesta: The game Lost Ruins of Arnak was released in 2021.
```

En este caso, se pueden visualizar varios problemas, en el primer caso de llama 3.2, que no realiza la búsqueda en la fuente de datos adecuado, y luego al encontrarla, termina su ejecución por iteraciones máximas, esto también puede ser a causa del contexto escueto que es el texto 2020, que es posible que el agente no sepa cómo interpretarlo, al igual que en las LLM, cuando no se le pasaba las instrucciones adecuadas.

El problema aquí es que si tiene las instrucciones necesarias para lidiar con casos de este índole.

Luego en la imagen siguiente, busca información en wikipedia, sin poder identificar, que en 2021, fue lanzada una expansión del juego, y no el juego original. lo que le sucede lo mismo a el modelo de gamma

Pregunta 7

Pregunta: What are the mechanics of the game?

Respuesta:

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step dd45fb07-8ab0-4ee4-89f5-7f62814be211. Step input: which are the mechanics of the game?
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': 'Game Mechanics'}
Observation: ['Item', 'Efficiency(Including VP)', 'Efficiency', 'Difference', 'Cost', 'Benefit', 'Recursive Inflation', 'Play Value', 'Points', 'Speed', 'Versati
> Running step 439f16c5-d777-437c-bb26-c5d860148a43. 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': 'Game Mechanics'}
Observation: ['Item', 'Efficiency(Including VP)', 'Efficiency', 'Difference', 'Cost', 'Benefit', 'Recursive Inflation', 'Play Value', 'Points', 'Speed', 'Versati
> Running step 4ff717af-7547-421f-a78f-d0ce889141f4. 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: graph_search
Action Input: {'query': 'Game Mechanics'}
Observation: ['Contracts', 'Deck, Bag, and Pool Building', 'Market', 'Multi-Use Cards', 'Once-Per-Game Abilities', 'Resource to Move', 'Solitaire Game', 'Turn O
> Running step 93b45a22-ddb4-4896-a071-f19ee01bbcd7. 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: graph_search
Action Input: {'query': 'Game Mechanics'}
Observation: ['Contracts', 'Deck, Bag, and Pool Building', 'Market', 'Multi-Use Cards', 'Once-Per-Game Abilities', 'Resource to Move', 'Solitaire Game', 'Turn O
> Running step 6f1457bf-9bb8-4aa0-bacd-7947cf4140e7. 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: The game mechanics include contracts, deck building and pool management, a market system, multi-use cards, once-per-game abilities, resource movement, so
Respuesta: The game mechanics include contracts, deck building and pool management, a market system, multi-use cards, once-per-game abilities, resource movement,
```

gamma

Trabajo Práctico N°2

```
=== Ejemplo de interacción con el agente ReAct ===  
> Running step 71d9b739-c36d-45b2-9f04-202826b3c284. Step input: What are the mechanics of the game?  
Thought: (Implicit) I can answer without any more tools!  
Answer: The game features resource management, action selection, and worker placement mechanics. Players  
Respuesta: The game features resource management, action selection, and worker placement mechanics. Play
```

En este caso ambos modelos consiguen llegar a la información adecuada, el modelo llama 3.2, consiguiendo de la fuente de datos tabulares, intentando la búsqueda en otras herramientas, mientras que gamma, utiliza wikipedia para encontrar esta mismas respuestas

Pregunta 8

Pregunta: what mechanics has the game the lost ruins of arnak in the individual mode?

Respuesta:

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===  
> Running step 48e456c5-2279-4a06-80a7-3d39465c8129. Step input: what mechanics has the game the lost ruins of arnak in the individual mode?  
Thought: I can answer without using any more tools. I'll use the user's language to answer  
Answer: In the individual mode, The Lost Ruins of Arkana features a deck-building mechanic where players build their own deck by collecting cards during gameplay. It  
Respuesta: In the individual mode, The Lost Ruins of Arkana features a deck-building mechanic where players build their own deck by collecting cards during gameplay.
```

En este caso, llama encuentra la información directamente desde lo que hipotetizó que es wikipedia, no siendo precisa en la información, además de ser muy corta, pudiendo haberla buscado en la base de datos vectorial, pudiendo encontrarla de manera amplia y correcta.

En este caso gamma, no fue capaz de dar resultados o de dar resultados con sentido a la pregunta que se le estaba realizando

Pregunta 9

Pregunta: What is the best initial move in the game?

Respuesta:

Trabajo Práctico N°2

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step ae5aa6ca-9098-4bb2-bda7-706467fcf2b7. Step input: What is the best initial move in the game?
Thought: I need to use a tool to help me answer the question about the best initial move in Lost Ruins of Arnak.
Action: table_search
Action Input: {'query': 'best initial move Lost ruins of arnak'}
Observation: ['Item', 'Efficiency(Including VP)', 'Cost', 'Benefit']
> Running step 338f1a0d-a141-457a-8c8f-8fa3866ad7a0. Step input: None
Thought: I can answer without using any more tools. I'll use the user's Language to answer
Answer: The best initial move in Lost Ruins of Arnak is a matter of debate among players, but some popular strategies inclu
Respuesta: The best initial move in Lost Ruins of Arnak is a matter of debate among players, but some popular strategies in
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 181a836d-4aaf-4cfc-bb2a-dbd5bc35108d. Step input: what mechanics has the game the lost ruins of arnak
Thought: (Implicit) I can answer without any more tools!
Answer: I am unable to provide information about the game's individual mode mechanics using the available tools.
Respuesta: I am unable to provide information about the game's individual mode mechanics using the available tools.
```

En este caso, gamma no fue capaz de encontrar la información ni en la base de datos vectorial, ni en wikipedia. llama 3.2 fue capaz de responder la pregunta de manera satisfactoria, utilizando la información que había conseguido antes buscando en wikipedia. Aunque en la segunda ejecución del agente con la misma pregunta, este no supo dar una respuesta.

Pregunta 10

Pregunta: Who are the publishers of the game?

Respuesta:

llama 3.2

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step d7dbde0a-1b87-4702-a755-02e1cb0651c7. Step input: who are the publishers of the game?
Thought: The current language of the user is English. I need to use a tool to help me answer the question.
Action: wikipedia_search
Action Input: {'properties': AttributedDict({'query': AttributedDict({'title': 'Publishers of Lost Ruins of Arnak'}), ('type': 'game'})}
Observation: Error: wikipedia_search() got an unexpected keyword argument 'properties'
> Running step f96dcd8-9d0b-4a01-a904-a8b8aec3e385. 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: wikipedia_search
Action Input: {'query': 'Publishers of Lost Ruins of Arnak'}
Observation: Lost Ruins of Arnak is a 2020 board game by the husband-wife duo Michal "Elwen" Štách and Michaela "Mín" Štachová.

== Gameplay ==
Lost Ruins of Arnak uses a combination of deck-building and worker placement mechanics for its core gameplay.
> Running step 2878862e-c6f3-49ef-8321-4322fdb0554. Step input: None
Thought: I can answer without using any more tools. I'll use the user's language to answer
Answer: The publishers of Lost Ruins of Arnak are Michal "Elwen" Štách and Michaela "Mín" Štachová, a husband-wife duo.
Respuesta: The publishers of Lost Ruins of Arnak are Michal "Elwen" Štách and Michaela "Mín" Štachová, a husband-wife duo.
```

gamma

```
=== Ejemplo de interacción con el agente ReAct ===
> Running step 57905c23-b73f-475b-8e1c-178bdcf68d23. Step
Thought: (Implicit) I can answer without any more tools!
Answer: The publishers of the game are CGE.
Respuesta: The publishers of the game are CGE.
```

En este caso, llama da una respuesta correcta, pero a otra pregunta que el agente mismo se hace, pero en el contexto de responder la pregunta original, falla, tomando el nombre de los diseñadores como si fueran los publicadores del juego.

En el caso de gamma, encuentra la respuesta correcta aunque esta también pudiese ser más extensa si hubiese intentado utilizar otro tipo de herramientas, donde esta información está contenida.

Conclusiones generales

Como conclusión y luego de ver las distintas pruebas con los modelos llama 3.2 y gamma, decide utilizar llama, ya que en general da respuestas más precisas, y también utiliza las distintas herramientas para llegar a una conclusión.

Esto en los caos que llegue a una respuesta en la primera ejecucion del codigo, ya que en varios casos, en la primera ejecución o en la segunda ejecución, el modelo alucina o no encuentra dónde buscar la información, o decide cambiar la query por su propia cuenta, tratando de buscar otro tipo de información.

Esto como se comentó antes, se debe a la baja cantidad de parámetros que tienen los modelos, pero esto también se debe al poco procesamiento que presta google colab para poder ejecutar estos modelos.

Agregando la comparación en este caso del agente con el RAG, el agente funciona de peor manera, teniendo problemas en una pregunta, donde luego al ejecutarlo otra vez, el modelo responde apropiadamente, o casos contrarios. A diferencia del RAG, que es siempre consistente con sus respuestas y no cambia lo que devuelve se ejecute las veces que se ejecute, esto también se debe a que se utiliza un modelo de 72 billones de parámetros contra 3 billones de llama 3.2, donde se logra ver la diferencia de ambos modelos.

En respuestas a los resultados esperados por el trabajo práctico, en las preguntas 3, 4, 5, 7, 9, se logra dar una respuesta utilizando más de una herramienta, aunque muchas veces la utilización de esta herramienta confunde al agente, o simplemente la query que busca sea diferente a la original

Luego, las preguntas 6, 8 y 10, dan información incorrecta o en general son poco precisas

En el caso de la 2, podría ser considerada como correcta si se aumentase la cantidad de iteraciones, para que el modelo no temiera la búsqueda de manera abrupta, ya que como se ve en la imagen, logra encontrar la información, pero se queda sin suficientes iteraciones

Luego en el caso de la pregunta 1, como muchos otros casos de preguntas, el agente devuelve la información sin utilizar más de 1 herramienta para buscar la información, o respondiendo según las preguntas anteriores que se le hicieron, respondiendo de manera correcta, por lo que resultó innecesario poner más ejemplos donde no contengan información relevante del agente usando distintas herramientas.

Las mejoras son en general, utilizar un modelo para el agente con una mayor cantidad de parámetros, aumentar la información de las bases de datos locales, para que, de esa manera el agente tenga un mayor repertorio donde poder buscar, también podrían sumarse otras herramientas de búsqueda si se quisiese dar otro enfoque con las preguntas.

En el caso de mejorar las instrucciones, no logré apreciar una gran mejora luego de cierto punto, intentando implementar varios pasos con ejemplos, por lo que tampoco me resulta una gran mejora la ampliación de las instrucciones

Instrucciones que se le dieron al agente:

```

"""
1. Analyze the user's query: Identify whether it is a numeric, textual, or relational query. Determine whether a tool is needed to answer it.

2. Use the correct format for each tool's input:
- Valid JSON format: Make sure to use double quotes (") for keys and values.
- Examples for table_search:
- Efficiency query:
Action Input: {"query": "efficiency > 300"}
- Price query:
Action Input: {"query": "price < 50"}

3. Error handling:
- If a tool returns an error, check:
1. Is the input format valid?
2. Is the query defined correctly?
- Only retry if you can correct the problem (e.g., by using double quotes or adjusting the syntax).

4. Avoid unnecessary iterations:
- Stop the process if the same error occurs more than two consecutive times.
- Use an iteration limit to prevent infinite loops.

5. Provide a clear final answer:
- Combine all valid observations obtained from the tools.
- Give a complete and well-structured answer to the user.

```

```

6. Format examples for tools:
- doc_search: Action Input: "text to find" / use it as principal tool
- wikipedia_search: Action Input: "topic"
- graph_search: Action Input: "relationships or names"
- table_search: Action Input: {"query": "numeric or tabular data"}

7. Context for queries on tables:
- The table contains columns such as: "Item", "Efficiency", "Cost", "Category".
- "Efficiency" refers to the output/input ratio of a process.

8. Validate inputs before running them:
- If the input contains filters or complex parameters, structure them in a valid JSON format.
- Corrected input example:
Action Input: {"query": "items with efficiency greater than 300"}

9. General interaction format:
- Always use the following format:
Thought: [Reasoning]
Action: tool_name
Action Input: [Input in the correct format]
Observation: [Tool output]
Final Answer: [Complete final answer]

10. Testing and validation (optional):
- Run tests to ensure that inputs are handled correctly. For example:

```

```

python
def test_table_search():
    input = {"query": "efficiency > 300"}
    assert is_valid_json(input), "Input is not valid JSON"
    result = run_tool("table_search", input)
    assert "Error" not in result, "Error in table_search"

```

ENLACE A LOS MODELOS Y LIBRERÍAS UTILIZADOS

Las librerías que se utilizaron son:

- **llama_index**
 - **Settings:** Se utiliza para ajustar parámetros relacionados con los modelos LLM
 - **Ollama:** Se utiliza para interactuar con los modelos LLM específicos de cómo llama3.2 o gemma
 - **ReActAgent:** para implementar un agente basado en ReAct
 - **FunctionTool:** Para definir las herramientas que el agente va a utilizar
 - **ReActChatFormatter:** Para estructurar los mensajes de entrada y salida del agente
 - **wikipedia:** Se utilizó como herramienta para extraer información extra antes las consultas del agente
 - **LLM:** Se utiliza para configurar los modelos de LLM que el agente va a utilizar

Páginas donde se busco la información para la creación del trabajo práctico:

- Los cuadernillos proporcionados por la cátedra
- https://docs.llamaindex.ai/en/stable/examples/agent/react_agent/ : Para configurar el agente
- <https://ollama.com/library> : Para buscar los diferentes modelos

Modelos que se utilizan:

- llama3.2
- gemma