UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA
INFORMÁTICA Y CIENCAS DE LA COMPUTACIÓN

Sistema Graph RAG de Extracción de Información para la S.S.D.D.H.H de Chile

---

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil Informático

Vicente Emilio Lermanda Candia

Profesores Guía

José Fuentes Sepúlveda

Juan Reutter de la Maza

Agosto 2024. Concepción, Chile

UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
INGENIERÍA CIVIL INFORMÁTICA

Agradecimientos

Me gustaría expresar mi más profunda gratitud a todos aquellos que han confiado en mi, y me han brindado su apoyo incondicional durante estos últimos años.

En primer lugar, quiero reconocer con gran aprecio a mi familia. Mis padres, los cuales siempre han entregado todo buscando mi bienestar y plenitud en la vida. Mis hermanos Julián y Amparo, quien junto a Lisa me brindan constantemente de felicidad y de su afecto. A mis abuelos Óscar e Isabel, cuya preocupación y cariño hacia mi ha sido una fuente de fortaleza y consuelo. A mi abuela Emilia, quién durante el tiempo que estuvo en mi vida, dejó un legado imborrable de cariño y perseverancia. A mis tías Inés y Macarena quien sin su apoyo no podría estar aquí. A mi primo Omar que ha sido un hermano mayor para mi. Y a todos mis demás familiares, quienes aunque no podré mencionar uno por uno, han contribuido significativamente con su amor, apoyo y ánimo en cada etapa de mi vida. Su presencia y respaldo han sido cruciales para alcanzar este logro.

Quisiera agradecer a aquellos profesores que jugaron un rol fundamental en mi desarrollo académico y personal. A Marcelo Ravanal y Jaime Urrutia del SIC por su simpatía, entusiasmo y motivación para siempre dar lo mejor de uno mismo. A Roberto Asín y Diego Seco a quienes tuve la fortuna de conocer antes de entrar a la universidad, y de poder compartir hasta su partida de la UdeC. Al profesor José Fuentes quien ha mantenido su alta estima y confianza en mi a pesar de las adversidades personales que he tenido durante mi etapa universitaria. Y al profesor Juan Reutter, quien sin dudarlo recibió a un alumno del quien nunca había escuchado, entregando su total confianza y apoyo en todo momento.

Me gustaría expresar mi gratitud al Instituto Milenio de Fundamentos de los Datos (IMFD), cuyo generoso patrocinio de mi beca de magíster ha sido fundamental para la realización de este proyecto. Su apoyo no sólo me ha brindado los recursos necesarios para continuar con mis estudios, sino que también ha sido una fuente constante de motivación y estímulo.

Agradezco también a la iniciativa estudiantil Open Source UC de la cual formo parte, donde he conocido excelentes compañeros, y donde también obtuve la plantilla de LaTeXen la cual basé el presente informe.

Finalmente, aunque no menos importante, deseo expresar mi total agradecimiento a mis amistades, quienes han estado presentes tanto en los momentos de alegría como en los de dificultad. Mostrando un sincero interés por mi bienestar y soportando con gran paciencia cualquier ingratitud de mi parte durante este periodo. Agradezco a Katheryne Alegría, Carolina Rubilar, Juan Venegas, Nicol Ortega, Lucas Torres, Carla Quero, Juan Nuñez, Ignacio Henriquez, Ignacio Rubilar, Sebastián Parra.

Resumen

La dictadura militar en Chile (1973-1990) se caracterizó, entre otras cosas, por violaciones sistemáticas a los derechos humanos, incluyendo la desaparición forzada de personas. La Subsecretaría de Derechos Humanos (S.D.D.H.H) en Chile ha desempeñado un rol fundamental en documentar estos casos, acumulando un vasto volumen de documentos, muchos aún en formato físico, lo cual presenta desafíos significativos para su análisis y acceso eficiente.

Se considera que la implementación de tecnologías modernas de Inteligencia Artificial, como los Modelos de Lenguaje de Gran Tamaño y los sistemas de recuperación como el "Retrieval Augmented Generation", puede mejorar la gestión y el análisis de estos documentos. Basándose en esta premisa, se ha desarrollado un primer prototipo software, de una plataforma digital que facilite el acceso a la información en el ámbito de derechos humanos en Chile, particularmente en documentos históricos chilenos en español.

La plataforma desarrollada aprovecha las capacidades de estas nuevas tecnologías para procesar, analizar y extraer información relevante de los documentos, complementándola con la transformación de texto a Grafos de Conocimiento, los cuales permiten una visualización y análisis más claro de entidades y sus relaciones. El objetivo es permitir la extracción eficiente de información relevante, asegurar la veracidad de los datos mediante la trazabilidad de fuentes y proporcionar herramientas para complementar y contrastar información de diversas fuentes.

El sistema creado ha demostrado su capacidad para recuperar contextos relevantes y generar respuestas coherentes con alta fidelidad a los datos originales en base a las consultas realizadas en lenguaje natural acerca de diversos hechos, lugares, personas, entre otros.

Finalmente, Aunque se identificaron algunas limitaciones, como la variabilidad en el nivel de detalle de los contextos y la posible pérdida de información en la conversión de texto a grafos, los hallazgos sugieren un gran potencial para perfeccionar este enfoque y aplicarlo en un contexto de investigación. En resumen, este prototipo muestra la viabilidad de utilizar tecnologías avanzadas de IA para manejar y analizar grandes volúmenes de datos históricos desestructurados en español, ofreciendo nuevas vías para la investigación y documentación de eventos críticos en la historia de Chile, al tiempo que proporciona una mejor comprensión de las capacidades y limitaciones de LLMs en casos de alta sensibilidad e importancia como este.

Palabras Clave: Ingeniería de Software, Inteligencia Artificial, Derechos Humanos, Historia de Chile, Recuperación de Información, Procesamiento del Lenguaje Natural, Redes Neuronales, LLM.

UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
INGENIERÍA CIVIL INFORMÁTICA

Abstract

The military dictatorship in Chile (1973-1990) was characterized, among other things, by systematic human rights violations, including the enforced disappearance of individuals. The Secretariat for Human Rights (S.D.D.H.H) in Chile has played a crucial role in documenting these cases, amassing a vast collection of documents, many of which remain in physical format, posing significant challenges for efficient analysis and access.

It is considered that the implementation of modern Artificial Intelligence technologies, such as Large Language Models and retrieval systems like Retrieval Augmented Generation, can enhance the management and analysis of these documents. Based on this premise, an initial software prototype has been developed, a digital platform designed to facilitate access to human rights information in Chile, particularly concerning chilean historical documents in Spanish.

The developed platform leverages these new technologies to process, analyze, and extract relevant information from the documents, complemented by transforming text into Knowledge Graphs. This allows for clearer visualization and analysis of entities and their relationships. The objective is to enable efficient extraction of relevant information, ensure data veracity through source traceability, and provide tools to complement and cross-reference information from various sources.

The system has demonstrated its ability to retrieve relevant contexts and generate coherent responses with high fidelity to the original data based on natural language queries about various events, places, and individuals, among other topics.

Finally, although some limitations were identified, such as variability in the level of detail of contexts and potential information loss in the conversion of text to graphs, the findings suggest a significant potential to refine this approach and apply it in a research context. In summary, this prototype demonstrates the feasibility of utilizing advanced AI technologies to manage and analyze large volumes of unstructured historical data in Spanish, offering new avenues for the investigation and documentation of critical events in Chilean history, while also providing a better understanding of the capabilities and limitations of Large Language Models in highly sensitive and important cases such as these.

Keywords: Software Engineering, Artificial Inteligence, Human Rights, Chilean History, Information Retrieval, Natural Language Processing, Deep Learning, LLM.

Índice de Contenido

Índice de Tablas

Índice de Figuras

Índice de Código

Glosario

Software Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

GUI (Graphical User Interface) Interfaz gráfica de usuario, que permite interactuar con el sistema a través de elementos visuales como ventanas, iconos y botones.

UI (User Interface) Interfaz de usuario es el medio por el cual el ser humano interactúa con alguna máquina. En este caso, con un software.

Testing Proceso de evaluación de un sistema o componente para verificar que cumple con los requisitos especificados y funciona correctamente.

HTTP (HyperText Transfer Protocol) Protocolo de transferencia de hipertexto, conjunto formal de estándares y normas utilizado para la transferencia de información en la web.

Servidor Web Software funcionando dentro de un servidor o computador físico, capaz de aceptar peticiones HTTP o HTTPS.

API (Application Programming Interface) Interfaz de programación de aplicaciones. Intermediario que permite la comunicación entre diferentes sistemas de software.

RESTful API (Representational State Transfer API) Estilo de arquitectura de APIs para sistemas distribuidos como la web, que utiliza un conjunto de operaciones estándar para interactuar con recursos.

IP (Internet Protocol) Dirección numérica única que identifica a un dispositivo en una red, permitiendo que los datos sean enviados y recibidos entre dispositivos a través de Internet.

URL (Uniform Resource Locator) Localizador uniforme de recursos, una dirección web que especifica la ubicación de un recurso en Internet.

Puerto Número que identifica un punto específico de entrada o salida para la comunicación en una

red, permitiendo que varios servicios funcionen en un mismo dispositivo.

Socket Abstracción de software que permite la comunicación entre dos procesos, que puede estar en la misma máquina o en máquinas diferentes a través de una red. Un socket es definido por una combinación de una dirección IP y un puerto.

Endpoint Punto de acceso a un servidor web, generalmente representado por una URL y un puerto específico donde se pueden realizar operaciones como GET, POST, PUT y DELETE.

IR (Information Retrieval) Recuperación de información, un área de la informática que estudia el proceso de obtención de información desde datos estructurados y no estructurados.

IA (Inteligencia Artificial) "IA es la ciencia de hacer que las máquinas realicen tareas las cuales, hechas por humanos, requerirían inteligencia." [15] Marvin Minsky et al. 1955

ML (Machine Learning) Aprendizaje automático, una subdisciplina de la inteligencia artificial que se centra en el desarrollo de algoritmos que permiten a las máquinas aprender de datos y mejorar su rendimiento con el tiempo.

DL (Deep Learning) Aprendizaje profundo, una subdisciplina del ML que utiliza redes neuronales profundas con muchas capas para aprender representaciones jerárquicas de los datos.

LLM (Large Language Model) Modelo de lenguaje de gran tamaño. Corresponden a modelos entrenados en grandes cantidades de datos textuales, capaces de generar y comprender texto de manera coherente.

NLP (Natural Language Processing) Procesamiento de lenguaje natural, un campo de la IA que se centra en la interacción entre computadoras y el lenguaje humano.

RAG (Retrieval Augmented Generation) Es un sistema de IA que combina la búsqueda de información con la generación de texto a través de un LLM.

IMFD (Instituto Milenio Fundamento de los Datos) Centro nacional multidisciplinario con la

meta de hacer investigación de frontera en ciencia de datos y así hacer frente a los grandes problemas que existen hoy en este campo.

D.D.H.H (Derechos Humanos) Derechos humanos, principios y normas que reconocen y protegen la dignidad de todos los seres humanos. Proclamados por la ONU en 1948.

S.S.D.D.H.H (Secretaría de Derechos Humanos) Secretaría de Derechos Humanos, entidad gubernamental encargada de la promoción y protección de los derechos humanos.

1. Introducción

La dictadura militar en Chile (1973 - 1990) trajo consigo, entre muchas otras cosas, violaciones sistemáticas a los derechos humanos por parte de instituciones o grupos, bajo el mando o el amparo del gobierno de Chile [7].

Es en este contexto que nace la figura del *detenido desaparecido*, es decir, víctimas de privación de libertad forzada en donde existe la intencionalidad por parte del perpetrador de ocultar el paradero de el o los afectados.

En base a los informes escritos por las llamadas comisiones *Rettig* y *Valech* [8], además de una reciente declaración en el año 2023 por el presidente Gabriel Boric [2], se contempla un total de 1.469 personas detenidas desaparecidas, de las cuales hasta el momento sólo se ha establecido el destino de 307.

La *Subsecretaría de Derechos Humanos* en Chile ha jugado un rol crucial en la documentación y el mantenimiento de registros pertinentes a estos casos, lo cual ha sido fundamental para continuar con las investigaciones y esfuerzos para esclarecer estos hechos.

De la gran cantidad de documentos que se maneja, muchos están únicamente en formato físico, y varios de los que están en digital corresponden a transcripciones echas a mano. Así, existen limitaciones logísticas para aprovechar todos estos datos en su totalidad.

En la última década, hemos sido testigos del desarrollo de tecnologías que han transformado radicalmente nuestra forma de vivir y trabajar. La *Inteligencia Artificial* ha sido una de las principales protagonistas, brindando soluciones que han mejorado la eficiencia, productividad y calidad de vida en diversos ámbitos. Desde asistentes virtuales hasta sistemas de diagnóstico médico, la IA ha demostrado su capacidad para ofrecer soluciones que antes parecían fuera de alcance, impactando, mayoritariamente, de manera positiva en la sociedad y en nuestras vidas.

En particular, un sistema de IA son los *Grandes Modelos de Lenguaje*, los cuales han demostrado

ser herramientas excepcionales en el procesamiento y análisis de grandes volúmenes de texto. Estos modelos, alimentados por grandes cantidades de datos, y entrenados mediante técnicas de aprendizaje automático, pueden “entender” y generar texto con una coherencia que antes parecía reservada sólo para el ser humano. La posibilidad de aplicar esto en la revisión y análisis de archivos históricos podría significar un gran avance en la manera en que se accede y se utiliza la información, facilitando así la tarea de investigadores en su esfuerzo por esclarecer y documentar lo ocurrido con las víctimas de desaparición forzosa durante la dictadura militar en Chile.

El IMFD ha estado implicado constantemente en el trabajo en conjunto a la S.D.D.H.H, desarrollando proyectos y probando distintos métodos en los que el uso de la IA generativa permita servir a la comunidad rescatando información de documentos históricos de Chile. Sin embargo, durante el desarrollo de este proyecto en particular, las organizaciones implicadas han ido cambiando.

En la actualidad, este trabajo se encuentra bajo el marco del proyecto *Nuestra MemorIA*[1] el cual en sus propios términos describe su objetivo como:

> "...pretendemos aprovechar el poder transformador de la Inteligencia Artificial (IA) para gestionar, analizar e interpretar archivos históricos de la época de la dictadura chilena. Impulsado por la necesidad crítica de consolidar y escudriñar miles de documentos, fotografías antiguas y grabaciones de audio dispersas en varios archivos y colecciones, este proyecto aborda los desafíos de los archivos fragmentados, una tarea casi imposible de lograr manualmente."

Nuestra MemorIA cuenta con convenios del *Instituto Nacional de Derechos Humanos* (INDH)[2], el *Museo de la Memoria y los Derechos Humanos*[3], y la *Vicaria de Solidaridad*[4].

[1] https://nuestramemoria.ing.uc.cl/
[2] https://www.indh.cl/
[3] https://mmdh.cl/
[4] https://www.vicariadelasolidaridad.cl/

2. Preliminares

La presente sección tiene como objetivo proporcionar el contexto y los fundamentos teóricos esenciales para este trabajo. Se busca ofrecer una base sólida que permita al lector comprender los conceptos y técnicas abordados en esta memoria de título.

Dado que estos conceptos y técnicas descritos en la sección estarán debidamente demostrados, y considerando que el trabajo se sustenta en esta base teórica, se asegura la correctitud teórica del proyecto.

Además, se ha hecho un esfuerzo por presentar la información de manera clara y comprensible, incluyendo la creación de un glosario, de modo que cualquier persona pueda entender a grandes rasgos el contenido y la relevancia de este trabajo. Sin embargo, una comprensión completa y detallada puede requerir conocimientos previos en las áreas de las ciencias de la computación y matemáticas en general.

2.1. Detenidos Desaparecidos durante la Dictadura Militar en Chile (1973 - 1990)

La desaparición forzada de personas y las violaciones a los derechos humanos durante la dictadura militar en Chile constituyen crímenes de lesa humanidad según las definiciones establecidas por la ONU, el derecho internacional, y académicos [6]. Estas acciones son consideradas graves violaciones a los derechos humanos, tal como se detalla en el Plan Nacional Verdad y Justicia, y el marco conceptual correspondiente [17, p. 39].

La validación de que estos eventos ocurrieron no debe ser puesta en duda, y está respaldada por múltiples informes, documentos, procesos y reconocimientos oficiales, entre los cuales se destacan los llamados Informe Rettig [7], e Informe Valech [8], informes del Servicio Médico Legal [27], reconocimiento por parte de la ONU, y juicios realizados contra personas encontradas culpables de

ser cómplices o perpetradores de crímenes de lesa humanidad, dada las pruebas en su contra o por sus propias confesiones.

Se destaca que en el año 2000, las fuerzas armadas de Chile reconocieron finalmente las graves violaciones a los derechos humanos cometidas durante la dictadura, acordando el resguardo de la identidad de quienes entregaran los antecedentes. Para el cumplimiento de este acuerdo, se aprobó la Ley N° 19.687 [5].

Históricamente, las búsquedas de los detenidos desaparecidos en Chile han seguido diversos métodos antes de la implementación del plan actual de búsqueda, aunque con distintos grados de éxito y compromiso.

Se debe considerar que solo nos referimos a esfuerzos hechos por el Estado, ya que de manera particular, y a veces apoyados por instituciones internacionales, las familias de las víctimas de desaparición forzada han creado organizaciones, desde tan temprano como 1974, donde realizaban la denuncia pública de estos hechos, y buscaban activamente a sus seres queridos.

Los procesos de búsqueda por parte de gobiernos anteriores no han estado exentos de críticas por parte de las familias afectadas, la sociedad, gobiernos actuales e investigadores. Criticando por un lado la demora de gestiones asociadas a las búsquedas, y por otro, el enfoque judicial de estas, ubicando el encontrar responsables como objetivo principal, mientras que encontrar el paradero de las víctimas, junto con la reparación a las familias afectadas, se delega a un segundo plano [26].

Se han creado diversas comisiones y programas para la búsqueda de los detenidos desaparecidos y ejecutados políticos en Chile. La primera de ellas fue la Comisión Nacional de Verdad y Reconciliación (Rettig), creada por el decreto supremo N° 355 de 1990 del Ministerio del Interior. Esta comisión tuvo como objetivo investigar las violaciones a los derechos humanos ocurridas entre el 11 de septiembre de 1973 y el 11 de marzo de 1990. Posteriormente, la aprobación de la Ley N.º 19.123 en 1992, crea la Corporación Nacional de Reparación y Reconciliación [4, art 6], que establece una pensión de reparación y otorga otros beneficios en favor de las personas señaladas en dicha ley. Esta corporación funcionó hasta 1996, siendo sucedida por el Programa de Derechos Humanos, creado por

el decreto supremo N° 1005 de 1997 del Ministerio del Interior [18]. Este programa tiene la función de prestar asistencia social y legal a los familiares de las víctimas de ejecutados políticos y detenidos desaparecidos. Además, facilita el acceso a los beneficios establecidos por la Ley N° 19.123 y asegura el derecho inalienable de conocer la ubicación de las personas detenidas desaparecidas o de los cuerpos de las personas ejecutadas, así como las circunstancias de dichas desapariciones o muertes. Esto se logra mediante la asistencia proporcionada o la puesta a disposición de los antecedentes reunidos y custodiados por los archivos conformados por las comisiones anteriores.

En el año 2003 se crea la Comisión Nacional sobre Prisión Política y Tortura (Valech I), mediante el decreto supremo N° 1040 del Ministerio del Interior [19], creado para esclarecer la identidad de las personas que sufrieron privación de libertad y torturas por razones políticas, por actos de agentes del Estado o de personas a su servicio, en el período comprendido entre el 11 de septiembre de 1973 y el 11 de marzo de 1990. Buscando así suplir carencias encontradas en el informe anterior (Informe Rettig) abordando casos que previamente no habían podido ser confirmados debido a falta de antecedentes.

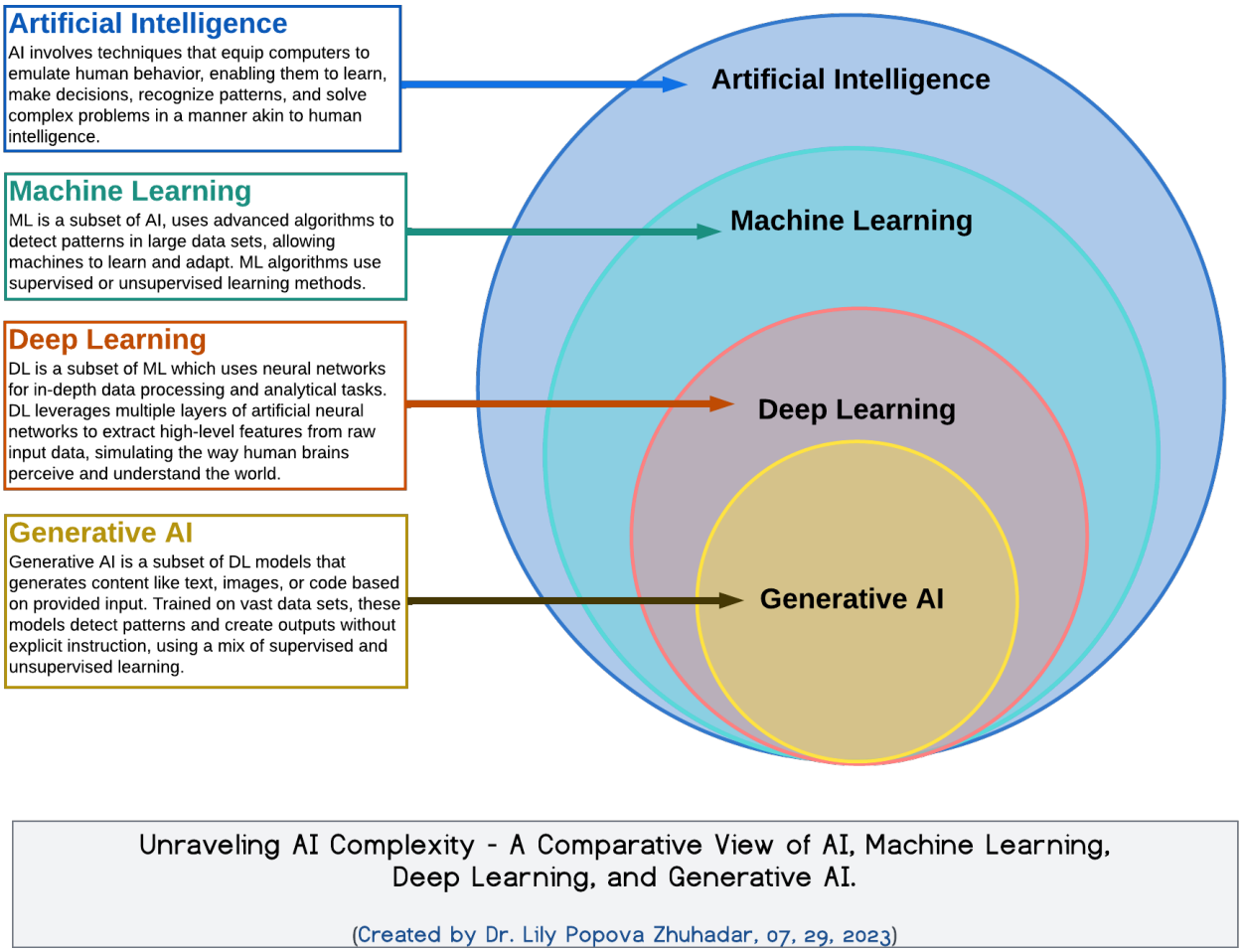
La última instancia previa al plan de búsqueda actual, se dio el 2010, cuando se creó la Comisión Asesora Presidencial para la Calificación de Detenidos Desaparecidos, Ejecutados Políticos y Víctimas de Prisión Política y Tortura (Comisión Valech II) [20].

Finalmente, se debe recalcar que la búsqueda de los detenidos desaparecidos no solo es una obligación ética y moral del Estado, sino que también tiene fundamentos legales. El derecho a conocer la verdad sobre la desaparición o muerte de las personas detenidas desaparecidas y ejecutadas es un derecho inalienable de los familiares de las víctimas y de la sociedad chilena en su conjunto, tal como se establece en el Art. 6º de la Ley N.º 19.123. Sumado a esto, Chile ha suscrito varios tratados internacionales que establecen esta obligación. Además, en el año 2023 fue promulgada, como una política pública, el plan de búsqueda actual ya mencionado [21].

2.2. Modelo del Lenguaje de Gran Tamaño

Dentro del área de estudio de la IA, si bien no hay una definición global de este concepto, existe una jerarquía bien marcada respecto a sus sub áreas, siendo los LLMs un término de alta especificidad, encontrándose dentro de lo que corresponde a la IA generativa (Generative AI) dentro de la figura 1.

Figura 1
Jerarquía de términos de IA [9]



Unraveling AI Complexity - A Comparative View of AI, Machine Learning, Deep Learning, and Generative AI.

(Created by Dr. Lily Popova Zhuhadar, 07, 29, 2023)

Un LLM es un modelo de IA entrenado en vastos volúmenes de datos textuales para procesar y generar lenguaje natural de manera coherente y contextualmente relevante. Utilizando DL, estos modelos capturan patrones y matices del lenguaje, permitiendo aplicaciones versátiles como asistentes

virtuales, chatbots, traducción automática y generación de contenido. Ejemplos notables incluyen chatGPT, GPT-4, Gemini, entre otros.

Si bien se le quiere dar un uso al servicio de la comunidad a estas tecnologías, se debe ser consciente que estos modelos no están exentos de problemas. En primer lugar, está el gran gasto energético que estos modelos requieren al entrenarse y utilizarse, lo que tiene un costo adicional intrínseco, dependiendo del valor de la electricidad en la zona donde se use, pero más importante aún es el impacto negativo directo que estos modelos tienen en el cambio climático.

Por otro lado más teórico, estos modelos se escapan de los marcos formales de aprendizaje automático, lo que tiene relación directa con su carencia de interpretabilidad y dificultad para entregar garantías analíticas.

La interpretabilidad en el DL se refiere a la capacidad de comprender y explicar cómo un modelo toma decisiones, rastreando sus salidas hasta las entradas y sus atributos clave. Esto es fundamental porque fomenta la confianza de los usuarios en las predicciones del modelo, especialmente en sectores críticos como la salud y las finanzas.

No obstante, estos modelos, suelen ser percibidos como "cajas negras". Aunque son precisos en sus predicciones, entender su funcionamiento interno y la lógica detrás de sus decisiones es complejo. Esta complejidad dificulta identificar qué características influyen en una decisión y de qué manera [31].

Un modelo puede ajustarse perfectamente a los datos de entrenamiento sin ofrecer claridad sobre lo aprendido, y rendir de buena manera incluso si ha memorizado o generado sobreajuste en su entrenamiento [22].

2.3. Embeddings

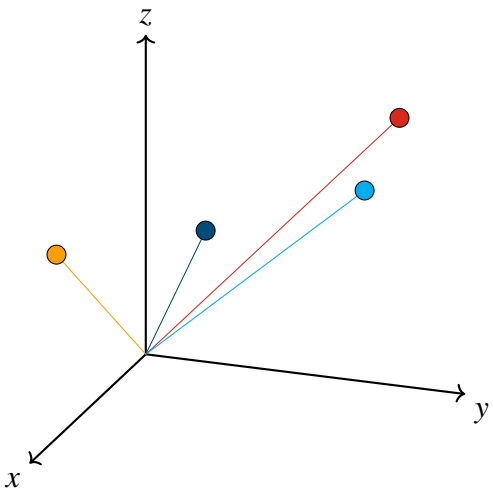
Embedding se refiere al acto de representar diversos tipos de datos, como imágenes, audio, texto, entre otros, como vectores, en un espacio vectorial, siendo el objetivo que datos que son similares en

su forma original estén cerca al medir la distancia de sus representaciones vectoriales.

En el área del NLP, y también en este trabajo, se utilizan los *Word Embeddings*, que como su nombre indica, son representaciones vectoriales de palabras (figura 2).

Figura 2
Ejemplo de Word Embedding

	x	y	z
Perro	1.4	1.7	1.8
Gato	1.3	1.5	1.4
Árbol	2.8	1.3	1.7
Casa	2.8	0.6	1.5



Existen diversas técnicas para crear un Word Embedding, como utilizar un modelo *Bag-of-Words*, el cual mide la co-ocurrencia de términos en un conjunto de documentos, sin embargo, los modelos más usados actualmente, están basados en redes neuronales, ya que han demostrado ser más efectivos para capturar relaciones semánticas complejas entre palabras.

2.4. Búsqueda Semántica

Como su nombre lo indica, corresponde al proceso de, a través de una consulta hecha por un usuario, obtener información cuyo significado es similar y relevante respecto a las palabras clave o términos presentes en la consulta inicial.

La búsqueda semántica no es algo nuevo, ya que fuera del contexto de la IA, este tipo de búsqueda es equivalente a un caso particular de *Modelo Vectorial* dentro del área del IR [1].

En este modelo, cada documento tiene una representación $d_i = (w_{i1}, w_{i2}, \cdot, w_{in})$. Es decir, un vector de n dimensiones donde cada elemento w es un peso. De manera similar, una consulta tiene un representación $q = (w_{q1}, w_{q2}, \cdot, w_{qn})$.

Lo fundamental de este tipo de modelo, es el de asignar pesos tal que al aplicar un función de rank sobre una consulta y el conjunto de documentos, se obtenga que los resultados relevantes son aquellos con mayor valor de rank.

Para asignar los pesos en este caso, se utilizan los modelos de embedding mencionados anteriormente, los cuales en la práctica entregan resultados superiores al de técnicas previas como Bag-of-Words o TF-IDF.

En cuanto a la funcion rank, debe corresponder a alguna medida de similitud entre vectores, por ejemplo, la similitud coseno.

El proceso de búsqueda termina al aplicar algún algoritmo que pueda retornar los top k términos que obtuvieron un mayor valor de rank, como lo es *k-Nearest Neighbors* (kNN), o alguna de sus versiones aproximadas.

2.5. Bases de Datos Vectoriales

Una Base de Datos Vectorial es un tipo de Base de Datos el cual almacena objetos junto con una representación vectorial del mismo. Así, permite realizar búsqueda no solo a través de un id como lo haría una Base de Datos Relacional, sino que es capaz de crear un embedding de la consulta realizada, para ejecutar un búsqueda semántica.

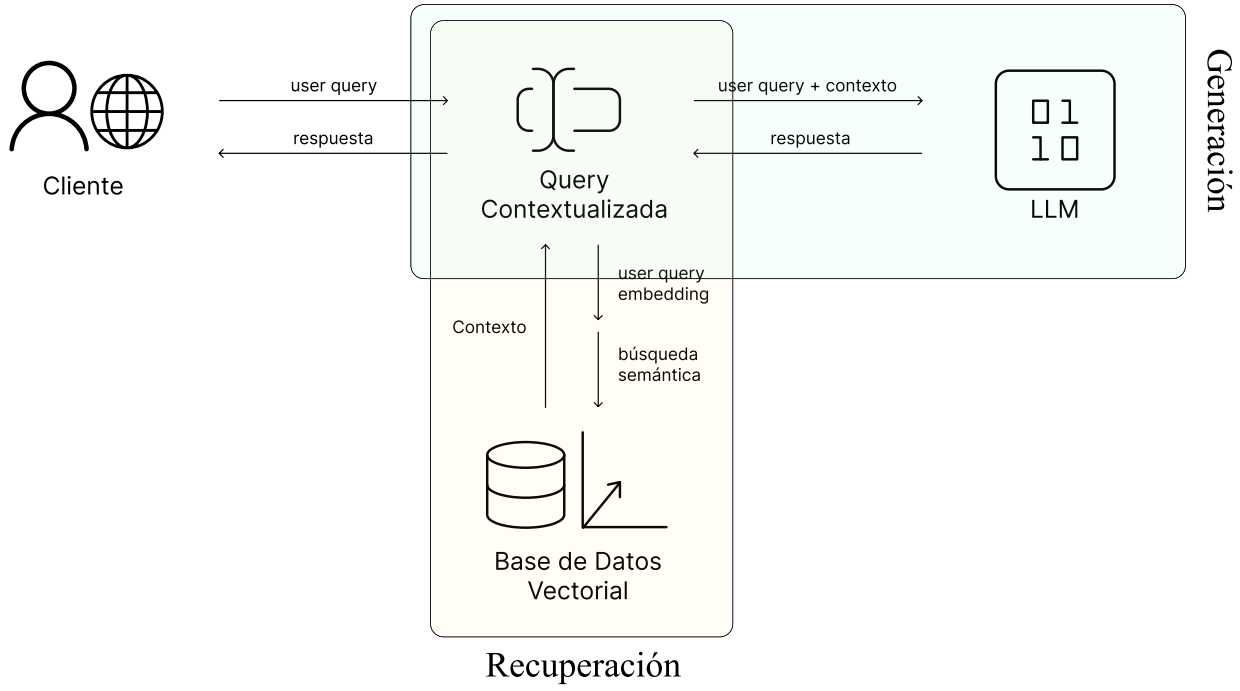
Usualmente para esta búsqueda se mantienen estructuras de datos probabilísticas que disminuyen la exploración del espacio vectorial donde existen los embeddings, y sobre estas mismas estructuras se ejecutan distintas versiones de kNN aproximado.

2.6. RAG

Retrieval-Augmented Generation (RAG) es un proceso de mejora de precisión y relevancia para resultados generados por un LLM al incorporar información externa recuperada de alguna fuente de datos. Su estructura básica se puede ver en la figura 3.

Combina dos procesos fundamentales en el manejo y generación de información, el proceso de Retrieval (recuperación) y el proceso de Generation (generación).

Figura 3
Forma General de un RAG



2.6.1. Recuperación

Dentro del RAG, la recuperación es el proceso por cual, a través de alguna técnica de recuperación de información (usualmente algún tipo de búsqueda semántica), se extraen datos relevantes para el tipo de consulta que hace el usuario. Estos datos pueden ser extraídos desde distintas fuentes, pero usualmente corresponde a alguna base de datos. Posteriormente, estos datos, denominados

"contexto", son entregados a la fase de generación.

2.6.2. Generación

De manera general, la generación corresponde al proceso por el cual, a través de una consulta o un "prompt" en lenguaje natural, el LLM genera una respuesta. A esto también se le conoce como "inferencia de LLM".

En el contexto de un RAG, el prompt corresponde a una combinación entre la consulta inicial del usuario, y el "contexto" obtenido en el proceso de recuperación.

2.7. Grafo de Conocimiento

Un KG es una estructura de datos que conecta información de manera semántica, representando entidades (como personas, lugares, cosas) y sus relaciones.

De manera formal, definimos un KG como una 3-tupla $G = (E, R, F)$, donde E corresponde al conjunto de entidades, R al conjunto de relaciones, y F al conjunto de triples (también llamado a veces "facts") tal que $(e_1, r, e_2) \in F$, y $e_1, e_2 \in E,\ r \in R$.

En cuanto a la representación concreta de un KG, existen muchos tipos y variantes, pero generalizamos en este caso a dos principales:

Resource Description Framework (RDF) Representación de KG que puede ser completamente representada a través de triples, ya que almacena relaciones entre recursos en la forma (sujeto-predicado-objeto).

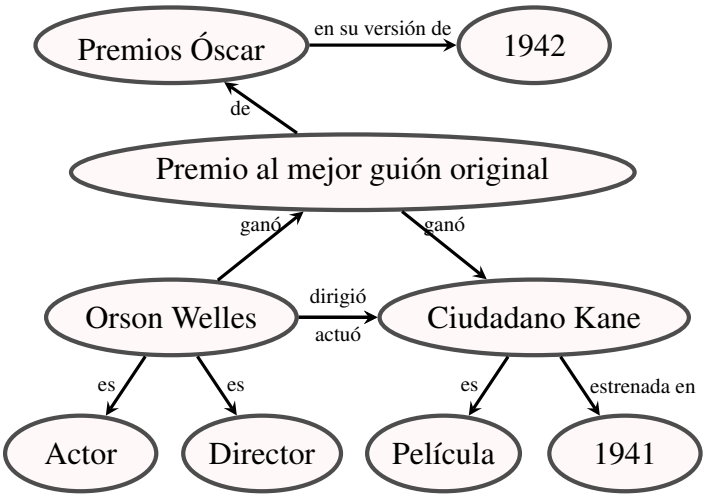
Property Graph Representación de KG donde tanto las entidades representadas como nodos, y las relaciones como aristas, pueden tener propiedades.

En este trabajo se utiliza la representación RDF debido a la simpleza de su modelo, y por lo mismo, simplicidad de serializar a otros formatos. La figura 4 (se deja el conjunto F de manera implícita en

el grafo) muestra como representar como un KG con estándar RDF, la información del siguiente párrafo de ejemplo:

> "El Ciudadano Kane fue una película estrenada en el año 1941, dirigida y protagonizada por Orson Welles. Esta película fue nominada en 9 categorías distintas en los Premios Óscar de 1942, llevándose únicamente el Premio al Mejor Guión Original."

Figura 4
Ejemplo de KG con estándar RDF



$$G = (E, R, F)$$

$$E = \{\text{"Orson Welles"}, \text{"Ciudadano Kane"}, \text{"Premios Oscar"}, \text{"Actor"}, \text{"Director"}, \text{"Premio al Mejor Guión Original"}, \text{"Película"}, \text{"1941"}, \text{"1942"}\}$$

$$R = \{\text{"es"}, \text{"dirigió"}, \text{"actuó"}, \text{"ganó"}, \text{"de"}, \text{"en su versión de"}, \text{"estrenada el año"}\}$$

2.8. Desarrollo de Software

Al desarrollar cualquier tipo de proyecto o sistema software, es necesario aplicar técnicas, herramientas, y metodologías de Ingeniería de Software. Así, aseguramos el correcto funcionamiento del sistema, además de facilitar y optimizar el trabajo en cuanto a reducción tiempo y complejidad.

Por ejemplo, a pesar que el equipo de desarrollo de este proyecto está compuesto por solo una persona, es de gran utilidad aplicar un enfoque de metodología ágil. Esta propone un trabajo iterativo e incremental y una descomposición de requerimientos desde lo general hasta lo programable, mejorando la gestión del tiempo y el cumplimiento de objetivos.

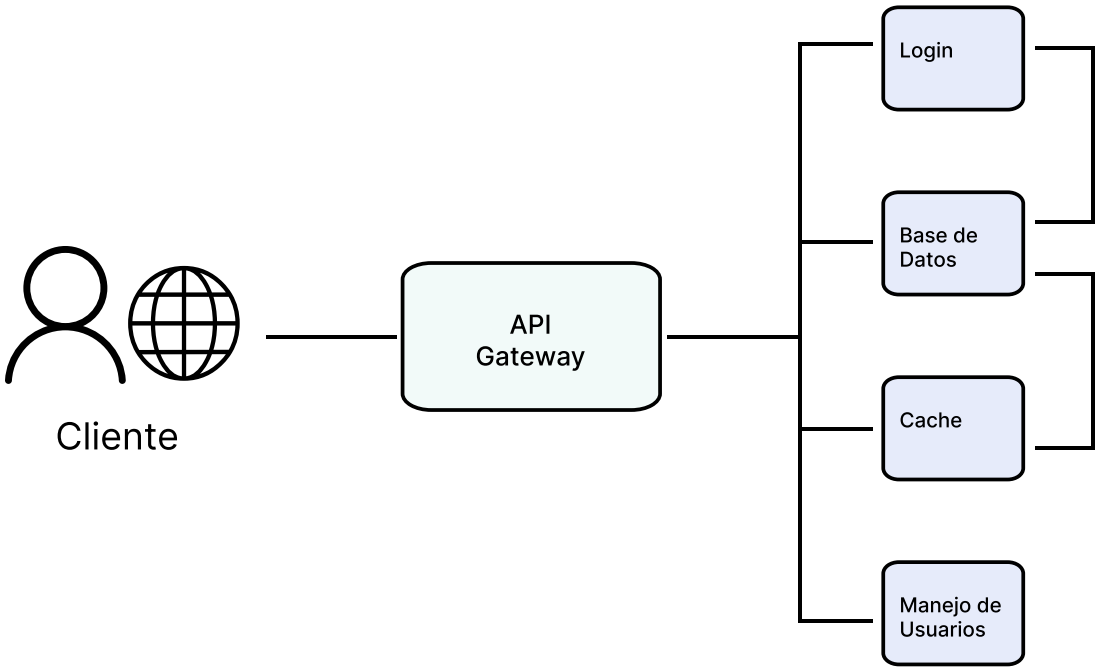
A continuación se describen algunos conceptos relevantes de la ingeniería y desarrollo de software utilizados durante el presente proyecto.

2.8.1. Arquitectura de Microservicios

Dentro de la etapa de *diseño de sistemas*, se deciden una variedad de elementos estructurales del proyecto como requerimientos del sistema, arquitectura de software, servicios a utilizar, lógica del sistema incluyendo casos de uso y flujo de los datos, y los requerimientos físicos necesarios (clusters, servidores, procesamiento, almacenamiento, etc).

Un tipo de arquitectura de software es la *arquitectura de microservicios*, en la cual una aplicación se estructura como un conjunto de servicios pequeños e independientes, cada uno de los cuales ejecuta un proceso único y se comunica con otros servicios a través de interfaces bien definidas, generalmente utilizando protocolos como HTTP a través de APIs REST o mensajería basada en eventos. Cada microservicio se implementa y despliega de manera autónoma, permitiendo que diferentes componentes de una aplicación puedan ser desarrollados, escalados y mantenidos de forma independiente. Esta arquitectura promueve la modularidad, facilitando la adopción de tecnologías diversas y mejorando la resiliencia y la capacidad de adaptación de la aplicación ante cambios y fallos. Además, al dividir la aplicación en servicios más pequeños, se facilita la implementación de prácticas de desarrollo ágil y DevOps, permitiendo ciclos de desarrollo más rápidos y eficientes. A continuación se muestra un diagrama de ejemplo de arquitectura de microservicios (figura 5)

Figura 5
Diagrama Arquitectura de Microservicios



2.8.2. Arquitectura del Cliente

El cliente es la interfaz por la cual un usuario interactúa con un sistema software, y la arquitectura del cliente establece como se estructuran los componentes involucrados, y como se relacionan entre ellos y con el resto del sistema.

Un tipo de cliente son las aplicaciones, o interfaces web, las cuales tienen como característica que se ejecutan a través de un navegador web (el cual a su vez es un cliente web ya que realiza peticiones http a servidores).

Existe una variedad de arquitecturas utilizadas en el desarrollo de aplicaciones web, como la arquitectura "Hexagonal", "Clean o Limpia", "De 3 capas", entre otras. Sin embargo, todos son formas de organizar un proyecto web en base a la separación de intereses de sus componentes, generando código mantenible y escalable en el tiempo.

A pesar de todo lo anterior, un navegador solo es capaz de renderizar HTML, CSS y Javascript,

por lo que de alguna forma toda la arquitectura, módulos, código y componentes de una aplicación web moderna deben ser capaces de funcionar en el navegador. Esto se logra a través de distintas herramientas de desarrollo:

Empaquetadores Al desarrollar con Javascript o similares, al igual que muchos otros lenguajes de programación, es común importar módulos o bibliotecas, sin embargo, el navegador web no tiene la capacidad de entrar a un servidor y buscar módulos. Ahí es cuando el empaquetador entra en juego y transforma todos los múltiples archivos en uno solo, el cual si puede ser interpretado en el navegador.

Transpiladores Corresponde a una herramienta software capaz de traducir de un lenguaje de programación a otro. En el contexto de desarrollo web, herramientas como babel o esbuild convierten el código escrito en TSX, TS, y JSX a Javascript normal, el cual es el único de estos lenguajes que puede ejecutar el navegador.

JS Frameworks Son conjuntos de bibliotecas y herramientas predefinidas que facilitan el desarrollo de aplicaciones web con JavaScript. Proporcionan estructuras y funcionalidades comunes para tareas como el manejo del DOM, la gestión de estados, y la comunicación con servidores, permitiendo a los desarrolladores enfocarse en la lógica de la aplicación en lugar de en los detalles de implementación. Ejemplos populares incluyen React, Angular y Vue.

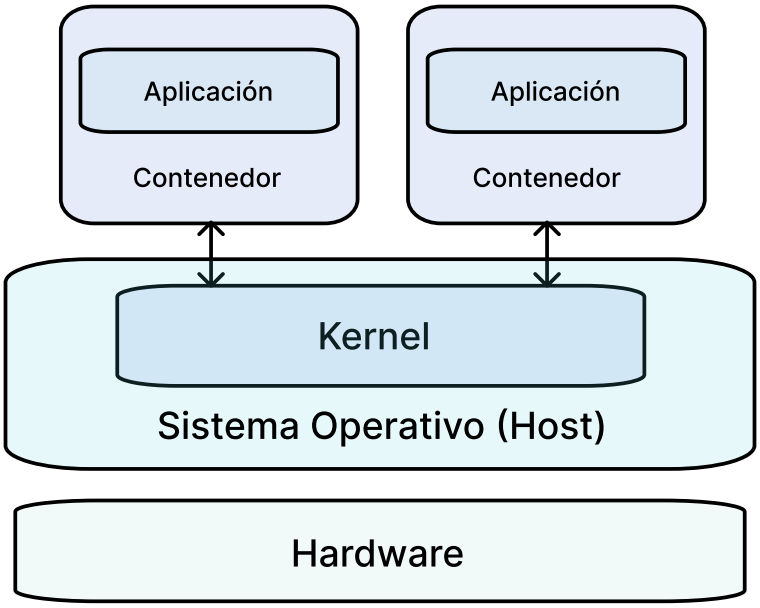
Meta-Frameworks Como su nombre indica, son frameworks construidos sobre otros frameworks, proporcionando una capa adicional de abstracción y funcionalidades específicas para simplificar y mejorar el desarrollo. Ejemplos populares son Next.js y Astro.

2.9. Docker

Docker es una plataforma que permite crear, desplegar y ejecutar aplicaciones en contenedores. Un contenedor es una unidad que empaqueta una aplicación junto con todas sus dependencias, asegurando que se ejecute de manera consistente en cualquier entorno. Los contenedores abstraen el sistema operativo subyacente, permitiendo que múltiples contenedores compartan el mismo

kernel del sistema operativo pero se ejecuten de manera aislada unos de otros (véase figura 6). Los contenedores pueden agruparse en clústeres, donde trabajan en conjunto, comunicándose entre si, para proporcionar un entorno de computación robusto y escalable.[1]

Figura 6
Funcionamiento de contenedores en Docker



Algunos beneficios del uso de docker son:

Portabilidad Ejecuta aplicaciones en cualquier entorno compatible con docker.

Aislamiento Al ser un sistema aislado, la máquina host se expone menos a riesgos externos y se mejora la gestión de dependencias, lo que implica mayor consistencia al ejecutarse en distintos tipos de máquinas.

Escalabilidad Facilita el escalado horizontal, es decir, agregando nuevos contenedores los cuales se comunican con aquellos ya existentes.

Sin embargo, también posee algunas características no deseables:

[1]Definición en el contexto de docker. En otros sistemas como *kubernetes*, o de manera general, *cluster* tiene un significado más amplio.

Complejidad Se agrega un capa extra de complejidad que se debe tener en mente a la hora de desarrollar un sistema software.

Uso de Recursos Ejecutar contenedores puede consumir recursos en exceso del sistema ya que no existe una reutilización de requerimientos entre los contenedores.

Fomenta Poca Mantención En muchas ocasiones, las aplicaciones que funcionan en docker se convierten en "cajas negras" las cuales no vuelven a ser intervenidas para actualizar paquetes o el sistema operativo, lo que puede generar vulnerabilidades, que en el peor de lo casos puede provocar que el sistema se quiebre.

No todo está asegurado A pesar de lo mencionado en los beneficios de docker, el sistema host no está libre de riesgos. Constantemente aparecen y se corrigen vulnerabilidades de ésta herramienta. Y respecto a la portabilidad, a diferencia de una máquina virtual que abstrae un sistema en su totalidad, los contenedores usan el kernel de la máquina host. Esto provoca que docker deba emular el kernel si un contenedor se ejecuta en un sistema distinto al que fue construido, pudiendo obtener resultados inconsistentes.

3. Revisión Bibliográfica

Uno de los objetivos del Plan Nacional de Búsqueda, Verdad y Justicia es "esclarecer las circunstancias de desaparición y/o muerte de las personas víctimas de desaparición forzada y su paradero" [17, p.104]. Este objetivo requiere una investigación exhaustiva en los ámbitos policial, jurídico y académico, para lo cual resulta crucial el empleo de todos los datos y tecnologías disponibles en la actualidad, por ejemplo, buscar la manera de aprovechar los novedosos LLMs para realizar tareas de extracción de información.

En este contexto de avances tecnológicos para el aprovechamiento de los datos, el área del IR ha buscado utilizar la capacidad de los LLMs de entender el lenguaje para mejorar distintos procesos involucrados a la hora de extraer y consultar por información [32]. Por ejemplo, uno de los problemas

de los sistemas de IR es la existencia de múltiples formas de hacer una consulta con la misma equivalencia lógica, que sin embargo, difieren en optimalidad a la hora de ser procesadas, llegando a un máximo de dificultad al trabajar con lenguaje natural, ya que los métodos tradicionales de optimización o reescritura de consultas, carecen de entendimiento semántico. Aquí, un LLM es capaz de mantener la intencionalidad de la consulta, e incluso lograr crear una consulta válida en otro lenguaje arbitrario.

Ahondando más en lo que es la combinación entre IR y LLMs, surgió de manera natural el interés por combinar el conocimiento inherente de estos modelos, conocido como memoria paramétrica, con información adicional extraída de diversas fuentes de datos a través de un sistema de IR, denominada memoria no paramétrica. Esta integración ha dado lugar al desarrollo de los sistemas de "Retrieval-Augmented Generation" [13] (RAG), los cuales aprovechan tanto la memoria paramétrica de los modelos de lenguaje como la memoria no paramétrica obtenida mediante técnicas de IR para ofrecer respuestas más precisas y contextualmente relevantes [11].

También se han trabajado en mejoras para el funcionamiento de los sistemas RAG, por ejemplo, creando una versión adaptativa que pueda decidir cuando recurrir a la recuperación de información [12], lo que implica menores tiempos y costo computacional en promedio. Lo anterior, es una propuesta interesante, y no muy compleja de implementar una vez se tiene la base del sistema, sin embargo, hay contextos, como en el caso de la solución de este proyecto (véase sección 7), donde se depende de obtener "metadatos" del proceso de recuperación para poder generar una respuesta.

Retomando el tema de utilizar LLMs para resolver problemas en distintas áreas, en lo que respecta al NLP, una tarea importante es la de encontrar "entidades" en un texto. La mayoría de propuestas de utilización de LLMs para encontrar entidades (Named entity recognition), está basada en el uso de "ontologías", es decir, una lista de clases a las que estas entidades pueden corresponder [23]. Sin embargo, en en el caso de este proyecto se opta por no asumir nada acerca de los datos, debido a la gran cantidad de documentos existentes, donde si bien cada uno está enmarcado por el tema de las violaciones a los derechos humanos en Chile, abarcan muchos tipos de contextos distintos. Además, dada la sensibilidad de los mismos, es preferible evitar clasificaciones erróneas que puedan

ser malinterpretadas por los usuarios.

Otro de los problemas que se desea abarcar en este trabajo, es en la conversión de texto a grafo, donde Neo Technology, la empresa detrás del popular motor de bases de datos de grafos, Neo4J [30] también ha trabajado ideas similares dentro de su ecosistema, que a través de LLMs, convierte datos desestructurados, en KGs que se pueden almacenar dentro de su sistema [24].

Finalmente, al revisar los avances relacionados al almacenamiento y extracción de información de datos estructurados como grafos, se tiene a MilleniumDB [28], el cual corresponde a una base de datos de grafos de código abierto implementada por académicos pertenecientes al IMFD, el cual demostró un mejor rendimiento que otras opciones similares tales como Blazegraph y Neo4J, entre otros [29]. MilleniumDB posee además la capacidad de búsqueda por similitud, por lo que se convierte en un candidato natural para, eventualmente, ser la BD utilizada en un trabajo similar al de esta memoria de título.

4. Problemática

La S.D.D.H.H enfrenta desafíos significativos debido al gran volumen de datos desestructurados que posee. Mayoritariamente textos en pdf, y otra parte en escaneos en pdf.

A continuación se listan las mayores dificultades identificadas:

1. Extracción de Información en Datos No Estructurados

   El gran volumen de los datos, la falta de estructura en su almacenamiento, y su poca clasificación según temáticas o palabras claves, dificulta la extracción de información relevante de manera rápida y eficiente acerca de temas y entidades específicas, como hechos, fechas, personas, instituciones y lugares.

2. Trazabilidad y Unificación de Información

   La dificultad para trazar información entre diferentes documentos es una de las principales

barreras para construir una narrativa coherente y completa de los eventos. La ausencia de una forma sistemática de relacionar datos entre múltiples fuentes impide la ratificación y el análisis exhaustivo de la información.

3. Documentación No Digitalizada

   Existe una gran cantidad de documentación histórica que se encuentra en formato de imágenes o escaneo de documentos físicos, lo que dificulta su aprovechamiento.

Las estrategias previamente utilizadas para solventar las dificultades listadas han sido incapaces de avanzar a un ritmo necesario para satisfacer las demandas y expectativas de la sociedad. Mientras que aquellas soluciones tecnológicas que se han desarrollado este último tiempo no han sido exploradas.

En particular, los últimos avances en IA han demostrado una gran capacidad a la hora de "entender" el razonamiento humano y así resolver tareas de NLP como la extracción de información en documentos, y complementar sus respuestas con distintas fuentes, sin embargo, existe la incertidumbre sobre si los famosos chatbots, asistentes inteligentes, IA generativa; LLMs, son capaces de aportar en este contexto de documentos históricos en español (el idioma es un dato no menor).

Otro cuestionamiento al uso de las LLMs en temas tan sensibles como estos, es el de la veracidad de las respuestas obtenidas, las llamadas "alucinaciones", que es cuando el modelo de IA entrega con seguridad una respuesta con información parcial o totalmente falsa.

Dado lo anterior, el IMFD, quien trabaja en conjunto con la S.D.D.H.H encargó un prototipo para conocer de que formas se pueden aprovechar estas tecnologías, y cuales son los límites asociados.

5. Definición del problema

En base a las problemáticas expuestas, se identifica el siguiente objetivo general:

- Desarrollo de una plataforma digital que facilite la labor de investigación en el ámbito de los D.D.H.H en Chile. La plataforma proporcionará acceso a información actualizada y verificada,

permitirá la extracción eficiente de datos relevantes, y ofrecerá herramientas para complementar y contrastar información de diversas fuentes. Estará diseñada para ser intuitiva y accesible, permitiendo a cualquier usuario utilizarlo sin obstáculos técnicos.

Ahora, se descompone este objetivo principal, en secundarios, cuyo objetivo es separar macro requerimientos lo más disjuntos posibles. Es decir, evitar que estos objetivos tengan resoluciones en común.

1. Almacenar, administrar y procesar los documentos dentro del sistema, para centralizar el acceso a los datos y automatizar sus transformaciones necesarias para poder utilizarlos eficientemente.
2. Garantizar que la información extraída proviene de fuentes confiables, para asegurar la veracidad y precisión de los hallazgos.
3. Implementar un sistema IR que permita consultar sobre entidades específicas, y genere respuestas en formatos que favorezcan y aceleren la obtención de conclusiones preliminares a medida que los documentos son procesados.
4. Complementar y contrastar información de distintas fuentes y documentos, para obtener una visión más completa y hallar patrones que no son evidentes a simple vista.
5. Desarrollar un sistema accesible y fácil de usar, para evitar lidiar con interfaces complicadas o conocimientos extras.

6. Propuesta

Para solucionar el problema definido, se propone la creación de una plataforma web que sea segura, de fácil uso, que almacene, organice, y permita el acceso a documentos, y, con esta información y utilizando diversas fuentes, sea capaz de responder a consultas acerca de hechos, fechas, lugares, personas, organizaciones e instituciones relevantes durante el periodo de 1973 a 1990, en el contexto de las violaciones a los D.D.H.H durante la dictadura militar en Chile.

El funcionamiento interno de esta plataforma se compondrá de dos partes principales, con los que se busca cumplir los requerimientos propuestos, un sistema RAG, y una solución al problema *texto a grafo*, donde el grafo corresponde a un KG asociado a la respuesta del sistema RAG creado en base a los mismos fragmentos de texto utilizados para generar la respuesta textual.

Se decide implementar un sistema RAG por varias razones. Primero, se desea eliminar el escepticismo existente para las respuestas que pueda entregar un LLM, dada la capacidad del sistema de mostrar cuales son las fuentes que fueron utilizadas para entregar una respuesta. Además, se tiene la habilidad de recuperar la información más relevante de entre todos los documentos disponibles, dado una consulta inicial hecha en lenguaje natural.

La visualización mediante un KG complementa la respuesta textual hecha por el LLM, ofreciendo una manera clara y comprensible de representar relaciones complejas entre diferentes elementos de los datos. Esta representación gráfica es especialmente útil en el contexto histórico-político con el que se trabaja, donde existe una gran cantidad de entidades como eventos históricos, personas, lugares e instituciones, permitiendo identificar patrones y conexiones que podrían no ser explicitas a través de métodos tradicionales de análisis de datos.

Además, el KG corresponde a una forma estructurada de almacenar la información, facilitando un guardado y recuperación eficiente, ya sea de manera no persistente en memoria primaria como en el caso de este trabajo, o utilizando bases de datos de grafos.

En la tabla 1, se desarrollan los objetivos secundarios listados en la sección anterior, lo cual se hará descomponiendo estos requerimientos más abstractos en algo más concreto, siguiendo lo que sería una asignación de *historias de usuario* de las cuales ya se pueda extraer tareas específicas de desarrollo de software. Considerando que solo hay un tipo de usuario, se omite la narrativa en primera persona, usual en este tipo de métodos de gestión.

Tabla 1
Asignación de historias de usuario para subobjetivos

Subobjetivo 1	
1. Quiero una plataforma que me permita subir, almacenar y administrar documentación para mejorar la organización y disponibilidad de los datos relacionados a los D.D.H.H en Chile.	1.1 Crear plataforma web que permita subir, acceder, eliminar y actualizar archivos. 1.2 Instalar y configurar una base de datos que almacene de manera persistente los documentos y permita manejarlos intuitivamente.
2. Quiero un sistema que al recibir datos, pueda procesarlos para hacerlos utilizables por el sistema.	2.1 Crear una ingesta de datos que procese los documentos al subirlos a la plataforma y permita al resto del sistema acceder a ellos.
Subobjetivo 2	
1. Quiero que la plataforma muestre las fuentes utilizadas en la información que se extraiga, para tener la certeza de que los datos provienen de una fuente confiable.	1.1 Hacer que la plataforma retorne referencia a las fuentes utilizadas para responder las consultas.
2. Quiero que la plataforma provea las fuentes que utiliza, para en caso de no conocer, o no estar seguro acerca de alguna, poder averiguar acerca de ella.	2.1 Listar las fuentes, una pequeña descripción y/o enlaces oficiales.
Subobjetivo 3	

1. Quiero poder visualizar los datos de forma clara y organizada, para facilitar la comprensión y extracción intuitiva e inmediata de información.	1.1 En base a las consultas recibidas, generar un KG con datos obtenidos de la recuperación de información del sistema.
2. Quiero poder obtener información de manera rápida y eficiente sobre entidades específicas (como personas, instituciones y lugares), para acelerar el proceso de investigación y análisis.	2.1 El sistema genera una respuesta textual utilizando un LLM a las consultas realizadas, lo que entrega información específica y puntual al preguntar sobre entidades.
Subobjetivo 4	
1. Al buscar información acerca de un tema, quiero obtener distintas fuentes donde se hable de aquello, para tener una visión amplia del tema desde múltiples perspectivas.	3.1 Al recuperar información, el sistema utiliza todos los documentos disponibles a través de búsqueda semántica y retorna las referencias a las fuentes.
2. Quiero que la plataforma sea capaz de inferir acerca de un tema usando todas las fuentes que tiene disponible, para obtener un análisis más profundo y detallado basado en la información recopilada.	3.1 El sistema entrega la información recuperada como contexto a un LLM, el cual es capaz de inferir y generar conocimiento en la respuesta entregada.
Subobjetivo 5	
1. Quiero que la plataforma sea auto explicativa en cuanto a su uso y funciones, para evitar perder tiempo adivinando o buscando instrucciones.	1.1 El diseño de la interfaz de usuario debe ser intuitivo, fácil de usar, y estar señalizado para dar a entender que función cumple cada componente.

2. Quiero que el uso de la plataforma no requiera conocimientos extras de computación o IA, para poder enfocarse en buscar información sobre D.D.H.H.	2.1 Las búsquedas o consultas en la plataforma serán hechas a través del lenguaje natural. 2.2 Para manejar la base de datos directamente, se hace disponible una "GUI" web que permite gestionar los datos sin tener que programar nada.

7. Desarrollo de la Solución

7.1. Preámbulo

7.1.1. Respecto a los Datos no Digitalizados

Antes de diseñar el sistema, se decide separar el proceso de digitalización de textos escaneados o imágenes de texto del sistema. Se considera que se debe trabajar como una tarea aparte, y que el sistema solo reciba archivos pdf válidos. Por lo tanto, se elimina del alcance de este proyecto.

Para realizar la conversión a formato pdf, se decide utilizar el OCR de Google Cloud[1], ya que pagar la API por un tiempo determinado es considerablemente menos costoso que ejecutar de manera local modelos de segmentación y OCR.

7.1.2. Uso de Frameworks: Langchain y LlamaIndex

Una pregunta natural que surge al diseñar el sistema es si usar frameworks como Langchain[1] o Llamaindex[2], cuyo objetivo es simplificar y acelerar la creación de estas aplicaciones para obtener

[1] https://cloud.google.com/use-cases/ocr
[1] https://www.langchain.com/
[2] https://www.llamaindex.ai/

un producto listo para producción (production-ready).

Estos frameworks o esquemas de trabajo para desarrollar aplicaciones con LLMs han explotado en popularidad desde el año 2023, y en la actualidad, para mucha gente su uso va por defecto a la hora de trabajar con LLMs. Sin embargo, se decide no utilizarlos debido a lo siguiente:

1. Dificultad para Modificar Funcionamientos Internos:

   La modificación de funcionamientos internos es difícil debido al uso de numerosas clases interdependientes, lo que complica ajustes y puede causar errores de compatibilidad. Esto limita significativamente la personalización.

2. Dependencia y Sostenibilidad:

   La complejidad de ciertos ecosistemas tecnológicos crea una dependencia significativa, ya que integrarse es fácil pero salir es difícil y costoso. Esta dependencia puede llevar a gastos adicionales cuando las organizaciones optan por servicios de pago para simplificar la configuración de herramientas externas ("langsmith"[2] como ejemplo de esto último). Mantener la independencia tecnológica es esencial para la sostenibilidad a largo plazo del proyecto, ya que minimiza riesgos relacionados con cambios en licencias, soporte o dirección del desarrollo (véase caso "redis"[3]). Esto permite que el proyecto evolucione y se adapte a nuevas tecnologías sin restricciones externas.

7.2. Diseño del Sistema

Ahora, en base a las tareas creadas y los componentes identificados, se diseña la arquitectura del sistema como se muestra en el diagrama de la figura 7.

En la parte superior de cada flecha se encuentra el método de comunicación utilizado por el componente desde donde nace la flecha, y en la parte inferior, el endpoint del componente apuntado. Los nombres

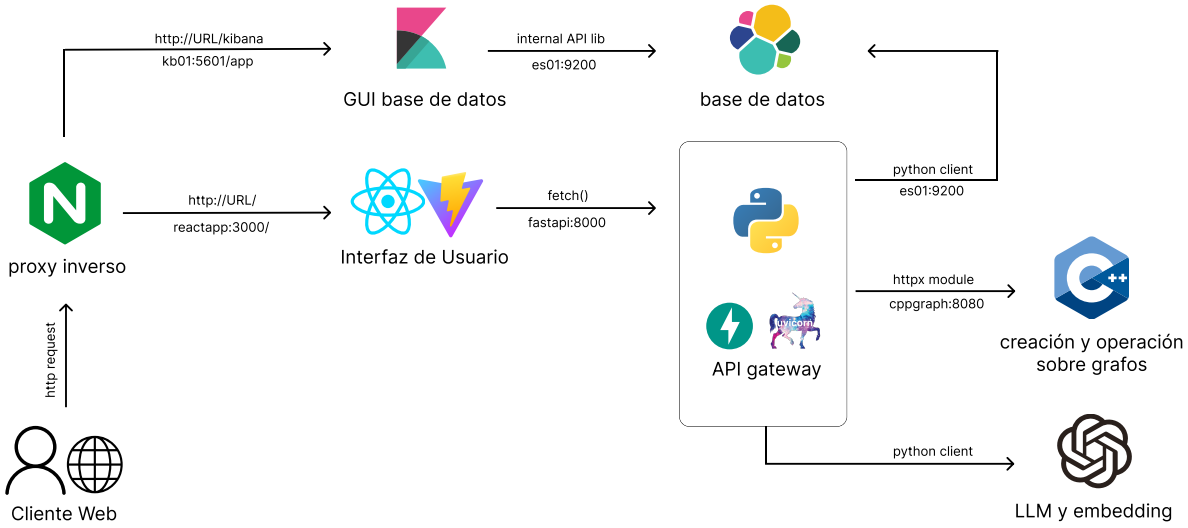
[2] https://www.langchain.com/langsmith
[3] https://redis.io/blog/redis-adopts-dual-source-available-licensing/

que tienen las URLs, como "es01", "reactapp", etc, son debido a que el sistema funcionará como un cluster en docker, con cada servicio correspondiendo a un contenedor independiente. Docker gestiona las redes dentro del cluster a través de una red virtual, con la que define los nombres de las URLs con los nombres de cada contenedor.

A través del navegador (cliente web) se puede acceder a dos enlaces distintos manejados por el proxy inverso *nginx*. Al requerir el subdominio de Kibana, se accede a la aplicación web homónima, mientras que al hacer la petición al enlace raíz ("/"), entraremos a la UI principal del sistema.

A través de la UI se envían peticiones a distintos endpoints del servicio de fastapi, el cual funcionando como API gateway, redirige las consultas hacia el servidor de C++, o a los clientes de OpenAI y de la base de datos.

Figura 7
Diseño del Sistema



7.3. Manejo de los Datos

A través del cliente web, un usuario puede subir un archivo desde su computador local, y la aplicación se encarga de codificar el archivo a base 64, para de esta manera, poder enviarlo a través de una petición de tipo POST al servidor corriendo la aplicación de fastAPI en Python.

7.3.1. Elasticsearch

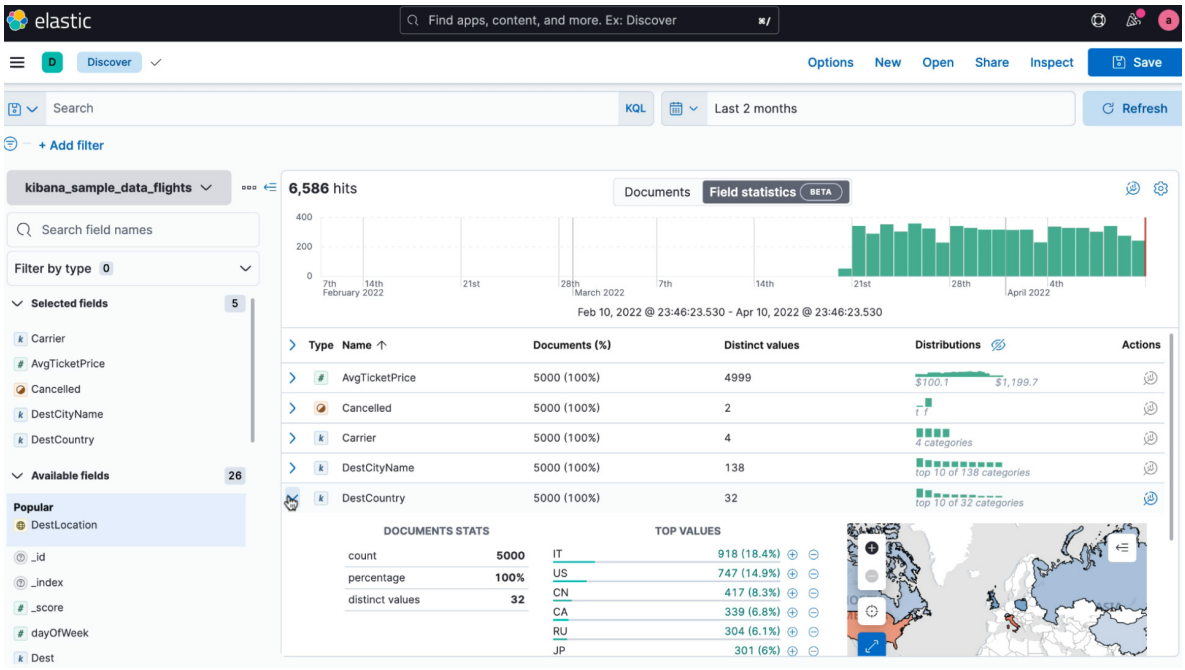
Elasticsearch corresponde a un motor de búsqueda y base datos NoSQL orientada a documentos con las siguientes características principales:

1. Indexación: Elasticsearch utiliza un esquema de indexación que permite una rápida recuperación de datos. Los datos se almacenan en índices, que son estructuras optimizadas para la búsqueda y recuperación eficiente. Cada índice puede tener múltiples documentos, y cada documento es una colección de campos clave-valor.
    - Documentos: La unidad básica de información que puede ser indexada. Un documento es un objeto JSON que contiene información estructurada.
    - Consultas: Al ser una base de datos NoSQL, los datos no se extraen a través de un lenguaje típico de SQL, sino que se utiliza un lenguaje propio (ES|QL) optimizado para la obtención de documentos. Este a su vez se abstrae con funciones disponibles en los clientes de Elasticsearch en distintos lenguajes.
2. Base de Datos Vectorial: Además de su capacidad de búsqueda de texto completo, Elasticsearch puede funcionar como una base de datos vectorial. Esto es especialmente útil para aplicaciones de ML y búsqueda semántica, donde los datos pueden ser representados como vectores en un espacio de alta dimensión.
    - Vectores: Elasticsearch puede almacenar y buscar datos representados como vectores, permitiendo la búsqueda por similitud en espacios de alta dimensión.
    - kNN: Elasticsearch soporta algoritmos de búsqueda de k vecinos más cercanos, lo que es útil para tareas como la búsqueda de imágenes, recomendaciones y análisis de datos no estructurados.

7.3.2. Kibana

Kibana es una herramienta de visualización, exploración, y manejo de datos que sirve como interfaz gráfica de usuario (GUI) para Elasticsearch (ejemplo en figura 8). Permite a los usuarios visualizar y analizar los datos almacenados en Elasticsearch mediante gráficos, tablas y mapas interactivos, además de permitir el uso de la base de datos directamente a través de la interfaz.

Figura 8
Screenshot de Kibana

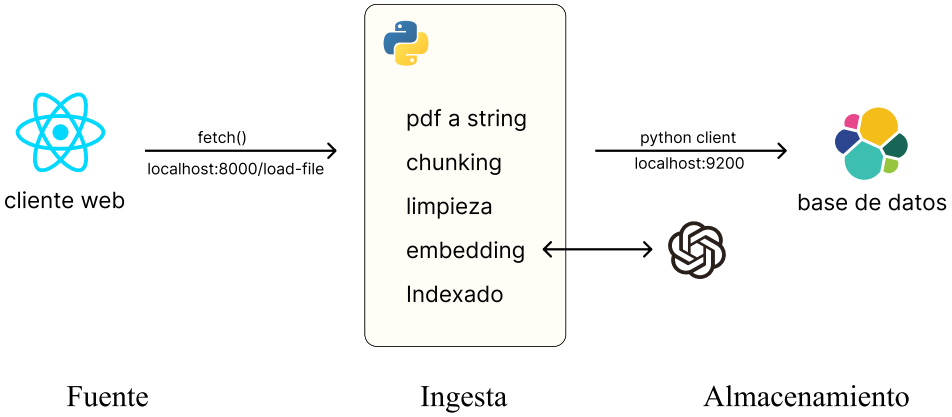


7.3.3. Ingesta de Datos

Por ingesta de datos nos referimos al conjunto de procesos desde subir un archivo al sistema, preprocesarlo y aplicarle transformaciones, hasta ser almacenado en la base de datos.

Existen tres componentes principales, la fuente de los datos, la ingesta o transformación, y el almacenamiento. A continuación se muestra el diagrama de este proceso dentro del sistema (figura 9):

Figura 9
Proceso de Ingesta de Datos



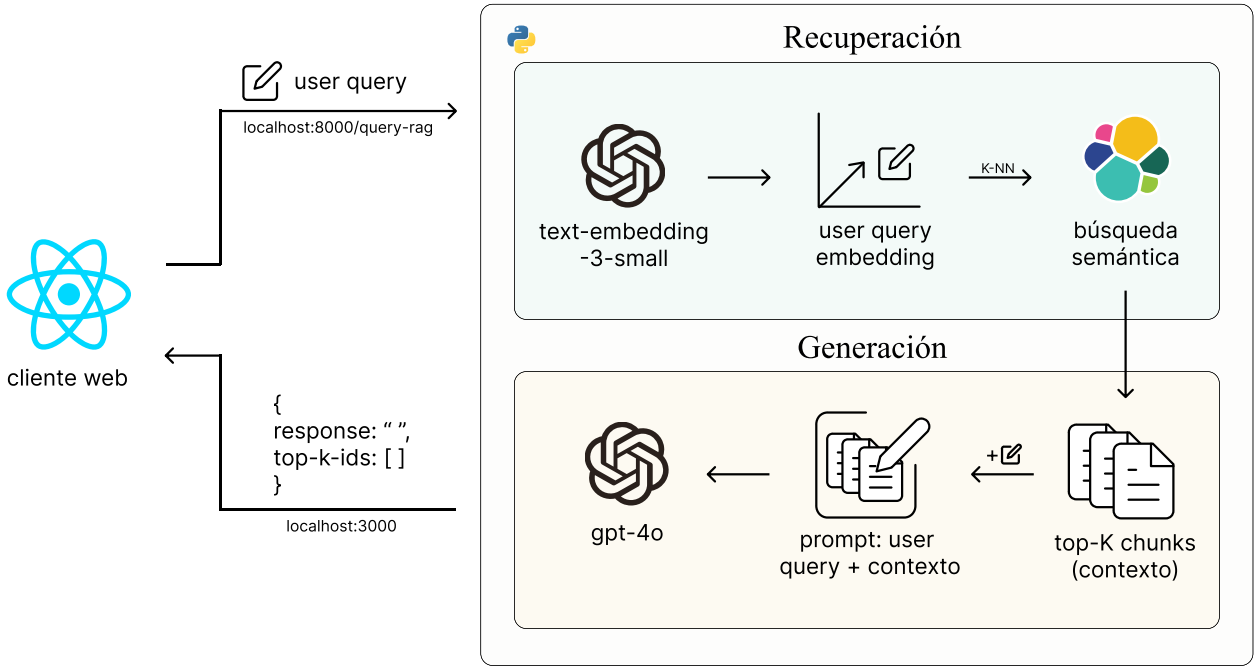
El elemento central, el cual es el servicio hecho en Python con FastAPI, corresponde a la ingesta. Una vez que recibe algún archivo desde el cliente web, realiza lo siguiente:

1. Guarda una copia del archivo en el sistema local.
2. Convierte el archivo pdf a tipo de dato string.
3. Le aplica "chunking" al texto, es decir, lo subdivide en trozos o fragmentos con la misma cantidad de caracteres.
4. Usando un modelo de embedding con la API de OpenAI, se genera una representación vectorial de cada chunk o fragmento.
5. Se crea un índice en elasticsearch con el nombre del documento original.
6. Cada chunk es indexado en elasticsearch, en el índice creado previamente.

7.4. RAG en el Sistema

Se describen los componentes utilizados para implementar el RAG dentro del sistema (véase figura 10).

Figura 10
RAG en el Sistema Creado



Para realizar la búsqueda semántica dentro del sistema RAG, la cual es un proceso dentro de la fase de recuperación, se crea una representación vectorial de la consulta del usuario utilizando el modelo de embedding. Posterior a esto, Elasticsearch permite buscar documentos en base a sus embeddings previamente creados en la ingesta de datos.

Sumado a lo anterior, para que esta búsqueda funcione, Elasticsearch solicita definir dos parámetros, una función de distancia, y un algoritmo de búsqueda de entre las opciones que tiene disponible. A continuación se describen las opciones elegidas:

Similitud Coseno. Corresponde a una medida de similitud entre vectores en un espacio con producto

punto definido. Se define como:

$$\text{Sean } u, v \text{ vectores con un ángulo } \theta \text{ entre ellos, similitud coseno}(u,v) = \cos\theta = \frac{u \cdot v}{\|u\|\|v\|}$$

kNN aproximado. Corresponde a una implementación del algoritmo Hierarchical Navigable Small Words (HNSW)[14], el cual utiliza la idea de búsqueda por capas de una *skip list*, donde ahora cada capa corresponde a un subgrafo con menor número de nodos que la capa inferior, llegando al grafo completo en la capa más profunda.

En el RAG implementado, se usan dos modelos utilizando la API de OpenAI, debido a la facilidad del uso, y que por el momento, es más costo-efectivo en tiempo de desarrollo y en gasto energético que utilizar un LLM de forma local o en un servidor propio.

text-embedding-3-small. Corresponde a la versión liviana y eficiente de los modelos actuales de embedding que ofrece OpenAI. No hay información oficial sobre la arquitectura o funcionamiento interno de estos modelos. Se tiene como característica que la dimensionalidad de este modelo es flexible hasta las 1536 dimensiones, es decir, a cambio de precisión, se pueden obtener embeddings de menos dimensión, lo que disminuye en gran medida la memoria y almacenamiento necesario para trabajar.

gpt-4o. Corresponde al modelo generativo multimodal de OpenAI con mejor rendimiento a la fecha[1]. Finalmente, ni la arquitectura específica, más allá del hecho que es un tipo de "transformer", ni la cantidad de parámetros que tiene este modelo, es información pública, pero los rumores de distintas fuentes parecen coincidir en que está por sobre el trillón (en el sentido anglosajón de la palabra) de parámetros.

Finalmente, el sistema RAG entrega dos respuestas. Una *respueta textual* que sería la típica respuesta en texto, en lenguaje humano, esperada por un LLM, y un respuesta correspondiente a un conjunto de triples en formato JSON.

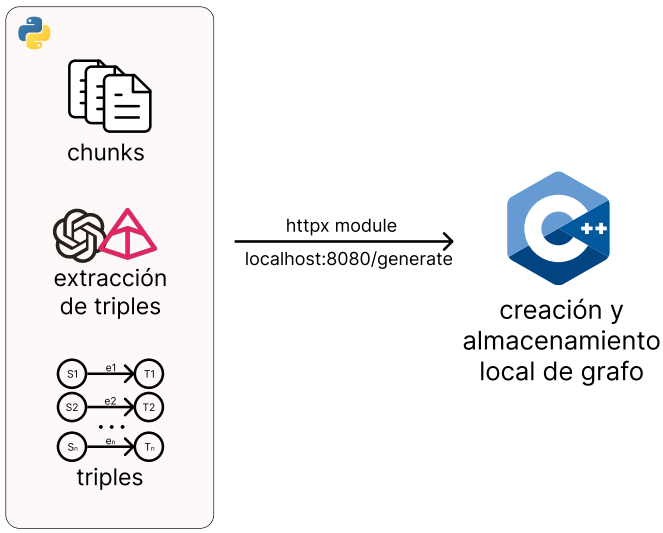
[1] https://openai.com/index/hello-gpt-4o/

Es sabido que no es tan fácil asegurar que el output de un LLM esté en el formato deseado, es por esto que se utilizan los modulos de python *Pydantic*[1] e *Instructor*[2]. Pydantic, de forma resumida, valida y/o exige formatos y tipos definidos en clases de Python (recordar que es un lenguaje no tipado). Instructor, por otro lado, sistematiza la obtención de clases hechas con Pydantic a través de LLMs.

7.5. Texto a Grafo de Conocimiento

7.5.1. Creación del Grafo

Figura 11
Creación de Knowledge Graph



Para que la UI sea capaz de renderizar el KG en el navegador, necesita recibir un grafo en formato especial en JSON, Graphology [25]. Es por esto que en el programa se almacena el KG de forma que la exportación a este formato sea lo más directo posible.

[1] docs.pydantic.dev
[2] python.useinstructor.com

En el código 1 se puede observar como se almacenan nodos y aristas en el programa, y su equivalencia, con valores "placeholder", en formato Graphology JSON en el código 2.

```cpp
struct Node {
  std::string key;
  Attributes_n attributes;
  Node(std::string key, Attributes_n attributes) : key(
    key), attributes(attributes) {}
};
struct Attributes_n {
  int id, size;
  double x, y;
  std::string label, color;
  Attributes_n(int id, int size, double x, double y,
    std::string label, std::string color)
    : id(id), size(size), x(x), y(y), label(label),
    color(color) {}
};

struct Edge {
  std::string source, target;
  Attributes_e attributes;
  Edge(std::string source, std::string target,
    Attributes_e attributes)
    : source(source), target(target), attributes(
    attributes) {}
};
struct Attributes_e {
  int chunk_id, size;
  bool forceLabel;
  std::string label, color;
  Attributes_e(int chunk_id, int size, bool forceLabel,
     std::string label, std::string color)
    : chunk_id(chunk_id), size(size), forceLabel(
    forceLabel), label(label), color(color) {}
};
```

Código 1
Estructuras de Nodos y Aristas en el código de C++.

```json
{
 "nodes": [
    {
      "key": "A",
      "attributes": {
        "id": 1,
        "label": "A",
        "color": "",
        "size": "",
        "x": "",
        "y": ""
      }
    },
    {
      "key": "B",
      "attributes": {
        "id": 2,
        "label": "B",
        "color": "",
        "size": "",
        "x": "",
        "y": ""
      }
    },
 ],
 "edges": [
    {
      "source": "A",
      "target": "B",
      "attributes": {
        "chunk_id": 1,
        "color": "",
        "forceLabel": "",
        "label": "AB"
      },
    }
 ]
}
```

Código 2
Ejemplo de output serializado a JSON en estándar Graphology.

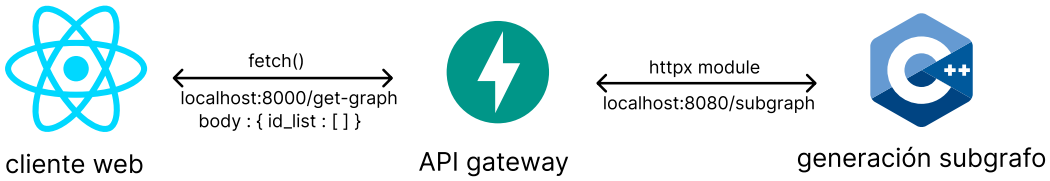
Una vez el servidor de C++ recibe los triples generados por la LLM como input, va creado los nodos y las aristas correspondientes tal como se describieron en el código anterior. Crea un vector de punteros a nodos, un vector de punteros a las aristas, y una tabla hash donde la llave es el id de algún chunk, y el valor son todas las aristas creadas a partir de ese chunk en particular.

La idea de almacenarlos como punteros, es la de siempre acceder al mismo objeto independiente de la estructura o clase a través de la que se acceda.

Adicionalmente, se crea una lista de adyacencia, la cual representa el digrafo subyacente del KG, la cual será utilizada para aplicar con ella los algoritmos de PageRank y de Layout.

7.5.2. Generación de Subgrafo

Figura 12
Creación de subgrafo en base a lista de ids



Una vez se obtiene la lista de ids de chunks para crear el subgrafo, se accede a las aristas a través de la tabla hash mencionada antes y estas aristas se agregan a un nuevo vector que corresponde a las aristas del subgrafo.

A través de cada arista, se accede a los nodos que forman parte de ellos, y verificando que no hayan repeticiones, se agregan a un nuevo vector de nodos para el subgrafo.

Finalmente, los dos vectores creados para almacenar el subgrafo son serializados a formato JSON en estándar de graphology.

7.5.3. Desafíos

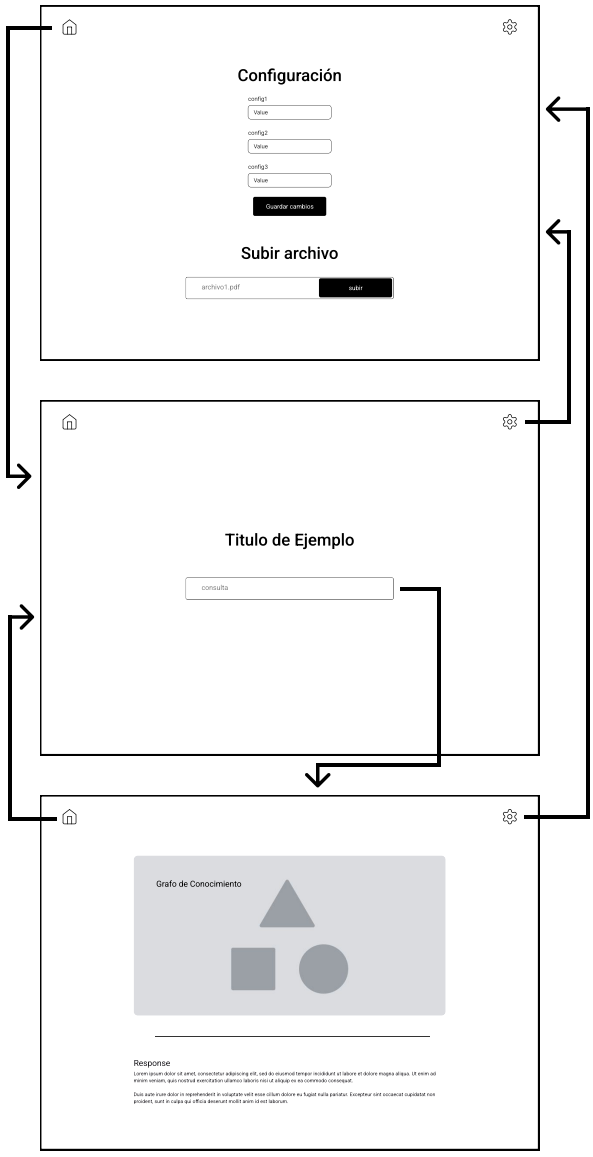
Algunos desafíos que quedan para esta solución del problema es la desambiguación de nodos cuando una entidad representa múltiples conceptos. La combinación de nodos que representan la misma entidad (por ejemplo el nombre completo de una organización versus sus siglas), y comparar si es mejor mantener todo el proceso en inglés, debido a que la mayor cantidad de datos de entrenamiento de las LLMs suele estar en este idioma.

7.6. Interfaz de usuario

7.6.1. Diseño

Se apuntó a crear un diseño y experiencia sencillo y minimalista. Se diseña un prototipo estilo ‘wireframe’.

Figura 13
Prototipo de la Interfaz



Dada la simpleza de la aplicación web, no se utiliza ninguna arquitectura en particular, pero si se tiene en consideración de tener una capa de acceso de datos separada, la cual se encarga de utilizar la función fetch de React para realizar peticiones al backend.

En el caso que eventualmente la interfaz web, más allá de necesitar cambios de diseño, requiera funcionalidades extras, se va a refactorizar para implementar alguna arquitectura que mantenga la

escalabilidad a largo plazo como *Clean* o *Hexagonal*.

7.6.2. Tech Stack

Por *tech stack* nos referimos a las tecnologías principales utilizadas para desarrollar la aplicación web.

Vite Es una herramienta para la construcción de proyectos web ("build tool" en inglés). Por un lado entrega facilidades durante el desarrollo de una aplicación web, como por ejemplo, para este proyecto se utilizó el script de vite para generar un esquema o plantilla para comenzar a desarrollar. Por otro lado, a la hora de utilizar el comando "build", transpila archivos *.tsx*, *.ts* y *.jsx* a Javascript "puro", y empaqueta los módulos de javascript para que el navegador los pueda interpretar.

React Es una biblioteca de javascript para construir interfaces de usuario. Permite crear componentes reutilizables que pueden manejar su propio estado, lo que facilita la construcción de aplicaciones web dinámicas y complejas. En este proyecto, React se utiliza para poder entregar una interfaz moderna, interactiva y fácil de usar.

TypeScript Es un superconjunto de JavaScript que añade tipos estáticos opcionales. Esto ayuda a detectar errores en tiempo de desarrollo, lo que mejora la robustez y mantenibilidad del código. En este proyecto, TypeScript se usa para asegurar que el código sea más predecible y menos propenso a errores. Sumado a que tanto vite como react soportan su uso nativamente.

Tailwind Es un framework de CSS que ayuda a dar estilo a interfaces de usuario de manera rápida y fácil. En lugar de escribir CSS personalizado, se utilizan clases predefinidas para aplicar estilos directamente en el HTML o JSX.

Shadcn Es una colección de componentes UI en React, con integración automatizada a través de comandos. En la descripción en su página web[1] explícitamente menciona que shadcn no es una biblioteca, ya que no se guardan todos los componentes como un módulo dentro del proyecto.

[1] ui.shadcn.com

En cambio, se utiliza la linea de comandos para agregar el componente específico que se quiera utilizar.

7.6.3. Visualización de Respuestas

Para visualizar la respuesta en texto del LLM solo se necesita agregarlo a algún componente de la aplicación. Sin embargo, para mostrar el KG si se requirió un mayor trabajo donde los siguientes tres componentes estuvieron presentes:

PageRank Es el famoso algoritmo introducido por Google [3] el cual tiene como objetivo medir la importancia de cada nodo de un grafo, basado en la idea que una página (nodo) es importante si muchas otras páginas importantes apuntan a él.

Este algoritmo resuelve la siguiente función para un nodo i:

$$PR(i) = \frac{1-d}{N} + d \sum_{j \in M(i)} \frac{PR(j)}{L(j)}$$

donde:

- $PR(i)$ es el PageRank del nodo i.
- d es el factor de amortiguación (damping factor), que se establece típicamente en 0.85.
- N es el número total de nodos en el grafo.
- $M(i)$ es el conjunto de nodos que apuntan al nodo i.
- $PR(j)$ es el PageRank del nodo j que apunta al nodo i.
- $L(j)$ es el número de enlaces salientes del nodo j.

El primer término, $\frac{1-d}{N}$, representa la probabilidad de que un usuario aleatorio llegue al nodo i directamente, mientras que el segundo término, $d \cdot \sum_{j \in M(i)} \frac{PR(j)}{L(j)}$, representa la probabilidad de que un usuario llegue al nodo i siguiendo los enlaces de otros nodos.

El algoritmo de PageRank se resuelve iterativamente, comenzando con una estimación inicial de $PR(i)$ (por ejemplo, $PR(i) = \frac{1}{N}$ para todos los nodos), y actualizando los valores de PageRank hasta que converjan a un valor estable.

El tamaño de cada nodo es proporcional a su PageRank.

Función de Layout Corresponde a una función que asigna posiciones a los nodos con el objetivo de ser visualizados.

Se utiliza una variación del algoritmo de Fruchterman-Reingold que toma en cuenta el PageRank de cada nodo.

En este algorimo, los nodos se repelen entre sí y las aristas actúan como fuerzas de atracción. Para incorporar el PageRank de los nodos, ajustamos la fuerza de repulsión para que sea proporcional al PageRank.

La fuerza de atracción f_a entre dos nodos i y j conectados por una arista se define como:

$$f_a(d) = \frac{d^2}{k}$$

Donde d es la distancia entre los nodos i y j, y k es una constante que controla la distancia ideal entre nodos conectados.

La fuerza de repulsión f_r entre dos nodos i y j se modifica para tomar en cuenta el PageRank y se define como:

$$f_r(d, PR(i), PR(j)) = \frac{k^2 \cdot PR(i) \cdot PR(j)}{d}$$

Donde $PR(i)$ y $PR(j)$ son los PageRanks de los nodos i y j respectivamente.

Este ajuste asegura que los nodos con mayor PageRank repelen a otros nodos con una mayor fuerza, lo que lleva a una disposición del grafo que favorece una mejor visualización.

Sigma.js Es una biblioteca de visualización de grafos para Javascript [1]. Lee grafos en un formato estandarizado; *Graphology* [25], y lo renderiza utilizando WebGL.

7.7. Dockerización

La "dockerización" corresponde a una jerga utilizada en el desarrollo de software para referirse al proceso por el cual alguna aplicación o proceso que se ejecuta de manera local es modificado o configurado para poder ejecutarse a través de un contenedor de docker.

Para este proyecto se decide optar por esta alternativa por la razón principal de facilitar el despliegue de este sistema software en distintas máquinas.

Dado el tipo de arquitectura del sistema, cada servicio corresponderá a un contenedor distinto, los cuales se logran comunicar entre si a través de una red virtual creada en docker.

A este tipo de conjunto se les conoce como "cluster", y se genera utilizando la herramienta "docker-compose" que permite configurar y construir clusters de manera descriptiva a través de un archivo yaml.

7.7.1. Cambios Realizados

Para cada servicio se agrego un archivo *Dockerfile*, el cual define las configuraciones a la hora de construir un contenedor.

Para cada Dockerfile se especifica la descarga de una imagen la cual es un paquete que contiene configuraciones, binarios, archivos y bibliotecas para ejecutar un contenedor.

Existen imágenes creadas por organizaciones o por la comunidad, las cuales tienen todo lo necesario para que algún tipo de aplicación específica pueda funcionar apenas el contenedor es creado, sin necesitar dependencias adicionales.

[1] https://www.sigmajs.org/

Ejemplificando con el caso de este proyecto, se utilizaron las imágenes de elasticsearch, node, debian y continuumio/miniconda3, las cuales están disponibles, con esos nombres, en la plataforma Docker Hub [1].

A nivel de comunicación entre los servicios, se crea una red virtual dentro del cluster de Docker la cual automáticamente configura que las URL de cada container, corresponde a su nombre (definido en el archivo docker-compose.yml) dentro del cluster. En la práctica, esto implica que a la hora de hacer solicitudes, ya no es al localhost (u otro enlace) donde previamente corrían estos procesos, si no que es a la nueva URL de cada servicio.

7.8. Testing

Se realiza el proceso de testing para garantizar la calidad y fiabilidad del sistema desarrollado. En este proyecto, se han utilizado diversas herramientas y enfoques para realizar pruebas exhaustivas de los diferentes componentes del sistema. A continuación, se detallan las herramientas y métodos utilizados para llevar a cabo el testing.

7.9. Pruebas Unitarias

7.9.1. Módulo unittest de Python

Para realizar pruebas unitarias en el contenedor *fastapi*, se utilizó el módulo unittest[5] de Python. Este módulo permite crear y ejecutar pruebas unitarias de manera estructurada y eficiente. Se desarrollaron pruebas unitarias para cada módulo implementado, asegurando que cada función y método se comportara según lo esperado.

Ejemplo de cómo se implementan pruebas unitarias utilizando unittest (código 3):

[1] https://hub.docker.com/
[5] docs.python.org/3/library/unittest.html

```
import unittest
from my_module import MyFunction

class TestMyFunction(unittest.TestCase):
    def test_case_1(self):
        self.assertEqual(MyFunction(input1), expected_output1)

    def test_case_2(self):
        self.assertTrue(MyFunction(input2))

if __name__ == '__main__':
    unittest.main()
```

Código 3: Ejemplo módulo unittest Python

7.9.2. ctest de CMake

Para realizar pruebas del servidor de C++, se utilizó la utilidad ctest de CMake. ctest permite ejecutar pruebas automatizadas y gestionar el proceso de testing de manera eficiente. Además, ctest permite usar scripts de shell que retornen 0 si la prueba es PASSED o 1 si es FAILED. Las pruebas se centraron en verificar que los diferentes módulos del sistema funcionen correctamente de manera independiente.

Se configuraron múltiples casos de prueba en el archivo CMakeLists.txt para ser ejecutados por ctest.

```
enable_testing()

add_executable(test_my_module test_my_module.cpp)
target_link_libraries(test_my_module my_module)

add_test(NAME TestMyModule COMMAND test_my_module)
```

Código 4: Ejemplo cómo agregar casos de prueba al archivo de CmakeLists.txt

7.10. Pruebas de Integración con Shell Scripts y curl

7.10.1. Endpoints de Servidor C++

Se probaron todos los endpoints del servidor en C++ y casos bordes.

La tabla 2 muestra estos casos bordes junto con el código de estado esperado para que el test sea correcto, y el significado de cada código.

Tabla 2
Tipos de casos de test con códigos de error esperados y sus descripciones

Tipo de Caso de Test	*Código*	*Descripción*
Request a endpoint inválido	404	El endpoint solicitado no existe en el servidor.
Header del Request inválido	400	El header del request no cumple con los requisitos esperados por el servidor.
Request body inválido	400	El cuerpo del request no es un JSON válido o está mal formado.
Request con JSON vacío	400	El cuerpo del request contiene un JSON vacío, lo cual no es aceptable.
JSON válido pero no parseable para generar grafo	422	El JSON proporcionado es válido pero no puede ser procesado para generar un grafo.
Generar grafo	201	El grafo se genera correctamente y se devuelve una respuesta de éxito.
Generar subgrafo	201	El subgrafo se genera correctamente y se devuelve el objeto generado.
Eliminar grafo	201	El grafo se elimina correctamente y se devuelve una respuesta de éxito.

7.10.2. Endpoints de FastAPI

Para probar los endpoints de FastAPI, se utilizaron scripts de shell haciendo solicitudes con curl para verificar su correcto funcionamiento.

A diferencia del servidor de C++, FastAPI facilita el manejo de errores y estados ya que realiza todas las verificaciones a las peticiones que recibe en base a las especificaciones del endpoint. Por lo tanto, errores como, solicitudes incorrectas (endpoint inexistente, header incorrecto) o cuerpo de la solicitud inválido (vacío, tipo de dato incorrecto, objeto no parseable) son automáticamente manejadas a través de mensajes de error en la respuesta enviada al cliente.

A continuación, se presenta un ejemplo de cómo se implementan estas pruebas:

```
#!/bin/bash

# Asegurarse que el proceso del servidor comienza y termina
../build/KnowledgeGraph &
SERVER_PID=$!
trap "kill $SERVER_PID" EXIT

test_file=../../searcher/tests/files/author-title.pdf

# Si la respuesta del servidor no es 200, se falla el test
check_http_status() {
  [ $1 -ne 200 ] && echo "Error Expected status 200, got $1" && exit 1
}

echo "Peticion 1: subir archivo"
status=$(curl -s -o /dev/null -w "%{http_code}" -X POST -F "file=@$test_file" http://localhost
:8000/load-file)
check_http_status $status

echo "Peticion 2: obtener grafo"
status=$(curl -s -o graph1.json -w "%{http_code}" -X POST -H "Content-Type: application/json" -d
```

```
    '{"index": "title", "values": [1]}' http://localhost:8000/get-graph)
    check_http_status $status

    status=$(curl -s -o /dev/null -w "%{http_code}" -X POST "http://localhost:8000/delete-index" -H "
    Content-Type: application/json" -d '{"request": "title"}')
    check_status $status

    # Si todas las pruebas pasan, retornar 0
exit 0
```

Código 5: Ejemplo caso de prueba para FastAPI

7.10.3. Pruebas del Front-End

Las pruebas del front-end se centraron en verificar que las funciones fetch, las cuales realizan las solicitudes correspondientes al API gateway, fueran manejadas correctamente. Es decir, que el backend retorne el tipo de respuesta esperado.

Se replicó 1 a 1 cada petición de curl hecha en las pruebas de FastAPI, en React a través de la función asíncrona fetch.

Dado que lo anterior correspondía al proceso más sencillo de realizar, no fue necesario usar herramientas de automatización. Manualmente se utilizaron las herramientas de desarrollo del navegador para monitorear las solicitudes y verificar que los datos se manejaran correctamente en la interfaz de usuario.

8. Demostración del Sistema

Esta sección presenta una demostración del funcionamiento y los resultados que se pueden obtener con el sistema a través de un ejemplo de uso con datos reales. Además, se evalúa y miden distintos aspectos de la ejecución y respuestas generadas.

8.1. Datos

Los datos a utilizar para esta prueba corresponden a 3 documentos:

Tabla 3
Dataset a utilizar en la prueba del sistema

Documento	*Autor*	*Año de pub.*	*Tamaño*	*N. de páginas*
Informe Rettig [7]	"Comisión Rettig"	1991	7,8 MB	460
Informe Valech [8]	"Comisión Valech"	2003	17,5 MB	638
Memoria Viva. Extractos [16]	Varios	Varios	0,371 MB	91

Dado el contexto del proyecto, se considera que esta muestra es representativa, y puede generalizar bien al tipo de textos que eventualmente los usuarios agregarán.

Luego, analizamos las palabras y entidades que contienen estos documentos en conjunto. Ambas se obtienen una vez los documentos son cargados al sistema. Simplemente se calcula y se guarda una copia de la frecuencia de las palabras mientras se realiza el "chunking", y de la frecuencia de entidades mientras se generan los triples.

A continuación se muestran estadísticas descriptivas de las palabras y entidades.

Tabla 4
Estadísticas de palabras y entidades en los documentos

	suma	*media*	*mediana*	*std dev*	*min*	*max*
Palabras	25.436	19,256	2,0	359,328	1	41.417
Entidades	3.959	1,627	1,0	4,407	1	156

Las estadísticas reflejan lo esperado. Un uso de palabras disperso, con una mediana baja y una alta desviación estándar, se corresponde a lo que se puede concluir a través de la "Ley de Zipf", ley empírica que cumplen los lenguajes, que, en palabras simples, para cualquier lenguaje, existen muchas palabras con muy poco uso, y pocas palabras con mucho uso.

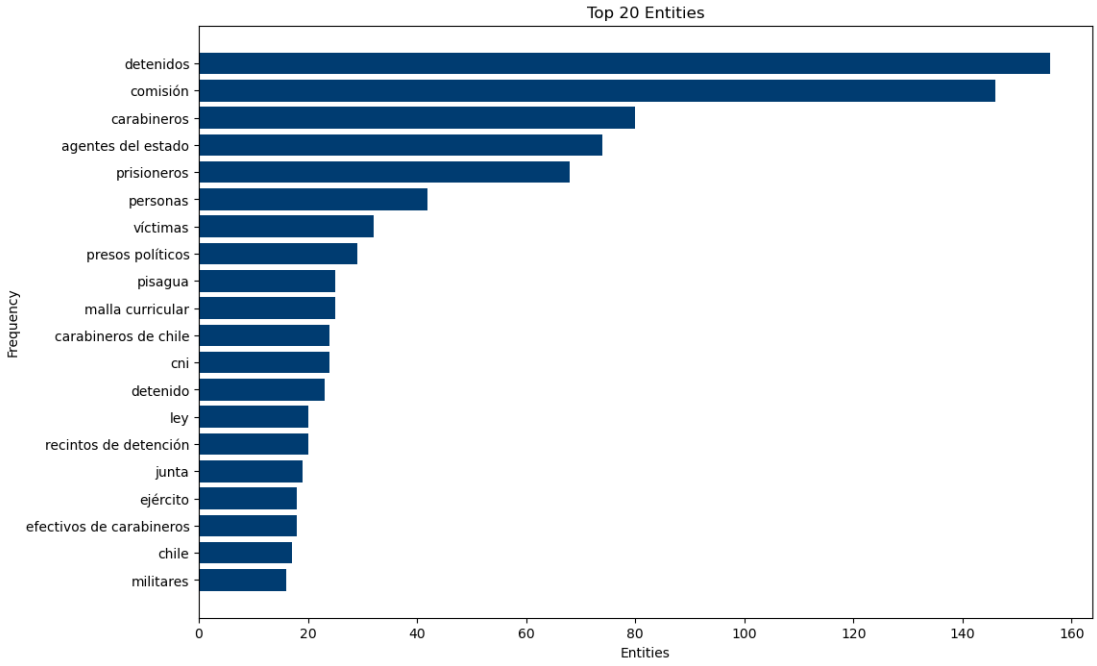
Para las entidades, si bien no hay un caso esperado como tal, ocurre un fenómeno similar al de las palabras, pero a menor escala. Aquí, la mitad de entidades aparecen una única vez, sin embargo, el hecho que haya un máximo tan alto en comparación puede sugerir que hay unas pocas entidades que se repiten muchas veces. Se analizará esto a continuación.

Eliminando primero las denominadas "stopwords"[1], las cuales corresponden a palabras tan usadas, que entregan muy poca información, por ejemplo, "la", "a", "un", "y", entre otras, se muestran las palabras y entidades con mayor aparición en los datos.

Figura 14
Palabras más usadas en los documentos

[1]Obtenidas en el archivo "spanish" de https://github.com/nltk/nltk_data/blob/gh-pages/packages/corpora/stopwords.zip

Figura 15
Entidades con más apariciones en los documentos



Los gráficos anteriores nos entregan información con la que se puede inferir las temáticas presentes en los datos.

Finalmente, a modo de interés acerca del comportamiento del modelo de embedding utilizado, que es el modelo de de OpenAI "text-embedding-3-small", se aplica la técnica de análisis de componentes principales (PCA) para poder visualizar las palabras y las entidades las cuales están representadas en un espacio de 200 dimensiones, en 2 dimensiones. Esto ocurre debido a que esta técnica de reducción de dimensionalidad busca las dimensiones con mayor varianza, es decir, donde la dirección de las palabras varían más, capturando así las diferencias significativas del espacio original.

Es interesante conocer estos resultados debido a que no hay claridad acerca de los datos con los que estos modelos fueron entrenados, y la única garantía existente respecto a su uso multilingüe, es un pequeño artículo en la página de ayuda de OpenAI donde solo se confirma la capacidad de la API de funcionar en distintos idiomas.

Figura 16

PCA de palabras más usadas en los documentos

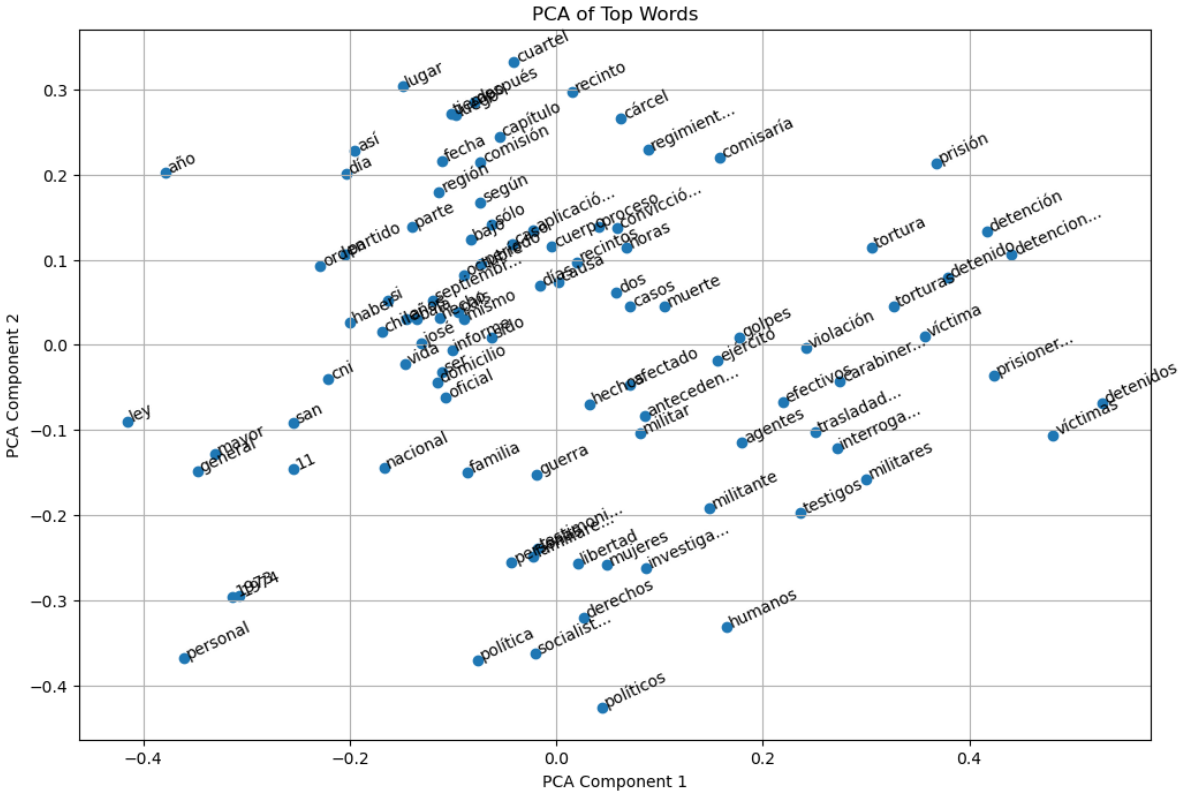
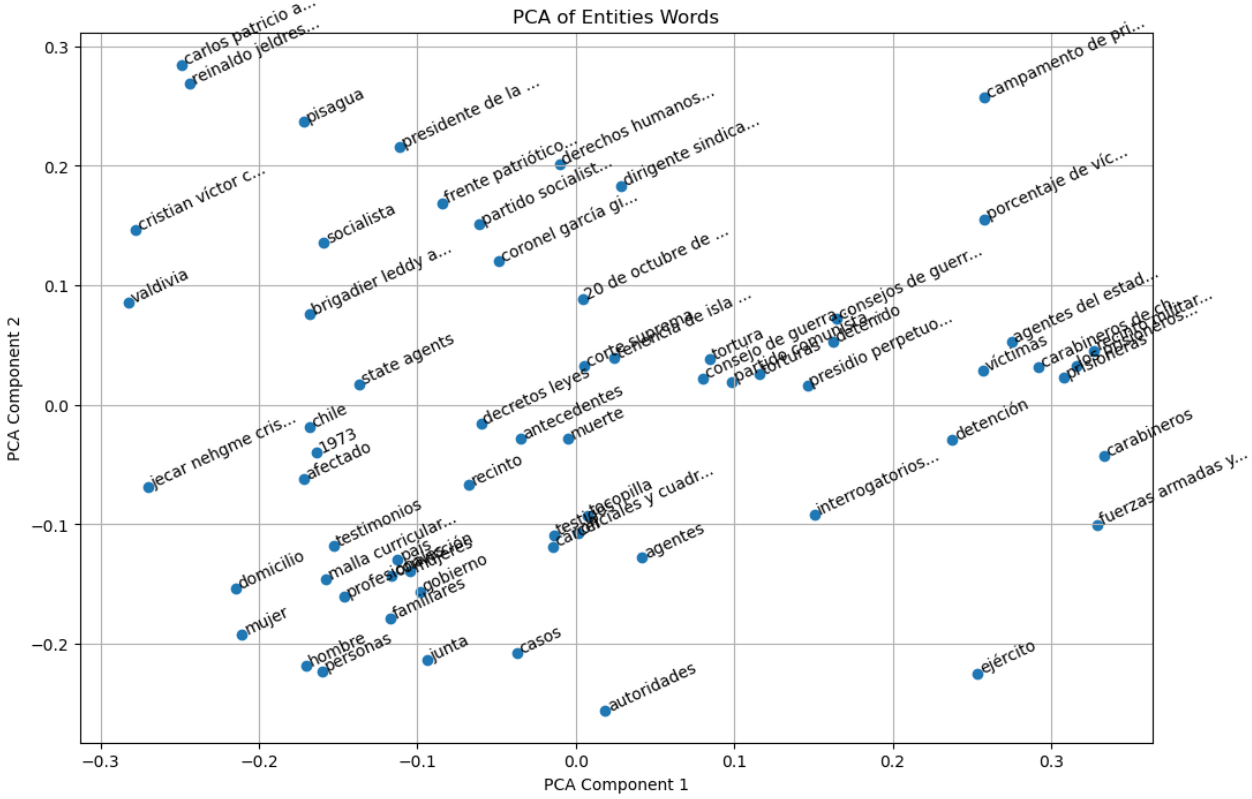


Figura 17
PCA de entidades con más apariciones en los documentos



En base al análisis de componentes, se puede visualizar que conceptos con significados similares están más cerca. Así concluimos que el modelo de embedding de OpenAI "text-embedding-3-small" con representaciones vectoriales de 200 dimensiones, cumple el objetivo de relacionar semánticamente conceptos en español. Sin embargo, también se evidencia que conceptos que dentro del contexto que se está trabajo sí deberían estar relacionados, no lo están. Esto es esperable dada la generalidad del uso de este embedding, lo que plantea como trabajo futuro el uso de uno propio, o de alguna alternativa que permita "fine-tuning".

8.2. Ejecución

La ejecución del sistema se realizó dentro de un cluster en docker, en una máquina local. Específicamente un Macbook Air con chip M1, 8G de RAM y 500G de almacenamiento SSD. Mientras los limites globales[1] de los recursos para docker corresponden a 4 cores de CPU, 4 GB de RAM y 1 GB de memoria swap. El único contenedor con un límite distinto al global es el *es01*, el cual tiene 1.093GB de RAM, el cual es el límite por defecto que dejaron los desarrolladores de Elasticsearch.

Respecto a las variables y parámetros existentes en el sistema (no confundir con los parámetros o pesos del LLM), se seleccionan en base a la teoría, al uso empírico de estos, y a los valores recomendados por los desarrolladores.

Al ser tantos para cada servicio, y en un sistema complejo y heterogéneo como este, no es factible aplicar métodos o técnicas de validación cruzada, o de análisis de sensibilidad.

Mencionando una alternativa reciente para ayudar a estos casos, se ha ido creando un paradigma de "desarrollo dirigido por métricas" (metric-driven development), en el cual, durante periodos extensos de tiempo, se monitorean y evalúan aplicaciones y se extraen las variables y el input utilizado para realizar pequeños ajustes que generen un cambio positivo en las métricas. Sin embargo, por razones evidentes, como el no contar con un periodo extenso de prueba, ni herramientas automatizadas de monitoreo, ni un alto volumen de usuarios, no se podría aplicar en este proyecto.

En la tabla 6 se muestran todas las variables, parámetros, hiperparámetros y configuraciones seleccionadas para la ejecución del sistema, junto a la justificación de su uso.

[1] Cada contenedor puede tener además su propio limite menor o igual al global.

Tabla 5
Uso de recursos de contenedores en procesos de Ingesta de Datos y Generación de Respuestas

	CPU %				*MEM USAGE (MB)*			
	media	*std.dev.*	*min*	*max*	*media*	*std.dev.*	*min*	*max*
Ingesta de Datos (tiempo promedio 1108,156 s)								
cppgraph	4,68	21,07	0,0	101,09	107,21	2,25	105,1	109,6
es01	3,56	4,69	0,46	51,24	1.066,59	1,23	1.060,86	1.069,06
fastapi	6,27	5,93	1,61	99,81	167,64	2,23	147,0	171,2
reactapp	0,03	0,64	0,0	12,2	7,32	0,81	6,94	11,45
Generación de Respuesta (tiempo promedio 94,044 s)								
cppgraph	80,4	41,25	0,0	101,09	104,95	0,37	104,1	105,1
es01	5,3	15,82	0,25	71,87	1.060,86	0.0	1.060,86	1.060,86
fastapi	2,86	2,32	1,13	9,3	144,85	0,1	144,8	145,1
reactapp	0,03	0,11	0,0	0,41	6,96	0,01	6,93	6,97

Antes de realizar consultas al sistema, primero se deben subir los documentos. Recordar que esto incluye lo mencionado en la ingesta de datos 7.3.3.

La tabla 5 muestra el resumen del uso de recursos de los contenedores, calculado extrayendo el estado actual de recursos cada 3 segundos durante el total del proceso, de duración de $1.108,156157$ segundos (scripts disponibles en Anexo C).

En los resultados mostrados en la tabla 5, se puede notar que no hay mayor variación en cuanto al uso de memoria. Sin embargo, se puede ver la diferencia en el uso del vCPU de cada contenedor, entre el estado inactivo o "idle" del sistema, versus cuando se empiezan a procesar los datos y se crea el KG.

8.3. Resultados

Una vez los datos ya se subieron a la plataforma, se realizan algunas consultas las cuales se describen y se muestran sus resultados a continuación:

Tabla 6
Variables utilizadas para la ejecución del sistema.

Variable	*Valor*	*Razón*
Elasticsearch Búsqueda Semántica		
Dimensión de Vectores	200	Igualar dimensiones para el modelo de embedding
Algoritmo	Approximate kNN	Eficiencia respecto al kNN tradicional.
k	5	Respuesta en tiempo razonable.
Función de Distancia	Similitud Coseno	Opción por defecto.
LLM		
Modelo	gpt-4o	Mejor modelo a la fecha.
Temperature	1,0	Valor por defecto.
Top-P	1,0	Valor por defecto.
Max Tokens	4.095	Máximo del modelo.
System Prompts	Anexo D	Entrega resultados esperados.
User Prompts	Anexo D	Entrega resultados esperados.
Ingesta de Datos		
Chunk Size	20,000 caracteres	Empírica
Chunk Overlap	1,000 caracteres	Empírica
Embedding Model	text-embedding-3-small	Costo-eficiencia.
Dimensión Embedding	200	Disminuir uso de memoria.

8.3.1. Prompts

1. *¿Qué es un centro de detención clandestino?*
2. *¿Qué hechos relacionados a la violaciones de los derechos humanos ocurrieron en Chile en el año 1980?*
3. *¿Qué rol cumplió la CNI y la DINA en la violación de los derechos humanos en Chile?*

8.3.2. Rendimiento

Se puede observar en la tabla 5 que la mayor carga en el proceso de generación de respuestas se lo lleva el servicio encargado de generar el subgrafo, y Elasticsearch, debido a la búsqueda semántica que realiza.

A continuación se muestran las respuestas obtenidas.

Notar que en las imágenes de los grafos pueden haber algunas deformidades en líneas o texto. Esto es debido a que para mejorar la visualización dentro del documento, se utilizó la herramienta "Upscayl"[1] que permite reescalar imágenes a través de modelos de IA.

8.3.3. Respuesta a Prompt 1

Respuesta Textual:

> "Un centro de detención clandestino es un recinto utilizado por organismos de seguridad del Estado para mantener detenidas a personas sin el debido proceso legal, frecuentemente involucrando actos de tortura y abuso físico y psicológico. Durante el régimen militar en Chile, estos lugares operaron de manera secreta, con condiciones inhumanas que incluían incomunicación, malos tratos, golpes, y diversas formas de tortura, como el uso de electricidad y violencia sexual. Los detenidos, en su mayoría prisioneros políticos, eran trasladados entre diferentes centros de tortura y secretos sin ser reconocidos oficialmente,

[1] upscayl.org/

creando un ambiente de terror y desamparo. Estos lugares mantuvieron a las personas en condiciones deplorables, privándolas de alimentos, agua y atención médica, y muchas veces, sus encerramientos eran ilegales y sin justificación.

Fuentes:

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 524 - 533) Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 233 - 241) Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 292 - 303) Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 321 - 330) Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 225 - 234)"

Respuesta KG

Figura 18
KG generado para prompt 1

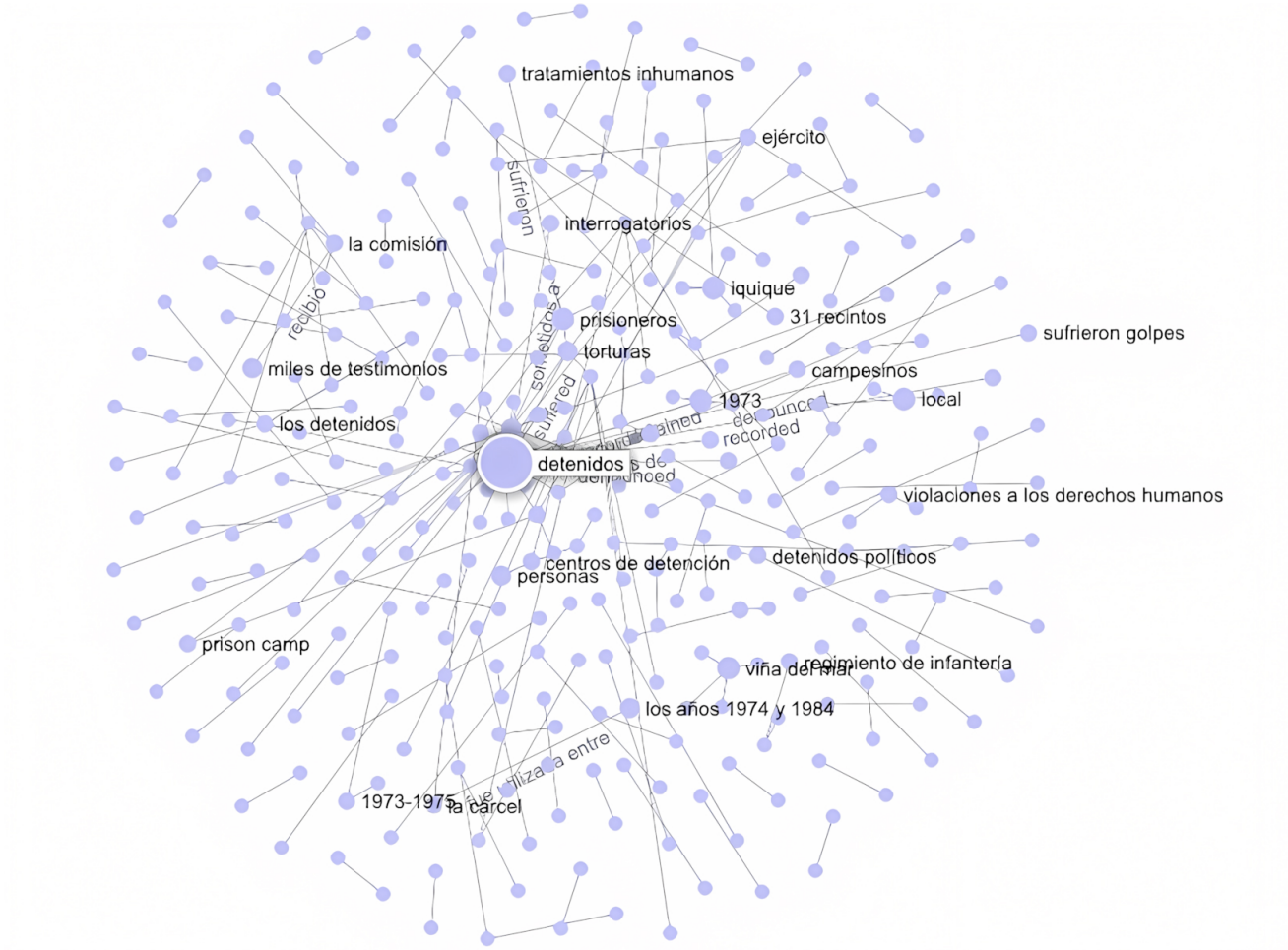
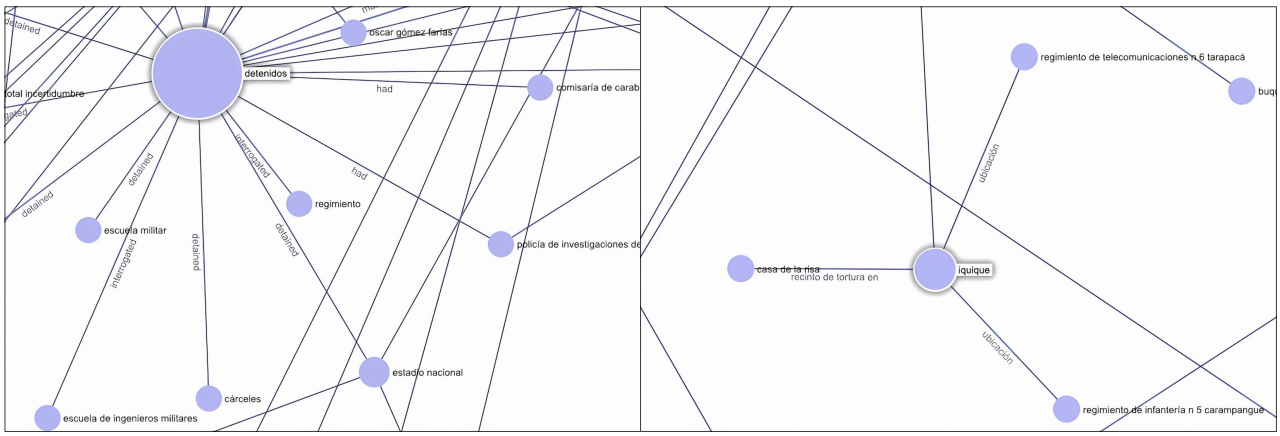


Figura 19
Zoom al grafo generado



En la figura 19 se destacan dos sectores dentro del KG. En la izquierda se observan relaciones con la entidad con mayor *PageRank*; "detenidos", junto con algunos lugares donde se mantuvieron (escuela militar, cárceles, estadio nacional), e instituciones que realizaron detenciones. En la imagen derecha se tiene un tipo de componente el cual no es único dentro de este grafo, donde se relaciona una comuna o ciudad, y sus lugares de detención más conocidos. En el caso de la imagen se tiene a la ciudad de Iquique, junto con tres centros clandestinos de detención y tortura.

8.3.4. Respuesta a Prompt 2

Respuesta Textual

> "En el año 1980, Chile se encontraba bajo el régimen militar encabezado por Augusto Pinochet, y se registraron varias violaciones de derechos humanos. Aunque el contexto específico de tu consulta se centra en ese año, es importante destacar que las violaciones de derechos humanos empezaron con el golpe de estado en 1973 y continuaron a lo largo de la dictadura.
>
> 1. **Constitución de 1980**: En septiembre de 1980, el régimen militar promulgó una nueva Constitución, que fue elaborada sin participación popular y bajo un clima de represión. Esta constitución contenía disposiciones que limitaban severamente los derechos políticos y civiles, incluyendo prohibiciones de actividad política por parte de grupos opositores.
>
> 2. **Detenciones Arbitrarias y Torturas**: A lo largo de 1980, las fuerzas de seguridad continuaron con la práctica de detenciones arbitrarias, secuestros y torturas. Diversas organizaciones de derechos humanos, tanto nacionales como internacionales, denunciaron casos de prisioneros políticos que eran torturados en centros de detención. La Comisión Interamericana de Derechos Humanos emitió informes y resoluciones que denunciaban esta situación.
>
> 3. **Censura y Control de Medios**: Los medios de comunicación estaban sometidos a

un rígido control y censura, lo que dificultaba la denuncia pública de las violaciones a los derechos humanos. La falta de información veraz impedía que la sociedad conociera plenamente la magnitud de las violaciones que estaban ocurriendo.

4. **Reacciones Internacionales**: Informes de organizaciones internacionales, incluyendo la ONU, denunciaron la situación de derechos humanos en Chile, incluyendo la práctica sistemática de tortura y detenciones arbitrarias. En 1980, se realizaron varias resoluciones de condena que reflejaron la preocupación internacional por la situación en el país.

En resumen, el año 1980 en Chile estuvo marcado por la continuación de una política de represión sistemática, incluyendo violaciones graves de derechos humanos, censura a los medios, y la promulgación de una constitución que consolidó el régimen autoritario. Esto generó importantes reacciones tanto a nivel interno, de organizaciones como el Comité Pro Paz, como a nivel internacional, donde se denunciaron estas violaciones ante diferentes foros y organismos.

Fuentes:

Comisión Nacional de Verdad y Justicia. Informe de la Comisión Nacional de Verdad y Justicia Tomo I 1991 (p. 1096 - 1098)

Comisión Nacional de Verdad y Justicia. Informe de la Comisión Nacional de Verdad y Justicia Tomo I 1991 (p. 1090 - 1096)

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 208 - 215)

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 215 - 225)

Comisión Nacional de Verdad y Justicia. Informe de la Comisión Nacional de Verdad y Justicia Tomo I 1991 (p. 668 - 673)"

Respuesta KG

Figura 20
KG generado para prompt 2

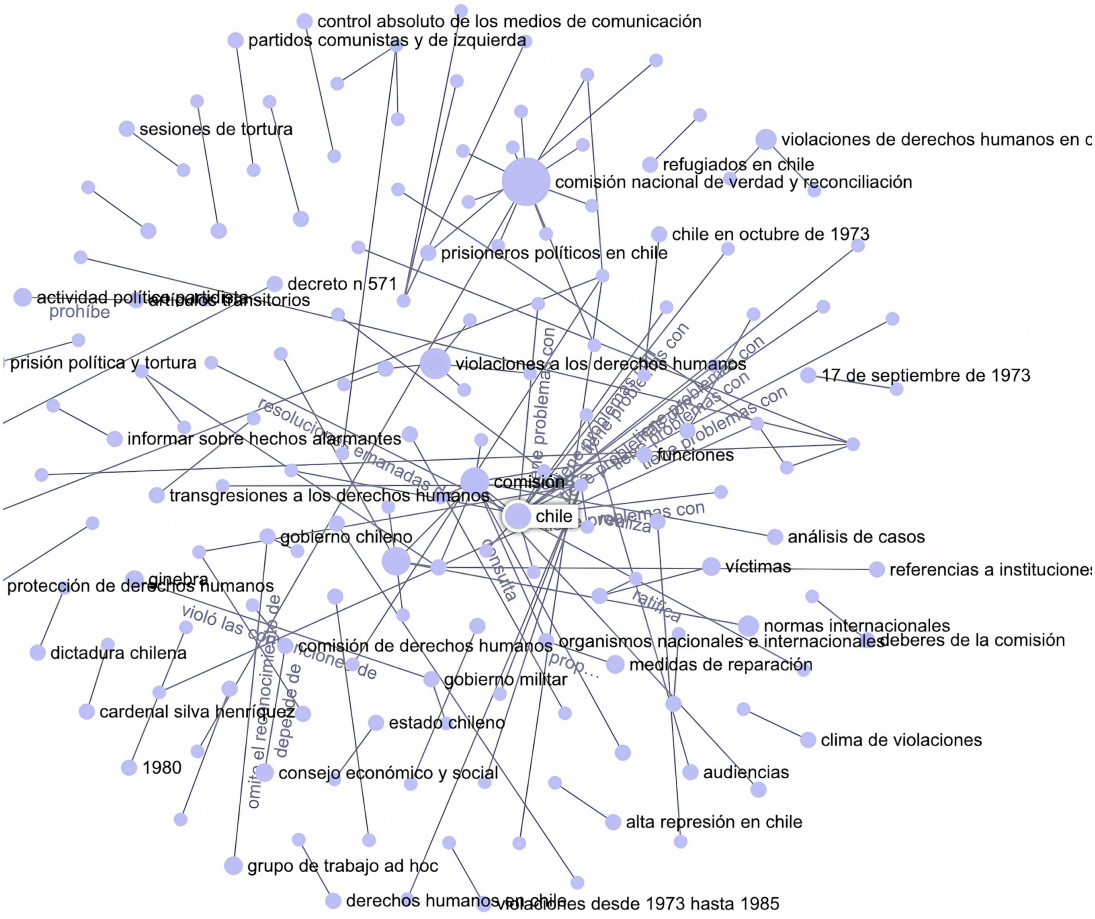
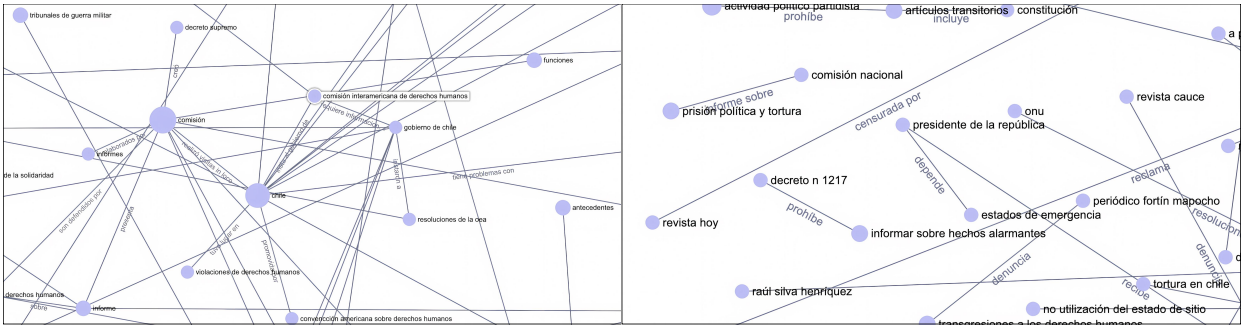


Figura 21
Zoom al grafo generado



Se puede observar que el LLM se escapa de la consulta inicial argumentando que el periodo importante es mayor que solo el año 1980. Sin embargo, entrega información interesante, ya que se puede notar en ambas respuestas que existía una preocupación global sobre lo que estaba ocurriendo en Chile a nivel de D.D.H.H y censura. Se puede ver en la imagen de la derecha en la figura 21 que se nombra al decreto 1217, el cual ademas de clausurar varios medios de comunicación, obligaba al resto de medios a tener la autorización directa de la Secretaría General de Gobierno para entregar información acerca del país, perpetuando así la censura periodística.

8.3.5. Respuesta a Prompt 3

Respuesta Textual

> "La CNI (Central Nacional de Informaciones) y la DINA (Dirección de Inteligencia Nacional) jugaron roles fundamentales en la violación de los derechos humanos en Chile durante la dictadura de Augusto Pinochet. Ambos organismos fueron responsables de llevar a cabo una represión sistemática y brutal contra aquellos considerados enemigos del régimen, que incluían no solo a militantes de partidos de izquierda, sino también a civiles sin vinculación política directa, lo que evidenció un amplio margen de arbitrariedad en la selección de sus víctimas.
>
> La DINA, creada en junio de 1974, se especializó en la represión política y fue el primer organismo de inteligencia del gobierno militar. Utilizó recursos significativos para llevar a cabo su labor, que incluía prácticas de tortura, detención arbitraria, desaparición forzada y asesinatos de prisioneros. La tortura era una práctica sistemática, utilizada tanto para obtener información como para imponer el terror, y se llevó a cabo en recintos clandestinos. La DINA no solo actuó dentro de Chile sino que también realizó operaciones internacionales, como el asesinato del ex canciller Orlando Letelier en Washington D.C.
>
> La CNI, que reemplazó a la DINA en 1977, continuó con las mismas prácticas represivas

y mantuvo, en gran medida, los métodos de la DINA, aunque con un enfoque más selectivo. Esto se tradujo en una intensificación del control sobre los opositores políticos y la neutralización de cualquier disidencia organizada. Ambos organismos operaron con una falta de rendición de cuentas y autonomía, contando con el respaldo del gobierno militar, lo que les permitió llevar a cabo abusos sistemáticos sin temor a represalias.

El legado de la CNI y la DINA se caracteriza por una política de estado que incluyó la tortura y la eliminación de opositores, y su accionar condujo a un ambiente de miedo y desesperación en la población chilena, violando gravemente los derechos humanos fundamentales.

Fuentes:

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 190 - 196)

Comisión Nacional de Verdad y Justicia. Informe de la Comisión Nacional de Verdad y Justicia Tomo I 1991 (p. 700 - 705)

Comisión Nacional de Verdad y Justicia. Informe de la Comisión Nacional de Verdad y Justicia Tomo I 1991 (p. 1096 - 1098)

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 183 - 190)

Comisión Nacional sobre Prisión Política y Tortura. Informe de la Comisión Nacional sobre Prisión Política y Tortura Valech I 2005 (p. 292 - 303)."

Respuesta KG

Figura 22
KG generado para prompt 3

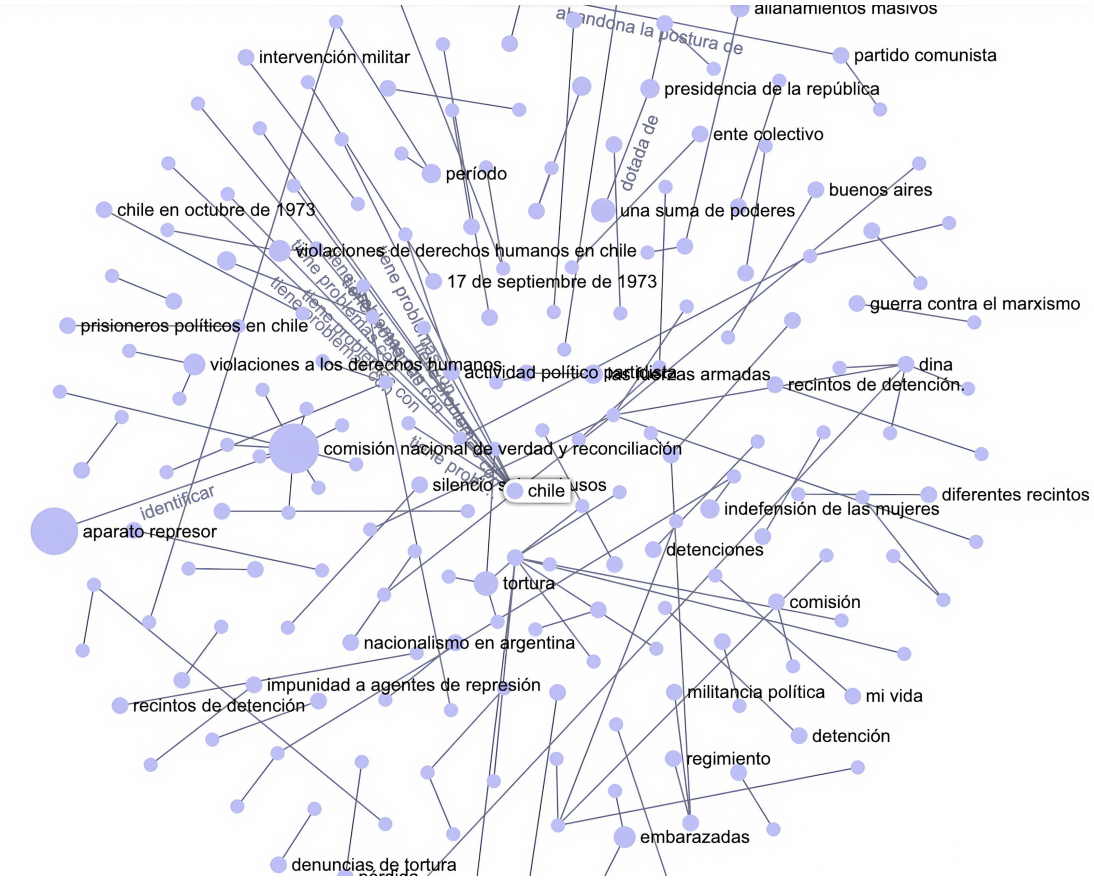
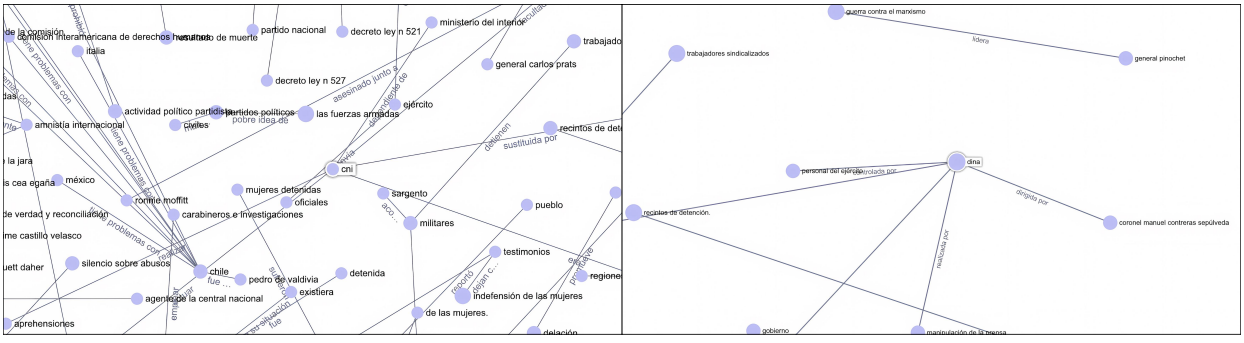


Figura 23
Zoom al grafo generado



Se puede observar en el KG que aparecen muchos términos relacionados a detenciones informales, torturas, recintos de detención, y violaciones a los D.D.H.H. Esto al haber consultado acerca del

CNI y la DINA, lo que indica la estrecha relación de estos conceptos con estos organismos.

8.4. Evaluación

8.4.1. Métricas

Se evalúan los resultados del sistema en cuanto a su recuperación de información [1], generación de respuesta textual, y generación de respuesta de triples.

Lo anterior se realiza utilizando el framework de evaluación de RAGs, "RAGAS" [10], proponiendo además ajustes para poder evaluar la generación de los grafos. Pero antes de definir las métricas seleccionadas, es crucial abordar el tema del tipo de variables involucradas en sus cálculos.

En problemas clásicos, como la clasificación, una vez que se obtiene una respuesta del algoritmo o modelo clasificador, las variables extraídas para calcular las métricas son constantes y bien definidas. Por ejemplo, en la clasificación binaria, las métricas como la precisión, la exactitud y el F1-score se calculan a partir de variables como verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. En este contexto, en cambio, las variables son conceptos más abstractos que dependen de normas gramaticales y lingüísticas, incluso siendo un problema de NLP (a baja escala) por si mismas, como la cantidad de afirmaciones, o la cantidad de entidades en una respuesta.

Además, algunas de estas variables deben ser generadas por un LLM, por lo que existe un componente de estocasticidad que añade una capa adicional de complejidad. Recordar que los LLMs, al ser modelos probabilísticos, pueden producir diferentes resultados para la misma entrada en distintas ejecuciones.

Entonces, volviendo a la selección de las métricas a utilizar, se consideran aquellas que se ajustan de mejor manera al contexto de este trabajo. A continuación se describen las métricas seleccionadas:[2] [3]

Fidelidad (Faithfulness)

[1] Los contextos recuperados se pueden acceder en en el Anexo D
[2] No existe una traducción oficial, o que sea altamente adoptada en el español de los nombres de las métricas
[3] Los nombres traducidos son de elaboración propia, intentando preservar de la mejor manera los significados.

Esta métrica mide la consistencia de los hechos de la respuesta generada en comparación con el contexto dado. Se calcula a partir de la respuesta y el contexto recuperado. La respuesta se escala en el rango de (0,1), siendo mientras mayor, mejor. La respuesta generada se considera fiel si todas las afirmaciones hechas en la respuesta pueden inferirse del contexto dado. Para calcular esto, primero se identifica un conjunto de afirmaciones de la respuesta generada. Luego, cada una de estas afirmaciones se verifica con el contexto dado para determinar si puede inferirse o no.
La puntuación de fidelidad se da por:

$$\text{Faithfulness} = \frac{|\text{Afirmaciones en la respuesta generada que pueden inferirse del contexto}|}{|\text{Total de afirmaciones en la respuesta generada}|}$$

Relevancia de la Respuesta (Answer Relevance)

La métrica de relevancia de la respuesta se centra en evaluar cuán pertinente es la respuesta generada con respecto al prompt dado. Se asigna una puntuación más baja a las respuestas que son incompletas o contienen información redundante, y puntuaciones más altas indican mejor relevancia. Esta métrica se calcula utilizando la pregunta, el contexto y la respuesta. La relevancia de la respuesta se define como la media de la similitud coseno de la pregunta original con una serie de preguntas artificiales, que se generan (inversamente) basándose en la respuesta:

$$\text{answer relevancy} = \frac{1}{N}\sum_{i=1}^{N}\cos(E_{g_i}, E_o)$$

$$\text{answer relevancy} = \frac{1}{N}\sum_{i=1}^{N}\frac{E_{g_i}\cdot E_o}{\|E_{g_i}\|\|E_o\|}$$

Donde:

- E_{g_i} es la incrustación de la pregunta generada.
- E_o es la incrustación de la pregunta original.
- N es el número de preguntas generadas, que por defecto es 3.

Aunque en la práctica la puntuación oscilará entre 0 y 1 la mayor parte del tiempo, esto no está garantizado matemáticamente debido a la naturaleza de la similitud coseno que varía de -1 a 1. Esta métrica no considera la factualidad, sino que penaliza los casos donde la respuesta carece de completitud o contiene detalles redundantes.

Precisión del Contexto (Context Precision)

Es una métrica que evalúa si todos los elementos relevantes del GT presentes en los contextos tienen un rango mayor o no. Idealmente todos los chunks relevantes deben aparecer en los primeros lugares del ranking. Esta métrica se calcula utilizando las consultas, un GT, y los contextos, con valores en el rango $[0,1]$, donde un mayor valor indica mejor precisión.

$$\text{Context Precision}_k = \frac{\sum_{k=1}^{K} (\text{Precision}_k \times v_k)}{\text{Total de elementos relevantes en los top } K \text{ resultados}}$$

$$\text{Precision}_k = \frac{\text{verdaderos positivos en } k}{(\text{verdaderos positivos en } k + \text{falsos positivos en } k)}$$

Donde K es el número total de chunks en el contexto, y $v_k \in \{0,1\}$ es el indicador de relevancia en el rango k.

Exhaustividad del Contexto (Context Recall)

En esta métrica se mide el grado en que el contexto recuperado se alinea con la respuesta generada, tratada como Ground Truth (GT). Se calcula en función del GT y el contexto recuperado, y los valores varían entre 0 y 1, con valores más altos indicando mejor rendimiento. Para estimar la exhaustividad del contexto a partir de la respuesta del GT, se analiza cada oración en la respuesta del GT para determinar si puede atribuirse al contexto recuperado o no. En un escenario ideal, todas las oraciones en la respuesta del GT deberían ser atribuibles al contexto recuperado.

$$\text{context recall} = \frac{|\text{Oraciones del GT que pueden atribuirse al contexto}|}{|\text{Número de oraciones en el GT}|}$$

Recuperación de Entidades del Contexto (Context Entity Recall)

Esta métrica proporciona una medida de la exhaustividad o recall del contexto recuperado, basada en el número de entidades presentes tanto en los GTs como en los contextos, en relación con el número de entidades presentes solo en los GTs. En términos simples, es una medida de qué fracción de entidades se recuerdan de los GTs. Esta métrica es útil en casos de uso basados en hechos como ayuda en turismo, QA histórico, etc. Esta métrica puede ayudar a evaluar el mecanismo de recuperación de entidades, basado en la comparación con las entidades presentes en los GTs, porque en casos donde las entidades son importantes, necesitamos los contextos que las cubran.

Para calcular esta métrica, utilizamos dos conjuntos, GE y CE, como el conjunto de entidades presentes en los GTs y el conjunto de entidades presentes en los contextos, respectivamente. Luego tomamos el número de elementos en la intersección de estos conjuntos y lo dividimos por el número de elementos presentes en GE, dado por la fórmula:

$$\text{context entity recall} = \frac{|CE \cap GE|}{|GE|}$$

Puntuación de Resumido (Summarization Score)

Esta métrica mide qué tan bien un resumen captura la información importante de los contextos. La intuición detrás de esta métrica es que un buen resumen debe contener toda la información importante presente en el contexto. Primero, extraemos un conjunto de frases clave importantes del contexto. Estas frases clave se utilizan luego para generar un conjunto de preguntas. Las respuestas a estas preguntas son siempre sí (1) para el contexto. Luego, hacemos estas preguntas al resumen y calculamos la puntuación de resumen como la proporción de preguntas respondidas correctamente respecto al total de preguntas.

Calculamos la puntuación de pregunta-respuesta (QA) utilizando las respuestas, que es una lista de 1s y 0s. La puntuación de pregunta-respuesta se calcula entonces como la proporción de preguntas respondidas correctamente (respuesta = 1) respecto al total de preguntas.

$$\text{QA score} = \frac{|\text{preguntas respondidas correctamente}|}{|\text{total de preguntas}|}$$

8.4.2. Resultados

Hay que destacar que es RAGAS quien automatiza la extracción de las variables necesarias para calcular las métricas, pero para funcionar, dependiendo de la métrica, se necesita armar un dataset que cumpla ciertas especificaciones (ver tabla 7).

Tabla 7
Conjuntos necesarios por métrica para poder evaluar

Métricas	*Conjuntos Necesarios*
Faithfulness y Answer Relevance	Questions, Contexts, Answer.
Context Precision, Context Recall, Context Entity Recall	Questions, Contexts, Ground-Truth.
Summarization Score	Contexts, Summary.

Como se mencionó al comienzo de esta sección, se desea evaluar el sistema considerando la respuesta textual, y la respuesta de triples. Es por esto que se crean dos datasets distintos, donde se describe a que corresponde cada uno de los conjuntos mencionados en la tabla 7 en cada dataset (ver tabla 8).

Se puede notar que en la tabla 8, el valor de los GT corresponde, al igual que el conjunto Answer, a las respuestas generadas por el mismo sistema. Esto se debe a dos razones. Primero, está el hecho que no existe un "Gold Standard" previo con el que poder comparar, y crear GTs a mano cada vez que se quiera realizar una evaluación, tiene poca factibilidad. Además, se tienen problemas como el hecho que realmente no existe una respuesta correcta, y que las mismas se pueden complementar arbitrariamente con información extra, la cual independiente de la intención de la consulta original, puede o no ser relevante para el usuario. En segundo lugar, se tiene que todas las respuestas generadas son verificables en base a las fuentes y los fragmentos específicos que referencia, por lo que finalmente se puede argumentar que correspondería a un GT generado, el cual su veracidad está asegurada.

Tabla 8
Datasets para evaluar respuestas

Conjunto	*Valor*
Respuestas textuales	
Questions	Consultas/Prompt del usuario.
Contexts	Contextos recuperados.
Ground-Truth	Respuesta textual generada.
Answer	Respuesta textual generada.
Summary	Respuesta textual generada.
Respuestas de triples	
Questions	Consultas/Prompt del usuario.
Contexts	Contextos recuperados.
Ground-Truth	Triples generados.
Answer	Triples generados.
Summary	Triples generados.

Tabla 9
Evaluación de Resultados

Prompt	*Context Precision*	*Context Recall*	*Context Entity Recall*	*Faithfulness*	*Answer Relevance*	*Summarization Score*
Respuestas textuales						
1	1,0	1,0	0,192308	1,0	0,936479	0,710786
2	1,0	1,0	0,066667	1,0	0,905807	0,497064
3	1,0	1,0	0,142857	1,0	0,941944	0,971908
Respuestas de triples						
1	1,0	0,408805	0,049180	0,891720	0,819966	0,950893
2	1,0	0,933333	0,424242	0,966387	0,886788	0,957308
3	1,0	0,801471	0,299320	0,906250	0,835951	0,948505

8.4.3. Comentarios Generales

- **Alta Precisión del Contexto**: Ambos métodos son muy efectivos en recuperar contextos relevantes sin falsos positivos, con una puntuación de precisión perfecta (1) en todas las pruebas. Esto indica que los fragmentos relevantes del contexto siempre están presentes en las respuestas generadas, ya sean textuales o de triples.
- **Recuperación de Entidades**: Aunque la recuperación de entidades es relativamente baja en ambas evaluaciones, se observa una mayor puntuación en los resultados de triples en comparación con las respuestas textuales (0,049180, 0,424242, 0,299320 frente a 0,192308, 0,066667, 0,142857). La estructura de triples (entidad1, relación, entidad2) probablemente facilita una mejor extracción de entidades relevantes del contexto, aunque aún hay margen de mejora.
- **Fidelidad de las Respuestas**: Las respuestas textuales mostraron una fidelidad perfecta (1) con respecto al contexto recuperado, lo que significa que todas las afirmaciones pueden inferirse directamente del contexto. En la evaluación de triples, aunque la fidelidad también es alta (0,891720 a 0,966387), hay ligeras inexactitudes. Esto sugiere que aunque los triples son mayormente fieles al contexto, hay algunos errores menores.
- **Relevancia de la Respuesta**: Ambas evaluaciones muestran alta relevancia en las respuestas generadas, con puntuaciones ligeramente menores para los triples (0,819966 a 0,886788) en comparación con las respuestas textuales (0,905807 a 0,941944). Esto indica que las respuestas textuales tienden a ser un poco más pertinentes con respecto al prompt dado.
- **Capacidad de Resumir Información**: Las respuestas de triples tienen consistentemente altas puntuaciones de resumen (0,950893 a 0,957308), superiores en comparación con las respuestas textuales (0,497064 a 0,971908). De nuevo, la estructura organizada de los triples (entidad1, relación, entidad2) probablemente ayuda a capturar y resumir de manera efectiva la información clave presente en el contexto.

- **Desempeño Variable en el Contexto**: Aunque la precisión del contexto es perfecta en ambas evaluaciones, la exhaustividad varía considerablemente en la evaluación de triples (0,408805 a 0,933333), en contraste con el valor perfecto en las respuestas textuales. Algunas hipótesis para explicar esto son:

  1. Al convertir texto a triples, se pierden detalles del contexto.

  2. La estructura de triples es menos expresiva que el lenguaje natural para describir información, generando triples en base a fragmentos del contexto donde el evaluador de RAGAS determina que no existe causalidad.

  3. La LLM es capaz de generar nuevos triples en base a los que ya creó, los cuales pueden no encontrarse de manera explicita en el contexto.

  Dado lo anterior, y a la precisión del contexto de 1 de los triples, se puede concluir que la información relevante se captura de manera efectiva, lo que a su vez aumenta la posibilidad que las hipótesis planteadas para explicar la baja exhaustividad sean correctas.

9. Conclusión

Se concluye que el desarrollo de este sistema entrega resultados interesantes en base a la bibliografía ingresada a la base de datos y a las consultas realizadas. Mostrando respuestas verificable a través de las fuentes provistas y mostrando una propuesta al problema de texto a KG utilizando el poder de las LLMs.

En la evaluación de los resultados se puede ver que el sistema obtiene contextos y respuestas coherentes y con concordancia entre sí, lo que da buenos indicios de la aplicabilidad de un sistema RAG para trabajar con una amplia bibliografía y producir distintos tipos de respuesta.

Como trabajo futuro, hay muchas pequeñas optimizaciones que se pueden realizar para mejorar el sistema sin hacer grandes cambios en el, como por ejemplo, automatizar el despliegue del sistema mediante *Infraestructura como Código* (IaC), mejora de prompts, optimización de respuesta de

peticiones, mejoras de UI, entre otras. Sin embargo, el mayor potencial se encuentra en (1) la mejora del modelo que realiza el "texto-a-grafo", (2) la validación del modelo, limitado actualmente debido a la carencia de métricas específicas para este problema, y (3) en la entrega de información una vez el grafo ha sido generado. Es decir, obtener propiedades del grafo, y aplicar algoritmos que puedan entregar medidas de centralidad, las cuales se puedan complementar con una interpretación natural, de cara a un usuario no experto en computación.

Además, al haber ya identificado que el sistema está compuesta de varias partes modulares, se pueden ir remplazando por distintas herramientas o servicios pensando en una mejora continua de los resultados. Como hacer cambios al tipo de recuperación y generación hechos, o al como se hacen las consultas, que además de lenguaje natural, podrían complementarse con algún lenguaje de consultas, pensando también en implementar una base de datos de grafos dentro de este sistema, aprovechando que ya se tienen datos estructurados como un KG, solo que de manera no persistente.

Si bien el trabajo realizado posee un aporte investigativo por si mismo, la razón primordial de su existencia es que pueda entregar un valor social, el cual no se podrá medir hasta que personas ajenas a todas las instituciones implicadas en el proyecto la prueben y den una opinión. Esto se puede lograr a través de desplegar de manera pública (o a través de invitaciones) el sistema y sistematizar una encuesta que los usuarios puedan llenar. Esto queda pendiente para fases futuras.

Referencias

[1] Baeza-Yates, R., Ribeiro-Neto, B., et al. *Modern information retrieval*, vol. 463. ACM press New York, 1999.

[2] Bonnefoy, P. Medio siglo después del golpe, chile lanza una búsqueda de sus desaparecidos. https://www.nytimes.com/es/2023/08/30/espanol/chile-golpe-estado-desaparecidos.html, 2023.

[3] Brin, S., and Page, L. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems 30*, 1-7 (1998), 107–117.

[4] Chile. Ley no. 19123 crea corporacion nacional de reparacion y reconciliacion, establece pension de reparacion y otorga otros beneficios en favor de personas que seÑala. *Diario Oficial [D.O]* (2000).

[5] Chile. Ley no. 19687 establece obligacion de secreto para quienes remitan informacion conducente a la ubicacion de detenidos desaparecidos. *Diario Oficial [D.O]* (2000).

[6] Cingranelli, D. L., and Richards, D. L. The cingranelli and richards (ciri) human rights data project. *Human rights quarterly 32*, 2 (2010), 401–424.

[7] Comisión Nacional de Verdad y Reconciliación. *Informe de Comisión Nacional de Verdad y Reconciliación (Rettig)*. Santiago: Secretaría de Comunicación y Cultura, Ministerio Secretaría General de Gobierno de Chile., 1991.

[8] Comisión Nacional sobre Prisión Política y Tortura. *Informe de la Comisión Nacional sobre Prisión Política y Tortura (Valech I)*. Santiago: Ministerio del Interior., 2004.

[9] Commons, W. File:unraveling ai complexity - a comparative view of ai, machine learning, deep learning, and generative ai.png — wikimedia commons, the free media repository, 2024. [Online; accessed 7-July-2024].

[10] Es, S., James, J., Espinosa-Anke, L., and Schockaert, S. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217* (2023).

[11] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., and Wang, H. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* (2023).

[12] Labruna, T., Campos, J. A., and Azkune, G. When to retrieve: Teaching llms to utilize information retrieval effectively. *arXiv preprint arXiv:2404.19705* (2024).

[13] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems 33* (2020), 9459–9474.

[14] Malkov, Y. A., and Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence 42*, 4 (2018), 824–836.

[15] McCarthy, J., Minsky, M. L., Rochester, N., and Shannon, C. E. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955.

[16] Memoria Viva. Archivo digital de las violaciones a los derechos humanos por la dictadura militar en chile (1973-1990). https://memoriaviva.com/nuevaweb/, 2024.

[17] Ministerio de Justicia y Derechos Humanos. *Plan Nacional de Búsqueda Verdad y Justicia*. Santiago: Diario Oficial de la República de Chile. Ministerio del Interior y Seguridad Pública., 2023.

[18] MINISTERIO DEL INTERIOR DE CHILE; SUBSECRETARIA DEL INTERIOR. Decreto 1005 reglamenta función asumida por el ministerio en materias que indica, de competencia de la ex corporación de reparación y reconciliación que creó la ley nº 19.123.

[19] MINISTERIO DEL INTERIOR DE CHILE; SUBSECRETARIA DEL INTERIOR. decreto 1040 crea comision nacional sobre prisión política y tortura, para el esclarecimiento de la verdad acerca de las violaciones de derechos humanos en chile.

[20] MINISTERIO DEL INTERIOR DE CHILE; SUBSECRETARIA DEL INTERIOR. Decreto 43 establece comisión asesora para la calificación de detenidos desaparecidos, ejecutados políticos y víctimas de prisión política y tortura entre el 11 de septiembre de 1973 y el 10 de marzo de 1990.

[21] MINISTERIO DEL INTERIOR DE CHILE; SUBSECRETARIA DEL INTERIOR. decreto 98 aprueba plan nacional de bÚsqueda de verdad y justicia respecto de las personas vÍctimas de desapariciÓn forzada ocurridas en chile entre el 11 de septiembre de 1973 y el 10 de marzo de 1990, designa al programa de derechos humanos como órgano ejecutor y crea el comité de seguimiento y participación.

[22] Nagarajan, V., and Kolter, J. Z. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems 32* (2019).

[23] Naraki, Y., Yamaki, R., Ikeda, Y., Horie, T., and Naganuma, H. Augmenting ner datasets with llms: Towards automated and refined annotation. *arXiv preprint arXiv:2404.01334* (2024).

[24] Neo4J. Llm graph builder. https://neo4j.com/labs/genai-ecosystem/llm-graph-builder/, 2024.

[25] Plique, G. Graphology, a robust and multipurpose graph object for javascript. *Zenodo. https://doi.org/10.5281/zenodo 5681257* (2022).

[26] Sferrazza Taibi, P. La búsqueda de personas desaparecidas en chile:¿ necesidad de un complemento humanitario? *Revista mexicana de ciencias políticas y sociales 66*, 243 (2021), 79–108.

[27] Unidad de Derechos Humanos, S. M. L. d. C. Informe de gestiÓn n° 4 “polÍtica de derechos humanos del servicio mÉdico legal”.

[28] Vrgoc, D., Rojas, C., Angles, R., Arenas, M., Calisto, V., Farías, B., Ferrada, S., Heuer, T., Hogan, A., Navarro, G., Pinto, A., Reutter, J., Rosales, H., and Toussiant, E. Millenniumdb: A multi-modal, multi-model graph database. In *Companion of the 2024 International Conference on Management of Data* (New York, NY, USA, 2024), SIGMOD/PODS '24, Association for Computing Machinery, p. 496–499.

[29] Vrgoč, D., Rojas, C., Angles, R., Arenas, M., Arroyuelo, D., Buil-Aranda, C., Hogan, A., Navarro, G., Riveros, C., and Romero, J. MillenniumDB: An Open-Source Graph Database System. *Data Intelligence 5*, 3 (08 2023), 560–610.

[30] Webber, J. A programmatic introduction to neo4j. In *Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity* (2012), pp. 217–218.

[31] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM 64*, 3 (2021), 107–115.

[32] Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., Dou, Z., and Wen, J.-R. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107* (2023).

Anexos

A. Código Fuente e Instalación

El proyecto desarrollado en el marco de la presente memoria de título es de carácter ”código abierto o libre”.

El código fuente, junto con instrucciones para su instalación, se encuentran en el siguiente repositorio: https://github.com/vlermandac/retrieval-augmented-graph-generation-webapp.

B. Prompts Utilizados

Primero, se muestran los prompts utilizados en la instancia del LLM encargada de generar los triples para cada chunk de texto.

System Prompt 1

> *You are an expert in extracting the most important information from text. You extract entities and relationships between them. Entities can be a person, organization, or location. Entities MUST NOT be actions or events. You extract the triplets in the below format:*
> *”’ {entidad1, relacion, entidad2} ”’*

User Prompt 1

Sea *{text}* el texto de cada chunk de un documento:

> *Obtén entidades y relaciones en el formato (entidad1, relación, entidad2) desde el siguiente texto: ”’{text}”’.*

Luego, se muestran los prompts utilizados para la instancia del LLM encargada de responder con el contexto obtenido en la fase de recuperación.

System Prompt 2

> *Eres un experto en historia y política de Chile.*

User Prompt 2

Sea *{query}*, la consulta del usuario, y *{retrieved_context}* el contexto generado a través de la búsqueda semántica en la fase de recuperación.

> *Responde la siguiente Query en base al Contexto de abajo:*
> *Query:*
> *”’ {query} ”’*
> *Contexto:*
> *”’ {retrieved_context} ”’*

C. Cálculo de Recursos

A continuación se muestran los script utilizados para ejecutar, monitorear y calcular estadísticas promedio respecto a la ejecución del sistema durante el proceso de ingesta de datos y de generación de respuestas.

```
#!/bin/sh

# Directory to save the CSV files
output_dir="docker_stats"
mkdir -p "$output_dir"

# Counter for file naming
```

```
counter=1

# Loop to run the docker stats command every 3 seconds
while true; do
  # Get the current timestamp
  timestamp=$(date +'%Y%m%d_%H%M%S')

  # Define the output file name
  output_file="$output_dir/docker_stats_$timestamp.csv"

  # Run the docker stats command and save the output to the CSV file
  docker stats --no-stream --format 'table {{.Name}},{{.CPUPerc}},{{.MemUsage}}' > "$output_file"

  # Print a message
  echo "Saved stats to $output_file"

  # Increment the counter
  ((counter++))

  # Wait for 3 seconds
  sleep 3

done
```

Código 6: Script para obtener recursos utilizados cada 3 segundos

```
import pandas as pd
import glob
import os

def parse_memory(mem_str):
    """Convert memory usage string to MiB."""
    if 'GiB' in mem_str:
        return float(mem_str.replace('GiB', '')) * 1024
```

```
    elif 'MiB' in mem_str:
        return float(mem_str.replace('MiB', ''))
    else:
        return float(mem_str)

def parse_cpu(cpu_str):
    """Convert CPU usage string to float."""
    return float(cpu_str.replace('%', ''))

# Path to the directory containing the CSV files
csv_files_path = './docker_stats'

# Get a list of all CSV files in the directory
csv_files = glob.glob(os.path.join(csv_files_path, '*.csv'))

# Initialize an empty DataFrame to store combined data
combined_df = pd.DataFrame()

# Read and concatenate all CSV files
for file in csv_files:
    df = pd.read_csv(file, skipinitialspace=True)
    combined_df = pd.concat([combined_df, df], ignore_index=True)

# Convert CPU % and MEM USAGE to numeric values
combined_df['CPU %'] = combined_df['CPU %'].apply(parse_cpu)
combined_df['MEM USAGE / LIMIT'] = combined_df['MEM USAGE / LIMIT'].apply(lambda x: parse_memory(
x.split('/')[0]))

# Group by NAME and calculate statistics
stats = combined_df.groupby('NAME').agg({
    'CPU %': ['mean', 'std', 'min', 'max'],
    'MEM USAGE / LIMIT': ['mean', 'std', 'min', 'max']
})
```

```
stats.to_csv('stats.txt')
```

Código 7: Obtención de estadísticas descriptivas dado un conjunto de datos de monitoreo

D. Contexto del Retrieval

Se adjunta el repositorio debido a que estos textos son muy extensos para dejarlos escritos en el informe.

D.1. Contexto Obtenido con Prompt 1

https://github.com/vlermandac/retrieval-augmented-graph-generation-webapp/blob/main/notebooks/results_to_evaluate/case1/context.json

D.2. Contexto Obtenido con Prompt 2

https://github.com/vlermandac/retrieval-augmented-graph-generation-webapp/blob/main/notebooks/results_to_evaluate/case2/context.json

D.3. Contexto Obtenido con Prompt 3

https://github.com/vlermandac/retrieval-augmented-graph-generation-webapp/blob/main/notebooks/results_to_evaluate/case3/context.json