UNIVERSIDAD DEL AZUAY

Facultad de Ciencias de la Administración

Carrera de Ingeniería en Sistemas y Telemática

Enriquecimiento de un conjunto de datos utilizando modelos de lenguaje grandes. Caso de estudio: Comercio Exterior del Azuay 2022

Trabajo de titulación previo a la obtención del grado de Ingeniero de Sistemas y Telemática

Autor:
José Esteban Valverde Dávila

Director:
Marcos Patricio Orellana Cordero

Cuenca – Ecuador
2026

DEDICATORIA

A mis padres Sonia y Esteban, quienes me impulsaron a formarme como una persona que aporte a la sociedad, mediante todo tipo de apoyo y amor que recibí durante toda su maravillosa crianza. A mi hermana menor Paz, por su fuerte disposición moral, la que me impulsó en momentos de debilidad. A mi primo Martín, junto a quien crecí y me enseñó una forma de perseverancia de la cual estoy orgulloso.

AGRADECIMIENTO

Agradezco a mi tutor, Marcos Orellana por la guía y paciencia en el proceso, a Jorge Zambrano por su aporte en consultas que llevaron a un mejor procedimiento y a Santiago García por facilitar el acceso a los datos, que fueron fundamentales durante todo el proceso.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Índice de Anexos

RESUMEN

Los conjuntos de datos carecen con frecuencia de documentación asociada, lo que impide que los modelos de lenguaje grandes (LLMs) los comprendan e interpreten correctamente sin contexto previo. Este problema limita la capacidad de los LLMs para responder consultas sobre data sets. El objetivo de este trabajo es diseñar y evaluar un sistema basado en cadena de prompts que enriquezca de forma autónoma un conjunto de datos cualquiera, generando información contextual a partir de sus propios atributos, sin requerir documentación. El sistema fue implementado en Python mediante Google Colab, utilizando LLMs accedidos a través de las APIs de Hugging Face y Groq, y aplicado al data set "COMEX_DATA_AZUAY_banco_central_2022.csv", de comercio exterior del Azuay del año 2022, con 94.018 registros y 78 columnas. El enriquecimiento se ejecutó en hasta nueve iteraciones de prompts encadenados, y los resultados se evaluaron mediante seis consultas SQL (tres simples y tres avanzadas) sobre cinco modelos: Llama-3.1-8B-Instruct, Llama-3.3-70B-versatile, Llama-3.2-1B-Instruct, ERNIE-4.5-0.3B y Qwen2.5-1.5B-Instruct. Todos los modelos alcanzaron una efectividad del 100%, generando respuestas sintácticamente válidas en todas las condiciones. La eficacia varía según el tipo de consulta: las simples obtuvieron un promedio del 81,09% sin enriquecimiento, frente al 42,92% de las avanzadas. El enriquecimiento tuvo un efecto inconsistente a nivel global, aunque resultó beneficioso en consultas avanzadas para modelos de mediana escala. Se concluye que el sistema es viable para transferir comprensión contextual a LLMs sobre data sets, y que la complejidad de la consulta constituye el factor de mayor incidencia sobre la eficacia del sistema.

Palabras clave: cadena de pensamiento, cadena de prompts, conjunto de datos, enriquecimiento, modelos de lenguaje grandes, prompts.

ABSTRACT

Data sets frequently lack associated documentation, preventing Large Language Models (LLMs) from correctly understanding and interpreting them without prior context. This issue limits the ability of LLMs to respond to queries about data sets. The objective of this work is to design and evaluate a system based on a chain-of-prompts approach that autonomously enriches any given data set by generating contextual information from its own attributes, without requiring external documentation. The system was implemented in Python using Google Colab, utilizing LLMs accessed through Hugging Face and Groq APIs, and applied to the "COMEX_DATA_AZUAY_banco_central_2022.csv" dataset, which contains foreign trade data from Azuay for the year 2022, consisting of 94,018 records and 78 columns. The enrichment process was executed through up to nine iterations of chained prompts, and the results were evaluated using six SQL queries (three simple and three advanced) across five models: Llama-3.1-8B-Instruct, Llama-3.3-70B-versatile, Llama-3.2-1B-Instruct, ERNIE-4.5-0.3B, and Qwen2.5-1.5B-Instruct. All models achieved 100% effectiveness, generating syntactically valid responses under all conditions. Efficacy varied depending on the query type: simple queries achieved an average of 81.09% without enrichment, compared to 42.92% for advanced queries. Enrichment had an inconsistent effect at a global level, although it proved beneficial for advanced queries in medium-scale models. It is concluded that the system is viable for transferring contextual understanding of datasets to LLMs, and that query complexity is the factor with the greatest impact on the system's efficacy.

Keywords: chain of prompts, chain of thought, datasets, enrichment, large language models, prompts.

1. Introducción

Los conjuntos de datos estructurados constituyen uno de los activos más valiosos en la gestión del conocimiento organizacional e institucional. Sin embargo, su valor potencial frecuentemente permanece subutilizado debido a una brecha fundamental: la información que explica qué contiene un data set, qué significa cada uno de sus atributos, cuál es su contexto de generación y cómo deben interpretarse sus valores, rara vez está integrada en el propio archivo de datos. Esta información contextual (denominada metadatos o información de enriquecimiento) suele encontrarse dispersa en documentos externos, en el conocimiento tácito de quienes construyeron el data set, o simplemente no existe en forma estructurada. Como consecuencia, un conjunto de datos sin documentación adecuada es, en la práctica, opaco para cualquier sistema o usuario que lo reciba sin ese contexto previo.

Esta opacidad representa un problema especialmente relevante en el dominio del análisis de datos con modelos de lenguaje grandes. Los LLMs han demostrado capacidades notables para comprender, razonar y generar texto en una amplia variedad de tareas; no obstante, cuando se les solicita operar sobre un data set de dominio específico (con columnas cuya nomenclatura es técnica, abreviada o propia de un sector) su desempeño puede degradarse significativamente si no disponen del contexto necesario para interpretar correctamente la información que se les presenta. El pre entrenamiento de estos modelos sobre corpus generales de texto no garantiza el conocimiento de dominio requerido para conjuntos de datos especializados, como los registros de comercio exterior de una región, que contienen terminología aduanera, códigos arancelarios y estructuras de datos propias del marco regulatorio ecuatoriano.

Frente a este problema, el enriquecimiento de data sets mediante LLMs emerge como una solución prometedora. Si un modelo de lenguaje es capaz de inferir, a partir del nombre de un archivo, los nombres de sus columnas y una muestra representativa de sus datos, qué información contiene el conjunto de datos y qué significa cada uno de sus atributos, entonces esa información generada puede utilizarse como contexto para habilitar a otros modelos (incluyendo modelos de menor capacidad paramétrica) para interactuar con el data set de manera informada. Este proceso de enriquecimiento, cuando se estructura como una cadena de prompts iterativa, permite acumular progresivamente capas de contexto semántico que enriquecen la comprensión del data set sin requerir intervención humana ni documentación pre existente.

El presente trabajo aborda precisamente este problema, proponiendo y evaluando un sistema de enriquecimiento de data sets basado en cadena de prompts e implementado sobre modelos de lenguaje grandes. Como caso de estudio se utiliza el conjunto de datos COMEX_DATA_AZUAY_banco_central_2022.csv, un data set de comercio exterior del Azuay del año 2022, provisto por el Banco Central del Ecuador, que contiene 94.018 registros distribuidos en 78 columnas de naturaleza técnica y especializada. Este data set no posee documentación estructurada de acompañamiento, lo que lo convierte en un caso representativo del problema descrito: rico en información, pero opaco en contexto.

La relevancia del trabajo se inscribe en una tendencia creciente en la investigación en inteligencia artificial: el uso de LLMs no únicamente como motores de generación de texto, sino como herramientas para la creación y gestión de conocimiento estructurado sobre datos. Trabajos como el de Giner-Miguelez et al. (2024) han explorado el encadenamiento de prompts para enriquecer la documentación de data sets a partir de textos descriptivos existentes, logrando precisiones superiores al 81% en la extracción de metadatos. La presente tesis extiende esta línea de investigación hacia escenarios donde dicha documentación no existe, construyendo el contexto directamente desde los datos, lo que amplía el alcance potencial del método a conjuntos de datos carentes de metadatos formales.

1.1 Objetivos

1.1.1 Objetivo General

Diseñar un sistema basado en Modelos de Lenguaje Grandes que enriquezca un conjunto de datos, integrando información contextual relevante a partir de los nombres de las columnas, los datos que cada columna, el nombre del archivo y cualquier fuente de datos no estructurada adicional, tomando como caso de estudio el comercio exterior del Azuay del año 2022.

1.1.2 Objetivos Específicos

a) Analizar la literatura existente sobre técnicas de enriquecimiento de datos utilizando LLMs, identificando técnicas, métodos, herramientas y mejores prácticas
b) Formular un método estructurado para el enriquecimiento de datos que guiarán la interacción del LLM con el conjunto de datos
c) Desarrollar un sistema que implemente el enriquecimiento de datos, utilizado los nombres de columnas, los datos del conjunto de datos, el nombre del archivo y

cualquier fuente no estructurada, como insumos para generar información enriquecida

d) Evaluar el desempeño del sistema mediante métricas estandarizadas que comparen los resultados con conjuntos de datos de referencia

2. Marco Teórico y Estado del Arte

2.1 Marco teórico

Procesamiento de Lenguaje Natural (NLP)

El Procesamiento de Lenguaje Natural (Natural Language Processing, NLP) es un campo interdisciplinario que combina lingüística computacional, ciencias de la computación e inteligencia artificial para permitir que las computadoras comprendan, interpreten y generen lenguaje humano de manera significativa y útil. Jurafsky & Martin (2008) definen el NLP como el conjunto de métodos computacionales para analizar, comprender y generar lenguaje humano, señalando que este campo se ha vuelto fundamental para una amplia gama de aplicaciones que van desde la traducción automática hasta asistentes virtuales.

El desarrollo histórico del NLP ha pasado por varias etapas evolutivas. En sus inicios, el campo se caracterizó por enfoques basados en reglas que requerían la codificación manual de gramáticas y conocimiento lingüístico. Con la llegada del aprendizaje automático, el NLP experimentó una transición hacia métodos estadísticos que aprenden patrones directamente de grandes cantidades de texto. Treviso et al. (2023) documentan que el trabajo reciente en NLP ha producido resultados atractivos al escalar los parámetros del modelo y los datos de entrenamiento, aunque esto también significa que el consumo de recursos como datos, tiempo, almacenamiento y energía también crece, motivando la investigación en métodos eficientes que requieran menos recursos para lograr resultados similares.

Las tareas fundamentales del NLP abarcan múltiples niveles de análisis lingüístico. En el nivel léxico, se incluyen tareas como tokenización, que divide el texto en unidades más pequeñas como palabras y análisis morfológico, que estudia la estructura interna de las palabras. En el nivel sintáctico, el análisis de partes del discurso (part-of-speech tagging) y el análisis sintáctico (parsing) identifican las relaciones gramaticales entre palabras. En el nivel semántico, tareas como el reconocimiento de entidades nombradas (NER), la desambiguación del sentido de las palabras y el análisis de dependencias semánticas, buscan extraer el significado del texto. Finalmente, en el nivel pragmático, se abordan aspectos

como el análisis de sentimiento, la resolución de correferencias y la comprensión del discurso.

Qin et al. (2025) exploraron cómo los LLMs se aplican actualmente a tareas de NLP en la literatura. Su investigación resalta que, aunque los LLMs como ChatGPT han mostrado capacidades impresionantes en tareas de NLP, una investigación sistemática de su potencial en este campo permanece en gran medida inexplorada.

En el contexto del enriquecimiento de datos, el NLP proporciona las herramientas fundamentales para extraer, transformar y generar información textual. Las técnicas de NLP permiten analizar metadatos textuales, inferir tipos semánticos de columnas en datasets, generar descripciones contextuales y realizar expansión semántica de términos abreviados. Estas capacidades son esenciales para sistemas que buscan agregar valor a conjuntos de datos existentes mediante la incorporación de información contextual relevante derivada de nombres de columnas, valores de datos y otras fuentes textuales no estructuradas.

Modelos de Lenguaje Grandes (LLMs)

Los Modelos de Lenguaje Grandes, conocidos en inglés como Large Language Models o LLMs, representan modelos de lenguaje pre entrenados a gran escala que han demostrado capacidades emergentes notables a medida que aumenta su tamaño. Zhao et al. (2025) definen los LLMs como modelos de lenguaje que contienen decenas o cientos de miles de millones de parámetros, los cuales son entrenados sobre cantidades de datos masivos. El lenguaje es esencialmente un sistema complejo de expresiones humanas gobernado por reglas gramaticales, y los LLMs han sido estudiados extensamente para la comprensión y generación del lenguaje, evolucionando desde modelos de lenguaje estadísticos hasta modelos de lenguaje neuronales.

Minaee et al. (2025) argumentan que la capacidad de comprensión y generación del lenguaje de propósito general de los LLMs, se adquiere mediante el entrenamiento de miles de millones de parámetros del modelo sobre cantidades masivas de datos de texto. El área de investigación de los LLMs, aunque muy reciente, está evolucionando rápidamente en muchas direcciones diferentes, incluyendo el desarrollo de técnicas para construir y aumentar LLMs, así como la creación de datasets para su entrenamiento, ajuste fino y evaluación.

En el contexto del enriquecimiento de datos, los LLMs han demostrado ser particularmente valiosos debido a sus capacidades avanzadas de procesamiento de lenguaje natural. Giner-Miguelez et al. (2024) exploraron el uso de LLMs para enriquecer la documentación de conjuntos de datos para aprendizaje automático, utilizando un método de encadenamiento de prompts de su propio desarrollo, mostrando que estos modelos pueden extraer dimensiones clave de documentos no estructurados y enriquecer las descripciones de conjuntos de datos con información contextual relevante. Los estudios comparativos entre modelos como GPT-3.5 y Flan-UL2 han reportado precisiones superiores al 81% en tareas de extracción de metadatos, aunque con diferente propensión a alucinaciones, según el modelo utilizado.

A pesar de los avances significativos en las capacidades de los LLMs, estos modelos enfrentan limitaciones fundamentales inherentes a su naturaleza computacional. Xu et al. (2025) demostraron formalmente que la alucinación, definida como la generación de información plausible pero incorrecta, es inevitable en los LLMs cuando se utilizan como solucionadores de problemas generales. Mediante el empleo de resultados de la teoría del aprendizaje, los autores establecieron que ningún LLM computable puede aprender todas las funciones computables, lo que implica que inevitablemente generarán salidas inconsistentes con la verdad fundamental en ciertos contextos. Estos hallazgos tienen implicaciones prácticas críticas para el despliegue seguro de LLMs, enfatizando la necesidad de supervisión humana, verificación externa y delimitación clara de los límites de capacidad de estos modelos, particularmente en aplicaciones críticas para la seguridad donde errores debido a alucinaciones pueden tener consecuencias inaceptables.

Prompts y Arquitectura de Instrucciones

Un prompt es una entrada proporcionada a un modelo de NLP, que contiene instrucciones del usuario indicando al modelo qué tipo de salida se desea. Brown et al. (2020) introdujeron formalmente el concepto de prompting en el contexto de GPT-3, donde el modelo analiza el prompt y basándose en su entrenamiento, produce un texto de salida. Este enfoque permite que los modelos de lenguaje realicen tareas específicas sin necesidad de ajuste fino, mediante lo que se denomina aprendizaje en contexto o in-context learning.

El diseño de prompts efectivos se ha convertido en un componente crítico para maximizar el rendimiento de los LLMs. Sahoo et al. (2025) señalan que el prompting permite la integración perfecta de modelos pre entrenados en tareas descendentes al provocar los comportamientos deseados del modelo únicamente basándose en el prompt proporcionado, sin modificar los parámetros centrales del modelo. Los prompts pueden ser instrucciones en lenguaje natural que proporcionan contexto para guiar al modelo, o representaciones vectoriales aprendidas que activan conocimiento relevante. A medida que las ventanas de contexto de los modelos de IA generativa crecen, los prompts pueden incluir cantidades cada vez mayores de texto, o incluso imágenes y videos, que proporcionan contexto para la solicitud del usuario. Esta característica resulta particularmente relevante para el enriquecimiento de datos, donde el prompt puede contener nombres de columnas, muestras de datos y metadatos del archivo como información contextual para generar descripciones o clasificaciones enriquecidas.

Ingeniería de Prompts

La ingeniería de prompts ha emergido como una técnica indispensable para aprovechar las capacidades de los LLMs. Chen et al. (2025) definen la ingeniería de prompts como el proceso de estructurar entradas para maximizar la utilidad y precisión de estos modelos de lenguaje, explorando metodologías fundamentales y avanzadas que incluyen técnicas como self-consistency, chain-of-thought y generated knowledge, las cuales mejoran significativamente el rendimiento del modelo.

Schulhoff et al. (2025) presentaron un estudio sobre ingeniería de prompts, estableciendo un vocabulario estructurado de 33 términos, una taxonomía de 58 técnicas de prompting para LLMs y 40 técnicas para otras modalidades. La investigación documenta que el rendimiento de los LLMs es altamente sensible a elecciones como el ordenamiento de ejemplos, la calidad de las etiquetas de demostración e incluso pequeñas variaciones en la redacción.

Knoth et al. (2024) investigaron la ingeniería de prompts, conceptualizándola como la capacidad de comunicar precisamente la esencia de un problema a un asistente de IA. Su estudio demostró que esta habilidad es tan crucial como el asistente mismo, dado que una ligera alteración en la redacción puede hacer la diferencia entre que un asistente malinterprete una instrucción o supere las expectativas. La investigación reveló que esta

competencia puede ser definida, enseñada y evaluada de manera similar a otras competencias genéricas.

Algunas de las técnicas más relevantes de la ingeniería de prompts, tales como Zero-shot, Chain of Thought o Few-shot están diseñadas para mejorar significativamente el rendimiento de los LLMs. Por ejemplo, el prompting zero-shot permite que los modelos realicen tareas sin ejemplos previos, confiando exclusivamente en su conocimiento pre entrenado para inferir respuestas apropiadas (Kojima et al., 2023). Esta técnica resulta particularmente valiosa debido a su simplicidad y eficiencia, aunque su efectividad depende de la complejidad de la tarea y la calidad del prompt. En contraste, el prompting few-shot proporciona al modelo algunos ejemplos de entrada-salida para mejorar su rendimiento en tareas específicas (Brown et al., 2020) La investigación ha demostrado que el espacio de etiquetas y la distribución del texto de entrada especificados por las demostraciones son ambos importantes, independientemente de si las etiquetas son correctas para entradas individuales (Min et al., 2022). La técnica de Chain of Thought

Chain of Thought (CoT)

Chain-of-Thought prompting es una técnica de ingeniería de prompts que mejora significativamente las capacidades de razonamiento de los LLMs. Wei et al. (2023) introdujeron esta técnica, demostrando que generar una cadena de pensamiento, es decir, una serie de pasos de razonamiento intermedios, mejora significativamente la capacidad de los LLMs para realizar razonamiento complejo. Los experimentos realizados en modelos de lenguaje grandes mostraron que el CoT mejora el rendimiento en una variedad de tareas de razonamiento aritmético, de sentido común y simbólico.

X. Wang et al. (2023) propusieron self-consistency para mejorar el razonamiento de cadena de pensamiento en modelos de lenguaje, una nueva estrategia de decodificación que reemplaza la decodificación greedy (codiciosa) ingenua utilizada en el CoT. Esta técnica primero muestrea un conjunto diverso de rutas de razonamiento en lugar de tomar solo la greedy, y luego selecciona la respuesta más consistente descartando las rutas de razonamiento muestreadas.

Existen variantes importantes de CoT, como el Zero-shot CoT, que implica simplemente agregar frases como "Pensemos paso a paso" al final del prompt original, induciendo al

modelo a razonar en voz alta sin ejemplos previos. También está el Few-shot CoT, que proporciona ejemplos explícitos de razonamiento paso a paso antes de presentar el problema a resolver. Meincke et al. (2025), en un estudio reciente, demostraron que la efectividad del prompting Chain of Thought puede variar bastante dependiendo del tipo de tarea y modelo, generalmente siendo más útil para modelos más antiguos o pequeños, mientras que para modelos de razonamiento dedicados, los beneficios adicionales del prompting CoT explícito parecen irrelevantes.

Few-Shot Prompting

Few-shot prompting es una técnica que involucra suministrar a un modelo de lenguaje unas muestras de ejemplos de entrada-salida dentro del prompt. Brown et al. (2020) demostraron formalmente esta capacidad con GPT-3, un modelo de lenguaje autorregresivo con 175 mil millones de parámetros, mostrando que escalar los modelos de lenguaje mejora enormemente el rendimiento agnóstico a la tarea en configuraciones de few-shot, a veces incluso volviéndose competitivo con enfoques previos de vanguardia basados en ajuste fino. El trabajo demostró que los humanos generalmente pueden realizar una nueva tarea de lenguaje desde solo unos pocos ejemplos o desde instrucciones simples, algo con lo que los sistemas de NLP actuales todavía luchan en gran medida.

El mecanismo de aprendizaje en contexto que permite que funcione el few-shot prompting opera mediante un proceso donde el modelo interpreta el prompt completo, consistente en ejemplos y nuevas consultas de entrada, como una secuencia continua de tokens. El modelo analiza patrones de los ejemplos de entrada y salida para guiar su respuesta a la nueva consulta. Las demostraciones sirven como condicionamiento para ejemplos subsecuentes donde se quiere que el modelo genere una respuesta.

Chain of Prompts (Cadena de Prompts)

La cadena de prompts es una técnica avanzada de procesamiento de lenguaje natural que aprovecha los LLMs mediante la generación de una salida deseada a través de una serie de prompts secuenciales. Ding et al. (2024a) describen este proceso como uno en el cual se proporciona una secuencia de prompts a un modelo de NLP, guiándolo para producir la respuesta deseada. El modelo aprende a entender el contexto y las relaciones entre los prompts, permitiéndole generar texto coherente, consistente y contextualmente rico.

Esta técnica resulta particularmente útil para abordar tareas complejas que un LLM podría tener dificultad en manejar si se le presenta un prompt excesivamente detallado. En esta técnica, los prompts encadenados realizan transformaciones o procesos adicionales en las respuestas generadas antes de alcanzar un estado final deseado. Según Ding et al. (2024a), además de lograr mejor rendimiento, la cadena de prompts ayuda a aumentar la transparencia de la aplicación LLM, incrementa la controlabilidad y la confiabilidad, facilitando la depuración de problemas con las respuestas del modelo y el análisis y mejora del rendimiento en las diferentes etapas que necesitan optimización.

Datasets (Conjuntos de datos)

Un dataset es una colección estructurada de datos relacionados organizados para análisis. En su forma más simple, un dataset podría ser un archivo que contiene información tabular. En su forma más compleja, podría ser una colección de varios terabytes de datos multimodales que alimentan sistemas de IA en tiempo real.

Los datasets se componen de elementos fundamentales que definen su estructura. Los registros o filas representan entradas individuales que corresponden a una sola observación, como por ejemplo, un usuario, producto o transacción. Los atributos o columnas son las características que describen cada registro, tales como edad, precio o marca temporal. Sin embargo, no todos los datasets encajan perfectamente en filas y columnas; algunos almacenan texto, medios o eventos basados en tiempo, mientras que otros son dinámicos, actualizándose constantemente desde APIs o flujos de datos.

En cuanto a la estructura interna, existen dos formas primarias: tabular y no tabular. Los datasets tabulares son estructuras donde las filas representan registros individuales y las columnas definen atributos. Las estructuras tabulares son ideales para datos estructurados, numéricos o categóricos que encajan en hojas de cálculo o bases de datos relacionales. Los datasets no tabulares incluyen datos no estructurados como documentos de texto, imágenes, audio o video que no encajan en un formato de filas y columnas tradicional.

Metadatos

Los metadatos se definen como la información estructurada y descriptiva sobre datos, ofreciendo contexto como fuente, tipo y relaciones con otros conjuntos de datos. Mejoran la capacidad de búsqueda, organizan el contenido y aseguran que los datos sean accesibles y

confiables. Al categorizar y anotar datos, los metadatos soportan funcionalidades avanzadas como personalización de contenido, interoperabilidad y gobernanza de datos.

Los metadatos comúnmente se categorizan en tres tipos primarios. Los metadatos descriptivos proporcionan información sobre el contenido intelectual de un objeto digital para descubrimiento e identificación, incluyendo elementos como título, autor, fecha de publicación, tema, editor y descripción. Los metadatos estructurales documentan cómo las piezas de componentes de recursos de información complejos encajan entre sí, como capítulos en un libro o archivos en una página web multimedia. Los metadatos administrativos soportan la gestión de recursos e incluyen información técnica, gestión de derechos y metadatos de preservación.

En el contexto específico de datasets, los metadatos pueden incluir información técnica como esquema, tipos de datos y conteo de filas o columnas; información de gobernanza que cubre propiedad, clasificación de datos y medidas de seguridad; información operacional que involucra dependencias, código e información de tiempo de ejecución; metadatos de colaboración que rastrean comentarios, discusiones y retroalimentación relacionada con datos; metadatos de calidad que contienen métricas de calidad como frescura y estado de validación; y metadatos de uso que capturan perspectivas como conteos de vistas y popularidad de datasets.

Para datasets tabulares, los metadatos estructurales son particularmente importantes. Estos describen la estructura, tipo y relaciones de los elementos de datos. En una base de datos relacional, los metadatos estructurales definirían cómo las tablas, filas, columnas y relaciones entre entidades están organizadas. Los metadatos estructurales pueden incluir información de elementos de datos como nombres, tipos, longitudes, definiciones y otra información de uso; información de tablas como descripciones de tablas, ubicaciones físicas, nombres de columnas, tipos y restricciones de validez.

Reconocimiento de Entidades Nombradas (NER)

Named Entity Recognition (NER), también conocido como extracción de entidades, es una tarea fundamental del procesamiento de lenguaje natural que se enfoca en identificar diferentes tipos de entidades dentro de un texto y categorizarlas en clases predefinidas. Li et al. (2020) definen NER como la tarea de identificar menciones de designadores rígidos del

texto pertenecientes a tipos semánticos predefinidos como persona, ubicación, organización, entre otros. NER siempre sirve como fundamento para muchas aplicaciones de lenguaje natural como respuesta a preguntas, resumen de texto y traducción automática.

El proceso de NER típicamente involucra varias etapas fundamentales. La tokenización es el primer paso, donde el texto se descompone en unidades más pequeñas y manejables como palabras o frases, que sirven como la base para reconocer entidades dentro del texto. En la fase de detección de entidades, el sistema escanea los tokens para detectar posibles entidades basándose en patrones predefinidos, reglas o modelos de aprendizaje automático. Una vez que las entidades son identificadas, se categorizan en clases predefinidas como "Persona", "Organización" o "Ubicación", a menudo utilizando modelos de aprendizaje automático entrenados en data sets etiquetados. En el contexto del sistema propuesto en esta tesis, el NER opera de forma implícita a través del LLM encargado del enriquecimiento: al procesar los nombres de columnas del data set (como RAZON_SOCIAL, PAIS_ORIGEN o REGIMEN), el modelo identifica las entidades semánticas que estos designan (empresa, país, clasificación aduanera) y las categoriza en tipos significativos para el dominio del comercio exterior. Esta capacidad de reconocimiento permite que las iteraciones posteriores de la cadena de prompts construyan descripciones contextuales más precisas sobre cada atributo del data set.

Enriquecimiento Semántico

Martinez-Rodriguez et al. (2020) definen el enriquecimiento semántico como el proceso de adicionar metadatos significativos y explícitos al contenido original, con el objetivo de dotarlo de una estructura que sea comprensible tanto para los seres humanos como para los sistemas computacionales. A diferencia del etiquetado tradicional, que puede ser ambiguo o plano, el enriquecimiento semántico vincula los términos dentro de un documento con conceptos definidos en bases de conocimiento o grafos de conocimiento externos. En el sistema desarrollado en esta tesis, el enriquecimiento semántico se materializa en la generación iterativa de descripciones estructuradas sobre el data set: a partir del nombre del archivo y los nombres de columnas, el LLM produce metadatos explícitos (como la descripción del propósito de cada columna, el tipo de valor que contiene y su relevancia en el contexto del comercio exterior) que dotan al conjunto de datos de una capa semántica accesible para modelos que, de otro modo, no dispondrían de ese conocimiento de dominio.

Inferencia Semántica

La inferencia semántica es el proceso mediante el cual los sistemas computacionales derivan nuevo conocimiento a partir de información existente utilizando razonamiento lógico sobre representaciones semánticas estructuradas. En el contexto de grafos de conocimiento y sistemas de enriquecimiento de datos, la inferencia semántica permite descubrir relaciones implícitas, validar consistencia de datos y generar perspectivas que no están explícitamente declaradas en los datos fuente. En el sistema propuesto, la inferencia semántica se activa en las iteraciones avanzadas de la cadena de prompts: al recibir como entrada las descripciones generadas en fases anteriores junto con una muestra estratificada de los datos, el LLM infiere relaciones no declaradas explícitamente, como la correspondencia entre columnas codificadas (COD_REGIMEN, SUBPARTIDA) y sus categorías funcionales dentro del sistema aduanero ecuatoriano. Esta capacidad inferencial es la que permite que el contexto acumulado en el proceso de enriquecimiento trascienda la mera descripción literal de columnas para constituir un modelo semántico del data set.

2.2 Estado del Arte

Evolución Histórica del Enriquecimiento de Datos

El enriquecimiento de datos, entendido como el proceso de mejorar el valor de un conjunto de datos mediante la adición de contexto, tiene sus raíces en las técnicas de Text Mining de las décadas de 1990 y 2000. En este periodo, predominaron enfoques como la bolsa de palabras (bag-of-words) y la frecuencia de término, los cuales permitían enriquecer el texto asignando pesos y frecuencias a los términos, aunque carecían de comprensión semántica profunda. Sin embargo, estudios fundamentales como los de Liu et al. (2020) comenzaron a sentar las bases para un enriquecimiento más sofisticado mediante el modelado de temas (topic modeling), permitiendo descubrir estructuras latentes en grandes corpus de texto.

Hacia la década de 2010, se produjo una transición significativa hacia el Aprendizaje Automático Supervisado. El enfoque principal se desplazó hacia el data augmentation (aumento de datos) para mitigar la escasez de datos etiquetados en tareas como el análisis de sentimientos. Shorten & Khoshgoftaar (2019) revisaron exhaustivamente estos métodos, destacando técnicas como el reemplazo de sinónimos y la traducción inversa para enriquecer corpus en NLP. Esta etapa marcó el paso de un enriquecimiento puramente estadístico a uno basado en reglas lingüísticas y aprendizaje supervisado.

Técnicas Clave en Enriquecimiento Textual

Con la maduración de las técnicas de aprendizaje profundo, surgieron métodos más complejos centrados en la semántica y la integración de datos heterogéneos. La extracción y análisis semántico que la literatura moderna enfatiza, es el reconocimiento de entidades nombradas (NER) y el enlace de entidades (entity linking) como pilares del enriquecimiento. Pen et al. (2020) demostraron cómo el aprendizaje profundo (deep learning) puede enriquecer textos con etiquetas afectivas complejas para el análisis de emociones. En dominios más técnicos, Francia et al. (2023) evaluaron el preprocesamiento en textos legales, subrayando que la calidad del enriquecimiento de los *inputs* determina la eficacia de los algoritmos de clasificación posterior.

Avances en IA y Modelos de Lenguaje Grandes

La irrupción de los Modelos de Lenguaje Grandes (LLMs) ha redefinido las capacidades de enriquecimiento, permitiendo una comprensión contextual que los métodos anteriores no podían alcanzar. Roth & Wermer-Colan (2023), exploran cómo la minería de texto con Machine Learning (ML) facilita la identificación de tópicos y el clustering en grandes volúmenes de literatura. La versatilidad de los modelos actuales se observa en diversos nichos. En el análisis de opinión, Prastyo et al. (2025) utilizaron modelos clásicos (Naive Bayes, SVM) potenciados con diccionarios de jerga (slang) enriquecidos. Asimismo, los autores Khan et al. (2025) cubrieron avances en entornos multi lenguaje, crucial para contextos de comercio internacional. Otros trabajos, como Elazar et al. (2022) y Kuzman & Ljubešić (2025), han validado la capacidad de los modelos para generalizar tareas de enriquecimiento sintáctico y de género en conjuntos de datos diversos.

La anotación automática con LLM ha emergido como una prometedora solución para reducir la dependencia de anotadores humanos, una tarea que tradicionalmente consume mucho tiempo y recursos. Un estudio exhaustivo realizado por Ding et al. (2022) evaluó la capacidad de GPT-3 como anotador de datos, comparando su rendimiento con los métodos de anotación tradicionales y con la precisión humana. La investigación abarcó una amplia gama de tareas de NLP, incluyendo la clasificación de sentimientos, la clasificación de temas, el reconocimiento de entidades nombradas (NER) y la respuesta a preguntas, para ofrecer una visión integral del potencial de GPT-3 en la anotación de datos. El estudio de Ding et al. (2022) se centró en un conjunto de datos de preguntas de opción múltiple en inglés, y los hallazgos iniciales fueron notablemente positivos. El modelo etiquetó respuestas

correctas con una precisión comparable a la de los anotadores humanos, alcanzando un 85% de coincidencia en algunas tareas. Este resultado sugiere que la anotación automática es una opción viable y eficiente para tareas bien definidas y directas. El enfoque de los autores se basó en el aprendizaje de pocos intentos (few-shot learning), donde el modelo recibe un pequeño número de ejemplos de alta calidad en el prompt para guiar su proceso de anotación, sin necesidad de un ajuste fino (fine-tuning).

Estudio Sobre Chain of Prompts

El método de Chain of Promtps de Giner-Miguelez et al. (2024) utilizando los modelos GPT3.5 (text-davinci-003) y Flan-UL2 mostró una precisión general con GPT de 81.21%, que es muy superior a la precisión de Flan-UL2 que es de 69.13%, sin embargo, los autores destacan que GPT3.5 tiende a ser excesivamente confiado en sus respuestas. El análisis de la precisión de cada dimensión de la documentación reveló que cada una tiene dificultades diferentes para extraerlas, donde las más complicadas resultan en una tendencia por parte de los LLM, de responder con menos exactitud. Giner-Miguelez et al. (2024) observaron que su método tiene la capacidad de detectar dimensiones que no se consideraron en la documentación. En cuanto a ciertas dimensiones con las que los LLM lo hacen peor, los autores observaron que es debido a información similar. Además, los modelos tienden a confundir los métodos de validación de todo el conjunto de datos, con los métodos de validación de etiquetas, que normalmente no se encuentran en la documentación. Sin embargo, el trabajo de Giner-Miguelez et al. (2024) opera sobre data sets que ya poseen documentación textual estructurada: el insumo de su cadena de prompts son los documentos descriptivos que acompañan al data set, y el proceso consiste en extraer de ellos dimensiones de metadatos predefinidas. Esta condición implica que el método no es aplicable cuando dicha documentación no existe, que es precisamente el escenario predominante en data sets generados por instituciones públicas o sistemas transaccionales de dominio especializado. El presente trabajo se diferencia en este aspecto fundamental: el insumo de la cadena de prompts es el propio data set (nombre del archivo, nombres de columnas y muestra estratificada de datos), por lo que el sistema genera el contexto semántico de forma autónoma sin requerir ningún documento de acompañamiento previo. Esta diferencia amplía el alcance del método hacia data sets carentes de metadatos formales, que constituyen la mayoría de los conjuntos de datos disponibles en contextos institucionales y gubernamentales, y representa la brecha que el presente trabajo viene a cubrir.

Estudios Sobre Ingeniería de Prompts

La ingeniería de prompts ha emergido como un campo de investigación fundamental en la era de los modelos de lenguaje grandes, experimentando una evolución acelerada en los últimos años. Sahoo et al. (2025) realizaron una revisión sistemática que documenta la transición de técnicas básicas de prompting hacia metodologías sofisticadas que maximizan el rendimiento de los LLMs sin necesidad de modificar los parámetros del modelo. Esta área de investigación se ha consolidado como una disciplina que combina elementos de lingüística computacional, aprendizaje automático y diseño de interacción, estableciendo nuevos paradigmas para la utilización efectiva de modelos pre entrenados.

El trabajo de Schulhoff et al. (2025) presenta un estudio sobre ingeniería de prompts, estableciendo por primera vez una ontología estructurada del campo. Los investigadores identificaron y catalogaron cincuenta y ocho técnicas distintas de prompting para LLMs basados en texto y cuarenta técnicas adicionales para modalidades no textuales, acompañadas de un vocabulario controlado de treinta y tres términos fundamentales. Este esfuerzo de sistematización resulta crucial dado que, como señalan los autores, el rendimiento de los LLMs muestra una sensibilidad extrema a variaciones aparentemente menores en la construcción de prompts. Los experimentos documentados revelan que modificaciones en el ordenamiento de ejemplos o pequeñas variaciones en la redacción pueden producir cambios de precisión superiores al cuarenta por ciento, evidenciando la necesidad crítica de metodologías rigurosas y estandarizadas en el diseño de prompts.

La investigación de Chen et al. (2025) complementa este panorama al explorar el potencial de las técnicas de prompting en aplicaciones prácticas, examinando metodologías que incluyen self-consistency, chain-of-thought y generated knowledge. Su análisis demuestra que estas técnicas avanzadas mejoran significativamente el rendimiento del modelo en tareas complejas que requieren razonamiento en varios pasos y comprensión contextual profunda. Los autores enfatizan que la ingeniería de prompts no se limita a la formulación de instrucciones simples, sino que constituye un proceso iterativo de refinamiento que considera múltiples dimensiones: la estructura sintáctica del prompt, la selección y ordenamiento de ejemplos demostrativos, el balance entre especificidad y generalización, y la alineación con las capacidades intrínsecas del modelo objetivo.

A pesar de los avances significativos documentados, persisten brechas importantes en la investigación. La mayoría de los estudios se han concentrado en modelos específicos como GPT-3 y GPT-4, limitando la generalización de los hallazgos a otras arquitecturas de LLMs.

Adicionalmente, existe una necesidad apremiante de metodologías estandarizadas para evaluar la efectividad de diferentes técnicas de prompting a través de diversos dominios y tareas. La investigación futura debe abordar la automatización del diseño de prompts, el desarrollo de herramientas de asistencia para usuarios no expertos y la creación de frameworks teóricos que expliquen por qué ciertas formulaciones de prompts resultan más efectivas que otras en contextos específicos.

Estudios Sobre Chain of Thought

El prompting Chain-of-Thought (CoT) ha emergido como una de las técnicas más influyentes para mejorar las capacidades de razonamiento de los LLMs. Wei et al. (2023) introdujeron esta técnica, demostrando que generar una cadena de pensamiento mejora significativamente la capacidad de los modelos de lenguaje grandes para realizar razonamiento complejo. Los experimentos originales en tres modelos de lenguaje mostraron que el prompting CoT mejora el rendimiento en una variedad de tareas de razonamiento aritmético, de sentido común y simbólico, con ganancias empíricas que resultan sorprendentes. Por ejemplo, al proveer a un modelo de lenguaje de quinientos cuarenta mil millones de parámetros con solo ocho ejemplares de cadena de pensamiento, se logró precisión de vanguardia en el benchmark GSM8K de problemas de palabras matemáticas, superando incluso a GPT-3 ajustado con un verificador.

La evolución posterior del campo ha generado una diversificación significativa de enfoques y variantes de CoT. Chu et al. (2024) sistematizan una investigación relevante mediante una clasificación meticulosa que ofrece perspectivas novedosas sobre el estado del campo. Su análisis revela que el campo ha transitado desde demostraciones simples de razonamiento paso a paso hacia sistemas complejos que integran múltiples estrategias de razonamiento, verificación automática de pasos intermedios y generación adaptativa de cadenas de pensamiento. Yu et al. (2023) contribuyeron con un análisis de múltiples factores que influyen en la efectividad del prompting CoT, examinando cómo aplicar mejor esta técnica en diferentes contextos aplicativos. Su investigación identifica variables críticas que modulan el rendimiento de CoT, incluyendo la longitud óptima de las cadenas de razonamiento, la especificidad requerida en cada paso intermedio, y la sensibilidad del método a la calidad de las demostraciones proporcionadas. Los autores proponen estrategias mejoradas de prompting que adaptan dinámicamente la estructura de la cadena de pensamiento según la complejidad de la tarea y las características del dominio de aplicación.

Una contribución particularmente significativa al entendimiento de CoT proviene de B. Wang et al. (2023), quienes demostraron hallazgos contraintuitivos que desafían suposiciones previas sobre el funcionamiento de CoT: el razonamiento CoT es posible incluso con demostraciones inválidas, donde el prompting con pasos de razonamiento incorrectos puede alcanzar entre ochenta y noventa por ciento del rendimiento obtenido usando CoT válido bajo diversas métricas, mientras continúa generando líneas de razonamiento coherentes durante la inferencia. Experimentos adicionales revelaron que otros aspectos de las racionales, como ser relevantes para la consulta y ordenar correctamente los pasos de razonamiento, resultan significativamente más importantes para el razonamiento CoT efectivo que la validez lógica estricta de cada paso intermedio. Estos hallazgos profundizan sustancialmente la comprensión del prompting CoT y plantean nuevas interrogantes respecto a la capacidad de los LLMs para aprender a razonar en contexto, sugiriendo que los modelos pueden estar utilizando las demostraciones CoT de maneras fundamentalmente diferentes a como inicialmente se conceptualizó.

Investigaciones recientes han explorado extensiones y variantes avanzadas de CoT que abordan limitaciones específicas del enfoque original. Jia et al. (2025) presentaron la generación de prompts mediante Chain-of-Thought guiado por conocimiento, abordando el problema de que los LLMs frecuentemente producen resultados inexactos cuando enfrentan tareas de razonamiento complejo, potencialmente debido a conocimiento insuficiente y pobre rendimiento en tiempo real que resulta en cadenas de inferencia incorrectas. Inspirados por el aprendizaje profundo aumentado con conocimiento y la generación aumentada por recuperación, los autores proponen que un enfoque más viable es la generación de prompting chain-of-thought guiado por conocimiento, que introduce grandes cantidades de conocimiento, incluyendo información común, lógica y fáctica, en el proceso de generar una cadena de razonamiento. Este trabajo identifica una brecha crítica en la literatura de surveys sobre generación de prompts chain-of-thought guiado por conocimiento y proporciona un marco sistemático para integrar bases de conocimiento externas en el proceso de razonamiento.

Limitaciones Actuales

Múltiples autores señalan desafíos persistentes relacionados con el sesgo (bias) en los LLMs, la escalabilidad en conjuntos de datos masivos y la privacidad de la información. Por

ejemplo, Kapugama Geeganage et al. (2024), discuten las dificultades de la validación humana en procesos automatizados. Existe una brecha notable en la integración multimodal y el enriquecimiento en tiempo real que sea a la vez preciso y eficiente computacionalmente. Las direcciones futuras y oportunidad de investigación según un enfoque híbrido, propuesto por Murdock (2023), que sugiere combinar la potencia de la IA con la supervisión humana (human-in-the-loop).

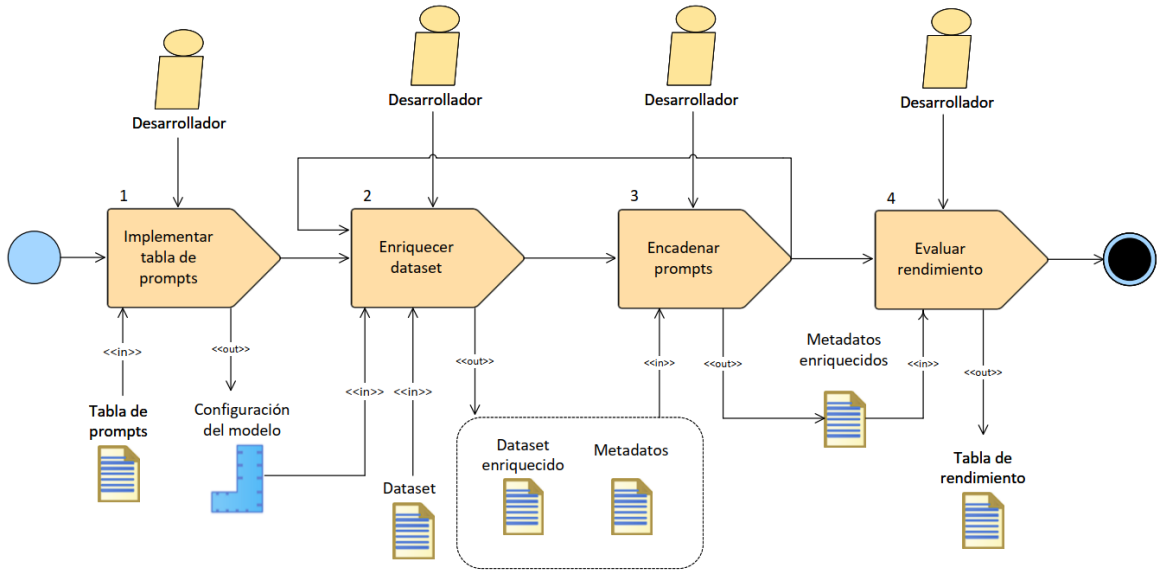
3. Métodos

El presente capítulo describe el diseño experimental y el sistema implementado para el enriquecimiento del conjunto de datos mediante cadena de prompts. El estudio adoptó un diseño experimental con grupos de comparación definidos por tres variables independientes: 1) el nivel de enriquecimiento aplicado al data set, con cuatro valores: sin enriquecimiento, Iteración 3, Iteración 6 e Iteración 9; 2) el modelo de lenguaje evaluado, con cinco instancias seleccionadas según criterios de diversidad de escala paramétrica y accesibilidad mediante APIs públicas; y 3) el tipo de consulta SQL, con dos niveles (simple y avanzada). Las variables dependientes fueron: (a) la efectividad, definida como la capacidad del modelo de producir una respuesta formalmente válida en términos de sintaxis y semántica SQL, expresada como valor binario (0% o 100%); y (b) la eficacia, definida como la diferencia porcentual entre el número de filas devuelto por el modelo y el número de filas de referencia validado mediante el motor DuckDB. La condición de control fue el grupo sin enriquecimiento, que estableció la línea base de desempeño de cada modelo ante el dataset sin contexto adicional. Todas las consultas fueron ejecutadas y verificadas mediante DuckDB previamente a la evaluación.

Para el enriquecimiento del conjunto de datos utilizando modelos de lenguaje grandes se implementó un programa escrito en Python mediante Google Colab. El proceso utilizado se representa mediante el diagrama SPEM (Software Process Engineering Metamodel) un estándar definido por el Object Management Group (OMG) que proporciona un marco para modelar, documentar y gestionar procesos de desarrollo de software (véase la Figura 1). El sistema se desarrolló mediante cinco fases: 1) Implementar tabla de prompts, 2) Enriquecer archivo, 3) Enriquecer datos, 4) Encadenar prompts y 5) Evaluar rendimiento. Cada fase es un proceso implementado en el programa.

Figura 1

Diagrama SPEM del Sistema



3.1 Implementar Tabla de Prompts

La primera fase del proceso consistió en la adquisición del conjunto de datos denominado COMEX_DATA_AZUAY_banco_central_2022.csv, el cual se ingresó en una estructura de datos llamada "data frame", de la librería Pandas. El data set inicialmente contenía un total de 94,018 entradas y 78 columnas. Se identificó que la mayoría de las columnas eran de tipo object (cadena de texto).

Se desarrolló una tabla de prompts en un archivo llamado cop_table.xlsx, el cual se ingresó al programa a través de la librería Pandas. Esta tabla se utilizó en el programa para implementar la cadena de prompts y guiar al LLM de manera iterativa. La tabla se compone de seis columnas: 1) Id, 2) Iteración, 3) Input, 4) Prompt, 5) Output y 6) Descripción.

El programa recorrió cada fila de la tabla (la cual se puede consultar en el Anexo 2), ingresando el prompt descrito al LLM (de la columna "Prompt"), pero reemplazando las variables que en la tabla se describen entre llaves "{}", de manera que el prompt que el modelo recibe, ya contiene los valores de las variables que corresponden. Estas variables se listan en la columna "Input". Después, la respuesta del modelo se estructuró en un archivo JSON, cuyo nombre se especifica en la columna "Output"; de este modo, en iteraciones siguientes se insertaron estas respuestas en el prompt que sigue. La columna "Iteración" simplemente enumera las iteraciones en orden ascendente. La columna "Descripción" no se

utiliza en el programa, simplemente es una anotación sobre el objetivo de la iteración.

3.2 Enriquecer Data Set

La implementación de esta fase se realizó utilizando la librería huggingface_hub de Python, que permite el acceso a modelos pre entrenados de última generación. El modelo utilizado fue Llama-3.1-8B-Instruct. El enriquecimiento se ejecutó de manera iterativa, procesando las filas de la tabla de prompts, de manera que el LLM, a través de una instrucción de prompt específica de la tabla, generó un resultado enriquecido y estructurado, en formato JSON por cada iteración de la tabla.

Para acceder a los modelos de la API de Hugging Face dentro del entorno de Google Colab, es necesario tener una cuenta de Hugging Face y luego crear un "Access Token" dentro de la web https://huggingface.co/settings/tokens con el permiso de escritura (write, en la opción "Token type") y el nombre "HUGGING_FACE_HUB_TOKEN" en la opción "Token name", después de lo cual es muy importante anotar el valor del Token que la página despliega luego de dar click en el botón "Create token". Después, en la web del modelo (https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct) se requiere pedir acceso al modelo. Una vez el usuario ha sido concedido con el acceso, en el entorno de Colab, se crea una variable en la opción de "Secrets" del panel izquierdo, donde el nombre se coloca "HUGGING_FACE_HUB_TOKEN" y el valor es el mismo valor del "Token" creado en la web de Hugging Face (https://huggingface.co/settings/tokens). Para acceder a las variables de Secrets, se utiliza el método userdata.get('HUGGING_FACE_HUB_TOKEN'), donde el argumento es el nombre de la variable. De esta manera el método devuelve el valor ingresado anteriormente en Secrets.

El programa inicializa un cliente de inferencia medinte el método InferenceClient(token=HF_TOKEN), donde la variable "Token" toma el valor de la variable que se configuró en Secrets anteriormente. Después se utilizó una función llamada llamar_llm(prompt, max_tokens=1500) donde el modelo se inicializa y se opera para utilizarlo en las iteraciones sobre la tabla de prompts. Para consultar la implementación, el código completo del sistema se encuentra en el Anexo 3

3.3 Encadenar Prompts

Mediante los resultados en los archivos JSON, cada iteración posterior (que en la tabla de prompts se describe desde el número 2) utilizó las respuestas generadas anteriormente para incorporar datos nuevos, tales como nombres de columnas y una muestra estratificada de valores únicos. De este modo, el modelo genera metadatos con información que él mismo produjo en fases previas, acumulando progresivamente capas de contexto semántico. Este diseño se fundamenta en el principio de cadena de prompts descrito por Ding et al. (2024), según el cual la provisión de una secuencia de prompts guía al LLM para producir una respuesta deseada de forma coherente y contextualmente rica, al tiempo que incrementa la transparencia del sistema, su controlabilidad y su confiabilidad. La elección de nueve iteraciones como límite del proceso de enriquecimiento respondió a criterios de exhaustividad y trazabilidad: las primeras tres iteraciones (Iteraciones 1 a 3) abordan la comprensión general del data set a partir del nombre del archivo, los nombres de columnas y una muestra estratificada de datos; las iteraciones 4 a 6 refinan esa comprensión integrando una muestra representativa de datos; y las iteraciones 7 a 9 sintetizan el contexto acumulado en una descripción experta y estructurada del data set. Los tres puntos de evaluación (Iteración 3, Iteración 6 e Iteración 9) permiten comparar el efecto de cada nivel de madurez del contexto sobre la eficacia de los modelos evaluadores, siguiendo así la lógica de medición incremental.

3.4 Evaluar Rendimiento

Para evaluar el impacto del enriquecimiento sobre la capacidad de comprensión de los LLMs, se seleccionaron cinco modelos que representan rangos paramétricos deliberadamente contrastantes. Se incluyeron dos modelos de gran escala (Llama-3.1-8B-Instruct, de 8 mil millones de parámetros y Llama-3.3-70B-versatile, de 70 mil millones de parámetros), accedidos a través de las APIs de Hugging Face y Groq respectivamente, con el propósito de establecer un referente de alto rendimiento. Se incluyeron además tres modelos de escala reducida (Llama-3.2-1B-Instruct, de mil millones de parámetros; Qwen2.5-1.5B-Instruct, de 1.5 mil millones de parámetros y ERNIE-4.5-0.3B, de 0.3 mil millones de parámetros), ejecutados de forma local mediante la librería Transformers, para evaluar si el contexto generado por el enriquecimiento es capaz de compensar la menor capacidad paramétrica de estos modelos. La incorporación de modelos de arquitecturas y familias distintas (Llama, Qwen, ERNIE) permite que los hallazgos no queden restringidos a una sola familia de LLMs, aportando mayor generalidad a los resultados. Después, se

seleccionaron seis consultas SQL que se validaron mediante el paquete DuckDB, para las cuales se pueden consultar en el Anexo 3. El número de seis consultas (tres simples y tres avanzadas) responde a la necesidad de cubrir dos niveles de complejidad sintáctica con al menos tres instancias por nivel, lo que garantiza que los resultados no dependan de una consulta particular sino que reflejen un patrón consistente. Las consultas simples se construyeron sin subconsultas, utilizando únicamente las sentencias Select, Where, Group By y Order By, y operan sobre columnas cuya identificación semántica es directa a partir del nombre. Las consultas avanzadas se construyeron con subconsultas, expresiones de tabla comunes (CTEs), funciones de ventana y operaciones de conversión de tipo (TRY_CAST, CASE, THEN, ELSE), lo que exige al modelo una comprensión más profunda de la estructura y el dominio del data set para producir estimaciones de resultado plausibles. Esta distinción permite observar si el efecto del enriquecimiento varía según la complejidad de la tarea, una variable de análisis fundamental para interpretar la utilidad práctica del sistema. El objetivo de cada consulta está descrito en la Tabla 1. Estas consultas se introdujeron a los LLMs junto con la petición del resultado a la consulta, mediante el prompt del Anexo 4. Además, se usó un prompt que no contiene el contexto del enriquecimiento para evaluar ambas muestras, que se encuentra en el Anexo 5.

Tabla 1

Descripción del Objetivo de las Consultas SQL

Numero	Complejidad	Objetivo de la consulta
1	Simple	Listar los 10 países de origen con más operaciones registradas
2	Simple	Mostrar las 10 empresas (por razón social) con mayor valor FOB exportado/importado en total, incluyendo también su número de operaciones.
3	Simple	Contar cuántas operaciones hay por año y régimen aduanero, mostrando los 50 resultados más recientes.
4	Avanzado	Calcular el CIF total y kilos netos por empresa, las jerarquiza a nivel nacional por CIF, y muestra las 20 más grandes que superen los 100.000 en CIF.
5	Avanzado	Para cada provincia, calcular cuánto aportó cada empresa al CIF provincial y qué porcentaje representa del total de su provincia. Filtra empresas con más de 50.000 en CIF.
6	Avanzado	Generar un resumen mensual con número de operaciones, FOB total y promedio por mes/año, y clasifica cada mes como de actividad "Alta" o "Baja" según si supera el promedio general histórico.

Los modelos respondieron a las consultas y el número de filas que resultó, se comparó con la verificación previa del DuckDB. Se utilizó una diferencia absoluta entre la cantidad de filas generadas tanto por los modelos, y la cantidad de filas generadas por el DuckDB. Después se calculó la diferencia porcentual, mediante la siguiente fórmula:

$$Diferencia\ porcentual = (\frac{Diferencia\ aboluta}{Número\ Filas\ del\ DuckDB}) \times 100$$

Los resultados se organizaron en una tabla comparativa con las siguientes columnas:

1. Consulta: se utilizaron dos tipos de consultas: una avanzada y una simple. Esta columna describe cuál de las dos consultas se evalúa.
2. Nombre_Modelo: nombre por el cuál se accede al modelo, que a su vez se acceden por las API tanto de HuggingF ace como de Groq.
3. Parámetros (Billones): La cantidad en billones de parámetros de los cuales está construido el modelo.
4. Con_Enriquecimiento: se refiere a tres posibles parámetros:
    a. Sin Enriquecimiento: el modelo no tiene contexto alguno.
    b. Iteración 3: el modelo tiene contexto hasta la iteración 3 de la tabla de prompts.
    c. Iteración 6: el modelo tiene contexto hasta la iteración 6 de la tabla de prompts.
    d. Iteración 9: el modelo tiene contexto hasta la iteración 9 de la tabla de prompts, es decir, el contexto completo.
5. Efectividad_%: se refiere a dos valores: 0% cuando el modelo no respondió a la consulta, y 100% cuando respondió con a la consulta correctamente en cuanto a sintaxis y semántica SQL.
6. Eficacia_%: diferencia porcentual entre el modelo y el DuckDB, como se describió con la fórmula.

4. Resultados

La evaluación se realizó sobre un total de 105 registros distribuidos entre cinco modelos de lenguaje, cuyo tamaño de parámetros los clasifica entre modelos grandes y pequeños: 1) Llama-3.1-8B-Instruct (grande, con 8 billones de parámetros), Llama-3.3-70B-versatile (grande, con 70 billones de parámetros), Llama-3.2-1B-Instruct (pequeño, con 1 billón de parámetros), ERNIE-4.5-0.3B (pequeño, con 0.3 billones de parámetros) y Qwen2.5-1.5B-

Instruct (pequeño, con 1.5 billones de parámetros). Los modelos fueron evaluados bajo dos tipos de consultas SQL (simple y avanzada) y en cuatro condiciones de enriquecimiento del conjunto de datos: sin enriquecimiento, Iteración 3, Iteración 6 e Iteración 9, donde cada iteración representa el número de fases de enriquecimiento aplicadas mediante prompts especializados.

Un primer hallazgo es que la efectividad alcanzó el 100% en todos los modelos, bajo todas las condiciones de enriquecimiento y en ambos tipos de consulta. Esto indica que ningún modelo incurrió en fallos de generación, lo que permite centrar el análisis comparativo exclusivamente en la eficacia, métrica que mostró que el tamaño del modelo influye levemente; la complejidad de la consulta influye mucho más, debido a que en las consultas avanzadas la eficacia cae considerablemente. El enriquecimiento tiene un efecto inconsistente en la eficacia, sin embargo, en la mitad de los casos de esta observación, es beneficioso, lo que se puede consultar en la Tabla 2, que contiene un resumen de los resultados. La tabla completa con los resultados, se encuentra en el Anexo 7.

Tabla 2

Resumen de los Resultados

Modelo	Consulta	Nivel de Enriquecimiento	Eficacia Promedio (%)
ERNIE-4.5-0.3B (Local)	simple	Sin Enriquecimiento	41.00
ERNIE-4.5-0.3B (Local)	simple	Iteración 3	54.00
ERNIE-4.5-0.3B (Local)	simple	Iteración 6	29.00
ERNIE-4.5-0.3B (Local)	simple	Iteración 9	20.00
ERNIE-4.5-0.3B (Local)	advanced	Sin Enriquecimiento	56.38
ERNIE-4.5-0.3B (Local)	advanced	Iteración 3	46.91
ERNIE-4.5-0.3B (Local)	advanced	Iteración 6	29.62
ERNIE-4.5-0.3B (Local)	advanced	Iteración 9	67.74
Llama-3.1-8B	simple	Sin Enriquecimiento	92.67
Llama-3.1-8B	simple	Iteración 3	74.67
Llama-3.1-8B	simple	Iteración 6	90.67
Llama-3.1-8B	simple	Iteración 9	86.67
Llama-3.1-8B	advanced	Sin Enriquecimiento	52.53
Llama-3.1-8B	advanced	Iteración 3	34.75
Llama-3.1-8B	advanced	Iteración 6	44.62
Llama-3.1-8B	advanced	Iteración 9	32.12
Llama-3.2-1B (HF)	simple	Sin Enriquecimiento	100.00
Llama-3.2-1B (HF)	simple	Iteración 3	100.00
Llama-3.2-1B (HF)	simple	Iteración 6	30.00
Llama-3.2-1B (HF)	simple	Iteración 9	10.00
Llama-3.2-1B (HF)	advanced	Sin Enriquecimiento	9.09
Llama-3.2-1B (HF)	advanced	Iteración 3	59.09
Llama-3.2-1B (HF)	advanced	Iteración 6	4.54
Llama-3.2-1B (HF)	advanced	Iteración 9	72.72
Llama-3.3-70B (Groq)	simple	Sin Enriquecimiento	83.00
Llama-3.3-70B (Groq)	simple	Iteración 3	86.50
Llama-3.3-70B (Groq)	simple	Iteración 6	62.00
Llama-3.3-70B (Groq)	simple	Iteración 9	66.67
Llama-3.3-70B (Groq)	advanced	Sin Enriquecimiento	40.19
Llama-3.3-70B (Groq)	advanced	Iteración 3	48.54
Llama-3.3-70B (Groq)	advanced	Iteración 6	44.89
Llama-3.3-70B (Groq)	advanced	Iteración 9	39.63
Qwen2.5-1.5B (Local)	simple	Sin Enriquecimiento	100.00
Qwen2.5-1.5B (Local)	simple	Iteración 3	11.00
Qwen2.5-1.5B (Local)	simple	Iteración 6	56.00
Qwen2.5-1.5B (Local)	advanced	Sin Enriquecimiento	40.72
Qwen2.5-1.5B (Local)	advanced	Iteración 6	18.18
Qwen2.5-1.5B (Local)	advanced	Iteración 9	9.79
Qwen2.5-1.5B (Local)	simple	Iteración 9	N/E*
Qwen2.5-1.5B (Local)	advanced	Iteración 3	N/E*

Nota: N/E: No evaluado. Las combinaciones Qwen2.5-1.5B — simple/Iteración 9 y Qwen2.5-1.5B — advanced/Iteración 3 no fueron ejecutadas durante el experimento debido a limitaciones de

disponibilidad del modelo local en esas condiciones. Los demás escenarios de la Tabla 2 corresponden a la totalidad de registros evaluados para cada modelo según el Anexo *7*.

La comparación directa entre la condición sin enriquecimiento y la Iteración 9 (que representa el máximo nivel de enriquecimiento del sistema) constituye el hallazgo central de la evaluación. El análisis por modelo y tipo de consulta revela que, en 7 de los 9 escenarios con datos completos, el enriquecimiento máximo produjo una desmejora de la eficacia respecto a la línea base; únicamente en 2 casos se observó una mejora. A continuación, se describe el impacto por modelo. En Llama-3.1-8B-Instruct, el enriquecimiento hasta la Iteración 9 redujo la eficacia en ambos tipos de consulta: de 92,67% a 86,67% en consultas simples (-6,00%) y de 52,53% a 32,12% en consultas avanzadas (-20,41%). En Llama-3.3-70B-versatile, la desmejora fue de 83,00% a 66,67% en consultas simples (-16,33%), mientras que en consultas avanzadas la diferencia fue mínima de 40,19% a 39,63% (-0,56%), configurando el único caso de efecto prácticamente nulo. En Llama-3.2-1B-Instruct se observa el patrón más polarizado: en consultas avanzadas el enriquecimiento produjo la mejora más pronunciada del experimento, de 9,09% a 72,72% (+63,63%); en cambio, en consultas simples la desmejora fue la mayor observada, de 100% a 10% (-90%). En ERNIE-4.5-0.3B, el enriquecimiento benefició las consultas avanzadas de 56,38% a 67,74% (+11,36%) pero perjudicó las simples, de 41,00% a 20,00% (-21%). Finalmente, en Qwen2.5-1.5B-Instruct, el enriquecimiento hasta la Iteración 9 redujo la eficacia en consultas avanzadas de 40,72% a 9,79% (-30,93%); la combinación simple/Iteración 9 no fue ejecutada durante el experimento. En síntesis, el enriquecimiento a su nivel máximo tiende a ser perjudicial en consultas simples (4 desmejoras de 4 casos evaluados) y presenta resultados mixtos en consultas avanzadas (2 mejoras y 3 desmejoras), lo que sugiere que el contexto acumulado en las nueve iteraciones resulta excesivo o introduce ruido que deteriora la capacidad inferencial del modelo en tareas de baja complejidad sintáctica.

La comparación entre la condición sin enriquecimiento y la Iteración 9 (que representa el máximo nivel de contexto generado) constituye el hallazgo central de la evaluación del sistema. De las nueve combinaciones modelo–tipo de consulta evaluadas, el enriquecimiento completo produjo una mejora en 2 casos (22%), una desmejora en 6 casos (67%) y no produjo efecto observable en 1 caso (11%). El análisis desagregado por modelo revela patrones diferenciados. Llama-3.1-8B-Instruct experimentó una desmejora en ambos tipos de consulta: su eficacia en consultas simples pasó del 92,67% (sin enriquecimiento) al

86,67% (Iteración 9), una reducción de 6 puntos porcentuales; en consultas avanzadas, la caída fue más pronunciada, de 52,53% a 32,12% (−20,41%). Llama-3.3-70B-versatile mostró desmejora en consultas simples de 83,00% a 66,67% (−16,33%) y prácticamente ningún cambio en consultas avanzadas de 40,19% a 39,63% (−0,56%). Llama-3.2-1B-Instruct presentó el comportamiento más contrastante: en consultas simples sufrió la mayor desmejora del experimento de 100% a 10% (−90%), mientras que en consultas avanzadas registró la mayor mejora de 9,09% a 72,72% (+63,63%). ERNIE-4.5-0.3B mostró desmejora en consultas simples de 41,00% a 20,00% (−21%) y mejora en consultas avanzadas de 56,38% a 67,74% (+11,36%). Qwen2.5-1.5B-Instruct, único modelo sin evaluación completa en todos los escenarios, registró desmejora en la combinación evaluable de consultas avanzadas de 40,72% a 9,79% (−30,93%). En conjunto, estos resultados indican que el enriquecimiento completo beneficia principalmente a los modelos pequeños en tareas de alta complejidad sintáctica, mientras que tiende a perjudicar el desempeño en consultas simples independientemente del tamaño del modelo.

La Tabla 3 resume la eficacia promedio obtenida por cada modelo a lo largo de todos los escenarios de enriquecimiento.

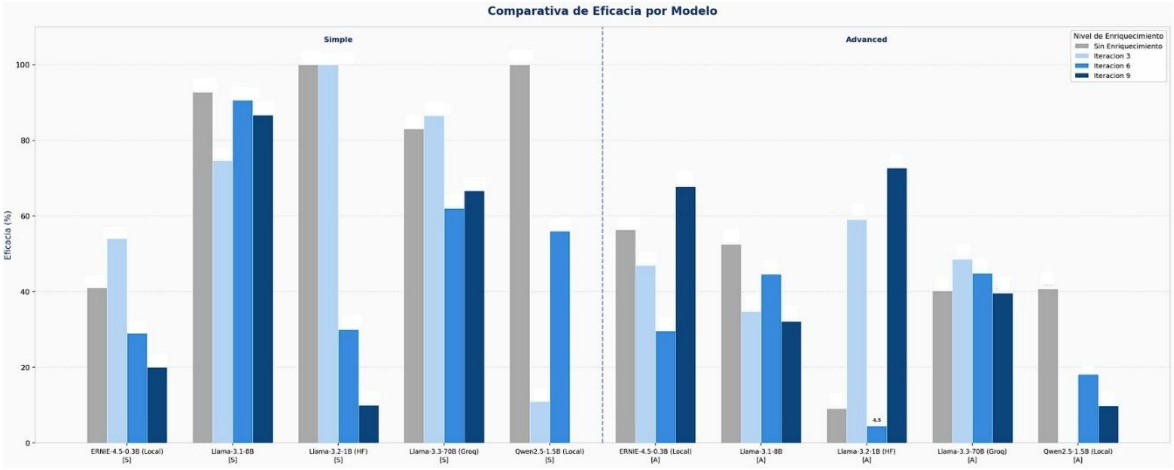
Tabla 3
Resumen del Rendimiento de los Modelos

Modelo	Eficacia Promedio (%)	Registros evaluados
Llama-3.1-8B-Instruct (Grande)	60,09	29
Llama-3.3-70B-versatile (Grande)	57,29	34
ERNIE-4.5-0.3B (Pequeño)	45,79	19
Llama-3.2-1B-Instruct (Pequeño)	44,24	12
Qwen2.5-1.5B-Instruct (Pequeño)	33,00	11

El modelo Llama-3.1-8B-Instruct obtuvo el mayor rendimiento global con un 60,09% de eficacia promedio, seguido de Llama-3.3-70B-versatile con un 57,29%. En el extremo opuesto, Qwen2.5-1.5B-Instruct resultó ser el modelo con menor desempeño general, alcanzando únicamente un 33,00%. Es relevante señalar que el tamaño del modelo en billones de parámetros no constituye un predictor directo de la eficacia: ERNIE-4.5-0.3B, con apenas 0,3 billones de parámetros, superó en eficacia a Llama-3.2-1B-Instruct (1B de parámetros) por un margen de 1,55 puntos porcentuales. Los resultados de la eficacia de los modelos, se representaron mediante la Figura 2.

Figura 2

Resumen de Resultados



5. Discusión

Los resultados del presente trabajo evidencian que el sistema de enriquecimiento implementado mediante cadena de prompts logra su objetivo fundamental: dotar a los modelos de lenguaje de un contexto suficiente para operar sobre un conjunto de datos de dominio específico (el comercio exterior del Azuay 2022) y producir respuestas a consultas SQL con coherencia sintáctica y semántica en el 100% de los casos. Este hallazgo, reflejado en la efectividad perfecta de todos los modelos bajo todas las condiciones, representa en sí mismo una confirmación de que la arquitectura de cadena de prompts es una técnica viable y robusta para guiar a los LLMs en tareas de comprensión de datos estructurados. Los resultados evidencian que la arquitectura de cadena de prompts implementada cumple su objetivo fundamental de viabilidad: en el 100% de los escenarios evaluados (cinco modelos, cuatro condiciones de enriquecimiento, dos tipos de consulta), todos los modelos produjeron respuestas formalmente válidas en sintaxis y semántica SQL. Sin embargo, la eficacia (medida como la proximidad entre los resultados del modelo y los valores de referencia obtenidos con DuckDB) muestra que la viabilidad del sistema no equivale a su eficacia universal. El análisis comparativo entre la condición base (sin enriquecimiento) y el máximo enriquecimiento (Iteración 9) revela que, en 7 de los 9 escenarios con datos disponibles, el enriquecimiento máximo produjo una desmejora de la eficacia, y únicamente en 2 casos (ERNIE-4.5-0.3B en consultas avanzadas, +11,36% y Llama-3.2-1B-Instruct en consultas avanzadas, +63,63%) representó una mejora. Este resultado, aunque contraintuitivo, resulta interpretable a la luz de la naturaleza de la tarea de evaluación: los modelos no ejecutan las

consultas sobre los datos reales, sino que deben estimar los resultados a partir del contexto semántico generado, lo que introduce un límite inherente a la utilidad del enriquecimiento independientemente de su profundidad.

El hallazgo más robusto del experimento es la incidencia del tipo de consulta sobre la eficacia. En la condición sin enriquecimiento, las consultas simples obtuvieron un promedio de eficacia del 81,09%, frente al 42,92% de las consultas avanzadas (una diferencia de 38,17 puntos porcentuales sobre la misma condición base). Esta asimetría se mantuvo en todos los niveles de enriquecimiento y en todos los modelos, lo que indica que la complejidad sintáctica de la consulta constituye el determinante primario de la eficacia del sistema, por encima del nivel de enriquecimiento o de la escala paramétrica del modelo. Las consultas simples operan sobre columnas de identificación directa (PAIS_ORIGEN, REGIMEN, RAZON_SOCIAL), cuyos nombres son suficientemente descriptivos para que un modelo pre entrenado en español infiera su contenido sin contexto adicional. Las consultas avanzadas, en cambio, involucran CTEs, funciones de ventana y conversiones de tipo que requieren no solo el nombre de las columnas sino una comprensión profunda de la distribución de los datos. Este resultado es congruente con los hallazgos de Meincke et al. (2025), quienes demostraron que las técnicas de prompting aportan mayor valor diferencial cuando la tarea supera la capacidad autónoma del modelo.

Una comparación central de este trabajo es la que se establece con el método de cadena de prompts propuesto por Giner-Miguelez et al. (2024). Ambos trabajos comparten el encadenamiento de prompts y el objetivo de enriquecer información asociada a data sets; sin embargo, difieren fundamentalmente en el objeto sobre el que opera el enriquecimiento. En Giner-Miguelez et al. (2024), el insumo es la documentación existente de los data sets (textos no estructurados que los acompañan), y el objetivo es extraer dimensiones clave de metadatos predefinidas. El método reporta una precisión del 81,21% con GPT-3.5 sobre 12 papers de data sets científicos con documentación disponible. En la presente tesis, el insumo es el data set mismo: nombres de columnas, nombre del archivo y muestra estratificada de datos. No existe documentación previa; el sistema infiere y construye la información contextual de forma autónoma. Esta diferencia es estructuralmente significativa: el método aquí propuesto puede aplicarse a cualquier data set, incluso los carentes de metadatos formales, como COMEX_DATA_AZUAY_banco_central_2022.csv. El propósito también difiere: Giner-Miguelez et al. (2024) orientan el enriquecimiento a facilitar la búsqueda e

interoperabilidad de data sets en repositorios de aprendizaje automático, mientras que este trabajo lo orienta a habilitar a modelos de menor escala para responder consultas sobre datos especializados.

La heterogeneidad observada entre modelos ante el enriquecimiento aporta evidencia sobre el rol de la arquitectura y la escala paramétrica. Llama-3.2-1B-Instruct presentó la mayor variabilidad del experimento: en consultas avanzadas, el enriquecimiento hasta la Iteración 9 elevó la eficacia de 9,09% a 72,72% (+63,63%); en consultas simples la redujo de 100% a 10% (-90%). Esta polaridad extrema sugiere que los modelos de menor capacidad aprovechan el contexto cuando la tarea lo requiere, pero sufren degradación cuando la tarea ya era resoluble sin él. Este comportamiento es congruente con lo documentado por Schulhoff et al. (2025), quienes reportan que pequeñas variaciones en el contenido o volumen del prompt pueden producir cambios de precisión superiores al 40%, siendo los modelos de menor capacidad los más susceptibles. Llama-3.3-70B-versatile, por su parte, mostró el comportamiento más predecible: una mejora discreta en consultas avanzadas con Iteración 3, de 40,19% a 48,54% (+8,35%) y un declive progresivo en iteraciones posteriores. Este patrón apunta a un punto de equilibrio entre cantidad de contexto y utilidad marginal, coherente con el riesgo de propagación de alucinaciones en pipelines iterativos señalado por Xu et al. (2025), donde la salida de una iteración se convierte en insumo de la siguiente.

El presente estudio presenta cuatro limitaciones que deben ser consideradas al interpretar y generalizar sus resultados. En primer lugar, la métrica de eficacia adoptada (diferencia porcentual en la cardinalidad del resultado respecto a DuckDB) constituye una aproximación parcial a la calidad de la respuesta: captura únicamente la capacidad del modelo para estimar el número de filas, pero no evalúa si los valores específicos devueltos son correctos, si el ordenamiento es adecuado o si los cálculos agregados son precisos. Futuras investigaciones deberían complementar esta métrica con medidas de similitud semántica entre resultados o con evaluación de la corrección de valores individuales. En segundo lugar, el diseño iterativo de la cadena de prompts introduce un riesgo inherente de propagación de errores: cualquier imprecisión generada en las primeras iteraciones se incorpora como insumo de las siguientes, pudiendo amplificarse. Los datos ofrecen evidencia indirecta de este fenómeno: en Qwen2.5-1.5B-Instruct, la eficacia en consultas avanzadas cayó de 40,72% (sin enriquecimiento) a 9,79% (Iteración 9), consistente con la acumulación de contexto inexacto a lo largo de nueve

fases. En tercer lugar, el diseño experimental no incluyó un grupo de control con un método alternativo de enriquecimiento no basado en LLMs (por ejemplo, mediante reglas léxicas o diccionarios de dominio). En ausencia de este grupo de control, no es posible aislar la contribución específica de la cadena de prompts respecto a la del enriquecimiento en general: los resultados demuestran que el sistema es viable, pero no demuestran que su arquitectura sea superior a enfoques alternativos más simples. En cuarto lugar, el alcance del caso de estudio se limita a un único data set de dominio altamente especializado (comercio exterior del Azuay 2022), con nomenclatura técnica y 94.018 registros. Los hallazgos no pueden generalizarse directamente a data sets de otros dominios, otros idiomas o estructuras relacionales más simples. La replicación sobre data sets de distintos dominios y niveles de complejidad es una condición necesaria para establecer el alcance real del método propuesto.

6. Conclusión

El presente trabajo diseñó, implementó y validó un sistema basado en cadena de prompts capaz de enriquecer de forma autónoma un conjunto de datos estructurado sin requerir documentación preexistente, tratándose de una diferencia fundamental respecto al método de Giner-Miguelez et al. (2024), cuyo insumo son textos descriptivos que acompañan a los data sets. El sistema fue aplicado al data set de comercio exterior del Azuay del año 2022 y genera contexto semántico directamente a partir de los atributos del propio archivo, lo que lo hace aplicable a cualquier data set carente de metadatos formales.

Los cuatro objetivos específicos del trabajo quedaron satisfechos. El primero (análisis de la literatura) permitió identificar el encadenamiento de prompts, el prompting few-shot y el Chain of Thought como las técnicas más pertinentes para el diseño del sistema. El segundo y tercero (formulación e implementación del método) se cumplieron con la tabla de prompts de nueve iteraciones, cuya arquitectura acumula contexto progresivamente al utilizar las salidas de fases tempranas como insumo de fases posteriores. El cuarto (evaluación del desempeño) se respondió mediante la comparación de cinco modelos bajo seis consultas SQL en cuatro condiciones de enriquecimiento, con DuckDB como referencia de validación.

Los hallazgos principales son tres. Primero, la efectividad fue del 100% en todos los escenarios: todos los modelos produjeron respuestas sintácticamente válidas independientemente del nivel de enriquecimiento o la escala paramétrica. Segundo, el tipo de consulta es el factor de mayor incidencia sobre la eficacia, con una brecha de 38,17 puntos

porcentuales entre consultas simples (81,09%) y avanzadas (42,92%) en la condición base, que se mantuvo en todos los modelos y condiciones. Tercero, el enriquecimiento máximo (Iteración 9) produjo desmejora en 7 de los 9 escenarios evaluados y mejora en 2, lo que indica que el efecto del contexto acumulado es condicionado: beneficia a modelos pequeños en tareas de alta complejidad sintáctica, pero tiende a introducir ruido en consultas simples donde el modelo ya es capaz de responder sin contexto adicional.

Las cuatro limitaciones centrales del estudio son: (a) la métrica de eficacia mide únicamente la cardinalidad del resultado y no evalúa la corrección de valores individuales; (b) el diseño iterativo propaga errores entre fases sin mecanismos de validación intermedia; (c) la ausencia de un grupo de control con método alternativo no basado en LLMs impide aislar la contribución específica de la cadena de prompts; y (d) el caso de estudio se limita a un único data set en español de dominio especializado, lo que restringe la generalización directa de los hallazgos.

El potencial del sistema desarrollado trasciende el caso de estudio presentado. Su aplicación más directa e inmediata es el enriquecimiento automático de registros gubernamentales, bases de datos institucionales, sistemas transaccionales heredados y conjuntos de datos sectoriales que carecen históricamente de metadatos estructurados, una condición prevalente en contextos públicos y organizacionales de América Latina. El método es estructuralmente extensible a data sets en otros idiomas, con otras nomenclaturas de columnas y en otros dominios (salud, educación, finanzas, logística) mediante la simple adaptación de la tabla de prompts al dominio objetivo, sin modificaciones arquitecturales. Futuras investigaciones podrían explorar mecanismos de validación intermedia en la cadena, criterios de parada adaptativos que detengan el enriquecimiento antes de que el contexto acumulado introduzca ruido, y la comparación sistemática con métodos alternativos de enriquecimiento no basados en LLMs. El sistema constituye una contribución concreta y replicable al desafío de hacer accesibles, para modelos de lenguaje de cualquier escala, los datasets especializados que de otro modo permanecerían semánticamente opacos.

7. Referencias

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei,

D. (2020). *Language Models are Few-Shot Learners*. https://doi.org/10.48550/arXiv.2005.14165

Chen, B., Zhang, Z., Langrené, N., & Zhu, S. (2025). *Unleashing the potential of prompt engineering for large language models*. https://doi.org/10.1016/j.patter.2025.101260

Chu, Z., Chen, J., Chen, Q., Yu, W., He, T., Wang, H., Peng, W., Liu, M., Qin, B., & Liu, T. (2024). *Navigate through Enigmatic Labyrinth A Survey of Chain of Thought Reasoning: Advances, Frontiers and Future*. http://arxiv.org/abs/2309.15402

Ding, B., Qin, C., Liu, L., Chia, Y. K., Li, B., Joty, S., & Bing, L. (2022). *Is GPT-3 a Good Data Annotator?* https://doi.org/10.48550/arXiv.2212.10450

Ding, B., Qin, C., Zhao, R., Luo, T., Li, X., Chen, G., Xia, W., Hu, J., Luu, A. T., & Joty, S. (2024a). *Data Augmentation using Large Language Models: Data Perspectives, Learning Paradigms and Challenges*. http://arxiv.org/abs/2403.02990

Ding, B., Qin, C., Zhao, R., Luo, T., Li, X., Chen, G., Xia, W., Hu, J., Luu, A. T., & Joty, S. (2024b). *Data Augmentation using LLMs: Data Perspectives, Learning Paradigms and Challenges*.

Elazar, Y., Basmov, V., Goldberg, Y., & Tsarfaty, R. (2022). *Text-based NP Enrichment*. https://doi.org/10.1162/tacl

Giner-Miguelez, J., Gómez, A., & Cabot, J. (2024). *Using Large Language Models to Enrich the Documentation of Datasets for Machine Learning*. http://arxiv.org/abs/2404.15320

Jia, Z., Geng, S., Zhao, Y., & Zhang, H. (2025). Comprehensive Survey on Prompts Generating via Knowledge-Guided Chain-of-Thought. *International Journal of Crowd Science*, *9*(4), 251–261. https://doi.org/10.26599/IJCS.2024.9100038

Jurafsky, D., & Martin, J. H. (2008). *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition Second Edition*.

Kapugama Geeganage, D. T., Wynn, M. T., & Ter Hofstede, A. H. M. (2024). Text2EL+: Expert Guided Event Log Enrichment Using Unstructured Text. *Journal of Data and Information Quality*, *16*(1). https://doi.org/10.1145/3640018

Khan, N., Rahman, A. A. H. A., Masud Rizvi, S., Yousef Alshareef, I., Nadzir Marsono, M., Paend Bakht, M., Shahrizal Rusli, M., & Sadiah, S. (2025). Systematic Literature Review of Machine Learning Models and Applications for Text Recognition. In *IEEE Access* (Vol. 13, pp. 177647–177670). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2025.3618109

Knoth, N., Tolzin, A., Janson, A., & Leimeister, J. M. (2024). AI literacy and its implications for prompt engineering strategies. *Computers and Education: Artificial Intelligence*, *6*. https://doi.org/10.1016/j.caeai.2024.100225

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2023). *Large Language Models are Zero-Shot Reasoners*. http://arxiv.org/abs/2205.11916

Kuzman, T., & Ljubešić, N. (2025). Automatic genre identification: a survey. *Language Resources and Evaluation*, *59*(1), 537–570. https://doi.org/10.1007/s10579-023-09695-8

Li, J., Sun, A., Han, J., & Li, C. (2020). *A Survey on Deep Learning for Named Entity Recognition*. https://doi.org/10.1109/TKDE.2020.2981314

Liu, F., Huang, X., Huang, W., & Duan, S. X. (2020). Performance evaluation of keyword extraction methods and visualization for student online comments. *Symmetry*, *12*(11). https://doi.org/10.3390/sym12111923

Martinez-Rodriguez, J. L., Hogan, A., & Lopez-Arevalo, I. (2020). Information Extraction meets the Semantic Web: A Survey. In *Semantic Web* (Vol. 11, Number 2). IOS Press. https://doi.org/10.3233/SW-180333

Meincke, L., Mollick, E., Mollick, L., & Shapiro, D. (2025). *Prompting Science Report 2: The Decreasing Value of Chain of Thought in Prompting*. https://doi.org/10.48550/arXiv.2506.07142

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., & Zettlemoyer, L. (2022). *Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?*

Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., & Gao, J. (2025). *Large Language Models: A Survey*. http://arxiv.org/abs/2402.06196

Murdock, V. (2023). *Mixed Methods Machine Learning*. 3–4. https://doi.org/10.1145/3555041.3589337

Prastyo, P. A., Berlilana, & Tahyudin, I. (2025). Sentiment Analysis on Slang Enriched Texts Using Machine Learning Approaches. *Journal of Applied Data Sciences*, *6*(2), 1076–1087. https://doi.org/10.47738/jads.v6i2.626

Qin, L., Chen, Q., Feng, X., Wu, Y., Zhang, Y., Li, Y., Li, M., Che, W., & Yu, P. S. (2025). *Large Language Models Meet NLP: A Survey*. http://arxiv.org/abs/2405.12819

Roth, S., & Wermer-Colan, A. (2023). Machine Learning Methods for Systematic Reviews: A Rapid Scoping Review. *Delaware Journal of Public Health*, *9*(3). https://doi.org/10.32481/djph.2023.11.008

Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2025). *A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications*. http://arxiv.org/abs/2402.07927

Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Schulhoff, S., Dulepet, P. S., Vidyadhara, S., Ki, D., Agrawal, S., Pham, C., Kroiz, G., Li, F., Tao, H., Srivastava, A., … Resnik, P. (2025). *The Prompt Report: A Systematic Survey of Prompt Engineering Techniques*. http://arxiv.org/abs/2406.06608

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0197-0

Treviso, M., Lee, J.-U., Ji, T., Van Aken, B., Cao, Q., Ciosici, M. R., Hassid, M., Heafield, K., Hooker, S., Raffel, C., Martins, P. H., Martins, A. F. T., Forde, J. Z., Milder, P., Simpson, E., Slonim, N., Dodge, J., Strubell, E., Balasubramanian, N., … Schwartz, R. (2023). *Efficient Methods for Natural Language Processing: A Survey*. https://doi.org/10.1162/tacl

Wang, B., Min, S., Deng, X., Shen, J., Wu, Y., Zettlemoyer, L., & Sun, H. (2023). *Towards Understanding Chain-of-Thought Prompting: An Empirical Study of What Matters* (Vol. 1). Long Papers.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E. H., Narang, S., Chowdhery, A., & Zhou, D. (2023). *Self-Consistency Improves Chain of Thought Reasoning in Language Models*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. http://arxiv.org/abs/2201.11903

Xu, Z., Jain, S., & Kankanhalli, M. (2025). *Hallucination is Inevitable: An Innate Limitation of Large Language Models*. http://arxiv.org/abs/2401.11817

Yu, Z., He, L., Wu, Z., Dai, X., & Chen, J. (2023). *Towards Better Chain-of-Thought Prompting Strategies: A Survey*. http://arxiv.org/abs/2310.04959

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., … Wen, J.-R. (2025). *A Survey of Large Language Models*. http://arxiv.org/abs/2303.18223

8. Anexos

Anexo A

Diccionario de datos del conjunto de datos "COMEX_DATA_AZUAY_banco_central_2022.csv"

Nombre del campo	Tipo	Descripción breve
ANIO	int	Año de la operación de comercio exterior.
MES	int	Mes de la operación de comercio exterior.
CAPITULO	int	Código del capítulo arancelario.
DESCRIPCION_CAPITULO	str	Descripción del capítulo arancelario.
PARTIDA_	int	Código de la partida arancelaria.
DESCRIPCION_PARTIDA	str	Descripción de la partida arancelaria.
RUC	str	Número de RUC de la empresa.
RAZON_SOCIAL	str	Razón social de la empresa.
RAZON_SOCIAL_DIRECCION	str	Dirección de la razón social.
REMITENTE	str	Nombre del remitente de la mercancía.
NOTIFY	str	Parte notificada en la operación.
EMBARCADOR_CONSIGNEE	str	Nombre del consignatario o embarcador.
EMBARCADOR_CONSIGNEE_ADDRESS	str	Dirección del consignatario o embarcador.
REFRENDO	str	Referencia o refrendo de la operación.
NUME_SERIE	int	Número de serie de la declaración.
TIPO_AFORO	str	Tipo de aforo aduanero aplicado.
COD_REGIMEN	int	Código del régimen aduanero.
REGIMEN	str	Descripción del régimen aduanero.
DISTRITO	str	Distrito aduanero de la operación.
AGENTE_AFIANZADO	str	Nombre del agente afianzado.
AGENCIA	str	Agencia involucrada en la operación.
LINEA	str	Línea naviera o de transporte.
MANIFIESTO	str	Número de manifiesto de carga.
MANIFIESTO_ADUANA	str	Número de manifiesto asignado por aduana.
BL	str	Número de conocimiento de embarque (Bill of Lading).
FECH_EMBAR	str	Fecha de embarque de la mercancía.
FECH_LLEGA	str	Fecha de llegada de la mercancía.
FECHA_INGRESO	str	Fecha de ingreso de la declaración.
FECH_PAGO	str	Fecha de pago de la declaración.
FECH_SALIDA	str	Fecha de salida de la mercancía.
FACTURA	str	Número de factura comercial.
NAVE	str	Nombre de la nave o medio de transporte.
ALMACEN_TEMPORAL	float	Código o nombre del almacén temporal. (Sin datos)
DEP_COMERCIAL	str	Depósito comercial involucrado.
SUBPARTIDA	int	Código de la subpartida arancelaria.
TNAN	int	Código TNAN (según nomenclatura nacional).
TASA_ADVALOREM	float	Tasa ad-valorem aplicada. (Sin datos)
DESC_ARAN	str	Descripción arancelaria del producto.
DESC_COMER	str	Descripción comercial del producto.
MARCAS	str	Marcas identificadoras de la mercancía.
CIUDAD_EMBARQUE	str	Ciudad de embarque de la mercancía.

PAIS_EMBARQUE	str	País de embarque de la mercancía.
PAIS_ORIGEN	str	País de origen de la mercancía.
TIPO_CARGA	float	Tipo de carga. (Sin datos)
UNIDADES	float	Cantidad de unidades de la mercancía.
TIPO_UNIDAD	str	Tipo de unidad de medida utilizada.
KILOS_NETO	float	Peso neto de la mercancía en kilogramos.
FOB	float	Valor FOB (Free On Board) en dólares.
FLETE	float	Valor del flete en dólares.
SEGURO	float	Valor del seguro en dólares.
CIF	float	Valor CIF (Cost, Insurance and Freight) en dólares.
CODIGO_LIBERACION	int	Código de liberación aduanera.
COD_LIBERACION	str	Descripción del código de liberación.
ADV_PAG_PARTIDA	float	Anticipo pagado por partida.
ADV_LIQ_PARTIDA	int	Anticipo liquidado por partida.
CARACTERISTICA	str	Característica adicional del producto.
PRODUCTO	str	Nombre o descripción del producto.
MARCA_COMERCIAL	str	Marca comercial del producto.
ANIO_PRODUCIDO	float	Año de producción del producto.
MODELO_MERCADERIA	str	Modelo de la mercadería.
FOB_UNITARIO	float	Valor FOB unitario.
VIA	str	Vía de transporte utilizada.
REGIMEN_TIPO	str	Tipo de régimen aduanero.
INCOTERM	str	Término INCOTERM de la operación.
CONSOLIDADORA	float	Código o nombre de la consolidadora. (Sin datos)
COD_PROVINCIA	float	Código de la provincia de la operación.
PROVINCIA	str	Nombre de la provincia de la operación.
FORMULARIO	float	Código de formulario. (Sin datos)
FORM_VIA_ENVIO	float	Código de vía de envío. (Sin datos)
ESTADO_MERCANCIA	str	Estado de la mercancía.
DIAS_SALIDA	float	Días hasta la salida de la mercancía.
FLETE2	float	Valor adicional de flete.
NUM_CIF2	float	Número adicional de CIF.
CFR	float	Valor CFR (Cost and Freight) en dólares.
ESTADO_DECLARACION	str	Estado de la declaración aduanera.
TIPO_REGIMEN	str	Tipo de régimen de la declaración.
PARTIDA_DESCRIPCION	str	Descripción de la partida arancelaria.
CAPITULO_DESCRIPCION	str	Descripción del capítulo arancelario.

Anexo B

Tabla de prompts

Id	Iteración	Input	Prompt	Output	Descripción
is11	1	{data_file}	Eres un experto en análisis de datos. Basado en el nombre del archivo de un dataset, deduce de qué se trata. Ejemplo 1: Nombre del archivo: ventas_2023.csv Respuesta esperada: El dataset parece contener datos sobre ventas realizadas durante el año 2023, posiblemente incluyendo información como fechas, productos, cantidades y montos. Ejemplo 2: Nombre del archivo: clima_ciudades.xlsx Respuesta esperada: El dataset trata sobre datos climáticos de varias ciudades, como temperaturas, precipitaciones y otras variables meteorológicas. Ahora, deduce de qué se trata el dataset cuyo nombre del archivo es {data_file}.	{resNombre.json}	Se deduce lo que contiene el archivo en base a su nombre
is12	2	{resNombre.json}, {df_columns}	Eres un experto en análisis de datos. Basándote en un análisis previo sobre un dataset y la lista de columnas, describe brevemente cada columna. Ejemplo 1: Análisis previo: El dataset contiene información sobre ventas de productos. Columnas: fecha, producto, cantidad, precio Respuesta esperada: fecha: Indica la fecha de la venta. producto: Nombre del producto vendido. cantidad: Número de unidades vendidas. precio: Monto por unidad del producto. Ejemplo 2: Análisis previo: Datos climáticos diarios. Columnas: ciudad, temperatura_max, temperatura_min, precipitacion Respuesta esperada: ciudad: Nombre de la ciudad. temperatura_max: Temperatura máxima registrada en el día. temperatura_min: Temperatura mínima registrada en el día. precipitacion: Cantidad de lluvia en mm. El análisis previo es el siguiente: {resNombre.json}. Las columnas del dataset son las siguientes: {df_columns}. Describe brevemente cada una.	{resColumns.json}	En base a los nombres de las columnas y la respuesta de la iteración 1, se deduce el concepto de cada columna

is13	3	{resColumns.json}, {dataSample}	Eres un experto en análisis de datos. Describe el concepto general de un dataset utilizando un análisis básico de columnas y una muestra de datos estratificada. Ejemplo 1: Análisis básico de columnas: id: Identificador único. nombre: Nombre del cliente. edad: Edad del cliente. Muestra de datos: id:1, nombre:Juan, edad:25; id:2, nombre:Maria, edad:30 Respuesta esperada: El dataset describe perfiles de clientes, incluyendo identificadores, nombres y edades, posiblemente para análisis demográficos. Ejemplo 2: Análisis básico de columnas: fecha: Fecha del evento. tipo: Tipo de evento. duracion: Duración en minutos. Muestra de datos: fecha:2023-01-01, tipo:Reunión, duracion:60; fecha:2023-01-02, tipo:Llamada, duracion:15 Respuesta esperada: El dataset registra eventos diarios, como reuniones y llamadas, con fechas, tipos y duraciones, útil para tracking de actividades. Utiliza la información siguiente: 1. Análisis básico sobre columnas: {resColumns.json}. 2. Muestra de datos estratificada: {dataSample}. Describe el concepto general del dataset.	{resConcept.json}	Utilizando una muestra estratiificada y la respuesta de la iteración 2, se deduce el concepto general del dataset
is21	4	{data_file}, {resConcept.json}, {resColumns.json}, {resNombre.json}	Eres un experto en análisis de datos. Utilizando un contexto generado anteriormente, describe breve y precisamente el concepto del dataset. Utilizando el contexto generado anteriormente: {resConcept.json}, {resColumns.json}, {resNombre.json}, describe breve y precisamente el concepto del dataset {data_file}.	{resNombre_i2.json}	Se utiliza contexto de la iteración 3, para refinar y optimizar la respuesta sobre el concepto del data set
is22	5	{resNombre_i2.json}, {df_columns}, {resColumns.json}, {resConcept.json}	Eres un experto en datos. Ahora que tienes un contexto sobre el dataset, analiza cada una de las columnas como un experto. El contexto es {resNombre_i2.json}, {resColumns.json}, {resConcept.json}. Las columnas del dataset son: {df_columns}. Analiza cada una.	{resColumns_i2.json}	Optimiza la iteración 2 con few-shot para análisis experto y estructurado de columnas.

Anexo C

Consultas SQL para evaluación

Numero	Complejidad	Consulta

1	Simple	SELECT
		  PAIS_ORIGEN,
		  COUNT(*) AS total_operaciones
		FROM df
		WHERE ANIO = 2022 -- Sin comillas, ANIO es numérico
		GROUP BY PAIS_ORIGEN
		ORDER BY total_operaciones DESC
		LIMIT 10;

2	Simple	SELECT
		  RAZON_SOCIAL,
		  COUNT(*) AS total_operaciones,
		  SUM(
		  TRY_CAST(
		  REPLACE(REPLACE(TRIM(FOB), '"', ''), ',', '.')
		  AS DOUBLE)
		  ) AS total_fob
		FROM df
		WHERE RAZON_SOCIAL IS NOT NULL
		  AND FOB IS NOT NULL
		  AND TRIM(FOB) != ''
		  AND TRY_CAST(REPLACE(REPLACE(TRIM(FOB), '"', ''), ',', '.') AS DOUBLE) IS NOT NULL
		GROUP BY RAZON_SOCIAL
		ORDER BY total_fob DESC
		LIMIT 10;

3	Simple	SELECT
		  ANIO,
		  REGIMEN,
		  COUNT(*) AS cantidad
		FROM df
		WHERE ANIO IS NOT NULL
		  AND REGIMEN IS NOT NULL
		  AND TRIM(REGIMEN) != ''
		GROUP BY ANIO, REGIMEN
		ORDER BY ANIO DESC, cantidad DESC
		LIMIT 50;

4	Avanzado	WITH DatosFormateados AS (
		  SELECT
		  RUC,
		  RAZON_SOCIAL,
		  -- Limpiar CIF: remover comillas, espacios, convertir comas a puntos
		  TRY_CAST(
		  REPLACE(REPLACE(TRIM(CIF), '"', ''), ',', '.')
		  AS DOUBLE) AS cif_numerico,
		  -- Limpiar KILOS_NETO
		  TRY_CAST(
		  REPLACE(REPLACE(TRIM(KILOS_NETO), '"', ''), ',', '.')
		  AS DOUBLE) AS kilos_numerico
		  FROM df
		  WHERE CIF IS NOT NULL
		  AND KILOS_NETO IS NOT NULL
		  -- Filtrar solo valores que sean convertibles a número
		  AND TRY_CAST(REPLACE(REPLACE(TRIM(CIF), '"', ''), ',', '.') AS DOUBLE) IS NOT NULL
		  AND TRY_CAST(REPLACE(REPLACE(TRIM(KILOS_NETO), '"', ''), ',', '.') AS DOUBLE) IS NOT NULL
		),
		AgrupacionEmpresas AS (
		  SELECT
		  RUC,
		  RAZON_SOCIAL,
		  SUM(cif_numerico) AS total_cif,
		  SUM(kilos_numerico) AS total_kilos
		  FROM DatosFormateados
		  WHERE cif_numerico > 0 -- Valores positivos
		  GROUP BY RUC, RAZON_SOCIAL
		)
		SELECT
		  RUC,
		  RAZON_SOCIAL,
		  total_cif,
		  total_kilos,
		  RANK() OVER(ORDER BY total_cif DESC) as ranking_nacional
		FROM AgrupacionEmpresas
		WHERE total_cif > 100000 -- Umbral más bajo por si 500k no tiene suficientes resultados
		ORDER BY ranking_nacional
		LIMIT 20;

5	Avanzado	WITH DatosPorEmpresa AS (
		  SELECT
		  PROVINCIA,

```
    RAZON_SOCIAL,
    SUM(
      TRY_CAST(
        REPLACE(REPLACE(TRIM(CIF), '"', ''), ',', '.')
      AS DOUBLE)
    ) AS cif_por_empresa
  FROM df
  WHERE PROVINCIA IS NOT NULL
   AND TRIM(PROVINCIA) != ''
   AND CIF IS NOT NULL
   AND TRY_CAST(REPLACE(REPLACE(TRIM(CIF), '"', ''), ',', '.') AS DOUBLE) IS NOT NULL
  GROUP BY PROVINCIA, RAZON_SOCIAL
),
TotalesProvincia AS (
  SELECT
    PROVINCIA,
    SUM(cif_por_empresa) AS total_provincia
  FROM DatosPorEmpresa
  GROUP BY PROVINCIA
)
SELECT
  d.PROVINCIA,
  d.RAZON_SOCIAL,
  d.cif_por_empresa,
  t.total_provincia,
  (d.cif_por_empresa / NULLIF(t.total_provincia, 0)) * 100 AS porcentaje_participacion
FROM DatosPorEmpresa d
JOIN TotalesProvincia t ON d.PROVINCIA = t.PROVINCIA
WHERE d.cif_por_empresa > 50000
ORDER BY d.PROVINCIA, porcentaje_participacion DESC
LIMIT 30;
```

6	Avanzado	WITH ResumenMensual AS (SELECT ANIO, MES, COUNT(*) AS num_operaciones, SUM(TRY_CAST(REPLACE(REPLACE(TRIM(FOB), '"', ''), ',', '.') AS DOUBLE)) AS total_fob,

```
      AVG(
        TRY_CAST(
          REPLACE(REPLACE(TRIM(FOB), '"', ''), ',', '.')
        AS DOUBLE)
      ) AS promedio_fob
    FROM df
    WHERE ANIO IS NOT NULL
     AND MES IS NOT NULL
     AND FOB IS NOT NULL
     AND TRY_CAST(REPLACE(REPLACE(TRIM(FOB), '"', ''), ',', '.') AS DOUBLE) IS NOT NULL
    GROUP BY ANIO, MES
)
SELECT
    ANIO,
    MES,
    num_operaciones,
    ROUND(total_fob, 2) AS total_fob,
    ROUND(promedio_fob, 2) AS promedio_fob,
    CASE
      WHEN total_fob > (SELECT AVG(total_fob) FROM ResumenMensual WHERE total_fob IS NOT NULL)
      THEN 'Alto'
      ELSE 'Bajo'
    END AS nivel_actividad
FROM ResumenMensual
WHERE total_fob IS NOT NULL
ORDER BY ANIO DESC, MES DESC
LIMIT 24;
```

Anexo D

Prompt con contexto utilizado en la fase de evaluación

"Eres un experto en análisis de datos con conocimiento profundo del dataset COMEX_DATA_AZUAY_banco_central_2022.csv.

CONOCIMIENTO DEL DATASET:
{contexto_enriquecimiento}

TAREA:
Proporciona el RESULTADO que produciría esta consulta SQL:
{consulta_sql}

INSTRUCCIONES:
- NO escribas código SQL
- Genera SOLO la tabla de resultados
- Formato: columna1 | columna2 | columna3
- Usa valores reales basados en tu conocimiento del dataset

Tu respuesta (SOLO la tabla):"

Anexo E

Prompt sin contexto utilizado en la fase de evaluación

"Eres un experto en análisis de datos.

Proporciona el RESULTADO de esta consulta SQL sobre el dataset COMEX_DATA_AZUAY_banco_central_2022.csv:
{consulta_sql}

INSTRUCCIONES:
- NO escribas código SQL
- Genera SOLO la tabla de resultados
- Formato: columna1 | columna2 | columna3

Tu respuesta (SOLO la tabla):"

Anexo F

Tabla de Resultados

Consulta	Nombre_Modelo	Parámetros (miles de millones)	Con_Enriquecimiento	Efectividad_%	Eficacia_%
simple	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	80
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	100
simple	Llama-3.2-1B-Instruct (HF)	1	Sin Enriquecimiento	100	100
simple	Llama-3.2-1B-Instruct (HF)	1	Iteracion 3	100	100
simple	Llama-3.2-1B-Instruct (HF)	1	Iteracion 6	100	30
simple	Llama-3.2-1B-Instruct (HF)	1	Iteracion 9	100	10
simple	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	76
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	70
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	48
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	72.73
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	18.18
advanced	Llama-3.2-1B-Instruct (HF)	1	Sin Enriquecimiento	100	18.18
advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 3	100	18.18
advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 6	100	9.09

advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 9	100	45.45
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	57.89
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	47.37
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	52.63
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	47.37
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	26.32
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	37.5
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	95.83
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	95.83
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	95.83
advanced	Llama-3.2-1B-Instruct (HF)	1	Sin Enriquecimiento	100	0
advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 3	100	100
advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 6	100	0
advanced	Llama-3.2-1B-Instruct (HF)	1	Iteracion 9	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	100
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	100
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	100
simple	Qwen2.5-1.5B-Instruct (Local)	1.5	Sin Enriquecimiento	100	100
simple	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 3	100	10
simple	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 6	100	100
simple	ERNIE-4.5-0.3B (Local)	0.3	Sin Enriquecimiento	100	40
simple	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 3	100	100
simple	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 6	100	40
simple	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 9	100	20
simple	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	78
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	24
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	72
simple	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	60

simple	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	76
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	76
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	0
simple	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	0
simple	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 3	100	12
simple	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 6	100	12
simple	ERNIE-4.5-0.3B (Local)	0.3	Sin Enriquecimiento	100	42
simple	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 3	100	8
simple	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 6	100	18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	18.18
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	0
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 6	100	18.18
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 9	100	18.18
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Sin Enriquecimiento	100	27.27
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 6	100	18.18
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 9	100	0
advanced	ERNIE-4.5-0.3B (Local)	0.3	Sin Enriquecimiento	100	45.45
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 3	100	54.55
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 6	100	63.64
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 9	100	54.55
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	68.42
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	52.63
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	42.11
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	42.11
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	21.05
advanced	llama-3.3-70b-versatile (Groq)	70	Iteracion 3	100	57.89
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 9	100	21.05
advanced	ERNIE-4.5-0.3B (Local)	0.3	Sin Enriquecimiento	100	73.68
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 3	100	73.68
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 6	100	21.05

advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 9	100	73.68
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Sin Enriquecimiento	100	100
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 3	100	50
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 6	100	100
advanced	meta-llama/Llama-3.1-8B-Instruct	8	Iteracion 9	100	50
advanced	llama-3.3-70b-versatile (Groq)	70	Sin Enriquecimiento	100	62.5
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Sin Enriquecimiento	100	54.17
advanced	Qwen2.5-1.5B-Instruct (Local)	1.5	Iteracion 9	100	8.33
advanced	ERNIE-4.5-0.3B (Local)	0.3	Sin Enriquecimiento	100	50
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 3	100	12.5
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 6	100	4.17
advanced	ERNIE-4.5-0.3B (Local)	0.3	Iteracion 9	100	75