ULPGC
Universidad de
Las Palmas de
Gran Canaria

ESCUELA DE INGENIERÍA
INFORMÁTICA

Grado en Ingeniería Informática

Trabajo Fin de Grado

Extracción Automática de Información de Documentos Estructurados Usando Modelos Extensos de Lenguaje

Javier Gómez Falcón

Tutor

Javier Sánchez Pérez

Las Palmas de Gran Canaria
11 de julio de 2025

ULPGC Universidad de Las Palmas de Gran Canaria | eii ESCUELA DE INGENIERÍA INFORMÁTICA

TFT04

SOLICITUD DE DEFENSA DE TRABAJO DE FIN DE TÍTULO

D./Dª **Javier Gómez Falcón**, autor del trabajo **Extracción Automática de Información de Documentos Estructurados Usando Modelos Extensos de Lenguaje**, correspondiente a la titulación **Grado en Ingeniería Informática**.

SOLICITA

que se inicie el procedimiento de defensa del mismo, para lo que se adjunta la documentación requerida, haciendo constar que

[X] se autoriza / [ ] no se autoriza la grabación en audio de la exposición y turno de preguntas.

Asimismo, con respecto al registro de la propiedad intelectual/industrial del TFT, declara que:
[ ] Se ha iniciado o hay intención de iniciarlo (defensa no pública).
[X] No está previsto.

Y para que así conste firma la presente. (fecha en firma electrónica)

El/La estudiante

Fdo. Javier Gómez Falcón

A rellenar y firmar **obligatoriamente** por el/la/los/las tutores

En relación con la presente solicitud, se informa:(firmar donde corresponda)

Positivamente (en caso de detección de copia, esta firma quedará invalidada)	Negativamente (justificación en TFT05)
Fdo. Javier Sánchez Pérez,	Fdo. Javier Sánchez Pérez,

DIRECTOR DE LA ESCUELA DE INGENIERÍA INFORMÁTICA

Extracción Automática de Información de Documentos Estructurados Usando Modelos Extensos de Lenguaje

Trabajo Fin de Grado

Javier Gómez Falcón

Tutor

Javier Sánchez Pérez

Grado en Ingeniería Informática

Escuela de Ingeniería Informática

Universidad de Las Palmas de Gran Canaria

Las Palmas de Gran Canaria
11 de julio de 2025

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Unas pocas palabras difícilmente pueden abarcar todo el amor y agradecimiento que siento hacia las personas que me han acompañado en esta etapa tan significativa de mi vida.

A mis padres, gracias infinitas por guiarme siempre por el camino correcto, por apoyarme en cada decisión y por estar presentes en cada paso dado. Su amor incondicional y sus sacrificios constantes son los pilares fundamentales que sostienen el templo que hoy soy. Nada de esto habría sido posible sin ustedes.

A mi hermano, mi referente y ejemplo constante, porque con cada objetivo que alcanzas me enseñas que no existen límites para quien persevera. Gracias por impulsarme siempre a crecer y a evolucionar. Tu apoyo, admiración y cariño me han hecho ser quien soy, y me impulsan a ser quien quiero ser.

A la universidad, por regalarme uno de los años más extraordinarios de mi vida en Praga, una ciudad que marcó un antes y un después en mi camino. Allí viví momentos inolvidables y conocí personas excepcionales que aún hoy continúan siendo fundamentales en mi vida.

A Hugo, por darme tantas alegrías y experiencias en ese año tan mágico. Llegamos siendo conocidos, y nos fuimos con una conexión y unos recuerdos de los que nunca me olvidaré. Gracias por enseñarme lo que es el compañerismo, y que juntos, se puede afrontar cualquier tormenta.

A Pijamitas, porque sin ustedes ese año no habría sido tan mágico. Praga siempre será sinónimo de ellos, y ustedes serán siempre mi recuerdo más bonito de Praga. Aunque estemos separados por kilómetros, siempre los llevo en el corazón.

A Armario & Co., gracias por convertir estos años de carrera en un auténtico *show* lleno de alegrías y momentos inolvidables. Han hecho que este cierre de etapa sea aún más especial. Con ellos, no solo termino con un título académico, sino con una segunda familia.

A Carla, gracias por estar siempre a mi lado, brindándome tu apoyo incondicional y tu cariño constante. Eres una de las razones por las que me siento muy afortunado.

A Carlos, gracias por ser ese amigo incondicional que siempre está ahí, sin importar el qué, el cómo o el cuándo. Gracias por acompañarme durante una gran parte de mi vida. Es un orgullo terminar como empezamos: juntos.

Finalmente, a mi tutor Javier, gracias por guiarme en este camino, por confiar en mí desde el principio y por aceptar dirigir este trabajo. Su acompañamiento y apoyo han sido claves para llevar este proyecto a buen puerto.

A todos ustedes, gracias.

Resumen

Este Trabajo de Fin de Grado (TFG) desarrolla un sistema basado en Modelos Extensos de Lenguaje (LLMs) para la extracción y clasificación automática de información contenida en documentos estructurados, como facturas eléctricas. Utilizando modelos basados en la nube accesibles al público y sin necesidad de reentrenamiento, el proyecto optimiza la precisión mediante técnicas de ingeniería de instrucciones y ajuste de parámetros (temperature, `Top-k`, `Top-p`). A través de una metodología comparativa, se analizan diferentes modelos y configuraciones para proponer una solución eficiente, adaptable y generalizable para la automatización de la gestión documental.

Abstract

This Bachelor's thesis proposes the development of a system based on Large Language Models (LLM) to automatically extract and classify information from structured documents, such as electric utility bills. Without requiring fine-tuning, the system leverages public cloud based models through prompt engineering and adjustment of inference parameters (temperature, `Top-k`, `Top-p`) seeking to optimize the precision of information extraction. The methodology includes comparative evaluation of different models and configurations, aiming to design a general and efficient approach to automated document management.

Capítulo 1

Introducción

Nos encontramos en una era definida por la transición irreversible hacia una economía del conocimiento, donde los datos han dejado de ser un mero subproducto de las operaciones empresariales para consolidarse como el activo estratégico más valioso de cualquier organización. Este paradigma se sustenta en la "datificación" de los procesos de negocio: la conversión de interacciones, transacciones y documentos, tradicionalmente analógicos, en flujos de información digital que alimentan el sistema nervioso central de la empresa moderna. En este contexto, la gestión de documentos transaccionales, como facturas, albaranes y contratos, ha trascendido su función puramente administrativa para convertirse en un desafío de inteligencia de negocio de primer orden. La escala de este desafío es monumental; se estima que anualmente se producen más de 550 mil millones de facturas a nivel global [Koch, 2019], una cifra que no solo evidencia un problema de volumen, sino que también subraya el inmenso valor latente atrapado en estos documentos, un valor que solo puede ser liberado mediante su procesamiento eficiente y preciso.

La eficacia de los sistemas de Planificación de Recursos Empresariales (ERP), que constituyen el corazón de la gestión moderna, depende de manera crítica de la calidad, velocidad y precisión de los datos que los nutren. Un proceso de entrada de datos deficiente, lento o propenso a errores no es solo una fuente de ineficiencia; es una vulnerabilidad estratégica que puede entorpecer las operaciones desde su base, llevando a análisis de negocio incorrectos, previsiones financieras erróneas y, en última instancia, a decisiones estratégicas mal informadas.

Históricamente, la respuesta tecnológica a este desafío ha seguido una clara línea evolutiva, con cada etapa representando un avance significativo, pero también introduciendo sus propias limitaciones. La era pre-digital, dominada por los archivos físicos, imponía costes de espacio, personal y tiempos de recuperación de información que son impensables en la economía actual. El primer salto cuántico llegó con el Reconocimiento Óptico de Caracteres (OCR), que prometía la digitalización del texto y la conversión de imágenes a datos legibles por máquina. Aunque revolucionario, el OCR tradicional demostró ser frágil, con una precisión muy dependiente de la calidad del escaneo, las fuentes tipográficas y la ausencia de ruido visual.

La siguiente generación de soluciones, los sistemas basados en plantillas, mejoró el OCR puro mediante la definición de zonas fijas y reglas preestablecidas para cada tipo de documento. Si bien estos sistemas pueden alcanzar una alta precisión para documentos altamente estandarizados, fracasan estrepitosamente ante la diversidad del mundo real, donde cada proveedor emite facturas con una maquetación única. Esto condujo a la creación de "islas de automatización" en un mar de excepciones que seguían requiriendo una

costosa y lenta intervención manual, un problema que, como ha demostrado la investigación, puede ser hasta un 2958 % más propenso a errores que los procesos completamente automatizados [Barchard and Pace, 2011].

Los enfoques posteriores, basados en técnicas de aprendizaje automático clásico como las Máquinas de Vectores de Soporte (SVM) o los Campos Aleatorios Condicionales (CRF) para el reconocimiento de entidades, introdujeron una mayor flexibilidad. Sin embargo, su principal cuello de botella persistía: requerían la creación de enormes conjuntos de datos etiquetados a mano para cada nuevo tipo de documento, haciendo que la escalabilidad y la adaptación a nuevos formatos siguieran siendo un objetivo económica y operativamente inalcanzable para la mayoría de las organizaciones. Incluso las soluciones comerciales más avanzadas ofrecidas por los hiperescaladores, como Azure Document Intelligence o AWS Textract, aunque potentes, suelen basar sus costes en un modelo de pago por página que puede resultar oneroso para el procesamiento masivo de documentos.

Esta rigidez inherente a las soluciones tradicionales evidencia la necesidad de explorar un nuevo paradigma tecnológico. Los recientes y exponenciales avances en el campo de la Inteligencia Artificial y, concretamente, la aparición de los Modelos Extensos de Lenguaje (LLM), plantean una oportunidad transformadora. Estos modelos, pre-entrenados sobre vastos corpus de texto e imágenes, poseen una capacidad sin precedentes para comprender el contexto y la semántica del lenguaje. Esto les confiere el potencial de interpretar documentos de forma flexible y generalizable, basándose en el significado del contenido ("¿qué es un 'importe total'?") más que en su posición física ("¿dónde está el importe total en esta plantilla específica?").

En este contexto de cambio de paradigma, el presente Trabajo de Fin de Grado se embarca en una investigación para analizar y medir la eficacia y el desempeño de estos modelos de propósito general, que carecen de entrenamiento especializado, en la tarea de extracción de información de documentos empresariales. Se examinarán de manera sistemática sus habilidades para procesar formatos estructurales diversos con el fin de evaluar su viabilidad como una nueva generación de herramientas para la automatización inteligente y escalable de procesos de negocio.

1.1. Motivación

La motivación de este Trabajo de Fin de Grado surge de la confluencia de una necesidad empresarial crítica, una oportunidad tecnológica y una brecha de conocimiento científico. La justificación de la investigación reside en validar un cambio de paradigma que podría redefinir la automatización de procesos de negocio. Para ello, la motivación se articula en tres ejes: la necesidad del mercado, la irrupción de una nueva solución tecnológica y la obligación académica de aportar rigor a un campo en plena evolución.

La extracción de datos de documentos como las facturas no es un desafío nuevo. Sin embargo, el paradigma dominante basado en Reconocimiento Óptico de Caracteres (OCR) y plantillas predefinidas (*templates*) es cada vez más inadecuado para la complejidad del comercio moderno. Este enfoque sufre de una rigidez que genera significativos costes ocultos.

El problema del mercado no se limita al coste de licencias o consultorías, sino que se manifiesta en tres áreas críticas:

- **Coste de Corrección de Errores:** Los sistemas basados en plantillas son frágiles; cambios menores en el formato provocan fallos que exigen una costosa intervención

manual. Este ciclo de error y corrección no solo merma la productividad, sino que también desmotiva al personal.

- **Coste de Inacción Estratégica:** En una economía de datos, la incapacidad de acceder a la información de forma ágil equivale a una ceguera estratégica. Cuando los datos quedan atrapados en procesos manuales, las decisiones críticas se retrasan, impidiendo analizar gastos en tiempo real u optimizar la cadena de suministro.
- **Coste de Oportunidad:** Para las Pequeñas y Medianas Empresas (PYMES), el alto coste de las soluciones tradicionales frena su crecimiento. Para las grandes corporaciones, la ineficiencia a escala genera pérdidas millonarias y vulnerabilidad estratégica. La dependencia de sistemas inflexibles es un riesgo empresarial que atenta contra la resiliencia y la adaptación.

Frente a este panorama, emerge la oportunidad de los Grandes Modelos de Lenguaje (LLM). La viabilidad de este proyecto se basa en la confluencia de tres avances clave:

1. **Avances Algorítmicos:** La arquitectura Transformer [Vaswani et al., 2017] fue un salto cuántico. Su mecanismo de auto-atención permitió a los modelos superar las limitaciones de arquitecturas previas, dándoles una capacidad sin precedentes para entender el contexto. Este avance impulsa la transición de un reconocimiento espacial (dónde está un dato) a una comprensión semántica (qué significa un dato).
2. **Disponibilidad de Datos y Computación:** El potencial del Transformer se materializó gracias a la disponibilidad de enormes corpus de texto para su entrenamiento y al acceso a cómputo masivo, principalmente a través de Unidades de Procesamiento Gráfico (GPUs).
3. **Democratización vía APIs:** La publicación de ChatGPT a finales de 2022 [Zhao et al., 2023] marcó el inicio de la IA Generativa accesible. La disponibilidad de estos modelos a través de Interfaces de Programación de Aplicaciones (APIs) ha democratizado el acceso a una capacidad antes reservada a gigantes tecnológicos, permitiendo transitar de un paradigma de *construcción de modelos* a uno de *orquestación de modelos*.

Estos modelos, con su capacidad de aprendizaje en contexto (*zero-shot* y *few-shot learning*), prometen superar la dependencia de las plantillas. La pregunta central de este trabajo es: ¿es este nuevo paradigma de IA generalista suficientemente robusto y preciso para un dominio de misión crítica como la extracción de datos de facturas?

La existencia de una tecnología prometedora no garantiza su idoneidad, lo que exige una validación científica rigurosa. Aquí reside la tercera motivación del trabajo:

- **Brecha de Conocimiento Específica:** Aunque la literatura académica tiene *benchmarks* para tareas de Procesamiento de Lenguaje Natural (PLN) generales, existe una escasez de estudios rigurosos sobre la aplicación de modelos de propósito general no especializados (*fine-tuned*) en la extracción de datos de documentos transaccionales, que exigen una alta precisión.
- **Sistematización de la Ingeniería de Instrucciones:** El proyecto contribuye a la emergente disciplina de la Ingeniería de Instrucciones (*Prompt Engineering*), un campo aún artesanal. Se busca responder no solo si los LLM funcionan, sino cómo maximizar su rendimiento mediante la comparación sistemática de distintas estrategias de *prompting*.

- **Necesidad de *Benchmarks* Replicables:** En un campo con tanto marketing, la comunidad científica necesita *benchmarks* públicos y replicables. Este estudio responde a esa necesidad usando un dataset público y métricas estándar (*Precision*, *Recall*, *F1-Score*) para generar resultados verificables.

En síntesis, este Trabajo de Fin de Grado se posiciona en la confluencia de estos tres ejes, respondiendo a la ineficiencia del mercado con la oportunidad de la IA Generativa, y asumiendo la responsabilidad de aportar rigor científico para validar esta nueva tecnología. La pregunta fundamental es si una IA generalista puede dominar un dominio tan especializado.

Una respuesta afirmativa, validada con métricas rigurosas, no sería un avance incremental, sino la base para una reconversión integral de la automatización de procesos con un impacto multidimensional:

- **A nivel operativo,** liberar al capital humano de tareas manuales para reorientarlo hacia la supervisión de IA y el análisis de alto valor.
- **A nivel estratégico,** transformar documentos en inteligencia de negocio accionable en tiempo real, mejorando la toma de decisiones y la resiliencia.
- **A nivel económico y social,** nivelar el campo de juego para las PYMES, catalizando un nuevo ecosistema de aplicaciones y fomentando una evolución de los roles laborales hacia la colaboración humano-IA.

Por todo ello, este trabajo representa un primer paso riguroso y necesario para cartografiar el potencial de esta tecnología y trazar la ruta hacia una nueva generación de automatización inteligente, flexible y democrática.

1.2. Objetivos

Para abordar la problemática y las preguntas de investigación descritas en la motivación, este Trabajo de Fin de Grado se articula en torno a una agenda de investigación clara. Esta agenda está guiada por un objetivo general que define el propósito último del estudio, y se desglosa en un conjunto de objetivos específicos y medibles, diseñados para construir, de manera sistemática, una respuesta empírica a las cuestiones fundamentales sobre la viabilidad, optimización y límites de los LLM en este dominio.

Objetivo General: Analizar y evaluar cuantitativamente el rendimiento de modelos extensos de lenguaje de propósito general en la tarea de extracción de información de un corpus de facturas del sector eléctrico español, comparando diferentes estrategias de ingeniería de instrucciones y configuraciones de parámetros.

Preguntas de Investigación y Objetivos Específicos: El objetivo general se materializa a través de los siguientes objetivos específicos, cada uno diseñado para aportar evidencia directa a las preguntas de investigación que impulsan este trabajo:

1. **Viabilidad Fundamental:** Establecer un marco teórico sólido mediante una revisión exhaustiva del estado del arte en Procesamiento Inteligente de Documentos

(PID) y Modelos Extensos de Lenguaje. Este objetivo busca contextualizar la novedad del enfoque propuesto frente a las limitaciones documentadas de las tecnologías precedentes (basadas en OCR y plantillas), definiendo así la brecha de conocimiento que este trabajo pretende cerrar.

2. **Optimización y Sensibilidad:** Diseñar y ejecutar un banco de pruebas sistemático y multifactorial para aislar y cuantificar el impacto de las variables clave en el rendimiento de la extracción. Este marco experimental se centrará en:

   a) **Ingeniería de Instrucciones:** Comparar un espectro de estrategias de *prompting*, desde el *zero-shot* (para medir la capacidad inherente del modelo), el *few-shot* con distintas configuraciones (para evaluar el aprendizaje en contexto y la generalización), hasta la extracción iterativa (como método para gestionar la complejidad documental).

   b) **Parámetros de Generación:** Evaluar la sensibilidad del rendimiento a los parámetros de inferencia (temperatura, `Top-p`, `Top-k`) para determinar su influencia relativa en una tarea que demanda alta fidelidad y precisión.

3. **Pipeline y Preparación de Datos:** Desarrollar y documentar un *pipeline* de preprocesamiento de datos completo y replicable, específico para la adaptación del corpus *Invoices database of the Spanish electricity market* (IDSEM). Este objetivo va más allá de la simple preparación; implica abordar los desafíos inherentes al formato PDF, justificar la transición a un formato optimizado para LLM como Markdown, y refinar el *ground truth* mediante un mapeo explícito de las etiquetas visibles por plantilla para asegurar una evaluación precisa y justa, estableciendo una metodología de referencia para trabajos futuros.

4. **Ejecutar de manera programática** la totalidad de los experimentos diseñados sobre los modelos seleccionados (Gemini 1.5 Pro y Mistral-small), recopilando sistemáticamente las extracciones generadas para su posterior análisis.

5. **Analizar cuantitativamente los resultados experimentales** mediante métricas estándar (*Precision, Recall* y *F1-Score*), con el fin de:

   a) Identificar las configuraciones de parámetros y, sobre todo, las estrategias de *prompting* que maximizan el rendimiento, revelando las mejores prácticas para interactuar con estos modelos en tareas de extracción estructurada.

   b) Comparar la eficacia y eficiencia de los modelos evaluados, estableciendo una base empírica para su selección en casos de uso reales según el equilibrio deseado entre coste, velocidad y precisión.

6. **Sintetizar la evidencia empírica recopilada para emitir un veredicto fundamentado** sobre la viabilidad de los LLM de propósito general en este dominio. Este objetivo final implica no solo resumir los hallazgos, sino también interpretar sus implicaciones prácticas, identificar los umbrales de rendimiento, las fortalezas, las limitaciones intrínsecas, y proponer líneas de investigación futuras que surjan de los resultados de este trabajo.

1.3. Organización del Documento

La presente memoria ha sido estructurada en diez capítulos que siguen una progresión lógica, diseñada para guiar al lector desde la contextualización inicial del problema hasta la presentación y síntesis de los hallazgos experimentales. Cada capítulo se construye sobre las bases del anterior, asegurando una argumentación coherente y una comprensión clara del proceso de investigación en su totalidad.

El Capítulo 1, "Introducción", sirve como punto de partida, donde se define el problema de negocio y tecnológico que motiva el estudio. En él se articulan la justificación, la relevancia y, fundamentalmente, los objetivos generales y específicos que marcan la dirección y el alcance de toda la investigación.

Sobre la base de la problemática establecida, el Capítulo 2, "Estado del Arte", contextualiza la investigación dentro del panorama científico y tecnológico actual. A través de una revisión crítica de la literatura sobre Modelos Extensos de Lenguaje y técnicas de Procesamiento Inteligente de Documentos, este capítulo identifica las limitaciones de los enfoques existentes y define la brecha de conocimiento que este trabajo se propone abordar, justificando así su originalidad.

A continuación, el Capítulo 3, "Metodología y Planificación", responde al vacío identificado en el capítulo anterior presentando el marco metodológico *CRoss-Industry Standard Process for Data Mining* (CRISP-DM), que estructura el ciclo de vida del proyecto. Se detalla la adaptación de cada una de sus fases, proporcionando un plan de trabajo estratégico que guiará las etapas prácticas posteriores.

Como paso previo a la ejecución, el Capítulo 4, "Recursos Empleados", cataloga de manera explícita los recursos de hardware y software utilizados. Esta sección garantiza la transparencia del proceso y proporciona la información necesaria para la futura replicabilidad del estudio.

Definida la metodología y los recursos, el Capítulo 5, "Conjunto de Datos", se centra en el objeto de estudio: el corpus IDSEM. Se describe su estructura y contenido, y se detalla el crucial *pipeline* de preprocesamiento desarrollado, incluyendo la conversión de los documentos a formato Markdown y el refinamiento del *ground truth* para asegurar una evaluación precisa.

Con los datos ya preparados, el Capítulo 6, "Redes Neuronales Empleadas", establece el fundamento teórico de las herramientas que se aplicarán. Se profundiza en la arquitectura del Transformer y las innovaciones específicas de los modelos Gemini 1.5 Pro y Mistral-small, proveyendo el contexto técnico indispensable para comprender las decisiones de diseño experimental y el análisis de resultados.

El Capítulo 7, "Desarrollo Práctico", documenta la transición de la teoría y la planificación a la implementación. Narra el proceso iterativo de construcción del sistema experimental, incluyendo los desafíos encontrados y las soluciones adoptadas, sirviendo de puente entre la preparación metodológica y la ejecución formal de los experimentos.

A partir de la implementación práctica, el Capítulo 8, "Configuración de Experimentos", formaliza el protocolo científico del trabajo. Se especifica con precisión cada variable a evaluar desde las 19 configuraciones de parámetros de inferencia hasta las 8 estrategias de *prompting*, garantizando el rigor y la claridad del diseño experimental que se ejecutará a continuación.

El Capítulo 9, "Resultados y Análisis de los Experimentos", constituye el núcleo empírico de la investigación. Ejecuta el protocolo definido en el capítulo anterior y presenta los datos cuantitativos obtenidos. De manera crucial, no se limita a exponer los resultados,

sino que los analiza e interpreta en profundidad para extraer los hallazgos significativos sobre el rendimiento de los modelos y las estrategias evaluadas.

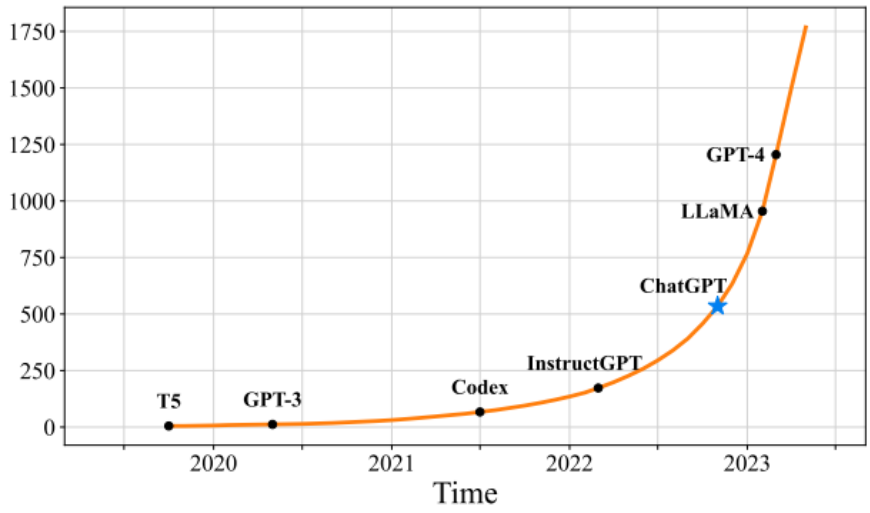
Finalmente, el Capítulo 10, "Conclusiones y Trabajo Futuro", cierra el ciclo de la investigación. En esta sección se sintetizan los hallazgos del Capítulo 9 para dar una respuesta directa y fundamentada a los objetivos planteados en el Capítulo 1. Se formulan las conclusiones principales del estudio y, a partir de éstas y de las limitaciones identificadas, se proponen líneas concretas de trabajo futuro para expandir el conocimiento generado.

Capítulo 2

Estado del Arte

Este capítulo tiene como propósito contextualizar el presente trabajo de fin de grado a través de una revisión de la literatura y tecnologías relevantes en la actualidad. Se abordarán los principales avances, marcos teóricos y herramientas en dos áreas fundamentales para el desarrollo de este proyecto: los Modelos Extensos de Lenguaje (LLM) y las técnicas tradicionales y modernas para la extracción de información de documentos.

2.1. Modelos Extensos de Lenguaje.



(b) Query="Large Language Model"

Figura 2.1: Evolución de la publicación de artículos relacionados con los Modelos Extensos de Lenguaje. Fuente: [Zhao et al., 2025]

El campo de los modelos extensos de lenguaje (LLM) ha experimentado una evolución notable desde finales de 2022, marcada por la publicación de ChatGPT, que catalizó un interés masivo tanto en el ámbito académico como en el industrial [Zhao et al., 2023]. La importancia que ha cobrado este tema en la comunidad de investigadores es muy notable. Como se muestra en la Figura 2.1 el número promedio de artículos sobre LLM publicados diariamente en la plataforma de investigación arXiv se disparó de 0.40 a 8.58 tras dicho lanzamiento. Si bien la era anterior se definió por la ley del escalado (un mayor

tamaño del modelo, datos y cómputo conducían a un mejor rendimiento), el problema se ha redefinido. La investigación a partir de 2023 ha girado hacia un nuevo paradigma centrado en la eficiencia computacional, la especialización de modelos y la robustez del razonamiento [Fan et al., 2024].

Las características principales del tema actual giran en torno a las limitaciones de la arquitectura Transformer, que sigue siendo fundamental [Zhao et al., 2023]. Su complejidad computacional cuadrática ($O(L^2)$) ha impulsado la exploración de arquitecturas alternativas. Además, el coste prohibitivo del reentrenamiento completo ha fomentado el desarrollo de técnicas de Ajuste Fino Eficientes en Parámetros (PEFT) [Hu et al., 2025]. A medida que los LLM se integran en aplicaciones críticas, sus fallos, como la generación de información falsa (alucinación) y los sesgos, han pasado a primer plano, lo que ha llevado a la creación de *benchmarks* más sofisticados para evaluar la robustez y la veracidad [Shang et al., 2025]. Los temas en los que profundizaremos para definir el estado del arte son, por tanto: arquitecturas emergentes, técnicas de entrenamiento y ajuste, evaluación avanzada y los desafíos actuales de fiabilidad y eficiencia.

2.1.1. Arquitecturas post-Transformer

La literatura científica desde 2023 revela un esfuerzo dedicado a superar las limitaciones presentadas por la tendencia del aumento masivo de parámetros en los modelos. Los avances se pueden agrupar en torno a la optimización de la arquitectura, la eficiencia de la adaptación y la calidad de la evaluación de los modelos. La complejidad cuadrática del mecanismo de auto-atención del Transformer, ha sido una barrera para escalar a contextos muy largos, motivando la búsqueda de alternativas subcuadráticas, desarrollándose las siguientes arquitecturas:

Modelos de espacio de estados y *Mamba*. Los Modelos de Espacio de Estados (SSM) han surgido como una alternativa prometedora, mapeando una secuencia de entrada a una de salida a través de un estado latente [Zhang, 2025]. Su principal ventaja es una dualidad que les permite ser formulados como una Red Neuronal Recurrente (RNN) para una inferencia en tiempo lineal o como un modelo convolucional para un entrenamiento paralelo [Han et al., 2024]. La arquitectura *Mamba*, propuesta en 2023, superó las limitaciones de los SSM anteriores mediante dos innovaciones clave. Primero, un mecanismo de selección de estado selectivo, que hace que los parámetros del modelo dependan de la entrada, permitiendo al modelo filtrar y propagar información selectivamente, de forma análoga a los mecanismos de compuerta. Segundo, un algoritmo paralelo consciente del hardware, que optimiza el cálculo para la jerarquía de memoria de las GPUs modernas, permitiendo un entrenamiento eficiente con escalado lineal, en vez de uno cuadrático. Como resultado, Mamba iguala o supera el rendimiento de Transformers del doble de su tamaño con una inferencia hasta 5 veces más rápida [Gu and Dao, 2023]. Su éxito ha impulsado extensiones multimodales como *Cobra* [Han et al., 2024] y arquitecturas híbridas como *mmMamba* [Li et al., 2025].

La arquitectura mezcla de expertos. La arquitectura Mezcla de Expertos (MoE) [Shazeer et al., 2017] tiene como objetivo aumentar la capacidad de un modelo, aumentando sus parámetros, sin incrementar el coste computacional. Esto se logra activando un pequeño subconjunto de "expertos" para cada *token* (subpalabras en las que se divide el texto de entrada que recibe el modelo) [Ilharco et al., 2025], sin activar toda la red del

Transformer, lo que genera un coste computacional mucho menor. La investigación reciente ha buscado escalar este enfoque a un número mucho mayor de expertos. *Test-Time Model Merging* (TTMM) entrena a expertos de forma independiente en clústeres de datos y, en el momento de la inferencia, fusiona los parámetros de los expertos más relevantes para crear un modelo dinámico y específico para la tarea [Ilharco et al., 2025]. Por otro lado, *Self-MoE* transforma un modelo pre-entrenado en un sistema MoE utilizando datos sintéticos auto-generados para especializar módulos expertos ligeros, demostrando mejoras significativas sobre el modelo base monolítico [Chen et al., 2025].

El LLM como un agente . Este es un marco funcional que utiliza un LLM como núcleo de razonamiento, pero lo aumenta con memoria, planificación y, fundamentalmente, la capacidad de usar herramientas externas [Fan et al., 2024, Wu et al., 2025]. Los agentes pueden aprender a descomponer tareas complejas mediante técnicas como la cadena de pensamiento, en la que dividen sus tareas paso por paso, y crean un plan estructurado para llegar a su objetivo final. Además, pueden interactuar con el entorno invocando APIs (para búsquedas web, ejecución de código, etc.) y gestionan contextos extendidos para mantener la coherencia en tareas largas [Wu et al., 2025]. Esto transforma al LLM de un generador de lenguaje a un solucionador de problemas proactivo [Fan et al., 2024].

2.1.2. Optimización del Ajuste Fino

El ajuste fino completo de los LLM es computacionalmente muy costoso, por lo que la investigación se ha centrado en buscar métodos que mejoren las técnicas de ajuste fino haciéndolas más eficientes. A continuación se describen las técnicas más relevantes en el estado del arte actual:

Ajuste fino eficiente en parámetros. Los métodos de Ajuste Fino Eficiente en Parámetros (PEFT) adaptan LLM actualizando solo una pequeña fracción de sus parámetros (1 %) [Lialina et al., 2025]. La técnica más influyente es *Low-Rank Adaptation* (LoRA), que parte de la hipótesis de que el cambio en los pesos durante la adaptación tiene un rango intrínseco bajo. Como se puede observar en la figura 2.2, LoRA divide la matriz de pesos entrenables en dos. En lugar de aprender la matriz de cambio de pesos completa, LoRA la factoriza en dos matrices de rango mucho más bajo, reduciendo drásticamente los parámetros entrenables [Hu et al., 2025].

La figura 2.2 muestra una gran matriz de pesos pre-entrenada y congelada (W) a la izquierda. A la derecha, ilustra cómo se añaden las dos matrices de rango bajo (A y B) que son las únicas que se entrenan. El diagrama visualiza cómo la salida final es la suma de la matriz de pesos pre-entrenados congelados (W) y las matrices de pesos A y B que sí han sido re-entrenadas.

Evolución de la eficiencia en el ajuste fino. A pesar de la eficiencia de LoRA, los modelos más grandes seguían siendo inaccesibles para muchos. *Quantized Low-Rank Adaptation* (QLoRA) fue un avance fundamental que redujo drásticamente la barrera de memoria para este re-entrenamiento [Dettmers et al., 2023a]. Su método consiste en retropropagar gradientes a través de un modelo base que ha sido congelado y cuantizado a 4 bits, mientras que solo se actualizan los adaptadores LoRA, que se mantienen en una precisión más alta [Dettmers et al., 2023a]. Esto se logra mediante tres innovaciones técnicas: un tipo de dato de 4 bits teóricamente óptimo llamado `NormalFloat` (NF4); la doble

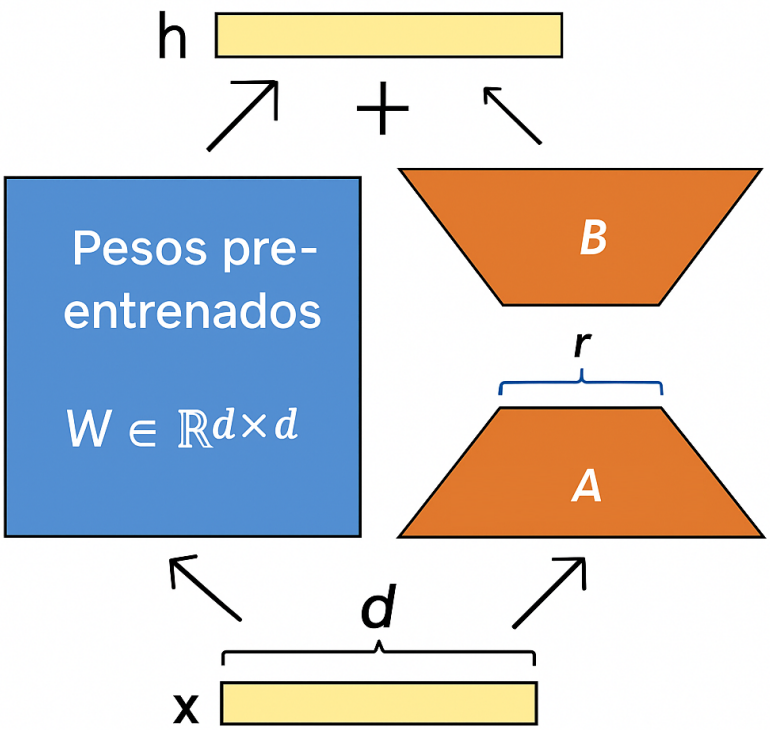


Figura 2.2: Funcionamiento de LoRA. Fuente: Modificado de [Hu et al., 2021]

cuantización para comprimir aún más las constantes de cuantización; y los optimizadores paginados para evitar desbordamientos de memoria de la GPU [Dettmers et al., 2023a, Dettmers et al., 2023b]. El impacto de QLoRA ha sido transformador, reduciendo la memoria necesaria para ajustar un modelo de 65B parámetros de más de 780 GB a menos de 48 GB, preservando el rendimiento de un ajuste fino de 16 bits [Dettmers et al., 2023b].

2.1.3. Evaluación del Progreso: *Benchmarks*

La evaluación ha pasado de medir tareas aisladas en un entorno controlado, a probar el razonamiento, la veracidad y la robustez de los modelos extensos de lenguaje en condiciones donde sus respuestas pueden estar sesgadas o ser incorrectas.

Evaluación del razonamiento y la veracidad. El *benchmark* GSM8K, que consiste en problemas de matemáticas de nivel escolar, se ha consolidado como una prueba clave para el razonamiento de varios pasos [Cobbe et al., 2021]. Para medir la veracidad y combatir la "alucinación", el *benchmark* TruthfulQA evalúa si los modelos evitan repetir conceptos erróneos comunes en sus datos de entrenamiento, revelando que a veces los modelos más grandes pueden ser menos veraces [Lin et al., 2021].

Subjetividad y opiniones sesgadas. Las evaluaciones de seguridad a menudo subestiman los riesgos al usar consultas que no son maliciosas [Shang et al., 2025]. El *benchmark* Fairness Benchmark in LLM under Extreme Scenarios FLEX aborda esto al aumentar los conjuntos de datos en temas de igualdad con *prompts* que presentan opiniones contrarias entre sí (p. ej., inyectando una persona sesgada o dando objetivos en conflicto)

[Shang et al., 2025, Zhang et al., 2025]. Los resultados muestran que los modelos considerados seguros en pruebas estándar exhiben un aumento significativo de sesgos bajo estas condiciones adversas [Shang et al., 2025]. Investigaciones más recientes utilizan herramientas inspiradas en la psicología, como el *LLM Word Association Test*, para detectar sesgos implícitos que persisten incluso cuando los sesgos explícitos han sido eliminados, revelando que los modelos de última generación todavía asocian, por ejemplo, ciertos géneros o etnias con profesiones específicas [Caliskan et al., 2024, Udupa, 2024].

2.1.4. Desafíos Persistentes

El Problema de la Alucinación. La alucinación, o la generación de contenido objetivamente incorrecto, sigue siendo un problema crítico [Ji et al., 2025]. Una causa fundamental es el entrenamiento con "etiquetas duras" (*hard labels*) que fomenta el exceso de confianza del modelo [Ji et al., 2025]. La estrategia de mitigación más usada para evitar esto es la Generación Aumentada por Recuperación (RAG), que ancla las respuestas del LLM en conocimiento externo verificable, recuperando documentos relevantes antes de generar una respuesta [Gao et al., 2023]. El paradigma ha evolucionado desde un RAG "ingenuo" hasta un RAG "modular" más flexible y potente, con variantes como *Graph-RAG* para razonar sobre datos interconectados [Gao et al., 2023, Gao et al., 2025].

El desafío del sesgo implícito. Se ha descubierto que la corrección de opiniones sesgadas a menudo solo crea una "fachada" de imparcialidad. Debajo de esta superficie, los modelos continúan albergando asociaciones implícitas que reflejan estereotipos sociales [Caliskan et al., 2024]. Esta subjetividad implícita es un problema más profundo que una correción superficial no elimina y representa un riesgo ético significativo.

2.2. Extracción de Información de Documentos

El campo de la extracción de datos de documentos empresariales está en medio de una transformación fundamental, evolucionando desde tecnologías de nicho hacia una capacidad estratégica esencial para la empresa moderna. Este capítulo analiza el estado del arte del Procesamiento Inteligente de Documentos (IDP), definido como una tecnología de automatización de flujos de trabajo que escanea, lee, extrae, categoriza y organiza información significativa de grandes volúmenes de documentos estructurados, semiestructurados y no estructurados. A diferencia de las tecnologías predecesoras que se limitaban a la digitalización, el IDP se enfoca en una verdadera comprensión del contenido sin intervención humana [Microsoft, 2025d].

La relevancia de indagar en este problema se fundamenta en imperativos empresariales claros y cuantificables. El procesamiento manual de documentos es lento, costoso, con estimaciones que sitúan el coste entre 6 y 8 dólares por documento, y propenso a errores [Microsoft, 2025d]. El IDP no solo reduce drásticamente estos costes y minimiza los errores, sino que también libera a los empleados de tareas repetitivas y de bajo valor, permitiéndoles centrarse en actividades que aportan más valor. [Microsoft, 2025d]. Las características principales del tema abarcan una amplia gama de aplicaciones que impulsan la eficiencia, la precisión y la agilidad en toda la empresa, desde la automatización de facturas en finanzas y la selección de currículums en recursos humanos, hasta la optimización de la logística y la mejora de la atención al paciente en el sector sanitario [Services, 2025].

El problema se define, por tanto, como la necesidad de automatizar la extracción de datos valiosos de formatos de documentos diversos (facturas, contratos, formularios, etc.) y transformarlos en conocimiento estructurado y procesado. [Microsoft, 2025d].

La revisión de la literatura y el análisis de las tecnologías existentes revelan una clara trayectoria evolutiva. El campo ha pasado de un conjunto de herramientas dispares a arquitecturas unificadas que representan el estado del arte. Esta evolución se puede estructurar en varias etapas clave, desde los componentes básicos hasta los objetivos de aprendizaje más avanzados.

2.2.1. La Importancia del Contexto

El progreso en IDP se ha construido sobre una base tecnológica que ha evolucionado con el tiempo, convergiendo hacia modelos más holísticos.

El reconocimiento óptico de caracteres. El Reconocimiento Óptico de Caracteres (OCR) es la tecnología base sobre la cual se construye todo el IDP. Su función es convertir imágenes de texto, ya sea de documentos escaneados o PDFs, en texto legible por máquina, actuando como un puente entre el mundo visual y el digital [Data, 2025]. Un motor de OCR moderno sigue un *pipeline* de múltiples etapas: preprocesamiento para limpiar la imagen (corrección de inclinación, eliminación de ruido), reconocimiento de caracteres (mediante comparación de patrones o extracción de características) y postprocesamiento para corregir errores usando modelos de lenguaje [IBM, 2025, DocuWare, 2025]. La vanguardia en este ámbito es el "OCR profundo" (*deep OCR*), que utiliza redes neuronales profundas para mejorar drásticamente la precisión con fuentes, tamaños y distorsiones complejas [DocuWare, 2025]. Un ejemplo de ello es el modelo *Read* de Azure AI, optimizado específicamente para documentos grandes y con mucho texto, capaz de detectar estructuras como párrafos y formularios [Microsoft, 2025b].

Análisis de la disposición de los elementos dentro de un documento. Si el OCR extrae el texto, el Análisis de la Disposición de los Elementos dentro de un Documento (DLA), impulsado por la visión por computador, proporciona la estructura [Wikipedia, 2025]. Como se puede observar en la imagen 2.3, su función es identificar y categorizar regiones lógicas en un documento (bloques de texto, tablas, encabezados) antes de la extracción [Code, 2025]. Históricamente, se utilizaron enfoques ascendentes (*bottom-up*), que agrupan píxeles para formar bloques, y descendentes (*top-down*), que dividen la página basándose en espacios en blanco [Wikipedia, 2025]. El estado del arte actual, sin embargo, ha sido revolucionado por el aprendizaje profundo, que enmarca el DLA como un problema de detección de objetos. Arquitecturas basadas en Transformer, como el Document Image Transformer (DIT), aprenden las relaciones espaciales y estructurales entre las diferentes partes del documento, ofreciendo una precisión y generalización muy superiores [Code, 2025, Face, 2025].

Procesamiento del lenguaje natural. Una vez extraído el texto y su estructura, el Procesamiento del Lenguaje Natural (PLN) se aplica para interpretar el significado y extraer información procesable [Data, 2025]. Las técnicas fundamentales incluyen el Reconocimiento de Entidades Nombradas (NER), que identifica y clasifica entidades clave como nombres, fechas o importes monetarios, la extracción de relaciones, que determina las conexiones semánticas entre estas entidades y la resolución de correferencias, que agrupa

Figura 2.3: Análisis de la estructura de documentos. Fuente: [Hu et al., 2021]

diferentes menciones de una misma entidad [GeeksforGeeks, 2025]. Se han utilizado muchas técnicas tradicionales de aprendizaje automático para la extracción de información, como regresión logística, SVM o *Random Forest* [Sánchez and Cuervo Londoño, 2024].

Modelos multimodales. La evolución más transformadora en IDP es el paso de un *pipeline* secuencial (OCR → DLA → PLN) a arquitecturas multimodales unificadas [Milvus, 2025]. El enfoque tradicional era frágil, ya que los errores en una etapa se propagaban sin posibilidad de corrección. Los modelos multimodales, casi siempre basados en la arquitectura Transformer, rompen estos silos al procesar simultáneamente texto, imagen (características visuales) y maquetación (coordenadas espaciales) [Lialin et al., 2023]. El modelo LayoutLM introdujo incrustaciones de posición 2D (coordenadas x, y) e incrustaciones de imagen, permitiendo al modelo "ver" la página y aprender conjuntamente el lenguaje y la maquetación, superando drásticamente a los modelos de solo texto [Face, 2025, Xu et al., 2019]. La generación actual de Modelos de Lenguaje Grandes (LLM) multimodales, como Gemini de Google, o modelos de código abierto como Donut y LayoutLLM, llevan este concepto más lejos, procesando la imagen de un documento directamente para generar una salida estructurada (JSON), eliminando la necesidad de un paso de OCR explícito [Face, 2025, Cloud, 2025a]. Esta convergencia define el estado del arte.

2.2.2. Extracción de Datos Mediante Instrucciones

Posiblemente la tendencia más impactante en el IDP actual es el cambio hacia el uso de instrucciones (*prompts*) a un LLM como método de extracción de datos. El enfoque

clásico del aprendizaje automático requería un ciclo más lento y costoso: recolectar miles de documentos de muestra y anotarlos manualmente para entrenar un modelo para cada nuevo tipo de documento [PaperOffice, 2025].

- **Aprendizaje *zero-shot***: Es la capacidad de un modelo para procesar un tipo de documento nunca antes visto sin ningún ejemplo de entrenamiento específico. El modelo aprovecha su vasto conocimiento sobre el lenguaje obtenido en el entrenamiento para inferir la estructura y semántica del nuevo documento [PaperOffice, 2025].
- **Aprendizaje *few-shot***: Es la capacidad que tiene un modelo de lograr una alta precisión de extracción cuando se le proporciona un número reducido de ejemplos etiquetados (típicamente de 5 a 10) con una estructura similar al documento que debe extraer [Gali and Agarwal, 2025].

El impacto de este cambio reduce el tiempo de implementación de meses a días, disminuye drásticamente los costes de anotación y proporciona una flexibilidad muy alta para manejar formatos de documentos variables [PaperOffice, 2025]. Esto puede alterar fundamentalmente el enfoque del modelo IDP, alejando el valor de los servicios de configuración tan personalizados y acercándolo a plataformas que permiten una implementación rápida manteniendo una fiabilidad buena [PaperOffice, 2025].

2.2.3. Soluciones Comerciales y de Código Abierto

El mercado de herramientas de IDP se ha bifurcado en dos campos principales: las ofertas de Plataforma como Servicio (PaaS) de los hiperescaladores (AWS, Microsoft, Google) y las plataformas especializadas de Software como Servicio (SaaS) [CloudThat, 2025]. Los hiperescaladores compiten en la potencia de su IA subyacente y la integración en sus ecosistemas de nube [Slashdot, 2025, Microsoft, 2025c], mientras que los especialistas se diferencian por ofrecer automatización de flujos de trabajo de extremo a extremo e interfaces de usuario superiores [Rossum, 2025, ABBYY, 2025].

Paralelamente, el ecosistema de código abierto ha madurado hasta convertirse en un competidor directo. Marcos como PaddleOCR ofrecen *pipeline*s integrales que rivalizan con las soluciones comerciales en tareas como el análisis de maquetación y el reconocimiento de tablas [Adevinta, 2025, PaddlePaddle, 2025]. *Hugging Face* se ha convertido en el epicentro de esta democratización, alojando modelos de vanguardia como la familia LayoutLM, Donut y DIT, junto con los conjuntos de datos y tutoriales necesarios para ajustarlos a tareas personalizadas [Face, 2025]. Esto ha hecho que a la decisión de construir vs. comprar se le añada la opción de usar una API comercial como motor y construir flujos de trabajo personalizados alrededor como un enfoque también accesible.

2.2.4. Desafíos y Limitaciones

El campo continúa avanzando, abordando las limitaciones de los sistemas actuales. Una evidencia de su madurez es la evolución de los *benchmarks*. Los *benchmarks* tempranos como FUNSD o SROIE, aunque importantes, eran pequeños y contenían sesgos de anotación, lo que llevaba a una brecha entre el rendimiento académico y el del mundo real [Boiangiu et al., 2021, Li et al., 2024]. La nueva generación, como *Document Understanding Evaluation* (DUE) y *Visually-rich Document Understanding* (VRDU), es mucho más desafiante y representativa, evaluando el rendimiento de extremo a extremo en documentos complejos y diversos [Boiangiu et al., 2021, Research, 2025].

Otro desafío clave abordado es la necesidad de una supervisión humana eficiente. El *Human-in-the-Loop* (HITL) se reconoce como un componente de diseño necesario. Su función es triple: actuar como control de calidad para casos de baja confianza, manejar la ambigüedad y los casos límite donde se requiere juicio humano, y, lo más importante, crear un círculo de retroalimentación y refuerzo (Aprendizaje Activo) donde cada corrección humana mejora continuamente el modelo de IA subyacente [Devoteam, 2025, Infrrd, 2025].

La investigación reciente se centra en encontrar una solución para los siguientes problemas:

- **Comprensión de un conjunto de documentos**: Ir más allá del procesamiento de documentos individuales para comprender el contexto y las relaciones que abarcan un paquete completo de documentos relacionados (p. ej., orden de compra, albarán y factura) [Science, 2024, Abolghasemi et al., 2023].

- **Fiabilidad y Mitigación de Alucinaciones**: Garantizar que cada dato extraído esté "fundamentado" (*grounded*), es decir, vinculado a una evidencia específica en el documento fuente, para mejorar la explicabilidad y la confianza, y evitar que los modelos generativos inventen información [V. A., 2024, Science, 2024].

- **Seguimiento de Instrucciones**: Desarrollar modelos, como LayoutLLM, que puedan seguir instrucciones en lenguaje natural para realizar tareas complejas, haciéndolos más flexibles e interactivos [Hong et al., 2024].

El análisis del estado del arte en la extracción automática de información de documentos revela un campo en pleno crecimiento, marcado por una rápida evolución tecnológica y un crecimiento económico. El descubrimiento principal es la transición de *pipeline*s secuenciales y frágiles a arquitecturas multimodales unificadas que procesan texto, estructura e imagen de forma integrada, siendo esta la metodología más avanzada actualmente [Face, 2025, Xu et al., 2019].

Una de las fortalezas de los estudios y tecnologías revisadas es la inmensa mejora en flexibilidad y velocidad de implementación gracias al aprendizaje *zero-shot* y *few-shot* [PaperOffice, 2025]. Esta capacidad, impulsada por los LLM, ha demolido el cuello de botella tradicional de la anotación masiva de datos, reduciendo costes y permitiendo una adaptación ágil a nuevos formatos de documentos. Sin embargo, una debilidad o desafío que emerge con estos modelos generativos es la necesidad de garantizar la fiabilidad, la explicabilidad y la mitigación de alucinaciones, un área que ahora es foco central de la investigación [V. A., 2024].

Los problemas que quedan por resolver definen la próxima frontera del IDP. El principal es el razonamiento a través de múltiples documentos y páginas para comprender procesos de negocio completos, no solo transacciones aisladas [Science, 2024]. La consolidación de la confianza en los modelos de IA a través de la "fundamentación" y la explicabilidad es otro reto crucial para su adopción en flujos de trabajo críticos [V. A., 2024].

El resultado esperado de la investigación continua en este campo es la creación de sistemas de IDP cada vez más autónomos, fiables e interactivos. El futuro no reside en la optimización de componentes individuales, sino en el desarrollo de modelos capaces de razonar a través de conjuntos de documentos complejos, garantizando al mismo tiempo una fiabilidad y transparencia que permita su plena integración en los procesos de negocio de mayor valor.

2.3. Posicionamiento del Trabajo y Contribuciones Propias

Tras la revisión de la literatura científica y las tecnologías que definen el estado del arte, esta sección tiene como objetivo conectar dicho marco general con la investigación concreta y aplicada de este TFG. Se busca posicionar el trabajo de manera explícita, contrastando su metodología y alcance con las tendencias generales para, así, destacar su enfoque único y sus aportaciones distintivas al campo del Procesamiento Inteligente de Documentos (IDP) mediante Modelos Extensos de Lenguaje (LLM).

El estado del arte actual está marcado por un intenso esfuerzo en el desarrollo de nuevas arquitecturas (Mamba, MoE), la optimización de métodos de entrenamiento (QLoRA) y la evaluación del rendimiento en *benchmarks* académicos de gran escala (DUE, VRDU). Si bien esta investigación fundamental expande las fronteras de lo posible, a menudo deja sin respuesta preguntas críticas sobre la aplicabilidad práctica de los modelos existentes en entornos de producción.

Este TFG se diferencia al adoptar un enfoque de validación empírica y aplicada. La contribución principal aquí no es la creación de una nueva arquitectura, sino la evaluación rigurosa y a gran escala de modelos de vanguardia de propósito general (Gemini 1.5 Pro, Mistral-small) en un corpus de documentos del mundo real, híper-específico y de alta relevancia industrial (facturas del sector eléctrico español). Mientras que los *benchmarks* académicos miden el rendimiento en una amplia variedad de tareas, este trabajo proporciona una visión granular y pragmática que dichos *benchmarks* no se centran en ofrecer. Responde a la pregunta fundamental sobre la viabilidad, robustez y los umbrales de rendimiento de estas tecnologías en un caso de uso concreto, generando conocimiento directamente aplicable para la industria.

2.3.1. Profundización en la Ingeniería de Instrucciones

La literatura identifica correctamente el *prompting* y el aprendizaje *zero-shot/few-shot* como una tendencia transformadora que reduce drásticamente los costes de anotación. No obstante, a menudo lo presenta como una capacidad inherente y algo monolítica de los modelos, sin desglosar sistemáticamente los factores que determinan su éxito.

Este TFG eleva la ingeniería de instrucciones de una simple técnica a una variable experimental central. La aportación más significativa de este trabajo es el diseño, la implementación y la medición cuantitativa de un abanico de estrategias de *prompting*: desde un *zero-shot* como línea base, pasando por múltiples variantes de *few-shot* con validación cruzada, hasta una compleja estrategia de extracción iterativa basada en la descomposición de tareas. El estudio aporta datos empíricos robustos (basados en métricas de *Precision*, *Recall* y *F1-Score* sobre miles de ejecuciones) que demuestran cómo y en qué medida la formulación detallada de la instrucción impacta directamente en la calidad y fiabilidad de la extracción. Así, se pasa de la afirmación cualitativa de que el *few-shot* funciona bien, a una guía cuantitativa sobre las mejores prácticas para su implementación.

2.3.2. Análisis Sistemático de la Configuración del Modelo para Tareas de Alta Fidelidad

El estado del arte aborda desafíos críticos como la mitigación de alucinaciones, proponiendo a menudo soluciones arquitectónicas o metodológicas complejas como la Genera-

ción Aumentada por Recuperación (RAG) o el ajuste fino. Sin embargo, se presta menos atención al impacto de la configuración más fundamental del modelo en el momento de la inferencia.

Este trabajo aborda el problema de la fiabilidad desde una perspectiva más básica y controlada: el análisis sistemático de los parámetros de inferencia (temperatura, `Top-p`, `Top-k`). La contribución aquí es proporcionar evidencia empírica clara, a través de 19 configuraciones experimentales, sobre cómo la configuración de estos parámetros afecta al equilibrio entre coherencia y precisión factual en una tarea que no tolera la creatividad ni los errores. Se demuestra que, para este dominio, la estructura del documento y la calidad del *prompt* dominan abrumadoramente sobre los parámetros de generación, un hallazgo pragmático que ofrece una guía práctica sobre dónde deben invertirse los esfuerzos de optimización en proyectos de implementación real.

En definitiva, este Trabajo de Fin de Grado se posiciona como un puente metodológico importante entre la investigación teórica de vanguardia y la aplicación práctica y rigurosa de los LLM en el ámbito empresarial. El trabajo toma conscientemente las capacidades y promesas generales identificadas en el estado del arte, el poder del aprendizaje en contexto y la flexibilidad de los modelos de propósito general, y las somete a una exhaustiva prueba de evaluación, diseñada para responder no solo si la tecnología funciona, sino cómo, por qué y bajo qué condiciones lo hace de manera óptima.

Esta prueba se articula en los tres niveles de análisis que, en conjunto, forman su contribución original. Primero, se ancla la investigación en la realidad de un dominio industrial específico, aportando una validación empírica que trasciende la abstracción de los *benchmarks* académicos. Segundo, se eleva la ingeniería de instrucciones a una disciplina científica, diseccionando y cuantificando cómo la estructura de la comunicación con el modelo se convierte en el factor más determinante del éxito. Finalmente, se desmitifica el impacto de los hiperparámetros de inferencia, demostrando empíricamente que su influencia es secundaria frente a la calidad de los datos de entrada y la inteligencia del prompt.

Por lo tanto, el valor añadido de esta investigación no reside en la creación de una nueva arquitectura, sino en la generación de conocimiento accionable. Ofrece un plano de ruta validado para ingenieros, desarrolladores y analistas de negocio, estableciendo una jerarquía clara de prioridades: la excelencia en la preparación de los datos y en el diseño de las instrucciones es fundamental, mientras que el ajuste de los parámetros de generación ofrece un rendimiento decreciente para este tipo de tareas.

En última instancia, este TFG no es un punto final, sino un manual de implementación fundamentado; un estudio que demuestra cómo pasar de la mera constatación de las capacidades de los LLM a la construcción de soluciones fiables, eficientes y medibles para la automatización de procesos de negocio en el mundo real.

Capítulo 3

Metodología y Planificación del Trabajo

En este capítulo se describe la estructura metodológica y la planificación que han guiado el desarrollo de este Trabajo de Fin de Grado. El objetivo es ofrecer una visión completa del proceso seguido, desde el marco de trabajo general hasta la organización práctica de cada una de las etapas completadas.

Para empezar, se detallará el marco metodológico seleccionado, el *CRoss-Industry Standard Process for Data Mining* (CRISP-DM). Se explicará cómo cada una de sus seis fases ha sido adaptada a las particularidades de este proyecto.

A continuación, la segunda parte del capítulo se centrará en la planificación práctica de las tareas. Se presentará tanto el cronograma inicial propuesto al comienzo del proyecto como el plan de trabajo final que refleja las labores realmente ejecutadas.

3.1. Metodología

Para la realización de este Trabajo de Fin de Grado se ha adoptado como marco metodológico CRISP-DM [Shearer, 2000]. Esta elección se debe a que CRISP-DM es el estándar de facto en la industria para el desarrollo de proyectos de ciencia de datos y minería de datos, proporcionando un enfoque estructurado, flexible e iterativo que guía el proyecto desde la concepción del problema hasta su despliegue final.

CRISP-DM descompone el ciclo de vida de un proyecto de minería de datos en seis fases principales, como se ilustra en la Figura 3.1. Dichas fases son: Comprensión del Negocio, Comprensión de los Datos, Preparación de los Datos, Modelado, Evaluación y Despliegue [Shearer, 2000]. Aunque el modelo presenta una secuencia lógica, su naturaleza es cíclica e iterativa, permitiendo retroceder a fases anteriores a medida que se obtienen nuevos conocimientos. A continuación, se detalla cómo cada una de estas fases ha sido adaptada y aplicada en el contexto de este proyecto.

3.1.1. Comprensión del Negocio

Esta fase inicial se centra en definir los objetivos del proyecto desde una perspectiva de negocio y traducirlos a un problema técnico de minería de datos. El objetivo principal, como se ha mencionado previamente en la redacción de esta memoria, es la automatización de la extracción de información clave de facturas del sector eléctrico español. Se busca

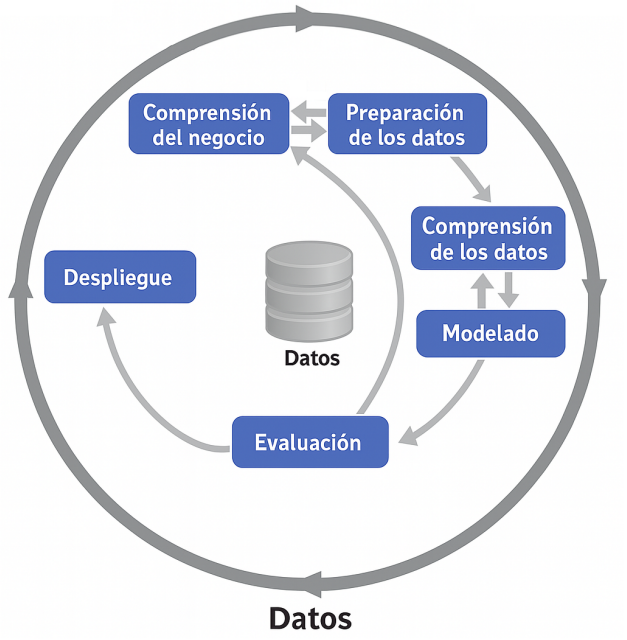


Figura 3.1: Metodología CRoss-Industry Standard Process for Data Mining (CRISP-DM) Fuente: Modificado de [Wikipedia, 2024]

reducir la carga de trabajo manual, minimizar errores y estructurar los datos para su posterior análisis o integración en otros sistemas.

El objetivo de la fase de minería de datos, se traduce de manera técnica en modelar las instrucciones y los parámetros para que modelos extensos de lenguaje, los cuales han sido entrenados sobre grandes cantidades de corpus de información, sean capaces de identificar y extraer con precisión un conjunto predefinido de campos a partir del texto de las facturas proporcionadas. La motivación detrás de centrarse concretamente en este tipo de documento de negocio, como es la factura de la luz, viene de la calidad del conjunto de datos de datos IDSEM, que facilita la evaluación de esta tarea, y proporciona una base sólida sobre la que realizar la evaluación de este proceso.

El éxito del proyecto se evaluará cuantitativamente mediante métricas de rendimiento estándar en tareas de extracción de información. Para obtener una visión detallada del rendimiento, se calcularán la *Precision*, *Recall*, y la *F1-score* para cada uno de los campos semánticos definidos.

Complementariamente, esta evaluación numérica se enriquecerá con un análisis de errores, diseñado para obtener una comprensión más profunda del comportamiento del modelo. Este análisis se centrará en identificar los patrones de fallo más comunes, como los campos que se confunden con mayor frecuencia o aquellos cuya extracción resulta sistemáticamente más difícil. Asimismo, se tendrá en cuenta la influencia de la diversidad estructural entre las distintas plantillas de factura para determinar cómo la organización del documento impacta en la precisión de los resultados.

3.1.2. Comprensión de los Datos

En la etapa de comprensión de los datos se realizan actividades para identificar, recopilar, describir, explorar y verificar la calidad del conjunto de datos para familiarizarse con el mismo. Además, estas actividades permiten encontrar problemas en los mismos, e

identificar subconjuntos de datos que revelen información que permita formular nuevas hipótesis. Esto ayudará a lograr la consecución del trabajo y los objetivos planteados en la fase anterior.

Este paso se ve muy simplificado al emplear el dataset IDSEM [Sánchez et al., 2022]. No fue necesaria una recopilación manual de facturas, sino un estudio de la documentación y estructura del corpus proporcionado en el conjunto de datos. Este conjunto de datos contiene las características necesarias para abordar los objetivos del trabajo, debido a que representa un tipo de documento de negocio concreto como es una factura de la luz. Este conjunto de datos ofrece 75,000 documentos divididos en 9 plantillas diferentes que han sido creadas a partir de facturas reales de las principales empresas del sector en España. Además, se han usado diccionarios con datos reales como nombres y apellidos españoles, calles y direcciones de municipios españoles, listados de empresas comercializadoras y distribuidoras registradas en la Comisión Nacional de los Mercados y la Competencia (CNMC).

Para una mayor comprensión de los datos, se analizó la estructura del conjunto de facturas, identificando la variedad de plantillas, la definición de las 107 etiquetas, la correspondencia entre los ficheros PDF y los archivos JSON que contienen los valores reales etiquetados en las facturas, y las distribuciones estadísticas de los datos simulados. Después de este análisis, podemos conlcuir que este extenso conjunto de datos se alinea con los objetivos del proyecto y su desarrollo.

En el capítulo 5 se profundiza más en el conjunto de datos utilizado en el proyecto, tanto en su estructura y contenido, como en el proceso seguido para su creación.

3.1.3. Preparación de los Datos

En esta fase se tiene como objetivo construir el conjunto de datos definitivo para el modelado. Dicho procesamiento engloba la selección de características pertinentes, la depuración de los datos para eliminar inconsistencias, y su transformación para adecuarlos a los requerimientos del modelo.

En este caso, una vez tenemos el conjunto de datos de facturas de la luz proporcionados IDSEM [Sánchez et al., 2022], debemos preparar y transformar los datos para que el modelo extenso de lenguaje sea capaz de procesarlo de una manera eficiente y eficaz, maximizando su capacidad de comprensión de los datos y por lo tanto, su capacidad de extracción. Además, se realizarán anotaciones adicionales a las etiquetas proporcionadas por el conjunto de datos como *ground truth*, usadas para evaluar el rendimiento en la extracción del modelo.

La tarea de preparación principal consistió en convertir las facturas, originalmente en formato PDF, a texto, utilizando el lenguaje de marcado Markdown. La elección del formato Markdown sobre el texto plano se fundamenta en su capacidad para preservar la estructura semántica y visual de los documentos originales. Mediante una sintaxis simple, este formato representa explícitamente elementos clave como títulos, listas y párrafos. Esta decisión se apoya en investigaciones que demuestran que proporcionar a los modelos extensos de lenguaje esta estructura textual, en lugar de una secuencia de texto simple, mejora significativamente su capacidad para comprender la estructura y las relaciones jerárquicas del documento. [Braun et al., 2025].

La herramienta de conversión usada para pasar de PDF a Markdown no extrae ciertos campos de datos presentes en el PDF original debido a la imposibilidad de convertir texto situado en los bordes del documento, imágenes o gráficas. Estas etiquetas se han anotado

para no ser evaluadas en la extracción del modelo, ya que empeoraría el rendimiento de este de manera sistemática. Además, de las 9 plantillas diferentes de facturas presentes en el conjunto de datos, cada una de ellas tiene un subconjunto diferente dentro de las 107 etiquetas. Por ello, se creó un mapeo explícito que documenta, para cada una de las plantillas de factura, qué subconjunto de las 107 etiquetas posibles está realmente presente. Este registro, junto al filtrado anterior, es indispensable para poder realizar una evaluación justa y precisa de los modelos.

3.1.4. Modelado

En la etapa de modelado, el objetivo es seleccionar la técnica de modelado a aplicar, configurando y calibrando sus parámetros, optimizando sus valores para obtener los mejores resultados posibles.

Dado que este proyecto se apoya en el uso de Modelos Extensos de Lenguaje (LLM) pre-entrenados, la fase de modelado de CRISP-DM se interpreta no como la creación de una arquitectura desde cero, sino como la aplicación y calibración de estos modelos existentes para la tarea específica de extracción de información.

El enfoque principal del modelado consistió en calibrar y guiar el comportamiento de los modelos de lenguaje seleccionados. Esta estrategia se implementó a través de dos ejes de experimentación:

- Diseño de *Prompts*: Se aplicaron diversas técnicas de ingeniería de *prompts* o instrucciones (*prompt engineering*) para crear un conjunto de instrucciones variadas. Cada instrucción fue diseñada para guiar al modelo de una manera específica hacia la tarea de extracción de información.

- Ajuste de Parámetros de Inferencia: Para cada instrucción diseñada, se evaluaron sistemáticamente diferentes combinaciones de los parámetros clave que controlan la generación de texto: temperatura, `Top-p` y `Top-k`. Estos parámetros son cruciales, ya que modulan el proceso de muestreo de *tokens* para ajustar el nivel de creatividad y determinismo en las respuestas del modelo [Renze and Guven, 2024, Brown and Mann, 2020].

Se considera que cada combinación de *prompt* y configuración de parámetros representa una instancia de modelo distinta. Para cada instrucción, se realizó un proceso iterativo en el que se evaluaba su funcionamiento. En función de los resultados de varias extracciones, se refinaba el *prompt*, hasta llegar al resultado deseado para su posterior evaluación en los experimentos. Los detalles sobre las formulaciones de las instrucciones, los valores de los parámetros evaluados y la justificación de cada configuración experimental se exponen en el siguiente capítulo, dedicado a la configuración de los experimentos.

3.1.5. Evaluación

La fase de evaluación se centra en la validación rigurosa tanto del proceso de construcción del modelo como de sus resultados cuantitativos. El objetivo primordial es verificar la alineación de dichos resultados con los objetivos de negocio definidos al inicio del proyecto. Es necesario comprobar si algún factor crítico de negocio fue omitido durante el ciclo de desarrollo. Finalmente, esta fase culmina con una decisión estratégica sobre los siguientes pasos: proceder a la fase de implementación, iniciar una nueva iteración para refinar el modelo, o descartar la solución actual y reiniciar el proyecto.

En este caso, la evaluación de los modelos se realizó con especial atención a las particularidades del conjunto de datos y del proceso de preparación. Como mencionamos en el apartado de preparación de los datos, después de realizar un procesamiento de las facturas se anotó qué etiquetas son las que se conservan en cada plantilla de las facturas post-procesadas. Una vez se tienen los valores reales con los que comparar la extracción para poder evaluarla, se usaron métricas estándar para evaluar el rendimiento de los modelos. Usamos las métricas de *Precision*, *Recall* y *F1-score* para medir la extracción de cada etiqueta.

Para formalizar estas métricas, es necesario primero definir los posibles resultados de una extracción para un campo determinado:

- **Verdadero Positivo (TP - True Positive):** El modelo extrae un valor para una etiqueta y este coincide con el valor real en el documento.
- **Falso Positivo (FP - False Positive):** El modelo extrae un valor para una etiqueta, pero este es incorrecto o no correspondía a esa etiqueta. Es un error de extracción.
- **Falso Negativo (FN - False Negative):** El modelo no extrae ningún valor para una etiqueta, a pesar de que esta existía en el documento. Es un error de omisión.

A partir de estos conceptos, las métricas se definen de la siguiente manera [Sokolova and Lapalme, 2009]:

La precision mide la calidad de las extracciones realizadas. De todos los valores que el modelo ha identificado y extraído para una etiqueta, la *Precision* calcula qué fracción de ellos eran realmente correctos. Una *Precision* alta indica que el modelo es fiable y comete pocos errores al afirmar que ha encontrado un dato.

El *recall* mide la cantidad de verdaderos positivos que encuentra el modelo, es decir, la cantidad de datos correctos que extrae. Un recall alto indica que el modelo es capaz de extraer bastantes datos correctos de las facturas.

El *F1-Score* proporciona un único indicador de rendimiento que equilibra la precision y el recall. Es la media armónica de ambas, penalizando los casos en los que una de las dos métricas es muy baja. Un *F1-score* alto indica un buen equilibrio entre la calidad y la cantidad de las extracciones.

3.1.6. Despliegue

En este caso, la fase de despliegue no implica la implementación de un sistema en un entorno de producción. En este contexto, el despliegue se orientara a la organización y presentación del conocimiento adquirido derivado de la realización de las pruebas ejecutadas y los resultados presentados. Esto incluye la redacción del informe final del proyecto, la exposición detallada de sus resultados de evaluación y la discusión de las conclusiones y las posibles futuras líneas de trabajo derivadas de la investigación.

3.2. Planificación

En lo que a la planificación de trabajo respecta, este plan debe reflejar la distribución en fases y sus respectivas tareas que se han dedicado para la realización del presente trabajo, tanto en la planificación inicial, como en la planificación real de las tareas que se ejecutaron durante el proyecto.

3.2.1. Planificación Inicial

En este apartado se describirá y detallará una estimación del contenido y las tareas abordadas en las distintas fases del proyecto. En la imagen 3.2, se muestra un diagrama que contiene las diferentes fases de nuestro trabajo, con sus respectivas tareas. La flecha bidireccional representa la naturaleza iterativa de la metodología CRISP-DM, en la que el desarrollo del trabajo no se completa de manera lineal, si no en un proceso cíclico e iterativo entre las diferentes fases. En la Tabla 3.1, se muestra con más detalle la distribución de fases y tareas inicialmente asignadas al proyecto junto a una estimación del tiempo requerido para completarlas frente al tiempo realmente invertido en cada una de las fases.

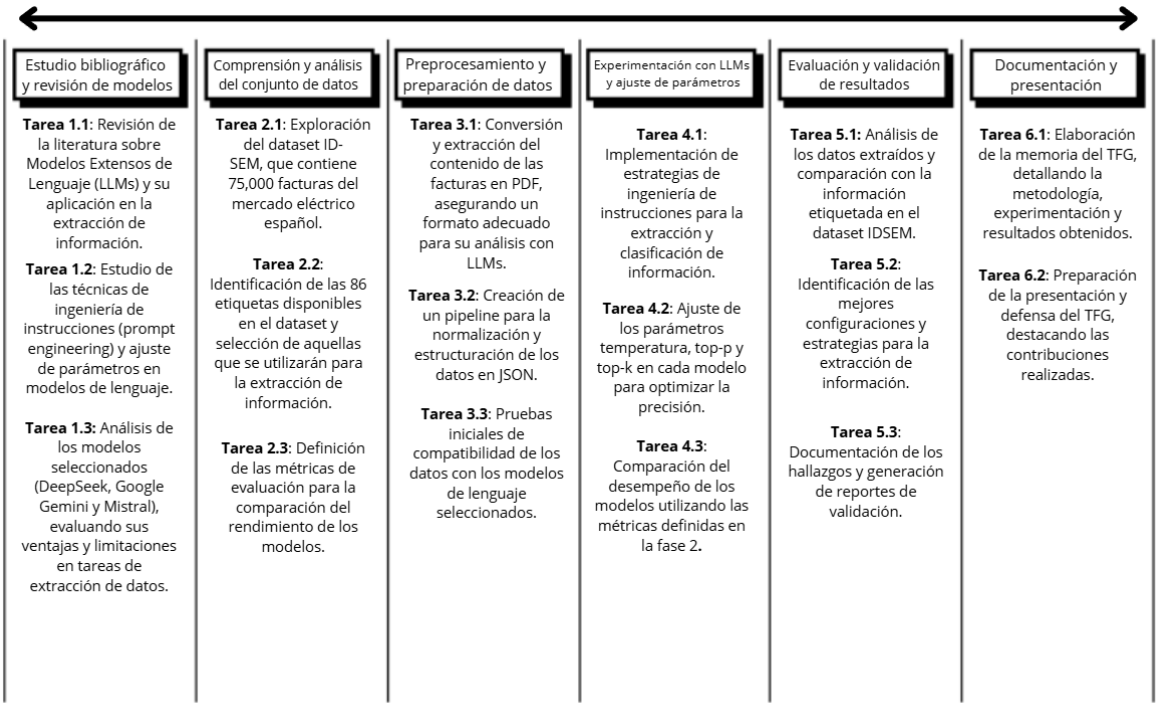


Figura 3.2: Diagrama de tareas planificadas basado en la Metodología CRISP-DM.

Tabla 3.1: Estimación de las fases y tareas del proyecto.

Fases	**Duración Estimada**	**Duración real**	**Tareas**
Estudio bibliográfico y revisión de modelos	40 horas	55 horas	Tarea 1.1: Revisión de la literatura sobre Modelos Extensos de Lenguaje (LLM) y su aplicación en la extracción de información. Tarea 1.2: Estudio de las técnicas de ingeniería de instrucciones (*prompt engineering*) y ajuste de parámetros en modelos de lenguaje.

Tabla 3.1– Continuación

Fases	Duración Estimada	Duración real	Tareas
Comprensión y análisis del conjunto de datos	40 horas	55 horas	Tarea 1.3: Análisis de los modelos seleccionados (DeepSeek, Google Gemini y Mistral), evaluando sus ventajas y limitaciones en tareas de extracción de datos. Tarea 2.1: Exploración del dataset IDSEM, que contiene 75,000 facturas del mercado eléctrico español. Tarea 2.2: Identificación de las 86 etiquetas disponibles en el dataset y selección de aquellas que se utilizarán para la extracción de información. Tarea 2.3: Definición de las métricas de evaluación para la comparación del rendimiento de los modelos.
Preprocesamiento y preparación de datos	50 horas	65 horas	Tarea 3.1: Conversión y extracción del contenido de las facturas en PDF, asegurando un formato adecuado para su análisis con LLM. Tarea 3.2: Creación de un *pipeline* para la normalización y estructuración de los datos en JSON. Tarea 3.3: Pruebas iniciales de compatibilidad de los datos con los modelos de lenguaje seleccionados.
Experimentación con LLM y ajuste de parámetros	80 horas	80 horas	Tarea 4.1: Implementación de estrategias de ingeniería de instrucciones para la extracción y clasificación de información. Tarea 4.2: Ajuste de los parámetros temperatura, `Top-p` y `Top-k` en cada modelo para optimizar la precisión. Tarea 4.3: Comparación del desempeño de los modelos utilizando las métricas definidas en la fase 2. Tarea 4.4: Análisis del impacto de los diferentes ajustes en la calidad de la extracción de información.
Evaluación y validación de resultados	40 horas	40 horas	Tarea 5.1: Análisis de los datos extraídos y comparación con la información etiquetada en el dataset IDSEM. Tarea 5.2: Identificación de las mejores configuraciones y estrategias para la extracción de información.

Tabla 3.1– Continuación

Fases	Duración Estimada	Duración real	Tareas
			Tarea 5.3: Documentación de los hallazgos y generación de reportes de validación.
Documentación y presentación	50 horas	50 horas	Tarea 6.1: Elaboración de la memoria del TFG, detallando la metodología, experimentación y resultados obtenidos. Tarea 6.2: Preparación de la presentación y defensa del TFG, destacando las contribuciones realizadas.

Con respecto al desarrollo de este trabajo, todas las fases y sus tareas han sido realizadas siguiendo la descripción establecida en la Tabla 3.1. El único cambio a mencionar es el orden de la ejecución de las tareas, ya que estas se realizaban de manera iterativa, y el tiempo tardado en realizar algunas tareas debido a imprevistos y problemas técnicos. El siguiente apartado expone la justificación de la extensión de tiempo para ciertas tareas con respecto a la planificación inicial

3.2.2. Tareas Realizadas

Como se ha comentado, la consecución de los objetivos del proyecto se adhirió al plan de trabajo inicial presentado en la Tabla 3.1, aunque sin mantener un orden estrictamente secuencial debido a la naturaleza iterativa de la investigación. Adicionalmente, la duración real de algunas fases excedió la estimación inicial debido a varios desafíos imprevistos, justificados a continuación.

Estudio y Selección de Modelos. En el caso de la fase de estudio bibliográfico y revisión de modelos, concretamente en la tarea 1.3, la selección del modelo no solo se basó en su capacidad técnica para la extracción de datos, sino que estuvo fuertemente condicionada por el factor económico. Los modelos extensos de lenguaje basados en la nube tienen costes asociados al uso de sus APIs, y la necesidad de procesar un gran volumen de facturas imponía una limitación importante. La mayoría de los proveedores de APIs de modelos de última generación eran de pago o, si ofrecían un plan gratuito, este tenía límites de frecuencia muy restrictivos. Por lo tanto, la búsqueda de un modelo que se encontrara en el estado del arte, pero que al mismo tiempo ofreciera límites de uso gratuitos y generosos, prolongó esta tarea más de lo planificado. Por ello, a pesar de que se planificó usar DeepSeek como uno de los modelos a investigar en este trabajo, debido a los pocos servicios que lo ofrecían y a sus servicios gratuitos extremadamente limitados se descartó como opción de experimentación.

Sumado a esto, la rápida evolución del campo de los LLM introdujo otra variable. Durante los meses de desarrollo, los modelos eran actualizados, (con más parámetros o cambios en su arquitectura), lo que dificultaba la realización de pruebas consistentes a largo plazo con una única versión del modelo, obligando a reevaluar constantemente los modelos disponibles hasta encontrar uno estable.

Preprocesamiento y Conversión de Documentos. La duración de la fase de preprocesamiento y preparación de datos, y en concreto de la Tarea 3.1, también se vio extendida por algunos desafíos técnicos. El requisito fundamental de esta tarea era convertir los documentos del formato PDF original a un formato textual que pudiera ser procesado por las APIs de los LLM, ya que estas no suelen admitir ficheros PDF directamente.

Inicialmente, se optó por utilizar un servicio API de terceros que realizaba una conversión a texto manteniendo la estructura visual del PDF, lo que en un principio parecía la solución ideal. Sin embargo, a mitad del proceso de conversión del subconjunto de datos del proyecto, el proveedor de dicha API restringió el acceso sin previo aviso. Se presume que esto se debió al alto volumen de uso diario, a pesar de que siempre se respetaron los límites de frecuencia establecidos en su documentación.

Este bloqueo imprevisto obligó a buscar una vía alternativa. Durante esta nueva fase de investigación, un artículo reciente destacó las ventajas del formato Markdown para el procesamiento por parte de los LLM, suponiendo un 23 % más de precisión respecto a textos escaneados por herramientas comerciales como AzureOCR en tareas que implican comprensión del documento y extracción de información [Braun et al., 2025]. Esto nos llevó a decantarnos finalmente por una librería de conversión de PDF a Markdown. Aunque esta librería ya se había considerado al principio, se descartó inicialmente debido a sus limitaciones conocidas, como la incapacidad de extraer texto de gráficas o de los bordes del documento. No obstante, tras el fallo de la primera estrategia, se convirtió en la solución más viable.

Paralelamente, también se exploró otra alternativa para una de las plantillas más complejas del conjunto de datos: se probaron y refinaron métodos de procesamiento de imágenes y reconocimiento óptico de caracteres (OCR). Sin embargo, los resultados ofrecidos por este enfoque fueron demasiado variables y no alcanzaron la precisión necesaria para realizar las pruebas deseadas. Dado que esta solución solo era aplicable a una de las nueve plantillas y su rendimiento era insuficiente, también fue descartada.

En definitiva, la implementación y posterior fallo de la herramienta inicial, la investigación de nuevas alternativas y la adopción y adaptación a una segunda solución, fue la causa principal por la que esta tarea excedió la estimación de tiempo original.

Comprensión y Refinamiento de Etiquetas. La fase de comprensión y análisis del conjunto de datos, y en particular la Tarea 2.2, también requirió más tiempo del previsto. La necesidad de esta extensión no se hizo evidente al principio, sino durante la fase final de evaluación y validación de resultados, lo que demuestra la naturaleza iterativa del proyecto.

Al analizar los resultados de las extracciones, se detectó que los modelos de lenguaje mostraban dificultades recurrentes con ciertas etiquetas. No las extraían correctamente debido a que no comprendían su finalidad dentro del documento. Una investigación más profunda reveló dos causas principales. Primero, la descripción oficial proporcionada por el dataset para algunas de las 107 etiquetas no era la correcta, o no era lo suficientemente precisa o contextual para que un LLM pudiera extraerla correctamente. Segundo, se identificaron campos que, debido a la naturaleza sintética del dataset, estaban anotados como existentes en la factura pero no contenían ningún valor (estaban "vacíos"), generando ambigüedad para el modelo.

Para solucionar esto, fue necesario regresar a la fase de comprensión de datos y llevar a cabo una revisión manual de cada una de las 107 etiquetas. El objetivo era verificar si la descripción proporcionada era suficiente para que el modelo entendiera el propósito del

campo y, en caso contrario, enriquecerla. Además, se revisó la incidencia de los campos vacíos para confirmar que se trataba de casos aislados y no de un problema sistémico. Este ciclo imprevisto, que nos llevó desde la fase de Evaluación de vuelta a la de Comprensión de Datos, fue un paso indispensable para asegurar la calidad del análisis y es un ejemplo de la flexibilidad usada en la metodología CRISP-DM.

Capítulo 4

Recursos Empleados en el Desarrollo

Este capítulo presenta una descripción detallada de los recursos técnicos utilizados en el proyecto, divididos en hardware y software.

La primera sección se centra en el hardware, explicando las características de los equipos informáticos empleados y la función que cada uno ha cumplido en el desarrollo del trabajo.

La segunda sección, dedicada al software, se desglosa en varias subsecciones. Se comienza por justificar la elección del lenguaje de programación y a continuación se explica el uso de Interfaces de Programación de Aplicaciones (APIs) para la comunicación con los modelos de lenguaje, se presenta un listado de las librerías utilizadas y, finalmente, se describen los entornos de desarrollo en los que se ha trabajado.

4.1. Hardware

En cuanto a los recursos hardware, las tareas de análisis y comprensión del dataset, así como la planificación, búsqueda de información y creación de scripts y programas completadas durante el desarrollo del proyecto han sido desarrolladas desde un ordenador portátil personal. Este ordenador portátil cumple con las características presentes en la Tabla 4.1. La ejecución de los programas desarrollados para realizar la tarea de extracción de facturas mediante la API de los modelos requerían de largos tiempos de ejecución, por lo que fueron ejecutados en el ordenador del laboratorio, ya que este se mantenía encendido todo el día. Este ordenador cumple con las características descritas en la Tabla 4.2.

Tabla 4.1: Especificaciones del equipo 1 utilizado en el proyecto.

Componente	Especificación
Sistema Operativo	Microsoft Windows 10 Pro (Versión 10.0.19045)
Fabricante	ASUS
Modelo	System Product Name
CPU	Intel® Core™ i9-10900F @ 2.80GHz
GPU	NVIDIA GeForce RTX 2060 SUPER
Memoria RAM	32,0 GB
Disco Principal	1,82TB

Tabla 4.2: Especificaciones del equipo secundario utilizado en el proyecto.

Componente	Especificación
Sistema Operativo	Microsoft Windows 10 Pro (Versión 10.0.19045)
Fabricante	HP
Modelo	OMEN by HP Laptop 17-cb0xxx
CPU	Intel® Core™ i7-9750H @ 2.60GHz
GPU	NVIDIA GeForce RTX 2060
Memoria RAM	16,0 GB
Disco Principal	475,71 GB
Disco Secundario	894,84 GB

4.2. Software

A continuación se presentan los recursos software usados para la consecución del proyecto. Se detallará cómo se usaron y el por qué de su selección para el desarrollo de éste.

4.2.1. Lenguaje de Programación

El lenguaje de programación seleccionado para la totalidad del desarrollo experimental ha sido Python, concretamente en su versión 3.11. La elección de Python ha sido debido a que es un lenguaje muy usado en el ecosistema de la ciencia de datos debido a su utilidad en tareas como análisis, procesamiento y transformación de datos. Su sintaxis, caracterizada por su sencillez y legibilidad, reduce significativamente los tiempos de desarrollo y facilita la implementación de sistemas complejos.

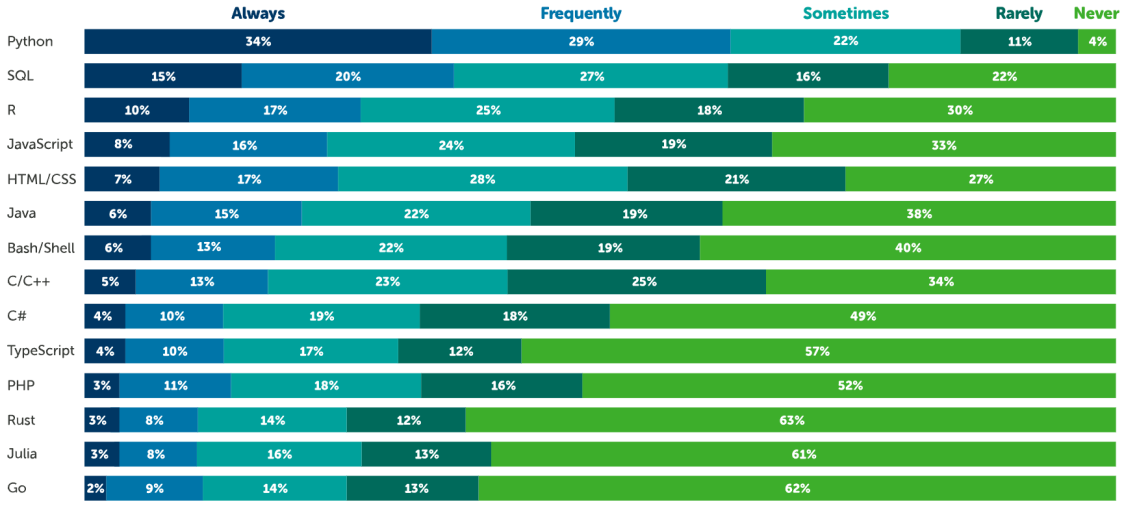


Figura 4.1: Encuesta sobre la popularidad de los diferentes lenguajes de programación. Fuente: [Anaconda, 2022]

.

Desde su concepción inicial por Guido van Rossum en 1991, Python ha experimentado una adopción masiva, consolidándose como una herramienta indispensable en los ámbitos científico y académico. Su popularidad se ha visto impulsada por una comunidad activa

y una filosofía de código abierto que ha fomentado la creación de un vasto ecosistema de recursos. En el contexto actual de la ciencia de datos, Python se erige como el lenguaje predominante. Como se puede observar en la Figura 4.1 encuestas como la realizada por Anaconda en 2022 [Anaconda, 2022], a 3104 científicos de datos, investigadores y profesionales del sector, reflejan esta realidad: más de la mitad de los profesionales lo utilizan de manera habitual en sus flujos de trabajo, y solo un 4 % determina no haberlo usado nunca.

Una de las fortalezas más significativas de Python, y un factor clave para su elección en este proyecto, es su extenso repertorio de librerías y módulos especializados. Estas librerías, de fácil instalación y muy bien documentadas , proporcionan facilidad en el desarrollo de una amplia gama de tareas. En este proyecto, Python ha sido usado para la el análisis y transformación del conjunto de datos, la comunicación con las Interfaces de programación de aplicaciones(APIs) de los modelos, la transformación y tratamiento de los datos obtenidos de estos modelos y la creación y visualización de los resultados.

4.2.2. Interfaces de Programación de Aplicaciones

Para la comunicación con los modelos extensos de lenguaje (LLM) se han utilizado sus Interfaces de Programación de Aplicaciones (APIs). Una API es un conjunto de reglas y herramientas que permite que diferentes programas de software se comuniquen entre sí, actuando como un intermediario que define cómo se deben solicitar y recibir los datos o servicios, en este caso, proporcionando la comunicación con el modelo. La naturaleza de los modelos empleados en este estudio es de sistemas basados en la nube (cloud-based). Su tamaño y el coste computacional asociado a su funcionamiento impiden su alojamiento y ejecución en equipos locales.

Los proveedores de estos modelos habilitan dos vías de acceso: una interfaz web para la interacción directa y un servicio API para la integración programática. El objetivo de este trabajo requería el procesamiento de un gran volumen de facturas, lo que demandaba la automatización del proceso de extracción. Por este motivo, se optó por el uso de la API mediante scripts de Python. En este proyecto, la API ha sido el mecanismo para enviar las instrucciones (prompts) con los datos de las facturas y recibir los resultados de la extracción de los documentos de forma estructurada. Concretamente se han usado dos proveedores de servicios API:

- Google Cloud: proporciona la comunicación con el modelo Gemini 1.5 Pro.
- Mistral AI: proporciona la comunicación con el modelo Mistrall-small.

El uso de estas APIs comerciales conlleva un coste económico asociado, el cual se calcula en función de la cantidad de *tokens* procesados. Los *tokens* son las unidades de texto que componen tanto la entrada de datos enviada al modelo, como la respuesta generada por este. Para este proyecto, el acceso al modelo Gemini 1.5 Pro fue financiado mediante un crédito de 263 € que Google Cloud Platform proporciona a cuentas de nueva creación, cubriendo así los costes derivados de su utilización [Cloud, 2025b]. Por su parte, Mistral AI ofrece acceso a sus modelos de mayor tamaño a través de una API de pago, pero también proporciona modelos más reducidos, como Mistral-small, sin coste. El plan gratuito de Mistral establece un límite de consumo mensual de mil millones de tokens, cifra que permitió la utilización de dicho modelo en el proyecto sin costes reales [Mistral, 2025b].

4.2.3. Librerías

Como mencionamos en el apartado previo, Python proporciona una gran cantidad de librerías y módulos muy bien documentados, sencillos de instalar y usar. A continuación, se listarán las librerías y módulos usados en el desarrollo de este proyecto, concretamente en las tareas de análisis y transformación del conjunto de datos, la comunicación con los modelo a través de la API y la creación y visualización de los resultados.

A continuación, se detallan las librerías y módulos de Python que han sido fundamentales para el desarrollo de este proyecto.

Librerías de la Biblioteca Estándar de Python.

- `os`: Módulo que proporciona una manera de usar funcionalidades dependientes del sistema operativo, como interactuar con el sistema de archivos (crear directorios, listar ficheros, etc.) [Foundation, 2025b].
- `argparse`: Utilizado para escribir interfaces de línea de comandos de manera sencilla. Permite definir los argumentos que requiere un programa y los procesa automáticamente a partir de la entrada del sistema [Foundation, 2025a].
- `json`: Proporciona las herramientas necesarias para trabajar con datos en formato JSON (JavaScript Object Notation), permitiendo la codificación (serialización) y decodificación (deserialización) de objetos de Python [Python, 2025b].
- `re`: Ofrece operaciones de coincidencia con expresiones regulares, permitiendo realizar búsquedas y manipulaciones de cadenas de texto avanzadas basadas en patrones [Python, 2025d].
- `pathlib`: Proporciona un enfoque orientado a objetos para interactuar con las rutas del sistema de archivos, facilitando su manipulación de manera independiente del sistema operativo [Python, 2025c].
- `time`: Suministra diversas funciones relacionadas con el tiempo, como obtener la hora actual o suspender la ejecución de un programa [Python, 2025f].
- `sys`: Permite el acceso a variables y funciones específicas del sistema que son mantenidas por el intérprete de Python, como los argumentos de la línea de comandos [Python, 2025e].
- `types`: Define nombres para varios de los tipos de objetos internos de Python y proporciona utilidades como la clase `SimpleNamespace` para agrupar atributos [Python, 2025g].
- `itertools`: Implementa un conjunto de herramientas para crear y manejar iteradores de manera eficiente [Python, 2025a].

Librerías de Procesamiento de Documentos.

- `pymupdf4llm`: Librería construida sobre `PyMuPDF`, diseñada específicamente para extraer texto de documentos PDF en un formato optimizado (como Markdown) para ser procesado por Modelos Extensos de Lenguaje (LLM) [PyPI, 2025].

Librerías de Análisis y Visualización de Datos

- `pandas`: Librería fundamental para la manipulación y el análisis de datos. Proporciona estructuras de datos de alto rendimiento y fáciles de usar, como el DataFrame, que es esencial para trabajar con datos tabulares [Pandas, 2024].
- `numpy`: Es el paquete fundamental para la computación científica en Python. Ofrece soporte para arrays y matrices multidimensionales de gran tamaño, junto con una amplia colección de funciones matemáticas de alto nivel para operar con ellos [NumPy, 2024].
- `scipy.stats`: Sub-módulo de la librería SciPy que contiene una gran cantidad de distribuciones de probabilidad y una creciente biblioteca de funciones estadísticas [SciPy, 2024].
- `seaborn`: Librería de visualización de datos basada en Matplotlib. Proporciona una interfaz de alto nivel para dibujar gráficos estadísticos atractivos e informativos [SeaBorn, 2024].
- `matplotlib.pyplot`: Es una colección de funciones que hacen que Matplotlib funcione como MATLAB. Cada función de pyplot realiza algún cambio en una figura, como crearla, añadir un área de trazado, trazar líneas, etc. [Matplotlib, 2024].
- `IPython.display`: Módulo utilizado dentro de entornos como Jupyter Notebook para mostrar objetos enriquecidos, como imágenes, sonido, vídeos o HTML, directamente en la salida de una celda [IPython, 2024].

Librerías para la Interacción con APIs de Modelos de Lenguaje.

- `google-generativeai`: Es la librería cliente oficial de Python para la API de Google Generative AI. Se utiliza para interactuar con los modelos de la familia Gemini [Google, 2025].
- `mistralai`: Es la librería cliente oficial de Python para la plataforma "La Plateforme" de Mistral AI, permitiendo el acceso y la comunicación con sus modelos de lenguaje a través de su API [Mistral, 2025a].

4.2.4. Entorno de Desarrollo

La selección de un entorno de desarrollo adecuado es un paso fundamental para garantizar la eficiencia en la implementación de cualquier proyecto. En este trabajo se optó por un flujo de trabajo que combina dos entornos complementarios, cada uno adaptado a una fase específica del desarrollo.

Visual Studio Code

Para la construcción y ejecución de los scripts de procesamiento y experimentación, se utilizó Visual Studio Code (versión 1.101.2)[Microsoft, 2025a]. Se trata de un editor de código fuente gratuito y de código abierto creado por Microsoft, que se ha consolidado como un estándar en la industria gracias a su rendimiento, su depurador integrado, el control de versiones con Git y un extenso mercado de extensiones que potencian su funcionalidad para múltiples lenguajes, incluido Python.

Jupyter Notebook

Para las fases de análisis exploratorio y visualización de resultados, se recurrió a `Jupyter Notebook` (versión 6.4.12)[Project, 2025]. Este entorno interactivo basado en web permite crear y compartir documentos que contienen código ejecutable, ecuaciones, visualizaciones y texto narrativo. Su principal ventaja, y la razón de su popularidad en ciencia de datos, es su capacidad para ejecutar código en celdas discretas, lo que permite un desarrollo iterativo y una rápida comprobación de los resultados. En el contexto de este proyecto, se utilizó específicamente para generar las gráficas y tablas que ilustran los datos y los resultados de las extracciones.

4.3. Análisis de la Pila Tecnológica y Alternativas Potenciales

La pila tecnológica seleccionada para este TFG, fundamentada en el lenguaje Python y su ecosistema de librerías (`pandas`, `pymupdf4llm`, etc.), junto con entornos de desarrollo como Visual Studio Code y Jupyter Notebook, representa un estándar de facto en la comunidad de ciencia de datos. Ha demostrado ser una elección acertada y perfectamente adecuada para la investigación, el prototipado rápido y la validación experimental que constituyen el núcleo de este trabajo. Sin embargo, un análisis desde la perspectiva de Machine Learning Operations (MLOps) y el desarrollo de software a escala puede revelar herramientas complementarias que, de integrarse, abordarían desafíos clave como la gestión avanzada de experimentos, la reproducibilidad estricta y la orquestación de flujos de trabajo complejos, preparando el proyecto para una transición más fluida hacia un entorno de producción.

Gestión y Trazabilidad de Experimentos. El proyecto implicó una considerable cantidad de ejecuciones experimentales, combinando múltiples modelos, un amplio abanico de estrategias de *prompting* y 19 configuraciones de parámetros distintas. La gestión de estas ejecuciones y sus resultados mediante scripts independientes y análisis posteriores en Jupyter, si bien es funcional para la investigación, presenta desafíos de escalabilidad, comparación y trazabilidad a largo plazo.

Las alternativas propuestas son herramientas de seguimiento de experimentos como MLflow [The MLflow Project, 2025] o Weights & Biases [Weights & Biases, 2025].

Estas plataformas están diseñadas específicamente para la trazabilidad del ciclo de vida de machine learning. En lugar de almacenar resultados en ficheros CSV o JSON dispersos, permiten registrar de forma automática y centralizada los parámetros de cada ejecución (modelo, temperatura, `Top-P`), las métricas de rendimiento (*F1-Score*, *Precision*, *Recall*), los *prompts* utilizados e incluso los artefactos de salida (los JSONs extraídos). Esto crea un panel de control interactivo para visualizar y comparar de manera instantánea qué combinación de factores produjo los mejores resultados, mejorando drásticamente la capacidad de análisis y la reproducibilidad de los hallazgos.

Frameworks de Desarrollo para LLM. La interacción directa con las APIs de Google y Mistral usando sus librerías cliente ofrece un control máximo, pero puede conducir a la escritura de código repetitivo (*boilerplate*), especialmente al implementar lógicas complejas como la "extracción iterativa" o al gestionar la lógica para alternar entre diferentes modelos. Las alternativas propuestas son Frameworks de orquestación de LLM como LangChain [Chase, Harrison, 2025] o LlamaIndex [Liu, Jerry, 2025]. Estos frameworks proporcionan abstracciones de alto nivel para patrones de diseño comunes en apli-

caciones con LLM. Por ejemplo, la sofisticada estrategia de extracción iterativa podría haberse implementado de forma más limpia, declarativa y modular utilizando las "cadenas" (*Chains*) de LangChain, que encapsulan la lógica de múltiples llamadas secuenciales a un LLM. Además, estos frameworks simplifican la integración y el cambio entre diferentes proveedores de modelos y ofrecen herramientas integradas para el parseo de salidas estructuradas y la gestión de plantillas de *prompts*, lo que habría reducido la cantidad de código de bajo nivel y aumentado la mantenibilidad del proyecto.

Reproducibilidad y Gestión del Entorno. El proyecto depende de una versión específica de Python y un conjunto de librerías con sus propias versiones. Para que otro investigador, o el propio autor en el futuro, pueda replicar los resultados exactamente, es necesario recrear este entorno a la perfección, una tarea que puede ser frágil y propensa a errores.

La alternativa propuesta [Docker, Inc., 2025] es una tecnología de "contenedorización" como Docker y el uso de Entornos de Desarrollo en Contenedores (`Dev Containers`) en Visual Studio Code.

Docker permite empaquetar la aplicación, junto con todas sus dependencias (el intérprete de Python, las librerías listadas en un fichero `requirements.txt`, e incluso variables de entorno), en una imagen de contenedor aislada y portátil. Esto garantiza que el entorno de ejecución sea idéntico en cualquier máquina, eliminando por completo el clásico problema de "en mi máquina funciona". La integración con VS Code a través de la extensión `Dev Containers` permite, además, desarrollar y ejecutar el código directamente dentro de este entorno controlado, mejorando drásticamente la reproducibilidad y facilitando la colaboración.

Orquestación de Flujos de Datos. En cuanto al flujo de trabajo completo (leer PDFs, preprocesar a Markdown, refinar *ground truth*, ejecutar experimentos, evaluar resultados y calcular métricas) constituye un *pipeline* de varios pasos. La gestión de esta secuencia mediante la ejecución manual de scripts individuales es funcional, pero puede ser engorrosa y frágil si un paso intermedio falla o si se necesita re-ejecutar solo una parte del proceso.

Las alternativas encontradas son orquestadores de flujos de trabajo ligeros como Prefect [Prefect Technologies, Inc., 2025] o Dagster [Elementl, Inc., 2025].

Estas herramientas permiten definir el *pipeline* completo como un Grafo Acíclico Dirigido (DAG) de tareas directamente en código Python. Ofrecen beneficios cruciales para la robustez del proyecto, como la observabilidad (un panel para ver qué tareas han fallado y por qué), la capacidad de configurar reinicios automáticos, la ejecución en paralelo de tareas independientes y una forma mucho más estructurada de definir las dependencias entre los pasos. Esto convierte una serie de scripts interconectados en un *pipeline* de datos resiliente y monitoreable.

En conclusión, es importante reiterar que la pila tecnológica elegida para este TFG fue exitosa y apropiada para los fines de la investigación. Sin embargo, la incorporación de las herramientas de MLOps y desarrollo avanzado aquí propuestas representa el siguiente paso natural en el ciclo de madurez del proyecto: el de transformar un prototipo de investigación funcional en una solución de software robusta, escalable, mantenible y lista para su potencial integración en un entorno de producción.

Capítulo 5

Conjunto de Datos

En este capítulo se realiza una descripción exhaustiva del conjunto de datos que constituye la base experimental de este Trabajo de Fin de Grado. La correcta selección y comprensión del corpus de datos es un pilar fundamental en cualquier proyecto de ciencia de datos, y en este apartado se busca proporcionar una visión completa del recurso utilizado, desde su concepción y estructura hasta los desafíos prácticos que presentó su adaptación para este proyecto.

Para empezar, se presentará una descripción general del dataset *Invoices Database of the Spanish Electricity Market* (IDSEM) [Sánchez et al., 2022], explicando su propósito, su escala y el método de simulación empleado para su creación. A continuación, se detallará la estructura interna de los documentos, mostrando ejemplos visuales de las facturas y desglosando el sistema de etiquetas semánticas que definen la información a extraer. Posteriormente, se dedicará una sección a describir el preprocesamiento y adaptación llevado a cabo, documentando los problemas encontrados y las soluciones implementadas. Finalmente, se abordará la validación técnica del dataset y se argumentará su idoneidad para los objetivos de esta investigación.

5.1. Descripción General de IDSEM

El desarrollo y la evaluación de modelos para la extracción automática de información dependen críticamente de la disponibilidad de conjuntos de datos a gran escala. Para este trabajo, se ha seleccionado el dataset IDSEM. Su principal fortaleza es que resuelve el obstáculo de la privacidad de datos mediante la generación de un corpus sintético, pero altamente realista, de facturas de electricidad. El resultado es una base de datos que consta de 75,000 documentos únicos, diseñada explícitamente para el entrenamiento y la validación de algoritmos de extracción de información .

El núcleo de IDSEM es un *pipeline* de generación automática que crea facturas indistinguibles en estructura y formato de las reales. El proceso se basa en plantillas de documentos (creadas a partir de facturas de grandes comercializadoras como Iberdrola o Endesa) y diccionarios con datos españoles (nombres, direcciones, empresas, etc.). El *pipeline* se ejecuta en tres etapas:

1. **Simulación de Datos**: Se generan los valores para un conjunto de etiquetas y se almacenan en un fichero JSON. Los datos numéricos clave, como la potencia contratada o el consumo, se muestrean de distribuciones estadísticas basadas en informes de la Comisión Nacional de los Mercados y la Competencia (CNMC) y Red Eléctrica Española (REE) .

2. **Población de Plantillas**: Un sistema automático rellena las plantillas en formato DOCX con los datos del fichero JSON, reemplazando marcadores de posición como `{{A1}}`.

3. **Conversión a PDF**: Los documentos DOCX se convierten al formato PDF, el estándar en el sector.

5.2. Estructura y Contenido de las Facturas

Una de las mayores fortalezas de IDSEM para la investigación es su diversidad estructural. A pesar de que el contenido es similar por regulación, el diseño (*layout*) de la factura es específico de cada compañía. El dataset captura esta variabilidad utilizando múltiples plantillas, como se puede observar en las Figuras **??** y **??**, que muestra los diferentes diseños de las facturas, donde la información se organiza en cajas, tablas y columnas de manera arbitraria. Esta diversidad de formatos presenta un desafío realista para los modelos de extracción de información, que deben aprender a generalizar más allá de una única estructura fija.

El conjunto de datos está bien estructurado, ya que para cada factura proporciona tres ficheros distintos: un PDF con la factura final, un fichero JSON que contiene el *ground truth*, y un segundo PDF anotado, donde el texto correspondiente a cada etiqueta está delimitado por marcadores (ej. `#A1 Juan Pérez #A1`), facilitando enormemente el entrenamiento supervisado .

Para ilustrar la estructura de estas anotaciones, la Figura 5.3 muestra un fragmento de uno de estos ficheros. Como se puede observar, el fichero consiste en una serie de pares clave-valor, donde la clave (ej. `A1`) corresponde al código de la etiqueta y el valor (ej. `Agamenón Arrollo Montalvo`) es el dato real de la factura simulada. Este fichero constituye el *ground truth* contra el cual se compara la salida del modelo durante la fase de evaluación.

Para organizar la gran cantidad de etiquetas, los creadores del dataset las agruparon en categorías semánticas identificadas por una letra, tal y como se ilustra en la Figura 5.4. Este sistema de agrupación visual ayuda a comprender la distribución de la información en una factura típica. Por ejemplo, el grupo "A" corresponde a los datos del cliente que recibe la factura, el grupo "C" a los de la comercializadora, y los grupos "J", "K", "M" y "N" agrupan los diferentes conceptos del desglose económico.

Este sistema se refleja directamente en el código de cada etiqueta. La primera letra de un código (ej. la "A" en "A1") indica la categoría a la que pertenece, facilitando así su identificación y comprensión.

Aunque el dataset documenta 86 conceptos semánticos fundamentales, el número total de etiquetas posibles asciende a 107. Esta discrepancia se debe a la existencia de 21 subetiquetas que representan variantes de formato para ciertos campos, principalmente fechas e importes. Por ejemplo, una misma fecha puede tener distintas representaciones, y una tarifa puede expresarse de forma anual o diaria. Como se detalla en la Tabla 5.1, el dataset utiliza un sistema de sufijos en los códigos de las etiquetas para diferenciar estas variantes.

A continuación, para ilustrar la estructura de los datos, las siguientes tablas (5.2, 5.3 y 5.4) detallan una selección representativa de las etiquetas, agrupadas por su función semántica. Si bien estos ejemplos cubren algunos campos, el listado completo de todas las etiquetas puede consultarse en la publicación original de IDSEM .

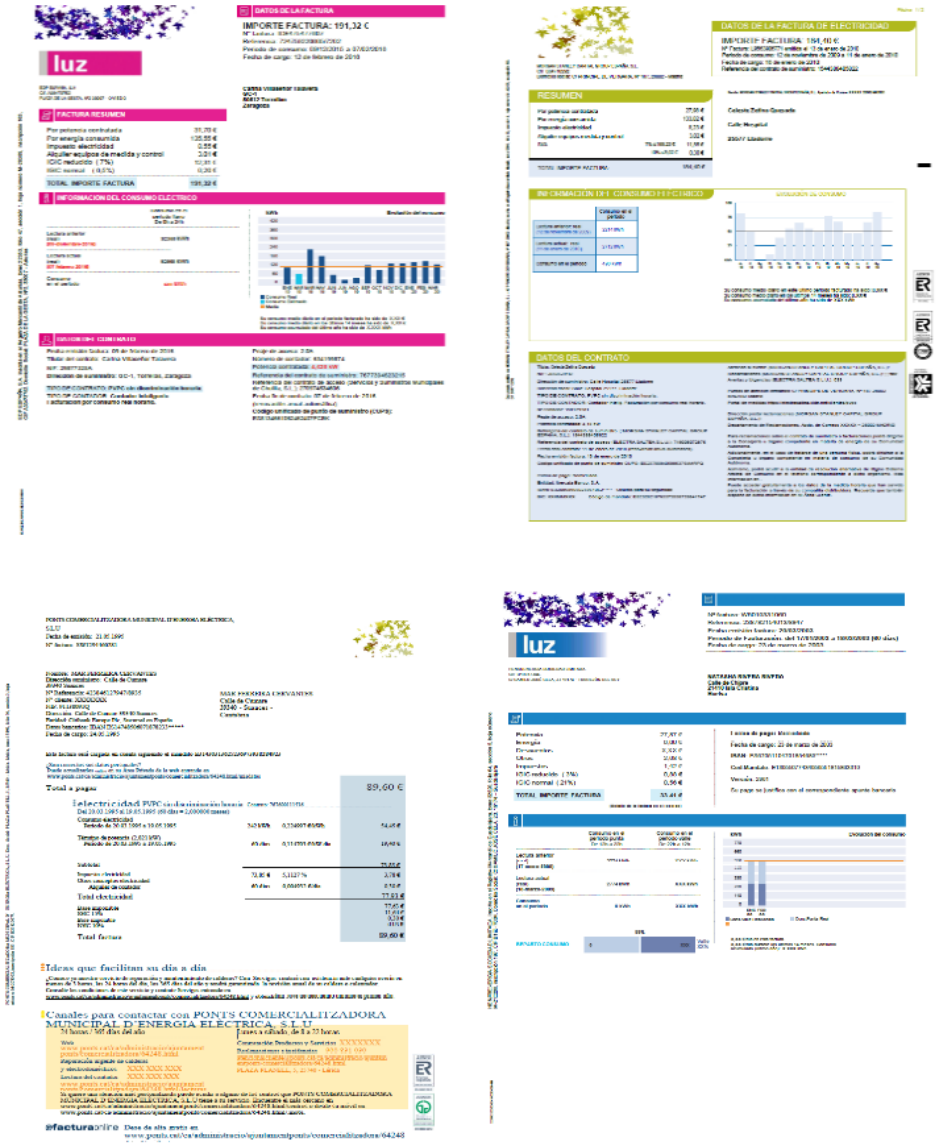


Figura 5.1: Ejemplo 1 de facturas presentes en el conjunto IDSEM.

Tabla 5.1: Modificadores de formato para las etiquetas de fecha.

Modificador	Formato	Ejemplo
l	%d de %B de %Y	12 de enero de 2021
s	%d/%m/%Y	12/01/2021
u	%d-%B-%Y	12-enero-2021
p	%d.%m.%Y	12.01.2021
d	€/día	0,050164 Eur/día

5.3. Adaptación al Proyecto

Para poder utilizar el dataset IDSEM en este trabajo, fue necesario realizar un preprocesamiento fundamental, durante el cual surgieron varios desafíos que requirieron la implementación de soluciones a medida.

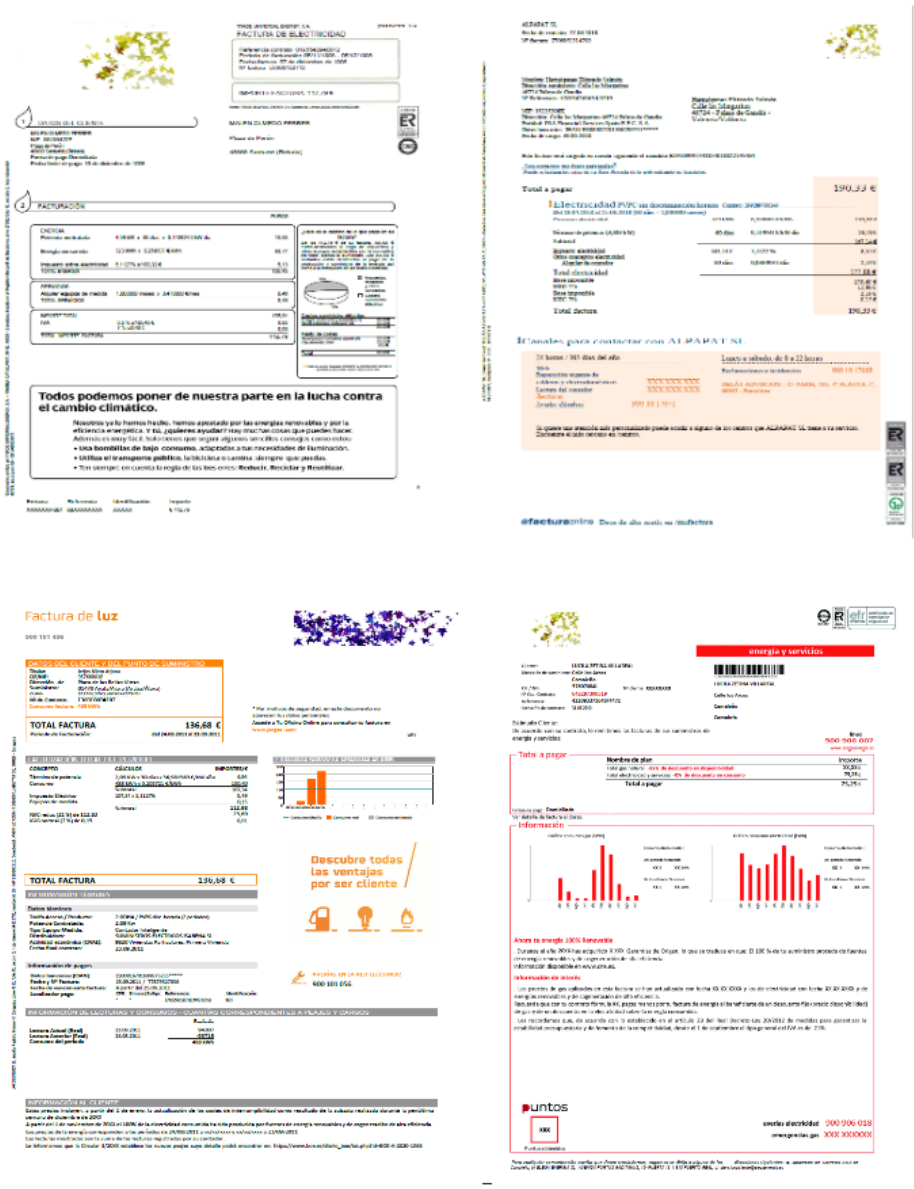


Figura 5.2: Ejemplo 2 de facturas presentes en el conjunto IDSEM.

Tabla 5.2: Etiquetas de los datos del cliente (Grupos A y B).

Etiqueta	**Descripción**
A1/B1	Nombre del cliente
A2/B2	Número de documento de identidad del cliente
A3/B3	Dirección del cliente
A4/B4	Código postal
A5/B5	Ciudad del cliente
A6/B6	Provincia del cliente

La tarea de preparación principal consistió en convertir las facturas, originalmente en formato PDF, a texto utilizando el lenguaje de marcado Markdown. La elección de este for-

```
{
    "A1": "ES0021000000000001FA",
    "B1": "01/01/2022",
    "B2": "31/01/2022",
    "C1": "Iberdrola S.A.",
    ...
    "M3": "2,05",
    ...
    ...
    "M4": "3,01",
    "N1": "5,1127",
    "N2": "8,55",
    "N3": "11,56",
    "N4": "7",
    "N5": "0,20",
    "N6": "6,5",
    "N7": "135,55",
    "N8": "12,31"
}
```


Figura 5.3: Ejemplo de la estructura de un fichero JSON del dataset IDSEM, que contiene el *ground truth* de una factura.

mato se fundamenta en su capacidad para preservar la estructura semántica y visual de los documentos. Esta decisión se apoya en investigaciones que demuestran que proporcionar a los LLM esta estructura textual mejora significativamente su capacidad para comprender la maquetación y las relaciones jerárquicas del documento [Braun et al., 2025].

Durante el preprocesamiento se identificaron varios problemas que tuvieron que ser abordados para garantizar una evaluación justa de los modelos:

- **Pérdida de Datos en la Conversión:** La herramienta utilizada para convertir de PDF a Markdown no era capaz de extraer texto situado en los bordes del documento, imágenes o gráficas.
- **Discrepancia entre JSON y Plantillas:** Se constató que los ficheros JSON (que actúan como *ground truth*) contienen claves para las 107 etiquetas posibles, mientras que cada plantilla de factura solo muestra un subconjunto de estas.
- **Gestión de Campos Vacíos:** Debido a la naturaleza sintética del dataset, algunos campos en ciertas facturas estaban anotados como existentes pero no contenían ningún valor, lo que generaba ambigüedad.

Para solucionar estos problemas, se creó un mapeo explícito que documenta, para cada una de las plantillas usadas en la evaluación del modelo, qué subconjunto de las 107 etiquetas está realmente presente y es extraíble en el formato Markdown final. Este registro fue indispensable para la fase de evaluación, asegurando que los modelos solo

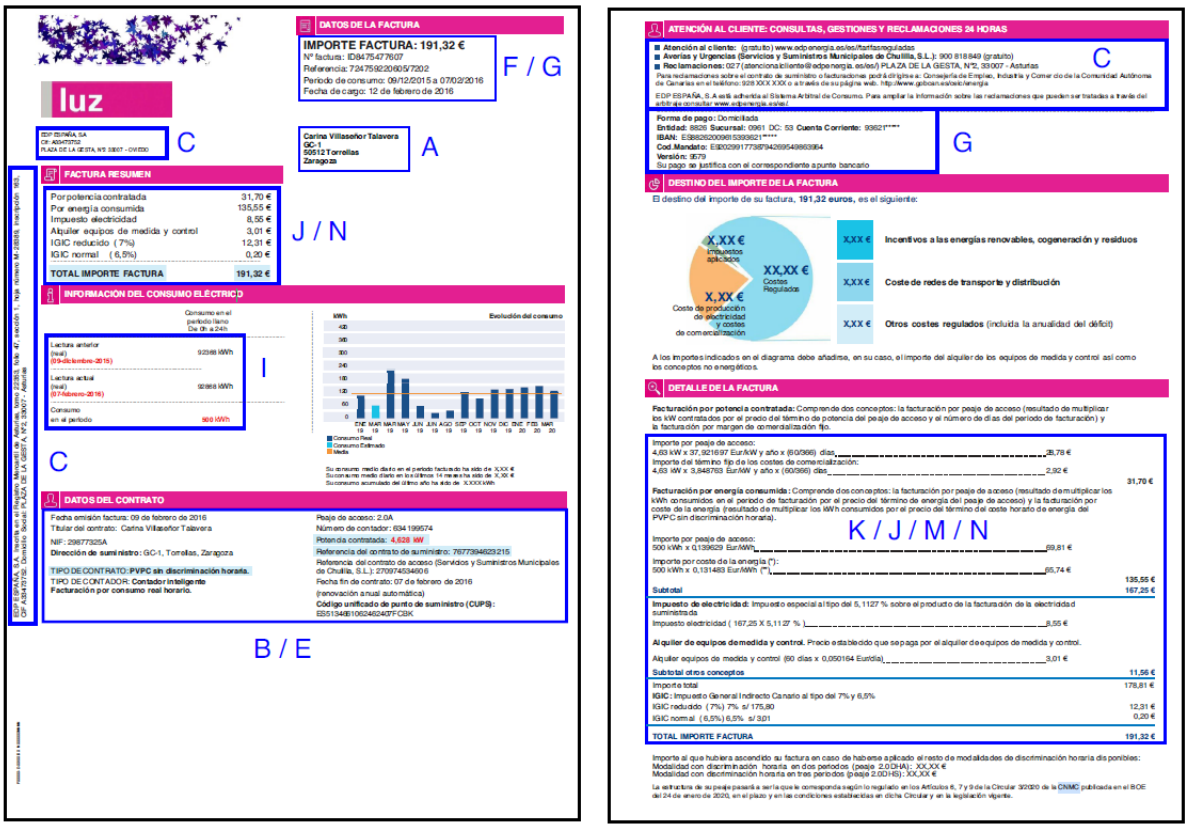


Figura 5.4: Ejemplo visual de la agrupación de etiquetas por categorías (A, B, C, etc.) en una de las plantillas del dataset IDSEM.

Tabla 5.3: Etiquetas de la comercializadora y distribuidora (Grupos C y D).

Etiqueta	**Descripción**
C1	Nombre de la comercializadora
C2	CIF de la comercializadora
C3	Dirección de la comercializadora
C4	Código postal
C5	Ciudad de la comercializadora
C6	Provincia de la comercializadora
C7	Información de la empresa en el registro mercantil
C8	Dirección de la administración de la empresa
C9	Sitio web de la comercializadora
CA	Email de la comercializadora
CB	Nombre corto de la empresa
CC	Teléfono de contacto de la comercializadora
CD	Teléfono de atención al cliente
CE	Teléfono de reclamaciones de la comercializadora
D1	Nombre de la distribuidora
D2	CIF de la distribuidora
D9	Sitio web de la distribuidora
DC	Teléfono de servicio público
DD	Teléfono para averías

Tabla 5.4: Etiquetas de otros conceptos e impuestos (Grupos M y N).

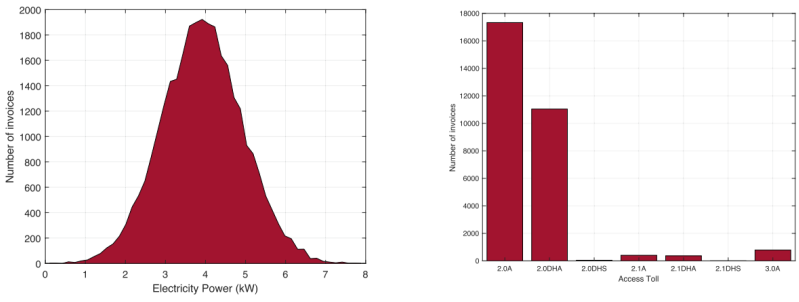
Etiqueta	Descripción
M3	Precio del alquiler del equipo (*)
M4	Precio del alquiler del equipo (total)
N1	Tasa del impuesto de electricidad
N2	Precio del impuesto de electricidad
N3	Suma de M4 y N2
N4	Tasa de impuesto normal
N5	Precio del impuesto reducido
N6	Tasa de impuesto reducido
N7	Suma de N2 y KF
N8	Precio del impuesto

(*) Aunque en el artículo de referencia se especifica que la etiqueta M3 corresponde a un precio diario, tras el análisis de las facturas (concretamente en la plantilla 5), se ha constatado que este valor representa una tarifa mensual, acompañada siempre de la unidad "€/mes". La variante que sí representa el valor diario es la sub-etiqueta "M3d".

fueran evaluados sobre los campos que realmente existían y eran visibles en los datos de entrada.

5.4. Validación Técnica y Relevancia

La fiabilidad de IDSEM como banco de pruebas está avalada por un riguroso proceso de validación técnica realizado por sus autores. Se llevó a cabo una validación estadística exhaustiva sobre los 30,000 documentos de entrenamiento. Los resultados del artículo confirman que las distribuciones de los datos generados se corresponden con los modelos teóricos y las estadísticas oficiales utilizadas en la simulación. Por ejemplo, la distribución de la potencia contratada y la distribución de las tarifas de acceso, presentes en la Figura 5.5, siguen la distribución Normal esperada y replican fielmente los porcentajes de clientes publicados por la CNMC.



(a) Contracted electricity power (E3) **(c)** Access toll (E6)

Figura 5.5: Distribución de los valores de las etiquetas E3 (Potencia contratada) y E6 (Peaje de acceso). Fuente: [Sánchez et al., 2022]

Adicionalmente, se realizó una validación manual sobre una muestra aleatoria de 10 facturas por cada plantilla, donde se verificó la coherencia de las fechas, la correcta inser-

ción de los datos, la precisión de los cálculos de importes y la fidelidad del formato del PDF final.

En resumen, a pesar de los desafíos del preprocesamiento, la gran escala del dataset, la diversidad de su estructura, su contenido basado en fuentes oficiales y la disponibilidad de un *ground truth* hacen de IDSEM un recurso de excepcional valor. Su estructura lo convierte en un banco de pruebas ideal para desarrollar y evaluar de manera rigurosa los métodos de extracción de información que se han desarrollado en este proyecto.

Capítulo 6

Redes Neuronales Empleadas

En el presente capítulo se establece el marco teórico de las redes neuronales que constituyen la base de este Trabajo de Fin de Grado. El objetivo es proporcionar una comprensión profunda de los Modelos Extensos de Lenguaje (LLM) utilizados, desde sus componentes fundamentales hasta las innovaciones más recientes. Dado que el proyecto aborda la extracción de información mediante modelos con arquitectura de Transformer, este capítulo se dedica íntegramente a su estudio.

El texto se organiza de manera progresiva. Se comienza con los fundamentos de la arquitectura del Transformer [Vaswani et al., 2017], incluyendo una descripción del proceso de *tokenización* y un desglose de la arquitectura codificador-decodificador, con énfasis en el mecanismo de atención. Posteriormente, el capítulo avanza hacia los modelos específicos empleados en la fase experimental: Gemini 1.5 Pro y su arquitectura de Mezcla de Expertos (MoE), y Mistral-small con sus optimizaciones de eficiencia.

6.1. Fundamentos de la Arquitectura del Transformer

En este trabajo se han empleado LLM con arquitectura de Transformer, que representan un avance fundamental en el Procesamiento del Lenguaje Natural (PLN). Estos modelos, entrenados con cantidades masivas de datos, son capaces de comprender el contexto léxico-semántico y manipular el lenguaje humano con gran coherencia.

El punto de inflexión fue el artículo "Attention is All You Need" [Vaswani et al., 2017], que introdujo el mecanismo de autoatención. Esta innovación permitió al modelo procesar todas las palabras de una secuencia simultáneamente, capturando relaciones semánticas a larga distancia y, crucialmente, permitiendo una paralelización masiva del cómputo durante el entrenamiento. Esta capacidad fue esencial para escalar los modelos a miles de millones de parámetros, sentando las bases para los LLM modernos.

6.1.1. Tokenización

La *tokenización* es el proceso que convierte el texto en una secuencia de unidades discretas (*tokens*) que las redes neuronales pueden procesar [Ali and Fromm, 2024]. Este proceso es precedido por un preprocesamiento del texto (normalización, supresión de *stopwords*, lematización) que reduce el ruido y estandariza la entrada, una etapa crítica para maximizar el rendimiento del modelo [Siino et al., 2023]. Los algoritmos de tokenización

más comunes, como BPE, WordPiece y SentencePiece, se basan en un corpus textual para aprender a construir sus vocabularios de *tokens*.

Byte-Pair Encoding (BPE)

El algoritmo BPE, adaptado al PLN por [Sennrich et al., 2016], descompone palabras no vistas en subunidades conocidas para manejar vocabularios abiertos, una técnica adoptada por modelos como GPT-2. Aprende un vocabulario de subpalabras fusionando iterativamente el par de *tokens* adyacentes más frecuente en el corpus hasta alcanzar un tamaño de vocabulario predefinido. Aunque eficaz, su estrategia voraz puede generar unidades que no siempre respetan los límites morfológicos de las palabras [Bostrom and Durrett, 2020].

WordPiece y SentencePiece

Utilizado en modelos como BERT [Devlin et al., 2019], WordPiece es similar a BPE pero emplea un criterio de fusión estadísticamente más riguroso, eligiendo el par que maximiza la verosimilitud del corpus. SentencePiece, por su parte, es una solución autónoma que opera sobre texto bruto, tratando los espacios como un carácter más, lo que lo hace ideal para lenguas sin delimitadores claros [Kudo and Richardson, 2018].

6.1.2. Arquitectura Codificador-Decodificador

Las arquitecturas para tareas de transducción de secuencias, como el Transformer, suelen seguir un esquema de codificador-decodificador [Vaswani et al., 2017]. El codificador transforma una secuencia de entrada en una representación continua contextualizada. Posteriormente, el decodificador utiliza esta representación para generar la secuencia de salida de forma auto-regresiva, donde cada nuevo *token* se predice condicionándolo a los *tokens* generados previamente [Graves, 2013].

El Transformer se distingue por basarse exclusivamente en capas de autoatención y redes *feed-forward*, permitiendo un procesamiento paralelo que captura eficientemente dependencias a largo plazo.

El Codificador y el Decodificador

El codificador del Transformer está compuesto por una pila de $N = 6$ capas idénticas. Cada capa contiene dos sub-capas: un bloque de autoatención multi-cabeza y una red neuronal *feed-forward*. Para facilitar el entrenamiento de redes profundas, cada sub-capa está envuelta en una conexión residual seguida de una normalización de capa [He et al., 2015, Ba et al., 2016].

El decodificador comparte una estructura similar, también con $N = 6$ capas, pero añade una tercera sub-capa. Esta sub-capa adicional implementa un mecanismo de atención que atiende a la salida del codificador. Una modificación crucial en el decodificador es el uso de un "enmascaramiento causal" en su propia capa de autoatención, que impide que una posición atienda a posiciones futuras, garantizando que el proceso de generación siga siendo estrictamente auto-regresivo [Vaswani et al., 2017].

6.1.3. Mecanismo de Atención

La atención es una función que mapea una consulta (*query*) y un conjunto de pares clave-valor (*key-value*) a una salida. El Transformer utiliza la "Atención por Producto

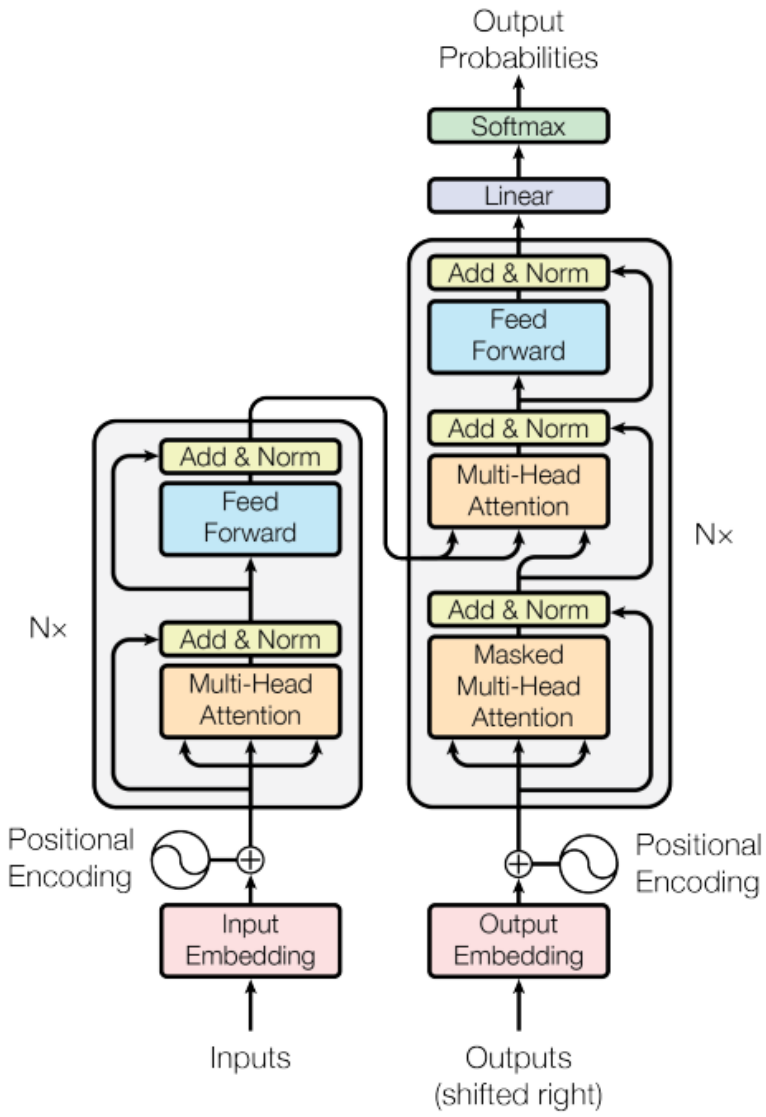


Figura 6.1: Estructura general codificador-decodificador del modelo Transformer. Fuente: [Vaswani et al., 2017]

Escalar Escalado. Para mejorar su capacidad, emplea la **atención multi-cabeza**, que ejecuta varias funciones de atención en paralelo. Esto permite que el modelo atienda simultáneamente a información de diferentes subespacios de representación, capturando distintas relaciones (sintácticas, semánticas, etc.) [Vaswani et al., 2017].

Este mecanismo se utiliza de tres formas en el modelo: autoatención en el codificador, atención cruzada en el decodificador y autoatención enmascarada en el decodificador.

Componentes Adicionales de la Arquitectura

Cada capa del Transformer incluye una red *feed-forward* posicional, que consiste en dos transformaciones lineales con una activación ReLU intermedia, añadiendo capacidad no lineal al modelo. Se utilizan *embeddings* entrenables para convertir los *tokens* en vectores. Para que el modelo pueda procesar el orden de la secuencia, se añaden codificaciones posicionales a los *embeddings* de entrada, usando funciones sinusoidales para generar un

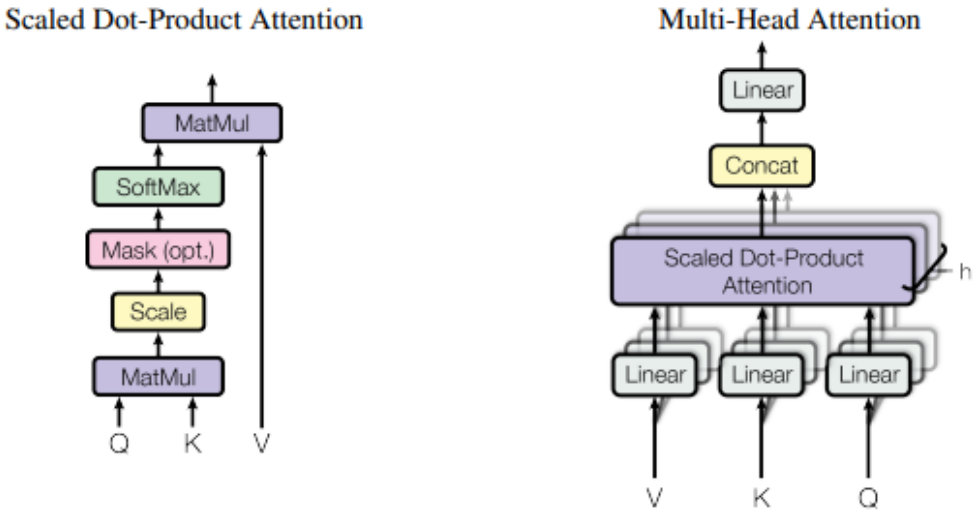


Figura 6.2: Atención por producto escalar escalado (izquierda). Atención multi-cabeza (derecha) Fuente: [Vaswani et al., 2017]

vector único para cada posición [Vaswani et al., 2017].

6.2. Evolución Post-Transformer: BERT y GPT

Tras el Transformer, el campo se bifurcó en dos filosofías principales:

Procesamiento Bidireccional: BERT El modelo BERT [Devlin et al., 2019] utiliza únicamente la pila de codificadores del Transformer para aprender representaciones de palabras profundamente bidireccionales, considerando simultáneamente el contexto izquierdo y derecho. Lo logra mediante dos tareas de pre-entrenamiento: el Modelo de Lenguaje Enmascarado (MLM) y la Predicción de la Siguiente Oración. BERT se convirtió en el estándar para tareas de Comprensión del Lenguaje Natural (NLU) mediante un paradigma de pre-entrenamiento seguido de un ajuste fino (*fine-tuning*) específico para cada tarea.

Generación Auto-regresiva: La Arquitectura GPT Paralelamente, los modelos GPT [Radford et al., 2018] se centraron exclusivamente en el componente decodificador del Transformer. Su naturaleza auto-regresiva los hace ideales para la generación de texto. El punto de inflexión fue GPT-3 [Brown and Mann, 2020], que demostró que el escalado masivo de parámetros daba lugar a la capacidad emergente del aprendizaje en contexto (*in-context learning*). Esto permite al modelo realizar tareas sin reentrenamiento, basándose en los ejemplos (*zero-shot, few-shot*) proporcionados en el *prompt*, lo que define el paradigma de uso actual de los LLM.

El escalado de estos modelos "densos" reveló un desafío de coste computacional. Para superarlo, la investigación se orientó hacia dos vías principales: arquitecturas dispersas como la Mezcla de Expertos (MoE) y la optimización de la eficiencia del mecanismo de atención con técnicas como GQA. A continuación, se analizan los modelos empleados en este trabajo, Gemini 1.5 Pro y Mistral-small, como representantes de cada una de estas soluciones.

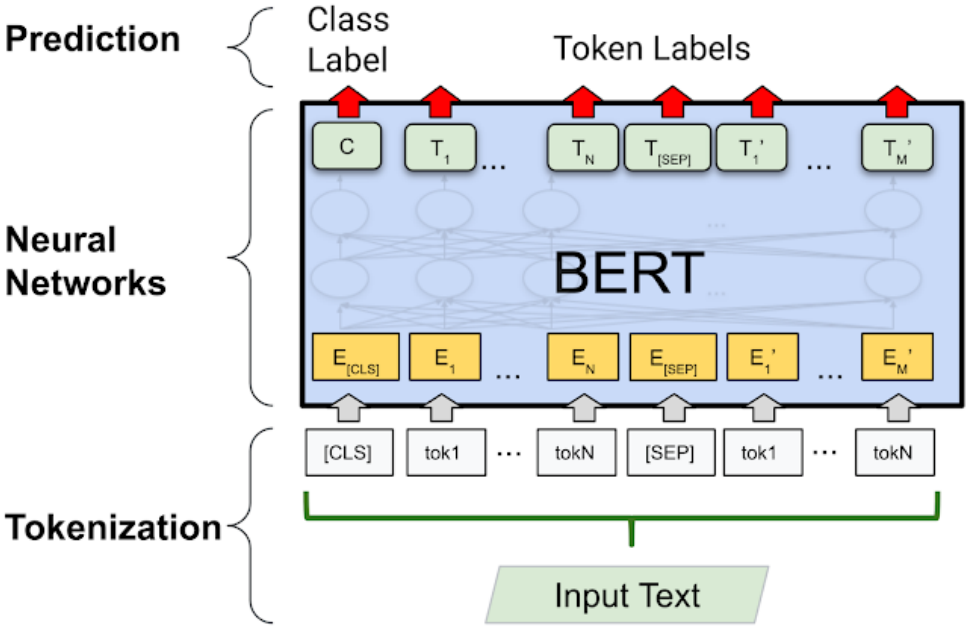


Figura 6.3: Arquitectura de entrada y salida de BERT para tareas de clasificación y etiquetado, mostrando el uso de los tokens especiales [CLS] y [SEP]. Fuente: Adaptado de [Devlin et al., 2019].

6.3. Modelos Utilizados

Tras haber recorrido la evolución de las arquitecturas neuronales en el capítulo anterior, desde los principios fundacionales del Transformer hasta la bifurcación en modelos de codificador y decodificador y los posteriores desafíos de escalabilidad, este capítulo se centra en los instrumentos concretos de la investigación. Se abandona la discusión teórica general para realizar un análisis detallado de los dos modelos de lenguaje extenso que han sido seleccionados para la fase experimental de este trabajo: Gemini 1.5 Pro y Mistral-small.

Por un lado, se analizará Gemini 1.5 Pro, un modelo de vanguardia que encarna la solución de la computación condicional y las arquitecturas dispersas a través de la Mezcla de Expertos (MoE) para alcanzar una capacidad masiva de manera eficiente. Por otro lado, se examinará Mistral-small, que ejemplifica la optimización de los modelos densos tradicionales mediante innovaciones en el mecanismo de atención, como la Atención de Ventana Deslizante (SWA) y la Atención de Consulta Agrupada (GQA), para lograr un equilibrio excepcional entre rendimiento y eficiencia.

Para cada modelo, se describirán sus características clave, las innovaciones arquitectónicas que lo sustentan y su rendimiento documentado en tareas relevantes, proporcionando así el contexto final necesario para comprender su aplicación y los resultados que se presentarán en los capítulos posteriores.

6.3.1. Gemini 1.5 pro

Los modelos de la familia Gemini representan la culminación de la línea evolutiva que parte de la filosofía de los modelos generativos a gran escala. El éxito de arquitecturas como GPT-3 demostró que el escalado masivo de los parámetros era un ingrediente clave para desbloquear capacidades emergentes como el aprendizaje en contexto

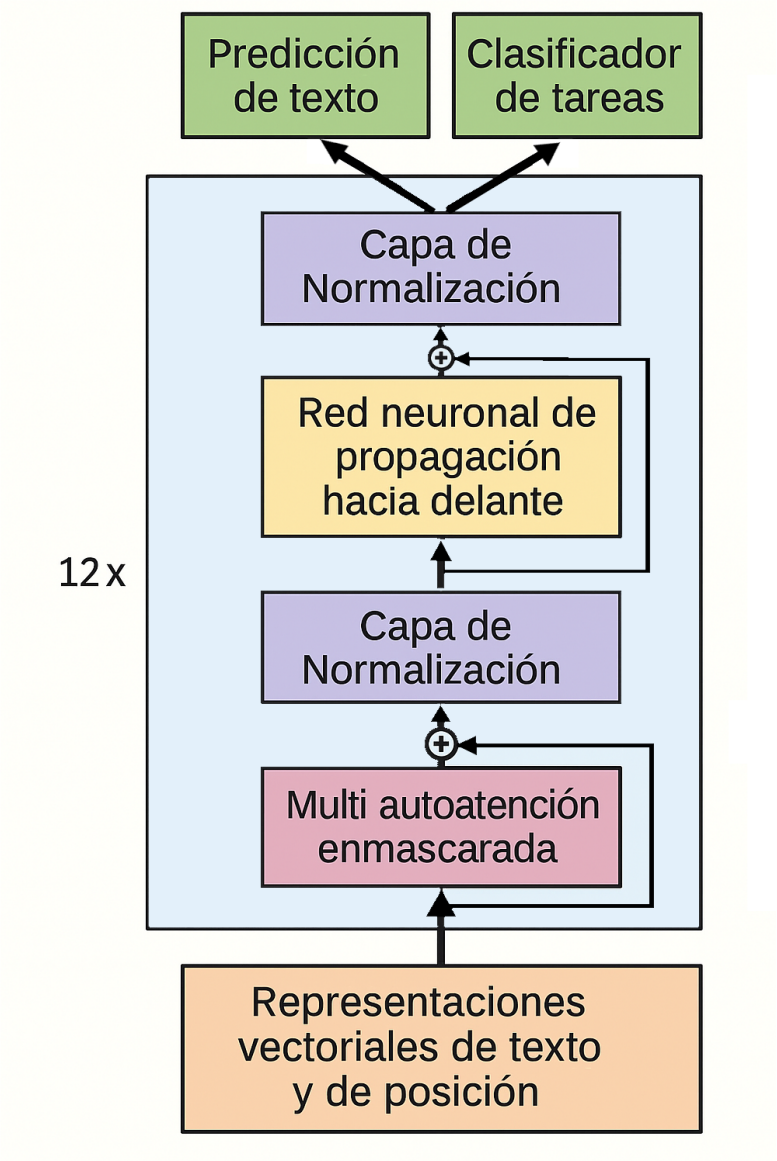


Figura 6.4: Diagrama de la arquitectura del modelo GPT. Fuente: Adaptado de [Radford et al., 2018].

[Brown and Mann, 2020]. Sin embargo, este paradigma se enfrentó rápidamente a un desafío de sostenibilidad: el prohibitivo coste computacional de entrenar y operar modelos "densos" de billones de parámetros.

Gemini 1.5 Pro, desarrollado por Google DeepMind [Reid et al., 2024], en este contexto, emerge como una solución a este desafío. Hereda la potente base auto-regresiva de la familia GPT pero la integra con una arquitectura de Transformer de tipo mezcla de expertos (MoE) dispersa. Como se detallará en la subsección 6.3.1, este enfoque de computación condicional permite que el modelo aumente significativamente su número total de parámetros mientras mantiene un coste computacional constante por inferencia. Esta eficiencia es la que le permite alcanzar una calidad comparable o superior a la de su predecesor, Gemini 1.0 Pro, con una cantidad significativamente menor de recursos de

entrenamiento, haciéndolo viable para abordar tareas a gran escala como el procesamiento de miles de documentos de negocio.

La innovación más destacada del modelo es su capacidad para procesar contextos extremadamente largos, demostrando una recuperación de información casi perfecta (superior al 99,7 %) en secuencias de hasta 1 millón de *tokens*, y manteniendo un 99,2 % de recuperación hasta los 10 millones de *tokens*. Esta ventana de contexto masiva le permite procesar de forma nativa colecciones enteras de documentos, como extensos informes financieros o miles de facturas en un solo lote, sin necesidad de dividirlos. En la práctica, su rendimiento en el análisis de datos no estructurados en lote, una tarea análoga al procesamiento de múltiples formularios, no solo mejora la precisión en la extracción de atributos en un 9 % absoluto sobre GPT-4 Turbo, sino que, a diferencia de este, su rendimiento mejora consistentemente a medida que aumenta el número de documentos en el contexto [Reid et al., 2024].

Capacidad	**Benchmark**	**Gemini**	
		1.0 Pro	**1.5 Pro**
Gráficos y Documentos	**ChartQA (test)** Comprensión de gráficos 0-shot [Masry et al., 2022]	74.1 %	**87.2 %**
	DocVQA (test) Comprensión de documentos 0-shot [Mathew et al., 2021]	88.3 %	**93.1 %**
	InfographicVQA (test) Comprensión de infografías 0-shot [Mathew et al., 2022]	75.2 %	**81.7 %**

Tabla 6.1: Resultados de los modelos Gemini en *benchmarks* que evalúan la comprensión de tablas y documentos. Los mejores resultados por *benchmark* se muestran en negrita. Tabla Modificada de [Reid et al., 2024]

Esta capacidad para manejar grandes cantidades de información se traduce en un rendimiento excelente en *benchmarks* de comprensión de documentos, como se puede observar en la Tabla 6.1. En tareas de lectura y extracción visual, el modelo alcanza una puntuación de 93,1 % en DocVQA, un 87,2 % en ChartQA, y un 81,7 % en InfographicVQA [Reid et al., 2024], lo que posiciona a Gemini como una herramienta en el Estado del Arte para la extracción de información en documentos estructurados.

Arquitectura de Mezcla de Expertos

La arquitectura de Mezcla de Expertos (MoE), en la que se fundamenta Gemini 1.5 Pro, representa un paradigma de computación condicional que permite aumentar drásticamente la capacidad de un modelo (su número total de parámetros) sin un incremento proporcional en el coste computacional [Reid et al., 2024]. Introducido originalmente por [Jacobs et al., 1991], este enfoque consiste en un sistema de múltiples redes "expertas" paralelas y una "red de compuerta" (*gating network*) que aprende a enrutar cada entrada al experto más adecuado, permitiendo una especialización que mejora la capacidad de modelado del sistema global.

En los modelos Transformer modernos, la capa MoE se utiliza típicamente para reemplazar la sub-capa de la red feed-forward (FFN). Una capa MoE en este contexto consta de dos componentes principales:

1. **Una pequeña red de compuerta:** Actúa como un enrutador aprendido (*learned routing function*). Para cada *token*, esta red (generalmente una transformación lineal seguida de una función softmax) produce una distribución de probabilidad sobre todos los expertos disponibles [Shazeer et al., 2017, Reid et al., 2024].

2. **Un gran número de expertos:** Cada experto es una red neuronal independiente, típicamente una FFN estándar. Aunque todos comparten la misma arquitectura, sus pesos son independientes y se aprenden de forma diferenciada [Shazeer et al., 2017, Reid et al., 2024].

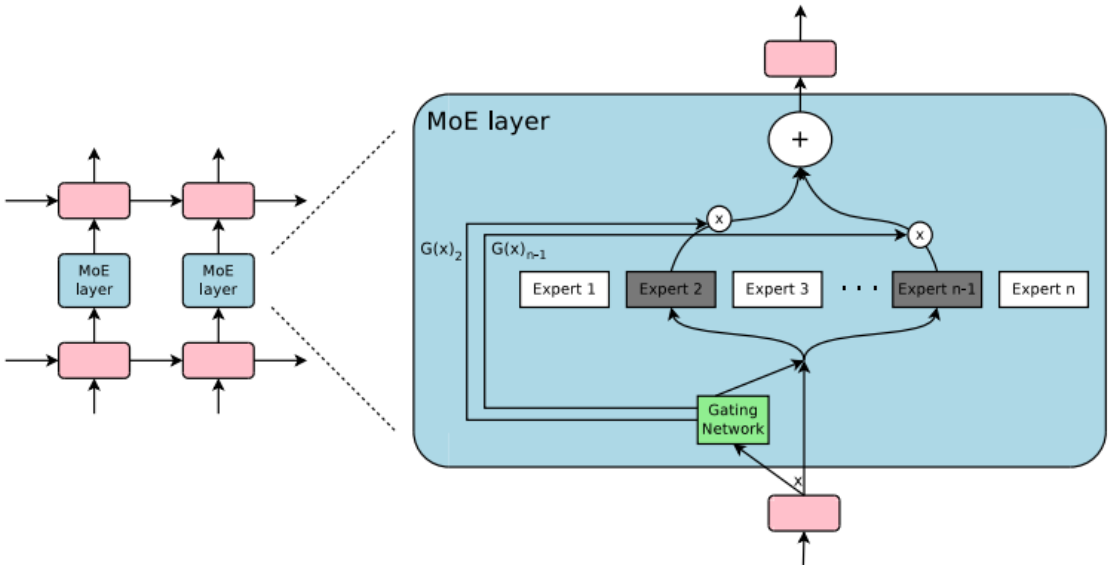


Figura 6.5: Arquitectura de la capa MoE integrada en un modelo extenso de lenguaje. Fuente: [Shazeer et al., 2017].

El mecanismo clave es la activación dispersa (*sparse activation*). En lugar de combinar las salidas de todos los expertos, la red de compuerta selecciona un pequeño subconjunto de ellos para cada unidad de texto generada. Como se observa en la Figura 6.5, la implementación de *Sparsely-Gated Mixture-of-Experts* [Shazeer et al., 2017] selecciona los k expertos con las probabilidades más altas (donde k es un número pequeño, a menudo 1 o 2). La salida final para ese *token* es la suma ponderada de las salidas de solo esos k expertos activados, con pesos normalizados por la compuerta [Reid et al., 2024]. Este enrutamiento a nivel de palabra permite que diferentes partes de la secuencia sean procesadas por distintos subconjuntos de expertos [Shazeer et al., 2017]. El trabajo de [Lepikhin et al., 2020] con GShard fue fundamental al adaptar y escalar masivamente esta técnica para reemplazar las capas *feed-forward* del Transformer, lo que se convirtió en el estándar para los modelos MoE de gran escala.

Desafíos y Soluciones: El Balanceo de Carga Un desafío inherente de la arquitectura MoE es la tendencia de la red de compuerta a favorecer consistentemente a un

pequeño número de expertos, dejando a otros infrautilizados. Esto crea un desequilibrio de carga computacional y reduce la efectividad del modelo. Para mitigar este problema, se introduce una pérdida auxiliar (*auxiliary loss*) durante el entrenamiento. Esta pérdida adicional incentiva a la red de compuerta a asignar una cantidad de *tokens* aproximadamente igual a cada experto, promoviendo un balanceo de carga uniforme y asegurando que todos los expertos contribuyan al aprendizaje [Shazeer et al., 2017, Fedus et al., 2021].

Innovaciones y Variantes La investigación en MoE, especialmente en Google, ha dado lugar a variantes y optimizaciones clave que han permitido escalar estos modelos a billones de parámetros [Reid et al., 2024]:

- **GShard:** [Lepikhin et al., 2020] aplicaron con éxito la arquitectura MoE para escalar un modelo de traducción a 600 mil millones de parámetros, demostrando su viabilidad al integrar las capas MoE en las capas FFN del Transformer.
- **Switch Transformers:** [Fedus et al., 2021] propusieron una simplificación donde cada palabra se enruta a un único experto ($k = 1$). Este enfoque redujo la sobrecarga computacional y de comunicación, permitiendo escalar modelos hasta el rango del trillón de parámetros.
- **GLaM y otros modelos:** Investigaciones como GLaM [Du et al., 2021] exploraron configuraciones con $k = 2$ para mejorar la calidad del modelo, mientras que otros trabajos [Zoph et al., 2022, Riquelme et al., 2021] han continuado escalando modelos MoE efectivos tanto para lenguaje como para tareas de visión.

6.3.2. Mistral-small: Eficiencia y Potencia en Modelos Densos

No toda la innovación en los LLM se ha centrado en arquitecturas dispersas. Una línea de investigación ha buscado optimizar la eficiencia de los modelos densos tradicionales. En este trabajo, el modelo Mistral-small se utiliza como representante de esta vía. Este es un potente modelo de 24B de parámetros, una evolución significativa sobre su predecesor fundacional, Mistral-7B [Mistral AI, 2025]. Aunque sigue la misma línea de ser un modelo de Transformer de tipo decodificador, denso y auto-regresivo, su mayor escala y sus optimizaciones arquitectónicas le permiten alcanzar un rendimiento alto para su categoría "*lightweight*".

La eficiencia de este modelo no se basa en la dispersión de la arquitectura MoE, como en Gemini, sino en la implementación y refinamiento de innovaciones en el propio mecanismo de atención, heredadas de la arquitectura original de Mistral-7B [Jiang et al., 2023].

Innovaciones Arquitectónicas para la Eficiencia

Las optimizaciones de la familia de modelos Mistral se centran en resolver dos de los principales cuellos de botella de la arquitectura Transformer estándar: el coste cuadrático de la autoatención en secuencias largas y la latencia generada por el tamaño de la caché de claves y valores (KV cache) durante la decodificación.

Atención de Ventana Deslizante Para manejar secuencias largas de manera eficiente, esta arquitectura implementa la Atención de Ventana Deslizante *(Sliding Window Attention - SWA)* [Jiang et al., 2023]. En lugar de permitir que cada *token* atienda a todos los

anteriores (con un coste computacional de $O(n^2)$), SWA restringe el alcance de la atención a una ventana de tamaño fijo de palabras recientes. Esto reduce la complejidad a ser lineal ($O(n)$) con respecto a la longitud de la secuencia, permitiendo al modelo procesar contextos mucho más largos de forma eficiente, como la ventana de 128,000 *tokens* que ofrece Mistral-small [Mistral AI, 2025, Jiang et al., 2023].

Atención de Consulta Agrupada Para acelerar el proceso de inferencia, que a menudo está limitado por el ancho de banda de la memoria, Mistral-small se apoya en la Atención de Consulta Agrupada *(Grouped-Query Attention - GQA)* [Jiang et al., 2023]. GQA es una solución intermedia entre la atención multi-cabeza estándar (MHA) y la atención multi-consulta (*MQA*). En *GQA*, las cabezas de consulta se agrupan, y cada grupo comparte un único conjunto de cabezas de clave y valor. Este enfoque ha demostrado acelerar significativamente la velocidad de generación de tokens al reducir los requisitos de memoria, manteniendo una calidad muy superior a la de *MQA* [Jiang et al., 2023].

Rendimiento y Aplicación

La combinación de estas innovaciones dota a Mistral-small de un perfil de rendimiento que lo hace especialmente adecuado para aplicaciones prácticas. Su alta eficiencia se traduce en una baja latencia, lo que lo convierte en una opción idónea para tareas interactivas como la asistencia conversacional y para flujos de trabajo automatizados que dependen de la ejecución ágil de llamadas a funciones (*function calling*) [Mistral AI, 2025].

Además de su velocidad, el modelo es notablemente ligero para su potencia, con la capacidad de ejecutarse en hardware de consumo como una única GPU RTX 4090, lo que abre la puerta a casos de uso donde se puede correr el modelo en el propio dispositivo. Además, el modelo es de código abierto, lo que permite a la comunidad realizar un ajuste fino (*fine-tuning*) para adaptar el modelo a dominios concretos, como el análisis de documentos legales o el soporte técnico, y utilizarlo como una base sólida para desarrollar capacidades de razonamiento más avanzadas [Mistral AI, 2025]. Este equilibrio entre rendimiento y eficiencia lo convierten en una propuesta interesante para comparar contra un modelo de mucho más superior en cuanto a parámetros, como es Gemini 1.5.

En conclusión, este capítulo ha trazado la trayectoria evolutiva de las arquitecturas neuronales empleadas en este trabajo, desde los principios fundacionales del Transformer original hasta los modelos de vanguardia. Se ha analizado la bifurcación de la arquitectura base en dos filosofías dominantes: los modelos bidireccionales basados en el codificador, como BERT, para la comprensión profunda del lenguaje, y los modelos auto-regresivos basados en el decodificador, como GPT, que sentaron las bases para la generación a gran escala. Asimismo, se ha examinado cómo los desafíos de escalabilidad posteriores fueron abordados mediante innovaciones clave, ilustrando las dos vías principales de optimización: la eficiencia de las arquitecturas dispersas con Mezcla de Expertos (MoE) en Gemini 1.5 Pro, y la optimización de los modelos densos con técnicas de atención avanzadas en Mistral-small. Con esta sólida comprensión del panorama arquitectónico, desde sus orígenes pasando por las mejoras en la arquitectura de los modelos, el presente trabajo se encuentra en disposición de abordar la siguiente fase: la aplicación y evaluación experimental de estos modelos en la tarea de extracción de información de documentos de negocio.

Capítulo 7

Desarrollo Práctico de la Investigación

En este capítulo se expone el desarrollo práctico de la investigación. El objetivo es narrar de manera detallada el proceso de trabajo seguido, describiendo cada una de las etapas ejecutadas, desde la definición de los objetivos y la preparación de los datos hasta la configuración final del marco de experimentación y evaluación.

Para empezar, se recapitularán las fases iniciales de comprensión del negocio y de los datos, estableciendo el punto de partida del proyecto: los objetivos de extracción de información y el análisis del corpus IDSEM. A continuación, se abordará en profundidad la laboriosa fase de preparación de los datos. Se detallará el desafío que supone el formato PDF y se narrará el proceso de exploración de diferentes herramientas de conversión, justificando la elección final del formato Markdown y la librería pymupdf4llm. Además, se explicará el crucial trabajo de refinamiento del *ground truth* para asegurar una evaluación precisa.

Posteriormente, la sección de modelado y experimentación describirá el recorrido práctico que llevó a la configuración final de los experimentos. Se detallará la exploración inicial de herramientas, desde interfaces web como OpenRouter hasta la prueba de modelos en local, explicando los desafíos y las limitaciones encontradas. Se explicará la estrategia de diseño de los parámetros de generación y se narrará el proceso iterativo de refinamiento de las distintas estrategias de prompting, incluyendo los contratiempos surgidos con la evolución de los modelos de la API de Google.

Finalmente, el capítulo concluirá con la descripción de la evaluación a medida que se diseñó para este proyecto. Se explicará el flujo de trabajo programático creado para clasificar los resultados, la lógica de comparación flexible mediante la distancia de Levenshtein, y el sistema de filtrado contextual que garantiza una medición justa del rendimiento de los modelos, sentando así las bases para el análisis que se presentará en el siguiente capítulo.

Las fases iniciales del proyecto se centraron en establecer unos cimientos sólidos, definiendo primero los objetivos del trabajo para después comprender en profundidad el material de datos sobre el que se iba a trabajar.

El primer paso consistió en establecer dos niveles de objetivos. El objetivo principal del proyecto se definió como la automatización de la extracción de información clave de facturas del sector eléctrico español. La meta es desarrollar un sistema que permita reducir la carga de trabajo manual, minimizar los errores humanos inherentes a estos procesos y, fundamentalmente, estructurar los datos extraídos para que puedan ser utilizados en análisis posteriores o integrados en otros sistemas de gestión empresarial.

Esto se traduce, a un nivel más técnico, a evaluar la capacidad de los modelos extensos de lenguaje (LLM) para identificar y extraer con precisión un conjunto predefinido de campos a partir del texto de las facturas. La motivación para centrarse en este tipo de documento de negocio se fundamenta en la alta calidad y la exhaustiva documentación del dataset IDSEM, que proporciona una base robusta y fiable para la evaluación de esta tarea.

Una vez definidos los objetivos, se establecieron los **criterios de éxito** del proyecto. Cuantitativamente, el rendimiento se medirá con las métricas estándar para tareas de extracción de información: *Precision*, *Recall* y *F1-score*, calculadas para cada uno de los campos semánticos. Cualitativamente, el éxito también se medirá a través de un análisis de errores que buscará identificar patrones de fallo comunes, como los campos que se confunden con mayor frecuencia y la influencia que tiene la diversidad estructural de las plantillas en la precisión de la extracción.

Posteriormente, nos dedicamos a la comprensión de los datos. En este proyecto, esta etapa se vio notablemente simplificada al emplear el dataset público IDSEM, lo que eliminó la necesidad de una laboriosa recopilación manual de facturas. El trabajo se centró en un estudio profundo de la documentación y la estructura del corpus. Se analizó su escala (75,000 documentos), la diversidad de sus 9 plantillas de factura (basadas en diseños de grandes comercializadoras españolas) y su detallado sistema de 107 etiquetas. Un punto clave fue comprender la relación entre los tres tipos de ficheros proporcionados para cada documento: el PDF visual, el PDF anotado y, de manera crucial, el fichero JSON que contiene el *ground truth* y que serviría como base para la evaluación. Tras esta exploración, se concluyó que la riqueza, estructura y realismo del conjunto de datos se alineaban con los objetivos del proyecto, validando su idoneidad como corpus experimental.

7.1. Preparación de los Datos: de PDF a Markdown

La fase de preparación de los datos fue una de las más importantes del proyecto, ya que abarcó todas las tareas necesarias para transformar el corpus de facturas en un formato final que fuera procesable y comprensible para los modelos extensos de lenguaje. Además, de él dependía la calidad de los datos que procesarían los LLM, afectando directamente a la calidad de las extracciones.

El punto de partida del proyecto fue el conjunto de datos IDSEM, que proporciona las facturas en formato PDF [Sánchez et al., 2022]. El PDF, a pesar de ser un estándar para la distribución de documentos, presenta desafíos significativos para la extracción automática de información. Su estructura es inherentemente compleja, ya que puede contener una mezcla de texto, imágenes, tablas con columnas irregulares y fuentes de diferentes tipos y tamaños, lo que dificulta un parseo consistente del contenido textual [Ho-Thanh-Tam and Tamine-Lechani, 2022].

Para que un LLM pudiera procesar eficazmente el contenido de las facturas, era indispensable convertir los documentos a un formato de texto más estructurado. La primera herramienta evaluada fue una API de terceros, LLMWhisperer [Unstract, 2025], que mostraba convertir los ficheros PDF a formato de texto plano manteniendo la disposición espacial del documento original, lo cual se consideró inicialmente como una solución.

Como se muestra en la Figura 7.1, este formato preservaba la maquetación visual mediante el uso de espacios en blanco.

Sin embargo, tras procesar una parte del subconjunto de datos, el proveedor del servicio restringió el uso gratuito, presumiblemente debido al alto volumen de uso, lo que obligó

```
                    DATOS DE LA FACTURA
IMPORTE FACTURA:                          191,32 EUR
N. factura: ID8475477607
Referencia: 724759220605/7202
Periodo de consumo: 09/12/2015 a 07/02/2016
Fecha de cargo: 12 de febrero de 2016

EDP ESPANA, S.A                            Carina Villasenor Talavera
Cif: A33473752                            GC-1
PLAZA DE LA GESTA, N.2 33007 - OVIEDO       50512 Torrellas
                                           Zaragoza

                    FACTURA RESUMEN
Por potencia contratada                  31,70 EUR
Por energia consumida                    135,55 EUR
Impuesto electricidad                    8,55 EUR
Alquiler equipos de medida y control     3,01 EUR
IGIC reducido ( 7%)                      12,31 EUR
IGIC normal ( 6,5%)                      0,20 EUR

TOTAL IMPORTE FACTURA                    191,32 EUR
```

Figura 7.1: Fragmento del formato de salida de la API LLMWhisperer.

a detener el proceso y buscar una nueva estrategia.

La búsqueda de alternativas llevó a la consideración de la librería `pymupdf4llm`, una herramienta diseñada específicamente para convertir documentos PDF a formato Markdown. La elección final de esta librería se fundamentó en lo siguiente: la comunidad de inteligencia artificial ha constatado que los LLM procesan el formato Markdown con mayor eficacia que el texto plano, y además, un artículo científico reciente de [Braun et al., 2025] demostró que proporcionar a los modelos esta estructura textual explícita que preserva elementos como títulos, listas y párrafos puede suponer una mejora de hasta un 23 % en la precisión en tareas de extracción, en comparación con el texto extraído por herramientas de OCR comerciales.

A pesar de sus ventajas, se tuvo en cuenta que esta librería presentaba ciertas limitaciones, como la incapacidad de extraer texto de imágenes, gráficas o de los bordes del documento. No obstante, tras el fallo de la estrategia inicial, se consideró la solución más robusta y se procedió a desarrollar un script en Python para convertir las 2,400 facturas seleccionadas al formato Markdown.

El resultado de esta conversión, como se puede ver en la Figura 7.2, es un texto mucho más limpio y estructurado semánticamente.

Una vez convertidas las facturas, la preparación de los datos se centró en el tratamiento de los ficheros JSON que contienen el *ground truth*. Un análisis detallado de estos ficheros reveló dos problemas fundamentales para la evaluación:

```
EDP ESPANA, S.A
Cif: A33473752
PLAZA DE LA GESTA, N.2 33007 - OVIEDO

### **DATOS DE LA FACTURA**
- **IMPORTE FACTURA:** 191,32 EUR
- **N. factura:** ID8475477607
- **Referencia:** 724759220605/7202
- **Periodo de consumo:** 09/12/2015 a 07/02/2016
- **Fecha de cargo:** 12 de febrero de 2016

---
**Carina Villasenor Talavera**
GC-1, 50512 Torrellas, Zaragoza

### **FACTURA RESUMEN**
| Concepto | Importe |
| :--- | :--- |
| Por potencia contratada | 31,70 EUR |
| Por energia consumida | 135,55 EUR |
| Impuesto electricidad | 8,55 EUR |
| Alquiler equipos de medida y control | 3,01 EUR |
| IGIC reducido ( 7%) | 12,31 EUR |
| IGIC normal ( 6,5%) | 0,20 EUR |

### **TOTAL IMPORTE FACTURA 191,32 EUR**
```

Figura 7.2: Fragmento de una factura convertida a formato Markdown.

1. **Discrepancia de Etiquetas:** Cada fichero JSON contenía claves para las 107 etiquetas posibles, pero el contenido visual de cada una de las 6 plantillas de factura solo mostraba un subconjunto diferente de estas.

2. **Campos Anotados Vacíos:** Debido a la naturaleza sintética del dataset, se encontraron casos en los que una etiqueta existía en el JSON, pero su valor era una cadena de texto vacía, generando ambigüedad.

Para realizar una evaluación justa y precisa, era indispensable saber exactamente qué etiquetas eran extraíbles en cada plantilla. Para ello, se utilizaron los PDFs anotados que proporciona el propio dataset. Estos documentos, que marcan cada campo con su etiqueta correspondiente (ej. `#A1 Javier Gómez #A1`), se procesaron con el mismo script de conversión a Markdown.

Posteriormente, se desarrolló un segundo script en Python que, mediante el uso de expresiones regulares, recorría estos nuevos ficheros Markdown anotados y realizaba dos tareas: primero, contabilizaba todas las etiquetas presentes en cada una de las plantillas; y segundo, verificaba que el contenido de dichas etiquetas no estuviera vacío. El resultado

de este proceso fue un **mapeo detallado en formato JSON**, que documenta, para cada plantilla, el subconjunto exacto de etiquetas que están realmente presentes y contienen valor. Este mapeo se convirtió en una pieza clave del proyecto, ya que permitió filtrar el *ground truth* en la fase de evaluación, asegurando que los modelos solo fueran evaluados por la extracción de los campos que realmente existían y eran visibles en los documentos de entrada.

7.2. Modelado y Configuración

Una vez preparado el corpus de datos, el núcleo de la investigación se centró en la fase de modelado y experimentación. Dado que este proyecto utiliza modelos de lenguaje pre-entrenados, el "modelado" no consistió en la creación de nuevas arquitecturas, sino en la **aplicación y calibración** de estos sistemas para la tarea específica de extracción de información.

Este proceso se articuló en torno a los dos ejes principales que permiten dirigir el comportamiento de un LLM: la **configuración de los parámetros de generación**, que modulan la aleatoriedad y diversidad de las respuestas, y el **diseño de las instrucciones (*prompts*)**, que definen la tarea y guían al modelo hacia el resultado deseado.

Las siguientes subsecciones narran de manera cronológica el proceso de trabajo seguido para definir la configuración final de los experimentos. Se comenzará detallando la exploración inicial de herramientas y los desafíos que llevaron a la selección del entorno de trabajo definitivo. A continuación, se explicará la estrategia diseñada para la configuración de los parámetros y, finalmente, se describirá el proceso iterativo de diseño y refinamiento de las distintas estrategias de prompting, que constituyó la parte más intensiva de esta fase.

7.2.1. Exploración Inicial de Herramientas y Desafíos

La primera etapa del trabajo consistió en una exploración de las herramientas disponibles para interactuar con los LLM. Inicialmente, se realizaron pruebas preliminares utilizando las interfaces web de modelos como ChatGPT, Gemini, DeepSeek y Mistral para obtener una primera impresión de su capacidad de extracción con bocetos iniciales de *prompts*. Sin embargo, estas interfaces no suelen permitir la configuración de los parámetros de generación (temperatura, `Top-p`, `Top-k`), lo que las hacía inviables para una experimentación sistemática y controlada.

Una excepción notable fue la plataforma **OpenRouter** [OpenRouter, 2025], que sí ofrecía la posibilidad de modificar estos parámetros directamente en su interfaz web, como se muestra en la Figura 7.4. En este panel, se muestra como se puede modificar el máximo número de *tokens* en su respuesta, la temperatura, los parámetros `Top-p` y texttttTop-k, entre muchos otros. Además, permitía realizar consultas a varios modelos simultáneamente, facilitando una comparación cómoda de sus respuestas para una misma instrucción (ver Figura 7.3). Aunque inicialmente pareció una opción muy atractiva, OpenRouter limitó su plan gratuito a 50 peticiones diarias debido a su creciente popularidad, una restricción que hacía imposible la ejecución de los miles de experimentos planificados.

Ante las limitaciones de las APIs en la nube, se exploró la alternativa de ejecutar un modelo de lenguaje en local. Esta vía ofrecía ventajas como la ausencia de costes por uso y una mayor privacidad de los datos. Sin embargo, presentaba un desafío técnico

Figura 7.3: Interacción con varios modelos a la vez en la Interfaz Web de Openrouter. Fuente: Captura de pantalla propia [OpenRouter, 2025].

importante: los recursos computacionales. Con una GPU NVIDIA RTX 2060 de 6GB de VRAM, solo era posible ejecutar modelos muy pequeños. Se realizaron pruebas con una versión cuantizada de 0.6B de parámetros del modelo Qwen3 [Qwen Team, 2024] utilizando el software Ollama [Ollama, 2025]. La cuantización reduce la precisión de los parámetros del modelo para disminuir su tamaño, a menudo a costa de una pérdida de rendimiento. Las pruebas confirmaron rápidamente que un modelo de estas características carecía de la capacidad y la ventana de contexto necesarias para comprender y extraer información de documentos tan complejos como las facturas.

7.2.2. Configuración y Estrategia de los Parámetros de Generación

La siguiente etapa del modelado consistió en definir los experimentos para evaluar el impacto de los parámetros de generación en la calidad de la extracción. Basándonos en la documentación oficial del modelo Gemini, que establece unos rangos operativos predeterminados (temperatura: 0.0-2.0, `Top-p`: 0.0-1.0, `Top-k`: 64), se diseñó una búsqueda sistemática para explorar un amplio espectro de valores dentro de dichos rangos.

Se partió de una hipótesis fundamental: para una tarea de extracción de datos de documentos estructurados, donde no se requiere creatividad ni generación de contenido nuevo, los resultados más precisos deberían obtenerse con configuraciones que minimicen la aleatoriedad. Por lo tanto, se esperaba que los valores de temperatura, `Top-p` y `Top-k` más bajos, que hacen que el modelo responda con las subpalabras más probables, arrojaran un mejor rendimiento. Para comprobar esta hipótesis, se diseñó tanto un experimento de búsqueda voraz (*greedy search*) con temperatura 0, que asegura la selección del *token* más probable, como una serie de 18 experimentos con valores progresivamente más altos de aleatoriedad. Para una descripción teórica detallada de cada parámetro, se puede consultar el Capítulo 8.

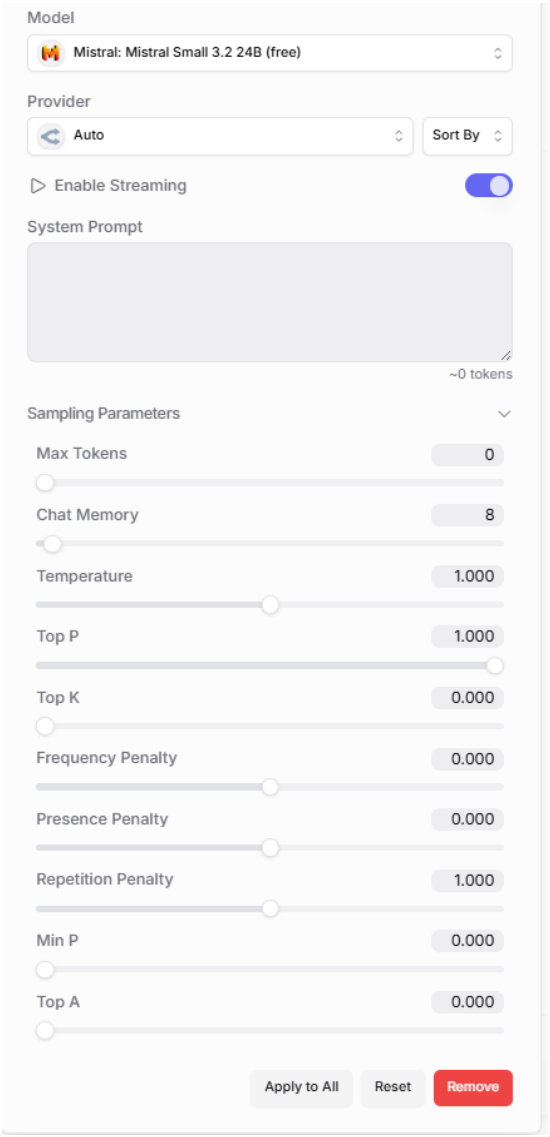


Figura 7.4: Parámetros de aleatoriedad del modelo en la Interfaz Web de Openrouter. Fuente: Captura de pantalla propia [OpenRouter, 2025].

7.2.3. Diseño y Refinamiento de las Estrategias de Instrucciones

La última y más intensiva etapa del modelado se centró en el diseño de los *prompts*. La *prompt engineering* o ingeniería de instrucciones es una disciplina crucial, ya que la forma en que nos comunicamos con un LLM determina en gran medida la calidad de sus respuestas. Para el diseño de los *prompts* de este proyecto, se tomaron como referencia las mejores prácticas publicadas por los principales laboratorios de investigación, como las guías de OpenAI y Google [OpenAI, 2025, Cloud, 2025c], y el influyente artículo de [Brown and Mann, 2020], que sentó las bases de las técnicas *zero-shot* y *few-shot*.

El desarrollo de cada tipo de instrucción fue un proceso iterativo. Para facilitar este ciclo de prueba y error, se desarrollaron dos scripts principales en Python que automatizaban la experimentación. El primer script, `Gemini_1.5_shot_prompting.py`, se diseñó para evaluar las estrategias *zero-shot* y *few-shot*, leyendo el *prompt* desde un fichero de texto y realizando una única llamada a la API por factura. El segundo script, `Gemini_1.5_iterativePrompting.py`, se creó para la estrategia de extracción iterativa, construyendo

dinámicamente un *prompt* único para cada etiqueta a partir de una plantilla y un fichero JSON de instrucciones detalladas. Con estos scripts, se realizaban pruebas con pequeños lotes de 10 facturas para observar el comportamiento del modelo y refinar la instrucción correspondiente antes de proceder a la evaluación a gran escala. Para realizar la extracción de Mistral-small se tuvieron que desarrollar scripts diferente, pero que realizaban el mismo mecanismo de extracción que Gemini. Esto es debido al distinto funcionamiento de la API entre estos modelos. Además, convenía tenerlo en dos scripts diferentes para ejecutarlos de manera independiente, pudiendo ejecutarse a la vez. El punto de partida fue el *prompt zero-shot*, construido a partir de las descripciones de las etiquetas del dataset IDSEM. Los *prompts* de tipo *few-shot* se crearon sobre esta base, añadiendo ejemplos y refinando las instrucciones para que el modelo siguiera el patrón deseado. Finalmente, el desarrollo del *prompt* iterativo fue la tarea más laboriosa, ya que requirió un análisis manual de las 107 etiquetas para crear instrucciones y ejemplos específicos para cada una, dado que la definición original del dataset a menudo resultaba insuficiente para que el LLM comprendiera el contexto de cada campo sin ambigüedad.

7.3. Ejecución de los Experimentos

Una vez teníamos los *prompts* afinados y la configuración de los experimentos, debíamos evaluar el rendimiento de los modelos con el corpus de 2400 facturas obtenido del conjunto de datos. Se usaron los propios scripts realizados para probar el rendimiento de los *prompts*, para que empezaran a procesar las 2400 facturas por experimento. Estos scripts realizaban, factura por factura, una llamada a la API que enviaba el *prompt*, la factura, y los parámetros temperatura, top p y top k en función del experimento realizado.

Para ello, se diseñó un flujo de trabajo a medida que permitiera no solo medir, sino también interpretar los resultados, teniendo en cuenta las particularidades del dataset y del preprocesamiento realizado. El proceso se automatizó mediante el desarrollo de varios scripts en Python.

El primer paso fue la creación de un script de comparación y clasificación de resultados. Este programa se diseñó para procesar las extracciones generadas por los LLM (en formato JSON) y compararlas directamente con los ficheros de *ground truth*. Para cada campo de cada factura, el script asignaba uno de los siguientes tres estados: Verdadero Positivo (TP), Falso Positivo (FP) o Falso Negativo (FN), que se almacenaban en un nuevo fichero de resultados.

Para determinar si una extracción era correcta (un Verdadero Positivo), la comparación no se limitaba a una simple igualdad de cadenas de texto. Se implementó una lógica más flexible para evitar penalizar al modelo por errores menores que no afectaban al valor semántico del dato. Concretamente, antes de la comparación, tanto la cadena extraída como la del *ground truth* se convertían a minúsculas. Esto se hizo para evitar falsos negativos debidos a diferencias de mayúsculas (por ejemplo, cuando un nombre aparecía en mayúsculas en una parte de la factura y en minúsculas en otra). Adicionalmente, se aplicó una distancia de Levenshtein con un umbral de 1, considerando como correcta una extracción que tuviera como máximo un carácter de diferencia con respecto al valor real. Esta medida se adoptó tras observar que, en ocasiones, el modelo extraía correctamente un valor pero añadía un carácter extraño, como un punto o una coma al final, un error que no debía invalidar la extracción.

Una vez clasificados todos los resultados, el segundo paso fue aplicar un filtrado contextual. Como se mencionó en la fase de preparación de los datos, cada una de las plantillas

de factura contiene un subconjunto diferente de las 107 etiquetas totales. Para asegurar una evaluación justa, era indispensable filtrar los resultados y eliminar aquellos campos que no eran aplicables a cada plantilla.

Para ello, se desarrolló un segundo script que leía los ficheros de resultados clasificados (con los TP, FP y FN) y, para cada factura, consultaba el mapeo de etiquetas previamente generado. El script identificaba la plantilla de la factura evaluada, recuperaba la lista de etiquetas válidas para esa plantilla y eliminaba del resultado cualquier campo que no perteneciera a dicha lista.

Este proceso de filtrado que aseguró que los modelos no fueran penalizados por no extraer información que, simplemente, no existía en el documento de entrada. El conjunto de datos resultante de este filtrado constituye la base final y depurada sobre la cual se calcularon las métricas de rendimiento y se realizaron los análisis que se presentan en el siguiente capítulo.

7.4. Evaluación

La fase final del proceso metodológico consistió en la evaluación del rendimiento de los modelos. Para ello, se diseñó un flujo de trabajo a medida que permitiera no solo medir, sino también interpretar los resultados, teniendo en cuenta las particularidades del dataset y del preprocesamiento realizado. El proceso se automatizó mediante el desarrollo de varios scripts en Python.

El primer paso fue la creación de un script de comparación y clasificación de resultados. Este programa se diseñó para procesar las extracciones generadas por los LLM (en formato JSON) y compararlas directamente con los ficheros de *ground truth*. Para cada campo de cada factura, el script asignaba uno de los siguientes tres estados:

- Verdadero Positivo (TP): El modelo extrae un valor para una etiqueta y este coincide con el valor real en el documento.
- Falso Positivo (FP): El modelo extrae un valor para una etiqueta, pero este es incorrecto y no corresponde a esa etiqueta.
- Falso Negativo (FN): El modelo no extrae ningún valor para una etiqueta, a pesar de que esta existía en el documento. Es un error de omisión.

Para determinar si una extracción era correcta (un Verdadero Positivo), la comparación no se limitaba a una simple igualdad de cadenas de texto. Se implementó una lógica más flexible para evitar penalizar al modelo por errores menores que no afectaban al valor semántico del campo. Concretamente, antes de la comparación, tanto la cadena extraída como la del *ground truth* se convertían a minúsculas. Adicionalmente, se aplicó una distancia de Levenshtein con un umbral de 1, considerando como correcta una extracción que tuviera como máximo un carácter de diferencia con respecto al valor real [Wagner and Fischer, 1974]. Esta medida se adoptó tras observar que, en ocasiones, el modelo extraía correctamente un valor pero añadía un carácter extraños, como un punto o una coma al final.

Esta clasificación en TP (*True Positive*), FP (*False Positive*) y FN (*False Negative*) se estableció como paso previo y necesario para poder aplicar las métricas estándar de evaluación de rendimiento en tareas de extracción, como son la *Precision*, el *Recall* y *F1-score* [Sokolova and Lapalme, 2009]. A partir de estos conceptos, las métricas se definen de la siguiente manera:

Precision Mide la calidad de las extracciones realizadas. De todos los valores que el modelo ha identificado y extraído para una etiqueta, la precisión calcula qué fracción de ellos eran realmente correctos. Una precisión alta indica que el modelo es fiable y comete pocos errores al afirmar que ha encontrado un dato. Su fórmula es:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall Mide la completitud o la capacidad del modelo para encontrar todos los datos relevantes. De todos los valores que realmente existían en el documento para una etiqueta, el recall calcula qué fracción de ellos fue encontrada por el modelo. Un recall alto indica que el modelo es capaz de extraer la mayoría de los datos presentes en la factura. Su fórmula es:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-Score Esta métrica proporciona un único indicador de rendimiento que equilibra la *Precision* y el *Recall*. Es la media armónica de ambas, penalizando los casos en los que una de las dos métricas es muy baja. Un F1-score alto indica un buen equilibrio entre la calidad y la cantidad de las extracciones. Su fórmula es:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precisión} + \text{Recall}}$$

Una vez clasificados todos los resultados, el paso final fue aplicar un filtrado de las etiquetas. Como se mencionó en la fase de preparación de los datos, cada una de las plantillas de factura contiene un subconjunto diferente de las 107 etiquetas totales. Para asegurar una evaluación justa, era indispensable filtrar los resultados y eliminar aquellos campos que no eran aplicables a cada plantilla. Para ello, se desarrolló un segundo script que, para cada factura, consultaba el mapeo de etiquetas previamente generado y eliminaba del cómputo cualquier resultado (TP, FP o FN) que no perteneciera a dicha plantilla.

Este proceso de filtrado aseguró que los modelos no fueran penalizados por no extraer información que, simplemente, no existía en el documento de entrada. El conjunto de datos resultante de este filtrado constituye la base final sobre la cual se calcularon las métricas de rendimiento y se realizaron los análisis que se presentan en el siguiente capítulo.

Capítulo 8

Configuración de Experimentos

En este capítulo se detalla la configuración de los experimentos llevados a cabo para evaluar el rendimiento de los modelos extensos de lenguaje (LLM) en la tarea de extracción de información. El diseño experimental busca analizar de manera sistemática cómo diferentes estrategias de generación de texto influyen en la precisión y calidad de los resultados.

La configuración se ha estructurado en dos componentes principales que se describirán en las siguientes secciones. Primero, se abordarán los parámetros de generación, que controlan la aleatoriedad y diversidad del texto generado por el modelo. A continuación, en una sección posterior, se detallarán las diferentes estrategias de ingeniería de instrucciones (*prompt engineering*) que se han diseñado para guiar al modelo en la tarea de extracción.

8.1. Parámetros y Estrategias de Muestreo

La generación de texto en los LLM es un proceso secuencial. En cada paso, el modelo calcula una distribución de probabilidad sobre todo su vocabulario para predecir cuál será el siguiente *token*. La estrategia utilizada para seleccionar una de las subpalabras de esta distribución, conocida como estrategia de decodificación o muestreo, es fundamental y tiene un impacto directo en la coherencia, diversidad y predictibilidad de la respuesta final. Estas estrategias se pueden agrupar en dos grandes familias: deterministas y estocásticas.

Mientras que las estrategias deterministas como la búsqueda codiciosa (*greedy search*) siempre seleccionan el *token* con la probabilidad más alta, resultando en respuestas muy predecibles pero a menudo repetitivas y poco naturales, las estrategias estocásticas introducen un grado de aleatoriedad. En este trabajo nos hemos centrado en evaluar ambas, controlando los parámetros de `temperatura`, `Top-K` y `Top-P`.

8.1.1. Temperatura, Top-P y Top-K

El parámetro de temperatura ajusta la aleatoriedad de la predicción de *tokens*. Técnicamente, funciona re-escalando los valores *logits* (las puntuaciones brutas antes de la función softmax) de la distribución de probabilidad de estos. Un valor de temperatura bajo (cercano a 0) agudiza la distribución, haciendo que los *tokens* de alta probabilidad sean aún más probables y los de baja probabilidad casi imposibles de seleccionar. Esto resulta en respuestas más deterministas y enfocadas, para tareas en las que se requiere un resultado más seguro, siendo ideal para condiciones en las que se van a realizar las mismas pruebas varias veces y en las que se quiere reducir lo máximo posible la aleatoriedad que

ofrecen los modelos extensos de lenguaje. En cambio, mientras más aumentamos el valor de la temperatura, más se aplana la distribución, aumentando la probabilidad de que se seleccionen palabras o subpalabras menos comunes, lo que conduce a respuestas más diversas, creativas y, a veces, inesperadas. Este principio de control de la aleatoriedad fue introducido en el contexto de las redes neuronales en el trabajo de [Ackley et al., 1985] para su aplicación en las Máquinas de Boltzmann.

El muestreo `Top-K` es una estrategia que limita el conjunto de *tokens* a considerar para la siguiente predicción. En lugar obtener una muestra que contenga todo el vocabulario, suponiendo un gran conjunto de subpalabras entre las que elegir, el modelo primero filtra los K tokens con la probabilidad más alta. A continuación, la probabilidad se redistribuye únicamente entre estos K tokens y se realiza el muestreo. Este método evita que se seleccionen subpalabras extrañas o sin sentido que, aunque con una probabilidad muy baja, podrían ser elegidos en un muestreo con temperatura alta. Sin embargo, su principal limitación es que el tamaño del conjunto de candidatos es siempre fijo (K), lo que puede ser demasiado restrictivo si la distribución de probabilidad real es muy plana [Fan et al., 2018].

A diferencia del anterior, el muestreo por núcleo, o `Top-P` (también conocido como Nucleus Sampling), es una alternativa más sofisticada y dinámica. En lugar de seleccionar un número fijo de *tokens* (K), `Top-P` selecciona el conjunto más pequeño posible de subpalabras cuya probabilidad acumulada sea superior al umbral p. De esta manera, el tamaño del conjunto de muestreo se adapta dinámicamente en cada paso: si el modelo está muy seguro del siguiente *token*, el conjunto ("núcleo") será pequeño; si está inseguro, el conjunto será más grande para incluir más opciones plausibles. Esta técnica evita la selección de palabras de la "cola" de la distribución y ha demostrado producir texto de mayor calidad que el muestreo `Top-K` [Holtzman et al., 2019].

8.1.2. Combinación de Parámetros Usada

Para evaluar el impacto de estos parámetros en la tarea de extracción, se diseñaron dos conjuntos de experimentos. El primero fue una búsqueda sistemática o *grid search*, con el objetivo de explorar un amplio rango de la aleatoriedad y diversidad en las respuestas, así como de entender las interacciones entre los parámetros. Estos valores se encuentran dentro del rango predeterminado para el funcionamiento de Gemini [Google Cloud, 2025b]. Manteniéndonos dentro del funcionamiento "óptimo" por parte del modelo según los desarrolladores, se definieron todas las combinaciones posibles de los siguientes valores: temperaturas de 1.0, 1.5 y 2.0; valores de `Top-P` de 0.5, 0.7 y 0.95; y valores de top_k de 32 y 64.

Para ilustrar cómo interactúan estos parámetros, se puede considerar el proceso de decodificación para la frase "El gato se sentó sobre la ". Inicialmente, el modelo genera una puntuación o *logit* para cada *token* potencial, como "alfombra" (4.5), "silla" (3.8) o "mesa" (3.5). A continuación, la temperatura (por ejemplo, un valor alto de 1.5) se aplica para re-escalar estas puntuaciones, aplanando la distribución de probabilidad y dando más oportunidad a palabras menos evidentes. Finalmente, se aplican los filtros de muestreo. Un valor de `top_k=3` restringiría las opciones a las tres subpalabras más probables, mientras que un `Top-P=0.7` seleccionaría el conjunto más pequeño de *tokens* cuya probabilidad acumulada supere ese umbral. La subpalabra final se elige aleatoriamente de este conjunto ya filtrado, demostrando cómo la combinación de los tres parámetros modela la respuesta final.

El segundo conjunto experimental consistió en una única configuración determinista para establecer una línea base. En este caso, se utilizó la estrategia de búsqueda voraz (*greedy sampling*) fijando la temperatura en 0. Como se ha explicado, este valor colapsa la distribución de probabilidad, haciendo que el modelo seleccione siempre el *token* más probable y, por consiguiente, anula el efecto de los parámetros top_p y top_k [Google Cloud, 2025a]. Este experimento es crucial, ya que sirve como punto de referencia para medir con precisión cómo la introducción de aleatoriedad afecta al rendimiento de la extracción.

Es importante destacar que, debido a las limitaciones de recursos económicos y de tiempo mencionadas en el capítulo de planificación, la totalidad de estos 19 experimentos (las 18 combinaciones de la búsqueda sistemática más el experimento determinista) se llevó a cabo exclusivamente con el modelo Gemini 1.5 Pro. Esta decisión, aunque limita la comparativa directa entre diferentes modelos, permitió realizar un seguimiento más profundo y exhaustivo. El objetivo fue analizar en detalle cómo la variación de los parámetros de generación afecta el comportamiento de un único modelo de alto rendimiento en la tarea específica de extracción de información de facturas, priorizando así la profundidad del análisis sobre una comparativa más superficial entre varios modelos.

8.2. Estrategias de Ingeniería de Instrucciones

Además de los parámetros de generación, el segundo componente fundamental en la configuración de los experimentos es el diseño de las instrucciones o *prompts*. La ingeniería de instrucciones se ha consolidado como la principal disciplina para guiar y especializar el comportamiento de los LLM en tareas específicas sin necesidad de reentrenamiento [Liu et al., 2021]. Un *prompt* bien diseñado no solo define la tarea que debe ejecutar el LLM, sino que también establece el formato de salida, las restricciones a seguir y el contexto necesario para que el modelo actúe de la manera deseada.

Para este proyecto, se han diseñado y evaluado sistemáticamente varias estrategias de *prompting*. En lugar de presentar el texto íntegro de cada *prompt*, se describirá su estructura conceptual. Cada estrategia parte de una instrucción de rol común, que establece el perfil del modelo. El contenido completo y detallado de cada *prompt* se puede consultar en el apéndice A.

8.2.1. System Prompt

Todos los experimentos se inician con un *system prompt* o una instrucción de rol. Esta técnica, también conocida como *role-play prompting*, consiste en enviar un mensaje inicial para establecer el contexto del modelo y definir un perfil o persona específicos para la tarea a realizar [Kong et al., 2023]. La investigación en este campo ha demostrado que asignar un rol de experto a un LLM no es un simple adorno, sino una estrategia que puede mejorar significativamente su rendimiento, especialmente en tareas que requieren razonamiento.

El mecanismo subyacente es que, al adoptar una persona, el modelo se condiciona para generar respuestas más coherentes y alineadas con las capacidades de dicho rol. Un hallazgo fundamental del trabajo de [Kong et al., 2023] es que el *role-play prompting* puede actuar como un desencadenante de la Cadena de Pensamiento (*Chain-of-Thought*). Es decir, al pedirle al modelo que actúe como un especialista, se le incita a estructurar su

"razonamiento" interno de una manera más lógica y metódica para llegar a la respuesta, incluso sin la instrucción explícita de "pensar paso a paso".

En el contexto de este proyecto, el objetivo del *system prompt* es precisamente activar este comportamiento de experto. Al definir al modelo como un "especialista en extracción de datos estructurados", se busca que aborde la tarea con mayor rigor y precisión. El prompt de sistema utilizado en todos los experimentos es el siguiente:

```
Eres un modelo avanzado de IA especializado en la extracción de datos
estructurados de facturas, recibos de la luz y documentos similares.
Tu tarea consiste en extraer con precisión la información basándote en
etiquetas predefinidas y respetando estrictamente las reglas de formato
e integridad. Debes responder en formato JSON según las instrucciones del
usuario.
```

8.2.2. Zero-Shot

La primera estrategia evaluada, y la más fundamental en la interacción con LLM, es el aprendizaje *zero-shot*. En este paradigma, se le pide al modelo que realice una tarea basándose únicamente en una instrucción en lenguaje natural, sin proporcionarle ningún ejemplo de demostración [Brown and Mann, 2020]. El éxito de esta técnica depende por completo de la capacidad del modelo para comprender la tarea y generalizar a partir del vasto conocimiento que ha adquirido durante su fase de pre-entrenamiento, sin ninguna guía contextual en el momento de la inferencia [Guide, 2024b].

Aunque es el método más directo y sencillo de implementar, también es el más exigente para el modelo. Como señalan los autores de GPT-3, este enfoque puede ser "injustamente difícil" para tareas complejas donde el formato de salida deseado o los matices de la tarea no pueden ser comunicados de manera inequívoca solo con una descripción [Brown and Mann, 2020]. A pesar de esta limitación, el rendimiento en *zero-shot* sirve como una línea base crucial, ya que mide la capacidad de razonamiento "pura" del modelo ante un problema nuevo.

En el contexto de este proyecto, el *prompt zero-shot* fue diseñado para ser lo más claro y completo posible dentro de estas limitaciones. Incluía una descripción detallada de la tarea de extracción, la lista completa de las 107 etiquetas con sus respectivas descripciones y un conjunto de reglas de formato explícitas. La estructura conceptual de esta instrucción es la siguiente:

- **Descripción de la Tarea:** Petición general para extraer información de una factura.
- **Reglas de Formato:** Instrucciones claras sobre cómo tratar unidades monetarias, fechas y saltos de línea.
- **Esquema de Salida:** Un bloque JSON con las 107 claves y una breve descripción de cada valor que el modelo debía extraer.
- **Entrada de la Factura:** El texto completo de la factura a procesar, claramente delimitado.
- **Instrucción de Salida:** Indicación final para que el modelo devuelva únicamente el objeto JSON completo.

En la Figura 8.1, se muestra la estructura de este *prompt*.

```
A continuacion se muestra el texto extraido de una factura...
Su tarea consiste en extraer informacion especifica...

### Category Matching:
Extraiga los datos de cada una de las 107 etiquetas...

### Uso de Unidades:
No incluya las unidades de los datos extraidos...

### Formatos de Fecha:
Si una etiqueta es una fecha, las etiquetas terminadas en:
- "s": Formato de barra oblicua (dd/mm/aaaa).
- "p": Formato punto (dd.mm.aaaa).
...

### Formato de salida:
La informacion extraida debe salir en formato \gls{JSON} valido.
Se estructurara de la siguiente manera:

{{
    'A1': 'Nombre del cliente',
    'A2': 'Numero de identificacion del cliente',
    ...
    'N8': 'Precio del impuesto'
}}

### Entrada:
El texto extraido de la factura de la luz se proporcionara a
continuacion.
### Salida:
Devuelve solamente el objeto JSON estructurado...
```

Figura 8.1: Estructura conceptual del *prompt* utilizado para la estrategia *zero-shot*, mostrando los diferentes bloques de instrucciones.

8.2.3. Few-Shot

El aprendizaje *few-shot*, también conocido como aprendizaje en contexto (*in-context learning*), es una técnica de *prompting* avanzada que se ha consolidado como uno de los paradigmas más efectivos para guiar a los LLM. A diferencia del enfoque *zero-shot*, esta técnica proporciona al modelo uno o más ejemplos completos de la tarea resuelta directamente dentro del *prompt*, en el momento de la inferencia y sin necesidad de realizar ninguna actualización de los pesos del modelo [Brown and Mann, 2020].

El concepto, popularizado en el influyente artículo sobre GPT-3, se basa en la idea de que los modelos de lenguaje a gran escala desarrollan durante su pre-entrenamiento una

capacidad de meta-aprendizaje. Esto les permite utilizar los ejemplos proporcionados en el contexto no como datos de entrenamiento, sino como una demostración que les enseña a reconocer y adaptarse rápidamente a un nuevo patrón o tarea [Brown and Mann, 2020]. La eficacia del aprendizaje *few-shot* depende en gran medida de la calidad y el formato de los ejemplos proporcionados. El modelo aprende a replicar el formato de salida y el tipo de razonamiento implícito en las demostraciones. Por ello, es crucial que los ejemplos sean consistentes y representativos de la tarea que se quiere realizar [Guide, 2024a]. La estructura de un *prompt few-shot* consiste típicamente en proporcionar 'K' pares de [contexto de ejemplo] y su [resultado deseado], seguidos por el nuevo contexto para el cual se espera que el modelo genere un resultado, imitando el patrón demostrado [Brown and Mann, 2020].

La principal ventaja de este enfoque es la drástica reducción en la necesidad de grandes conjuntos de datos específicos para cada tarea, superando así una de las mayores limitaciones del ajuste fino tradicional [Brown and Mann, 2020]. Como demostraron los autores, el rendimiento de los modelos en una amplia variedad de tareas suele escalar positivamente con el número de ejemplos proporcionados en el contexto, hasta el límite que permite la ventana de atención del modelo [Brown and Mann, 2020]. La estructura general de estos *prompts* está listada a continuación.

- **1. Descripción de la Tarea y Reglas de Formato.**
- **2. Esquema de Salida.**
- **3. Ejemplos de Referencia:** Uno o más pares completos de [Texto de Factura de Ejemplo] y su [Extracción JSON Correcta].
- **4. Entrada de la Nueva Factura:** El texto de la factura a procesar.

Para este proyecto, se aplicó este paradigma para guiar al modelo en la compleja tarea de extracción. Se diseñaron y evaluaron sistemáticamente las siguientes variantes de la estrategia *few-shot* para explorar diferentes hipótesis sobre qué constituye una demostración efectiva.

Few-Shot con Ejemplo Anotado. Esta es la primera versión de la estrategia. En el *prompt*, además de las instrucciones, se incluye un ejemplo completo de una factura cuyo texto ha sido anotado con marcadores que señalan qué parte del texto corresponde a cada etiqueta (ej. #A1 Javier Gómez #A1). Junto a esta factura anotada se proporciona su extracción JSON correcta. La hipótesis es que estas anotaciones explícitas facilitan al modelo el aprendizaje de la correspondencia entre el texto y las etiquetas. A continuación se muestra un ejemplo de la estructura de este *prompt* usada en el presente trabajo:

Few-Shot con Ejemplo No Anotado. En esta segunda variante, el ejemplo proporcionado en el *prompt* consiste en una factura sin anotar (texto plano) y su correspondiente JSON. El propósito de este experimento es determinar si el modelo es capaz de inferir las correspondencias por sí mismo sin la ayuda de las anotaciones explícitas, y comparar si esta capacidad de "auto-aprendizaje" es tan efectiva como la instrucción que ofrece el ejemplo anotado. La estructura es similar a la anterior, pero el texto de la factura de ejemplo se presenta sin los marcadores `#...`.

```
[Instrucciones Generales y Esquema de Salida (Idéntico al Zero-Shot)]

### Ejemplo de Referencia (Factura Anotada y Extraccion Correcta)
Para ayudarte a entender la tarea, aqui tienes un ejemplo de una
factura que HA SIDO ANOTADA y su extraccion \gls{JSON} correcta.

[
#C1 ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L. #C1
Cif: #C2 B47770656 #C2
...
(Resto de la factura anotada)
...
]

### Extraccion \gls{JSON} Correcta para la factura de Ejemplo:
[
{
  "A1": "Ulises Vasquez Lagunal",
  "C1": "ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L.",
  ...
  (Resto del \gls{JSON})
  ...
}
]

### Tarea de Extraccion: Nueva Factura
Ahora, aplicaras lo aprendido a la siguiente factura...
```

Figura 8.2: Estructura conceptual del *prompt* para la estrategia *Few-Shot* con ejemplo anotado.

Few-Shot con Validación Cruzada. Para evaluar de una manera más robusta la capacidad de generalización del modelo, se diseñó una serie de *prompts few-shot* que utilizan tres ejemplos por *prompt*, siguiendo una lógica de validación cruzada (*cross-validation*). El objetivo era medir cómo el aprendizaje a partir de un conjunto diverso de ejemplos influye en el rendimiento sobre plantillas no vistas. Estos experimentos se dividieron en dos enfoques:

El primer enfoque se basó en una agrupación secuencial de plantillas. Se crearon dos *prompts*: uno con ejemplos de las plantillas 1, 2 y 3; y otro con ejemplos de las plantillas 4, 5 y 6. La evaluación se realizaba de forma cruzada, utilizando el primer *prompt* para extraer datos de las facturas de las plantillas 4, 5 y 6, y viceversa. Esto permite medir la capacidad del modelo para generalizar a partir de un conjunto de entrenamiento a un conjunto de prueba completamente distinto.

El segundo enfoque se centró en la similitud estructural de las facturas. Tras un análisis visual de los documentos, (teniendo en cuenta la similitud por naturaleza que tienen de

por sí estos tipos de documentos de negocio), se identificaron dos grupos en cuanto a similitud estructural de la factura. Como se puede observar en la Figura 8.4, el primer grupo lo conforman las plantillas 1, 2 y 4 (fila superior), mientras que el segundo grupo está formado por las plantillas 3, 5 y 6 (fila inferior). Se crearon dos *prompts*, cada uno con ejemplos de una única familia estructural, y se utilizaron para evaluar la extracción en las facturas del grupo contrario. El propósito de este diseño es medir de forma explícita la capacidad del modelo para generalizar a partir de estructuras de documento no vistas previamente.

La comparación entre los resultados de estos dos enfoques de validación cruzada permitirá aclarar una cuestión fundamental: si el modelo obtiene un rendimiento significativamente mejor cuando los ejemplos proporcionados en el *prompt* son estructuralmente similares a la factura objetivo, o si, por el contrario, la diferencia no es relevante, lo que indicaría una sólida capacidad de adaptación por parte del LLM a maquetaciones que no ha visto previamente en el contexto del aprendizaje *in-context learning*. En la Figura 8.3 se muestra la estructura de este *prompt*.

```
[Instrucciones Generales y Esquema de Salida (Idéntico al Zero-Shot)]

### Ejemplo de Referencia (Factura y Extraccion Correcta)
Para ayudarte a entender la tarea, aqui tienes 3 ejemplos de facturas
junto a su extraccion correcta en formato \gls{JSON}.

### Factura de Ejemplo 1:
[ Texto de la factura de la Plantilla 3 ]
### Extraccion \gls{JSON} Correcta para la factura de Ejemplo 1:
[ \gls{JSON} correspondiente a la factura de la Plantilla 3 ]

### Factura de Ejemplo 2:
[ Texto de la factura de la Plantilla 5 ]
### Extraccion \gls{JSON} Correcta para la factura de Ejemplo 2:
[ \gls{JSON} correspondiente a la factura de la Plantilla 5 ]

### Factura de Ejemplo 3:
[ Texto de la factura de la Plantilla 6 ]
### Extraccion \gls{JSON} Correcta para la factura de Ejemplo 3:
[ \gls{JSON} correspondiente a la factura de la Plantilla 6 ]

### Tarea de Extraccion: Nueva Factura.
...
```

Figura 8.3: Estructura conceptual del *prompt* para la estrategia *Few-Shot con Validación Cruzada*, mostrando la inclusión de tres ejemplos distintos.

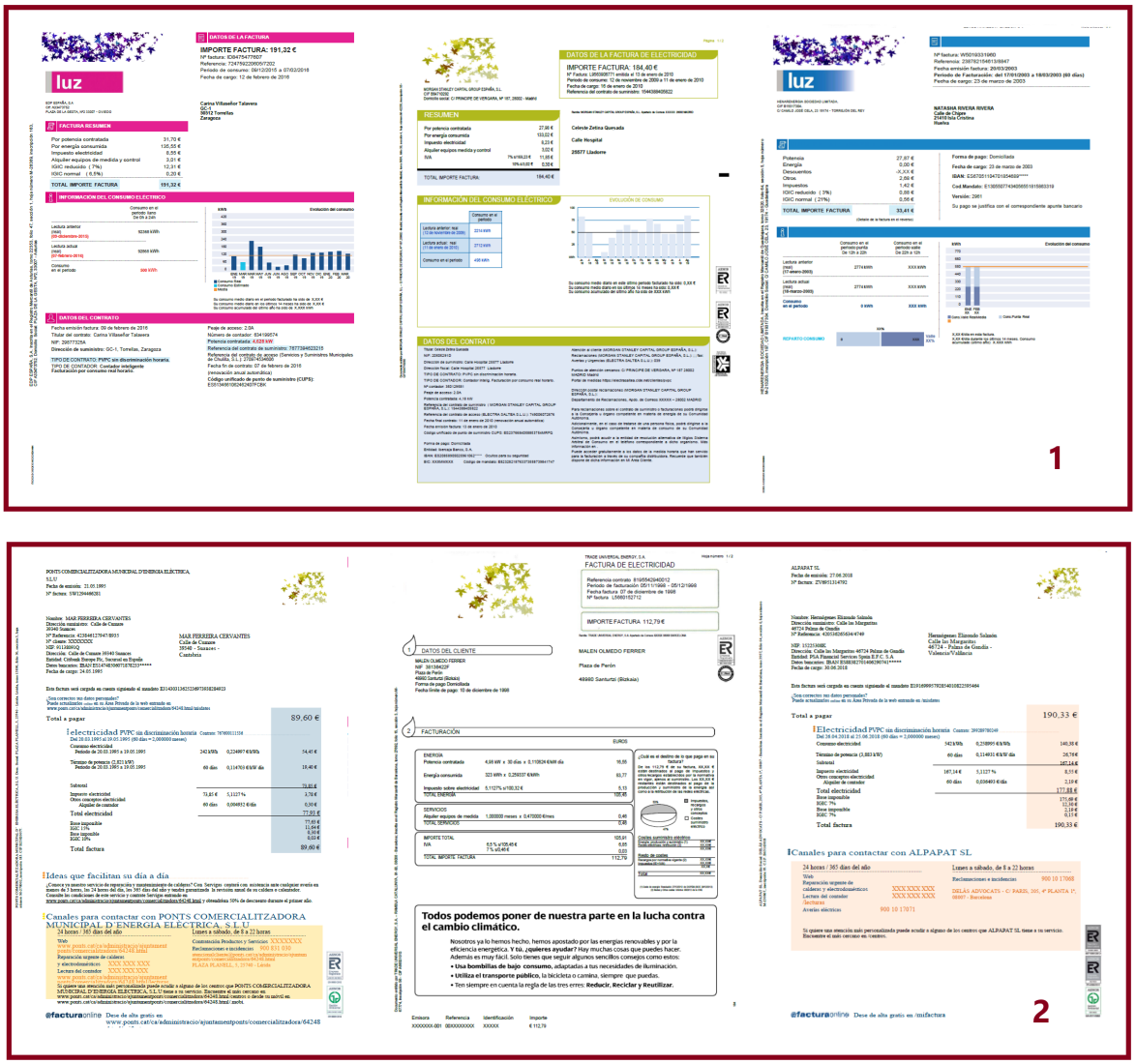


Figura 8.4: Agrupación de las plantillas de factura por similitud estructural para los experimentos de validación cruzada. La fila superior muestra las plantillas del Grupo 1 (1, 2 y 4) y la fila inferior las del Grupo 2 (3, 5 y 6).

8.2.4. Extracción Iterativa

Finalmente, se diseñó una estrategia diferente, fundamentada en el principio de **descomposición de tareas**, una idea central en la investigación sobre el razonamiento en los LLM. Este enfoque se inspira en la técnica de *Chain-of-Thought (CoT) Prompting*, que ha demostrado que los modelos de lenguaje a gran escala mejoran drásticamente su rendimiento en tareas de razonamiento complejo cuando se les incita a descomponer el problema en pasos intermedios [Wei et al., 2022].

Aunque la extracción de información no es una tarea de razonamiento aritmético o simbólico, comparte una característica fundamental: su complejidad. Pedirle a un modelo que extraiga 107 campos de un documento denso en una única pasada es una tarea de alta complejidad cognitiva que puede llevar a errores, omisiones o a que el modelo "alucine" valores. La hipótesis de la estrategia iterativa es que, al igual que en el CoT, descomponer el problema principal en una serie de sub-tareas más simples y enfocadas puede mejorar

significativamente la fiabilidad y precisión del resultado final [Wei et al., 2022].

En lugar de una única llamada a la API, esta técnica descompone el problema en múltiples llamadas, la mayoría de las cuales se enfocan en la extracción de una única etiqueta. Para ello, se creó un *prompt* iterativo que, para cada campo, proporcionaba un contexto altamente especializado:

- **Descripción de la Tarea:** Analiza la factura y extrae una única información.
- **Dato Específico a Extraer:** Descripción humana de la etiqueta (ej. "Nombre del cliente").
- **Instrucción Específica:** Indicaciones detalladas sobre cómo localizar ese dato concreto en el documento.
- **Formato Esperado:** Descripción del formato del valor a extraer.
- **Ejemplo de Extracción:** Un ejemplo concreto para esa única etiqueta.
- **Contexto de la Factura:** El texto completo de la factura.
- **Instrucción de Salida:** Petición de un JSON con una única clave y su valor.

La plantilla base, mostrada en la Figura 8.5, define la estructura de la petición que se envía en cada llamada. Contiene marcadores que se rellenan dinámicamente.

```
Analiza el siguiente texto de una factura y extrae la
informacion solicitada.

**Dato a extraer**: {{descripcion_humana}}
**Instruccion para el LLM**: {{instruccion_especifica_llm}}
**Formato esperado DEL DATO EXTRAIDO**: {{formato_esperado}}
**Ejemplo de extraccion DEL DATO**: {{ejemplo_extraccion}}

Contexto de la factura:
---
{texto_factura}
---

IMPORTANTE: Responde UNICAMENTE con un objeto \gls{JSON} que contenga
una sola clave llamada "{{etiqueta}}", y cuyo valor sea el dato
extraido para "{{descripcion_humana}}".
```

Figura 8.5: Estructura del *prompt* base o plantilla para la estrategia de extracción iterativa.

El contenido para rellenar los marcadores de la plantilla se almacena en un fichero de configuración JSON, del cual se muestra un fragmento en la Figura 8.6. Este fichero contiene un objeto para cada etiqueta, con una descripción humana, instrucciones específicas, el formato esperado y un ejemplo concreto.

```
[
  {
    "id_etiqueta": "A1",
    "descripcion_humana": "Nombre del cliente",
    "instruccion_especifica_llm": Localiza y extrae el
    nombre completo del titular del contrato o cliente.
    Suele estar en la sección de 'Datos del cliente',
    a menudo en la parte superior derecha de la factura y
    puede estar precedido por una etiqueta como 'Titular'
    o 'Cliente'.",
    "formato_esperado": "[Nombre] [Primer Apellido]
    [Segundo Apellido]",
    "ejemplo_extraccion": "De la línea que contiene
    'EDP ESPAÑA, S.A  Carina Villaseñor Talavera',
    extrae la string 'Carina Villaseñor Talavera'"
  },
  {
    "id_etiqueta": "A2",
    "descripcion_humana": "Numero de identificacion del cliente",
    "instruccion_especifica_llm": "Localiza y extrae el NIF...",
    "formato_esperado": "[DNI / NIF / NIE]",
    "ejemplo_extraccion": "Si en los datos... extrae '...'"
  },
  ...
]
```

Figura 8.6: Fragmento del fichero de configuración JSON con las instrucciones detalladas para cada etiqueta.

Al reducir la complejidad de cada petición, el modelo puede centrar todos sus recursos en una única tarea, minimizando la confusión y aumentando la precisión. No obstante, esta estrategia de "una etiqueta por llamada" cuenta con excepciones diseñadas para optimizar el proceso. Ciertas etiquetas que se encuentran muy próximas físicamente en el documento y que pertenecen a un mismo concepto semántico (como los diferentes componentes de una dirección o los datos del comercializador) se agrupan en una única petición. Este enfoque aprovecha el contexto local de la información y reduce el consumo de *tokens*.

Un caso de estudio particular fue el de los números de teléfono del comercializador (`CC_CD_CE`). Durante las pruebas iniciales, se observó que al solicitar estos números por separado, el modelo los confundía con facilidad. Al agruparlos en una única petición que solicita explícitamente el teléfono de "Atención al Cliente", el de "Averías" y el de 'Reclamaciones', se fuerza al modelo a diferenciar entre ellos, mejorando así la precisión.

Aunque este método mixto implica un mayor número de llamadas a la API y un consumo de *tokens* muy elevado en comparación con otras estrategias, busca maximizar la calidad del resultado final, siguiendo el principio de que la simplificación de la tarea a través de la descomposición conduce a un razonamiento más robusto y a mejores resultados

[Wei et al., 2022].

8.3. Subconjunto de Datos para la Experimentación

Para la ejecución de los experimentos, no se utilizó la totalidad de los 75,000 documentos disponibles en IDSEM [Sánchez et al., 2022], sino que se seleccionó un subconjunto representativo. Este subconjunto se extrajo exclusivamente del conjunto de datos de entrenamiento, ya que este proporciona los ficheros JSON con el *ground truth* y las facturas anotadas, ambos indispensables para la evaluación y el preprocesamiento.

Concretamente, se trabajó con las facturas correspondientes a las plantillas de la 1 a la 6. De cada una de estas seis plantillas, se seleccionaron los primeros 400 documentos. Esto conformó un corpus de experimentación de 2,400 facturas diferentes. La elección de las facturas anotadas del conjunto de entrenamiento se realizó para verificar de manera programática qué subconjunto de las 107 etiquetas posibles existía realmente en cada una de estas seis plantillas, una tarea necesaria para la correcta evaluación de los resultados.

8.4. Diseño y Ejecución de los Experimentos

Una vez definidos los parámetros de muestreo y las estrategias de *prompting*, en esta sección se detalla cómo se han combinado para construir el conjunto de experimentos del proyecto. El diseño experimental se dividió en dos grandes bloques: un análisis exhaustivo de los parámetros de generación sobre un único modelo, y un análisis comparativo de las diferentes estrategias de *prompting* con una configuración de parámetros fija.

8.4.1. Análisis de Parámetros de Generación (Gemini 1.5 Pro)

El primer bloque de experimentos se centró exclusivamente en el modelo Gemini 1.5 Pro con el objetivo de realizar un análisis profundo de cómo los parámetros de generación afectan a la calidad de la extracción. Para ello, se ejecutaron las 19 configuraciones de parámetros descritas (las 18 combinaciones de la búsqueda sistemática más la configuración determinista) sobre dos tipos de *prompt*:

- **Prompt Zero-Shot:** Se evaluó sobre el subconjunto completo de 2,400 facturas (400 por cada una de las 6 plantillas).
- ***Prompt Few-Shot* (usando un ejemplo anotado):** Para profundizar en el efecto de los parámetros en un contexto más guiado, se evaluó sobre un subconjunto de 800 facturas (400 de la plantilla 1 y 400 de la plantilla 2).

Este bloque de experimentos supuso un total de 60,800 llamadas a la API de Google.

8.4.2. Análisis Comparativo de Estrategias de Prompting

El segundo bloque de experimentos tuvo como objetivo comparar el rendimiento de todas las estrategias de *prompting* diseñadas. Para aislar el efecto del *prompt* y poder realizar una comparación justa, todas las ejecuciones de este bloque se realizaron con la configuración de parámetros determinista (temperatura, top p y top k a 0), tanto para Gemini 1.5 Pro como para Mistral-small. Para el modelo Gemini 1.5 Pro, se evaluaron

todas las estrategias de *prompting* que no habían sido completamente testeadas en el bloque anterior sobre el corpus de 2,400 facturas, y para Mistral-small, se evaluaron las 8 estrategias de *prompting* descritas sobre el corpus completo de 2,400 facturas.

Es importante destacar el caso de la estrategia de extracción iterativa, que requiere 83 llamadas a la API para procesar una única factura.

La Tabla 8.1 resume la configuración de los experimentos y el volumen total de llamadas a la API realizadas para cada modelo, reflejando el alcance de la evaluación llevada a cabo en este trabajo.

Tabla 8.1: Resumen de la configuración de los experimentos y el número de llamadas a la API por modelo.

Fase Experimental	**Configuración**	**Modelo**	**Nº Llamadas API**
Parámetros	*Prompt* Zero-Shot (19 combinaciones x 2400 facturas)	Gemini 1.5 Pro	45,600
	Prompt Few-Shot (Anotado) (19 combinaciones x 800 facturas)	Gemini 1.5 Pro	15,200
Prompts	*Prompts* No-Iterativos (5) (5 *prompts* x 2400 facturas)	Gemini 1.5 Pro	12,000
	Prompt Few-Shot (Anotado) (1 *prompt* x 1600 facturas restantes)	Gemini 1.5 Pro	1,600
	Prompt Iterativo (83 llamadas/factura x 2400 facturas)	Gemini 1.5 Pro	199,200
	Total Gemini 1.5 Pro		**273,600**
	Prompts No-Iterativos (7) (7 *prompts* x 2400 facturas)	Mistral-small	16,800
	Prompt Iterativo (83 llamadas/factura x 2400 facturas)	Mistral-small	199,200
	Total Mistral-small		**216,000**

Capítulo 9

Resultados y Análisis de los Experimentos

Tras haber detallado la metodología del proyecto y la configuración de los experimentos, este capítulo se centra en la presentación y el análisis de los resultados obtenidos. El objetivo es evaluar de manera cuantitativa y cualitativa el rendimiento de los modelos en la tarea de extracción de información, interpretando los hallazgos para extraer conclusiones significativas sobre las estrategias empleadas.

La exposición de los resultados se ha estructurado en tres apartados principales. En primer lugar, se presentará un análisis del impacto de los parámetros de generación, donde se examinará cómo la variación de la `temperatura`, `Top-P` y `Top-K` influyeron en la precisión de las extracciones realizadas con el modelo Gemini 1.5 Pro.

A continuación, se llevará a cabo un análisis comparativo de las estrategias de *prompting*. En esta sección, se comparará el rendimiento de los diferentes enfoques de ingeniería de instrucciones , desde el *zero-shot* hasta la extracción iterativa utilizando una configuración de parámetros determinista para aislar y medir el efecto de cada tipo de instrucción en la calidad de los resultados.

Finalmente, se concluirá con una discusión de los hallazgos principales, donde se sintetizarán los resultados de ambos bloques de experimentos para ofrecer una visión global, conectar los diferentes hallazgos y destacar las conclusiones más relevantes de esta investigación.

9.1. Impacto de los Parámetros de Generación

En este apartado se presentan los resultados del primer bloque de experimentos, cuyo objetivo era evaluar la sensibilidad del rendimiento del modelo Gemini 1.5 Pro frente a la variación de sus hiperparámetros de inferencia: `temperatura`, `Top-K` y `Top-P`. Para ello, se analizaron los resultados de las 19 configuraciones experimentales sobre el subconjunto de datos definido.

La observación más destacada, visible en la Figura 9.1, es la alta estabilidad del rendimiento del modelo para una plantilla de factura dada, independientemente de la configuración de los parámetros de muestreo. El análisis de la gráfica revela dos conclusiones fundamentales: primero, una baja variabilidad entre plantillas, ya que las curvas de precisión para cada plantilla son notablemente planas; y segundo, una jerarquía de rendimiento constante, pues el orden de rendimiento entre las plantillas se mantiene a lo largo de todos los experimentos, siento la plantilla 1 la que mejor precisión tiene a lo largo de todos los

experimentos, y la plantilla 5 la que peor rendimiento muestra. Estos resultados sugieren de forma contundente que las características inherentes a cada plantilla son un factor mucho más dominante para determinar la precisión que los hiperparámetros de inferencia.

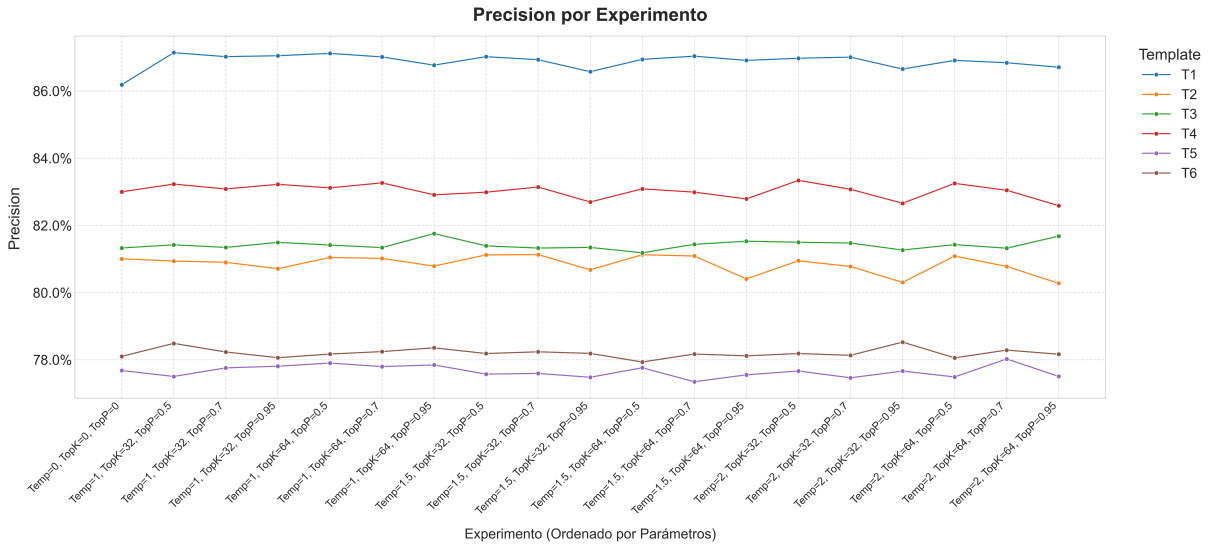


Figura 9.1: Gráfica de tendencias de la Precisión Media por plantilla a lo largo de los 19 experimentos de parámetros. Cada línea de color representa una de las seis plantillas (T1-T6).

La Tabla 9.1 muestra las métricas globales agregadas para cada uno de los 19 experimentos, lo que permite cuantificar la magnitud del impacto de los parámetros.

Tabla 9.1: Resultados de los experimentos de ajuste de parámetros, mostrando las métricas de *Precision*, *Recall* y *F1-Score*.

Temp.	**Top-k**	**Top-p**	**Precision**	**Recall**	**F1_Score**
0	0	0	80.87 %	75.31 %	77.99 %
1	32	0.5	81.40 %	75.86 %	78.54 %
1	32	0.7	81.33 %	76.00 %	78.57 %
1	32	0.95	81.34 %	75.59 %	78.36 %
1	64	0.5	81.40 %	75.89 %	78.55 %
1	64	0.7	81.40 %	75.84 %	78.52 %
1	64	0.95	81.34 %	75.63 %	78.38 %
1.5	32	0.5	81.32 %	75.94 %	78.54 %
1.5	32	0.7	81.36 %	75.96 %	78.57 %
1.5	32	0.95	81.11 %	75.41 %	78.16 %
1.5	64	0.5	81.31 %	75.97 %	78.55 %
1.5	64	0.7	81.29 %	75.87 %	78.49 %
1.5	64	0.95	81.16 %	75.43 %	78.19 %
2	32	0.5	81.40 %	75.90 %	78.56 %

Continúa en la página siguiente...

Tabla 9.1– Continuación

Temp.	Top-k	Top-p	Precision	Recall	F1_Score
2	32	0.7	81.27 %	75.81 %	78.45 %
2	32	0.95	81.09 %	75.21 %	78.04 %
2	64	0.5	81.34 %	75.80 %	78.47 %
2	64	0.7	81.33 %	75.71 %	78.42 %
2	64	0.95	81.08 %	75.35 %	78.11 %

El rango de variación entre el mejor y el peor rendimiento para cada métrica es bastante pequeño:

- **Precision:** El rango de variación es de un 0.53 %, entre 81.40 % y 80.87 %.
- **Recall:** El rango de variación es de un 0.79 %, entre 76.00 % y 75.21 %.
- **F1-Score:** El rango de variación es de un 0.58 %, entre 78.57 % y 77.99 %.

Para contextualizar estas cifras, se calculó el número medio de etiquetas presentes en las seis plantillas utilizadas, resultando en un promedio de aproximadamente 60 etiquetas por factura. Una variación del 1 % en la precisión equivaldría, de media, a una diferencia de solo 0.6 etiquetas. Por lo tanto, la variación máxima observada en la *Precision* (0.53 %) se traduce en una diferencia de apenas 0.32 etiquetas. De manera similar, la máxima variación en el *Recall* (0.79 %) equivale a 0.48 etiquetas, y en el *F1-Score* (0.58 %) a 0.35 etiquetas.

Este análisis cuantitativo confirma la conclusión del análisis visual: el impacto práctico del ajuste de los hiperparámetros de inferencia es muy poco significante, ya que la máxima diferencia de rendimiento entre la mejor y la peor configuración equivale, de media, a menos de media etiqueta correcta. A pesar de ello, se ha identificado una tendencia clara: tanto los extremos de determinismo absoluto (Temperatura 0) como de alta aleatoriedad (Temperatura 2) son subóptimos, mientras que una configuración de aleatoriedad moderada, ofrecida en el experimento con los parámetros `Temp=1.0`, `Top-k=32`, `Top-p=0.7` ofrece el mejor equilibrio.

9.2. Estrategias de Prompting

El segundo bloque de experimentos tuvo como objetivo comparar el rendimiento de las diferentes estrategias de *prompting* diseñadas. Para aislar el efecto de la instrucción y poder realizar una comparación justa, todas las ejecuciones de este bloque se realizaron con la configuración de parámetros determinista (temperatura 0), tanto para Gemini 1.5 Pro como para Mistral-small. El análisis se estructura presentando primero los resultados para cada modelo de forma individual, para concluir con una discusión comparativa de ambos.

Es importante notar que, para una evaluación equitativa, los resultados globales de los *prompts Few-shot_v2* y *Few-shot_v3* excluyen los datos de la Plantilla 1, ya que esta fue utilizada como ejemplo en dichas instrucciones, lo que podría inflar artificialmente el rendimiento. De manera similar, los resultados del *prompt* iterativo (*IP*) excluyen las Plantillas 1 y 2, pues sus datos sirvieron de base para la creación de los ejemplos de las etiquetas. Todo lo anterior mencionado aplica a los experimentos de ambos modelos.

Los resultados de la evaluación para el modelo Gemini 1.5 Pro se resumen en las métricas globales de la Tabla 9.2 y en la precision desglosada por plantilla de la Tabla 9.3.

Tabla 9.2: Métricas Globales para Gemini 1.5 Pro.

Prompting Strategy	**Precision**	**Recall**	**F1_Score**
Zero-shot	81.27 %	75.71 %	78.39 %
Few-shot_v2*	92.44 %	87.88 %	90.10 %
Few-shot_v3*	91.81 %	87.30 %	89.50 %
Few-Shot_Cross_valid_1	96.18 %	99.09 %	97.61 %
Few-Shot_Cross_valid_2	95.24 %	98.72 %	96.95 %
IP	94.45 %	96.51 %	95.47 %

Tabla 9.3: Precision por Template para Gemini 1.5 Pro.

Prompting Strategy	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
Zero-shot	86.91 %	80.92 %	81.44 %	83.08 %	77.67 %	78.20 %
Few-shot_v2	99.36 %	89.79 %	89.35 %	97.57 %	91.43 %	92.66 %
Few-shot_v3	98.42 %	91.25 %	90.36 %	96.54 %	85.61 %	93.24 %
Few-Shot_Cross_valid_1	97.78 %	92.64 %	98.14 %	97.67 %	92.72 %	97.48 %
Few-Shot_Cross_valid_2	95.92 %	92.66 %	97.54 %	96.73 %	92.03 %	96.21 %
IP	nan %	nan %	92.46 %	97.29 %	94.13 %	92.98 %

La tendencia general del rendimiento, ilustrada en la Figura 9.2, muestra un claro salto cualitativo desde la estrategia *Zero-shot* hacia las variantes *Few-shot*. El *prompt Zero-shot* actúa como una línea base, obteniendo el F1-Score más bajo (78.39 %), lo que confirma la dificultad de la tarea sin ejemplos contextuales.

Las estrategias que incorporan ejemplos (*in-context learning*) mejoran notablemente los resultados, superando todas el 90 % en *Precision*. Además, para las técnicas *Few-shot_v2* y *Few-shot_v3*, los resultados muestran un rendimiento muy parecido, resultando en

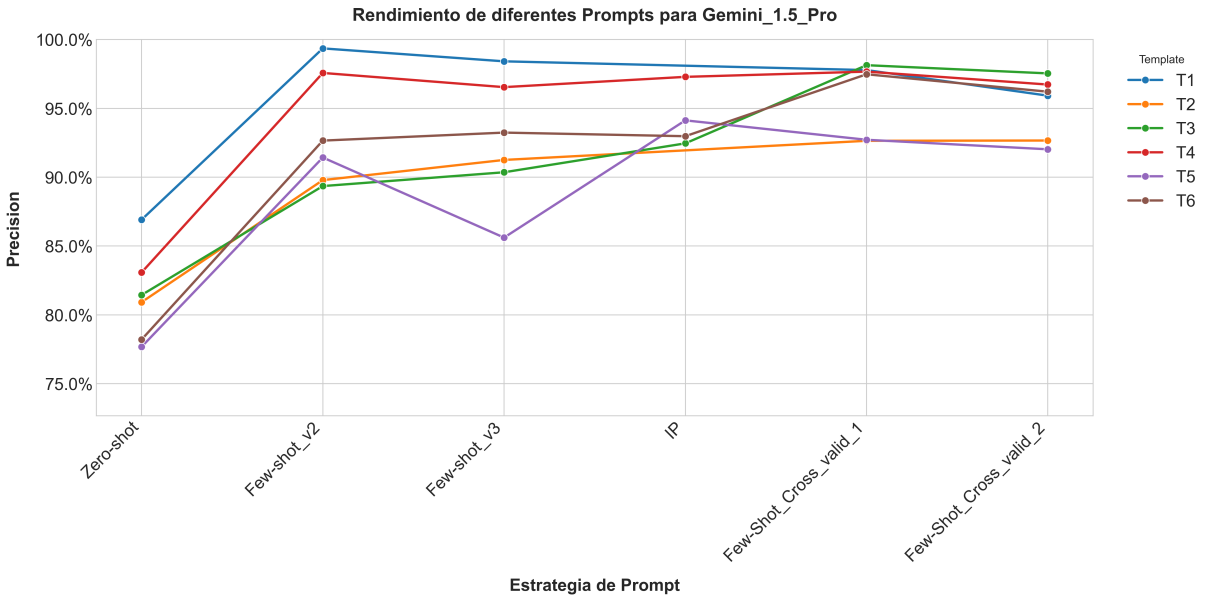


Figura 9.2: Gráfica de tendencias de la precision media por plantilla para Gemini 1.5 Pro según la estrategia de *prompt*.

menos de un 1 % de diferencia en las métricas globales *Precision*, *Recall* y *F1-Score*. Esto da a entender que la diferencia de extracción entre *prompts* de tipo few-shot con un ejemplo anotado y un ejemplo sin anotar es prácticamente nula y no afecta a la precisión de extracción para estos experimentos. Los mejores resultados se obtienen con los *prompts* de validación cruzada. Concretamente, el *prompt Few-Shot_Cross_valid_1* alcanza el rendimiento más alto, con un F1-Score de 97.61 %, impulsado por un Recall casi perfecto del 99.09 %. Esto sugiere que proporcionar al modelo múltiples ejemplos de diversas plantillas es una estrategia altamente efectiva para la generalización en plantillas que tienen un contenido similar, pero una estructura variable.

Los resultados para el modelo Mistral-small se resumen en las métricas globales de la Tabla 9.4 y la *Precision* desglosada por plantilla en la Tabla 9.5. De nuevo, los resultados más bajos son los que usan la instrucción *Zero-shot*, mostrándose una mejora de un 16 % en la *Precision* entre esta estrategia y la más alta, *Few-Shot_Cross_valid_1*. Además, en la Figura 9.3 se puede observar de manera más visual como la tendencia de la recta muestra un salto entre el primer *prompt* y el resto, que mantienen una precision similar entre ellos (+-3 %). Estos resultados siguen un patrón muy similar al de Gemini 1.5 Pro, lo que refleja la importancia de la calidad de los *prompts* proporcionados para estos modelos, independientemente de su arquitectura o cantidad de parámetros. Debemos tener en cuenta que para los ejemplos *Few-Shot_v2* y *Few-Shot_v3* no se deben tener en cuenta los resultados de la plantilla 1, ya que este tipo de facturas fueron las usadas en los ejemplos de estos *prompts*, por lo que el modelo ya conocía su estructura y extracción antes de completar la tarea. Además, podemos observar en la Tabla 9.5 cómo la *Precision* de la plantilla 4 mejora para el prompt IP hasta un 97.58 %. Esto ocurre también en los resultados de Gemini 1.5 Pro, donde obtiene un 97.29 % de *Precision* para la plantilla 4 usando el método IP, siendo el porcentaje de extracción más alto para todas las plantillas, sin contar a la plantilla 1, que se encontraba en el ejemplo proporcionado al modelo.

Tabla 9.4: Métricas Globales para Mistral-small.

Prompting Strategy	**Precision**	**Recall**	**F1Score**
Zero-shot	76.59 %	90.39 %	82.92 %
Few-shot_v2*	90.31 %	100.00 %	94.91 %
Few-shot_v3*	89.55 %	99.98 %	94.48 %
Few-Shot_Cross_valid_1	92.61 %	99.88 %	96.11 %
Few-Shot_Cross_valid_2	91.50 %	99.75 %	95.45 %
IP	92.55 %	98.14 %	95.26 %

Tabla 9.5: Precision por Template para Mistral-small.

Prompting Strategy	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
Zero-shot	82.98 %	80.41 %	70.94 %	81.12 %	68.70 %	70.10 %
Few-shot_v2	98.15 %	90.06 %	92.20 %	94.14 %	85.28 %	88.60 %
Few-shot_v3	97.49 %	87.86 %	92.27 %	92.02 %	84.50 %	90.58 %
Few-Shot_Cross_valid_1	93.44 %	89.66 %	93.28 %	93.67 %	90.53 %	94.89 %
Few-Shot_Cross_valid_2	90.64 %	92.67 %	93.64 %	88.24 %	92.87 %	92.07 %
IP	nan %	nan %	89.84 %	97.58 %	89.94 %	91.33 %

Nuevamente, la estrategia *Zero-shot* obtiene el F1-Score más bajo (82.92 %), consolidándose como la línea base menos efectiva. Un hallazgo notable es el extraordinario valor de Recall obtenido por los *prompts Few-shot*, alcanzando un 100 % en el caso de *Few-shot_v2*. Esto indica que Mistral-small, cuando se le proporcionan ejemplos, es extremadamente bueno encontrando los campos correctos, aunque a costa de una precision ligeramente menor que la de Gemini, lo que sugiere que puede generar más falsos positivos.

Al igual que con Gemini, los *prompts* de validación cruzada demuestran ser los más robustos, con *Few-Shot_Cross_valid_1* alcanzando el F1-Score más alto (96.11 %).

La Figura 9.4 ofrece una comparación directa de la Precision ajustada entre ambos modelos para cada estrategia de *prompting*. El análisis de esta comparativa, junto con los datos anteriores, revela los hallazgos principales de la experimentación.

Primero, se confirma que la jerarquía de las estrategias de *prompting* es consistente en ambos modelos: el rendimiento mejora progresivamente desde *Zero-shot*, pasando por las variantes de *Few-shot* con un ejemplo, hasta alcanzar su máximo con los *prompts* de validación cruzada, que incluyen múltiples ejemplos diversos.

Segundo, un hallazgo especialmente relevante es el rendimiento del *prompt Few-Shot_Cross_valid_2*. En este experimento, los modelos fueron entrenados con ejemplos de una familia estructural para luego ser evaluados en facturas de una familia estructuralmente diferente. A pesar de esta dificultad, ambos modelos obtuvieron resultados muy sólidos, con Gemini superando el 95 % de *precision* y Mistral el 91 %. Esto demuestra una notable capacidad de generalización, indicando que los modelos son capaces de aprender la tarea de extracción de forma abstracta, más allá de la maquetación específica de los ejemplos vistos en su instrucción.

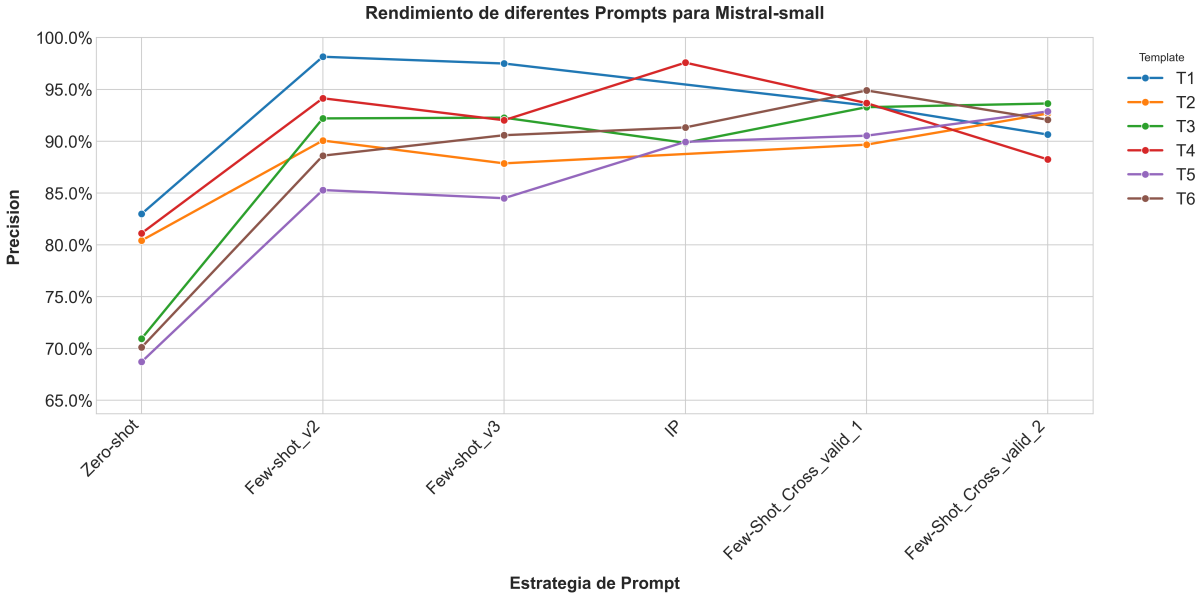


Figura 9.3: Gráfica de tendencias de la Precision media por plantilla para Mistral-small según la estrategia de *prompt*.

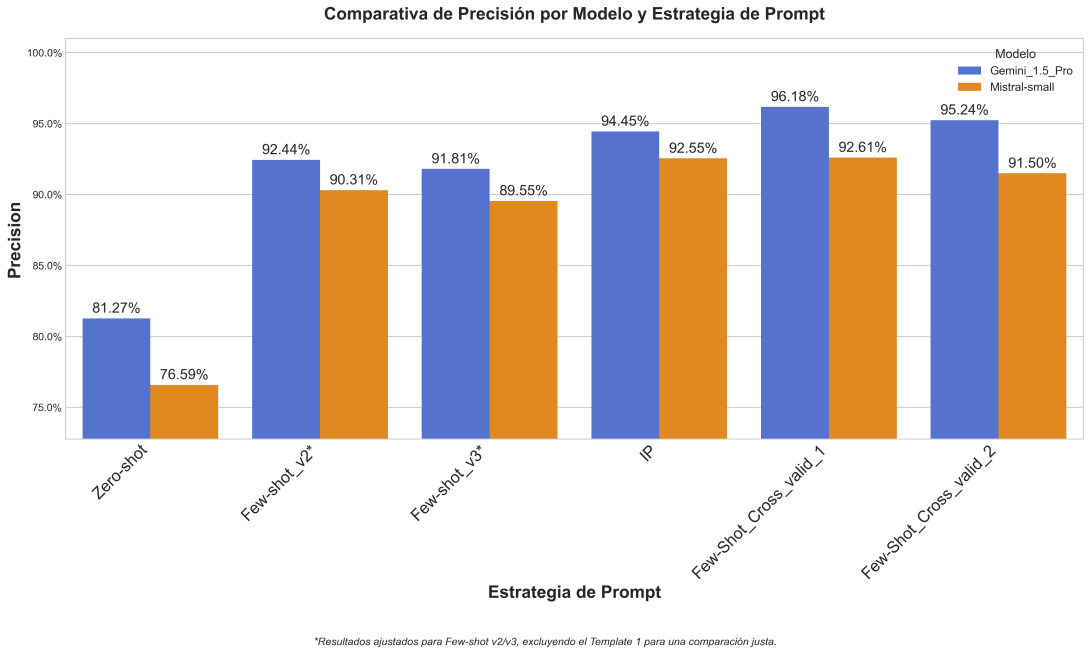


Figura 9.4: Comparativa de la Precision ajustada por Estrategia de Prompt para los modelos Gemini 1.5 Pro y Mistral-small.

Finalmente, la comparativa directa evidencia que, si bien Gemini 1.5 Pro obtiene una precision superior en la mayoría de las configuraciones, el rendimiento de Mistral-small es notablemente competitivo. A pesar de ser un modelo considerablemente más pequeño y ligero en cuanto a consumo de recursos, logra resultados muy cercanos a los de un modelo de vanguardia como Gemini, especialmente en las configuraciones *few-shot*. Esto posiciona a Mistral-small como una alternativa altamente eficiente y viable para tareas de extracción de información en entornos donde los recursos computacionales o económicos son una limitación.

9.3. Precision por Etiquetas

Para profundizar en la capacidad de generalización de los modelos, se realizó un análisis detallado del rendimiento por etiqueta utilizando los resultados del experimento *Few-Shot_Cross_valid_2*. Este experimento que evalúa la habilidad de los modelos para aplicar los patrones aprendidos a plantillas con una estructura diferente a la de los ejemplos proporcionados en el *prompt*. Las Tablas 9.6 y 9.7 desglosan la precision obtenida para cada etiqueta, agrupándolas en rangos de rendimiento.

Tabla 9.6: Clasificación de Etiquetas por Rango de Precision (Gemini 1.5 Pro). Prompt analizado: Few-Shot_Cross_valid_2.

Rango de Precision	**Etiquetas**	**Porcentaje**
[99-100 %]	A1, A3, A4, A5, A6, B1, B2, B4, B5, B6, C2, D1, E1, E3, E4, E5, E6, E7, E7p, E9, F1, F2, F3, F3p, F4, F4p, F4u, F4s, F5, F5p, F5s, F5u, F6, F7, F8i, G1, G2, G3, G3p, G4, G7, G8, G9, I1, I2, I3, I5, K3, K5, K6, K9, KA, KB, KC, KD, M4, N1, N2	61.9 %
[90-99 %]	B3, C1, C3, C4, C6, C8, CC, CE, D9, E2, E8, G6, GA, J1, J2, J3, M3d, N5, N7, N8	20.6 %
[80-90 %]	C5, CA, G5, I4, N3, N4, N6	7.2 %
[70-80 %]	C9, K2	2.1 %
[60-70 %]	C8, K4	2.1 %
[50-60 %]	DD, M3	2.1 %
[0-50 %]	CD, E8, K2d, K4d	4.1 %

Tabla 9.7: Clasificación de Etiquetas por Rango de Precision (Mistral-small). Prompt analizado: Few-Shot_Cross_valid_2.

Rango de Precision	**Etiquetas**	**Porcentaje**
[99-100 %]	A1, A3, A4, A5, A6, B1, B2, B3, B4, B5, B6, C2, D1, E1, E3, E5, E6, E7, E7p, E8, F1, F2, F3, F3p, F4, F4p, F4u, F4s, F5, F5p, F5s, F5u, F6, F7, F8i, G1, G2, G3, G3p, G4, I1, I2, I3, I5, K3, K5, KA, KB, KC, N1, N2	55.7 %

Tabla 9.7– Continuación

Rango de Precision	Etiquetas	Porcentaje
[90-99 %]	C1, C3, C4, C5, C6, C8, GA, J3, K2, K4, M4, N5	12.4 %
[80-90 %]	CA, E2, G6, G7, G8, G9, J1, J2, KD, M3d, N7	11.3 %
[70-80 %]	C8, CC, CE, D9, E4, K6, N3, N4, N6, N8	10.3 %
[60-70 %]	C9, DD, J4	3.1 %
[50-60 %]	E9, K9, M3	3.1 %
[0-50 %]	CD, G5, K2d, K4d	4.1 %

Para Gemini 1.5 Pro, el modelo demuestra una robustez excepcional en esta tarea de generalización. Como se detalla en la Tabla 9.6, el modelo clasifica un 82.5 % del total de las etiquetas con una precision superior al 90 %, con un notable 61.9 % de ellas alcanzando una precision casi perfecta (rango del 99-100 %). Esto indica que su capacidad de razonamiento abstracto le permite identificar correctamente la mayoría de los campos incluso en maquetaciones no familiares.

Mistral-small (Tabla 9.7) muestra una precision un poco menor, clasificando un **68.1 %** de las etiquetas con más del 90 % de *precision*. Aunque este porcentaje es inferior que el de Gemini, sigue siendo un resultado muy sólido que valida su capacidad de generalización. La principal diferencia reside en una mayor dispersión del rendimiento, con más etiquetas en los rangos intermedios.

La conclusión más relevante de este análisis es la identificación de un conjunto de etiquetas que resultan problemáticas para ambos modelos, lo que sugiere que su dificultad es inherente a la naturaleza de los datos y no a una debilidad específica de una de las arquitecturas.

Las etiquetas con una precision consistentemente baja (80 %) en ambos modelos incluyen C9, K2, C8, K4, DD, M3, CD, E8, K2d y K4d. Para entender la causa de este bajo rendimiento, es útil correlacionar estas etiquetas con su frecuencia de aparición en el conjunto de datos de entrenamiento, como se detalla en la Tabla A1 del artículo de [Sánchez and Cuervo Londoño, 2024]. Una selección de estas frecuencias se muestra en la Tabla 9.8.

Tabla 9.8: Número de ocurrencias de una selección de etiquetas del DataSet.
Fuente: Modificado de [Sánchez and Cuervo Londoño, 2024].

Código de Etiqueta	Nº de Ocurrencias
C1 (Nombre Comercializadora)	564,700
E2 (Tipo de Contrato)	210,400
J5 (Total Factura)	105,000

Tabla 9.8– Continuación

Código de Etiqueta	**Nº de Ocurrencias**
— *Etiquetas de rendimiento intermedio* —	
A6 (Provincia Cliente)	28,000
F2, KD, K6 (Referencia, Tasa Energía)	20,000
— *Etiquetas problemáticas* —	
E7s, F5u, F3p, **K2**, **K5**, N3, etc.	10,000
C6, F3s, G6, G7, G8, **K2d**, **K4d**, **K4**, **M3**, etc.	5,000
A2, D2, DC, E7l, F3l, F4l, F5l, etc.	0

El análisis de la Tabla 9.8 revela una clara correlación: las etiquetas con peor rendimiento son aquellas con una frecuencia de aparición muy baja en el dataset. Por ejemplo, etiquetas como K2d y K4d, que se encuentran entre las más difíciles para ambos modelos, solo aparecen 5,000 veces en los 30,000 documentos. Al tener una presencia tan escasa, es muy poco probable que estas etiquetas estuvieran incluidas en los tres ejemplos de los *prompts few-shot*. En consecuencia, el modelo no tuvo una demostración directa de cómo extraerlas, y su dificultad para generalizar a estos casos raros es la causa más probable de su bajo rendimiento.

En resumen, aunque ambos modelos demuestran una impresionante capacidad de generalización, su rendimiento en el aprendizaje en contexto sigue dependiendo de una representación suficiente de los conceptos en los ejemplos proporcionados. La superioridad de Gemini 1.5 Pro en este escenario radica en su capacidad para mantener una alta precision en un mayor número de estas etiquetas desafiantes, mientras que Mistral-small, aunque muy competente, muestra una mayor dificultad para discernir el valor correcto en estos casos más ambiguos y menos frecuentes.

9.4. Influencia de la Estructura en el Rendimiento

Los resultados agregados presentados en las secciones anteriores demuestran que la plantilla del documento es un factor determinante en el rendimiento de la extracción. Para ilustrar de manera concreta las causas subyacentes, esta sección realiza un análisis de caso de estudio comparativo que se centrará exclusivamente en el rendimiento obtenido con el *prompt zero-shot*. Al no proporcionar ningún ejemplo para ayudar al modelo a realizar la extracción, esta estrategia evalúa la capacidad de los modelos obtenida durante el entrenamiento para interpretar documentos basándose únicamente en su conocimiento pre-entrenado y en la calidad de los datos de entrada. Esto magnifica la importancia de la estructura y el contenido del documento, convirtiéndolo en el escenario ideal para desentrañar por qué ciertos diseños son inherentemente más sencillos o complejos de procesar.

Se realizará una comparación directa entre la factura de la plantilla con el rendimiento *zero-shot* más alto (T1) y la que obtuvo el más bajo (T5). Para Mistral, la precisión fue del 82.98 % para T1 frente a un 68.70 % para T5 (ver Tabla 9.4); para Gemini, la brecha fue similar, con un 86.91 % para T1 frente a un 77.67 % para T5 (ver Tabla 9.2).

La factura T1, que consistentemente obtuvo las métricas más altas, se caracteriza por una maquetación excepcionalmente clara y tabular, como se puede observar en la Figura 9.5. La disposición de los datos se organiza en bloques visualmente distintos y bien definidos ("DATOS DE LA FACTURA", "FACTURA RESUMEN"). De manera crucial, el diseño utiliza un formato explícito de clave-valor, donde las etiquetas descriptivas (ej. "Nº factura:") están situadas en inmediata proximidad a sus valores. Esta estructura lógica se preserva con alta fidelidad durante la conversión a formato Markdown, presentando al modelo un texto donde la relación semántica es inequívoca.

Desde la perspectiva del contenido, un análisis del mapeo de etiquetas revela que la factura T1 es notablemente densa en información, conteniendo 75 etiquetas distintas. Esta riqueza de campos, al estar bien estructurada, proporciona al modelo un contexto amplio y consistente. Aunque incluye algunas etiquetas de dificultad media como "C9" (sitio web) o "K2" (Tasa Energía), evita la mayoría de las etiquetas identificadas como problemáticas por su baja frecuencia o ambigüedad, como "M3", "CD" o "E8". La combinación de una estructura limpia y un conjunto de campos mayoritariamente estándar y bien definidos facilita que el modelo generalice correctamente a partir de su conocimiento pre-entrenado.

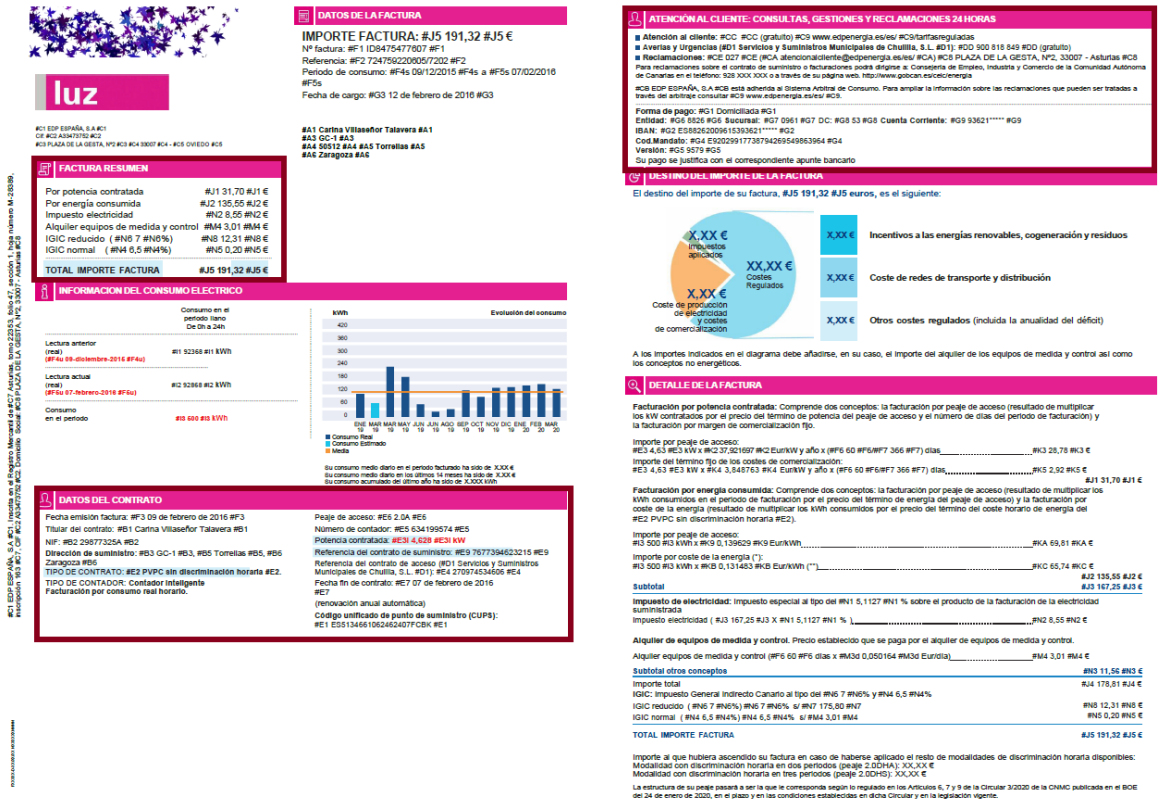


Figura 9.5: Disposición visual de las páginas 1 y 2 de la factura T1, que obtuvo el mayor rendimiento promedio en *zero-shot*. En rojo se rodean los bloques de información más claros y compactos.

Con una estructura relativamente similar, la factura T5 (ver Figura 9.6) presenta una confluencia de desafíos tanto estructurales como de contenido que explican su bajo rendimiento en el escenario *zero-shot*. Visualmente, su diseño es denso y se basa en un formato multicolumna que, al ser "aplanado" a Markdown, provoca una desvinculación espacial entre etiquetas y valores. Como se puede observar en la Figura 9.6 marcado con un cuadrado rojo, está la sección "FACTURACIÓN". Los conceptos (ej. "Potencia

contratada") y sus valores monetarios aparecen en columnas distintas debido a que se distribuyen a lo largo de columnas más espaciadas entre ellas, rompiendo la proximidad semántica en el texto procesado a formato Markdown.

El análisis del contenido de esta factura da pie a entender mejor su bajo porcentaje de extracción. La factura T5 es más dispersa, conteniendo solo **51 etiquetas**. Más importante aún, incluye varias de las etiquetas identificadas como problemáticas en el análisis de rendimiento general por su ambigüedad y baja frecuencia en el dataset, como "C9" (sitio web), "E8" (CNAE), "CD" (Teléfono de atención al cliente) y, de manera destacada, "M3" (Precio del alquiler del equipo), presente también en la sección de "FACTURACIÓN". La línea correspondiente a "M3", "Alquiler equipos de medida 1,000000 meses x 0,470000 €/mes 0,46", es un ejemplo claro de ambigüedad, ya que contiene múltiples valores numéricos a lo largo de una línea que es dividida cuando se pasa a Markdown, forzando al modelo a una inferencia compleja y propensa a errores sin la guía de un ejemplo.

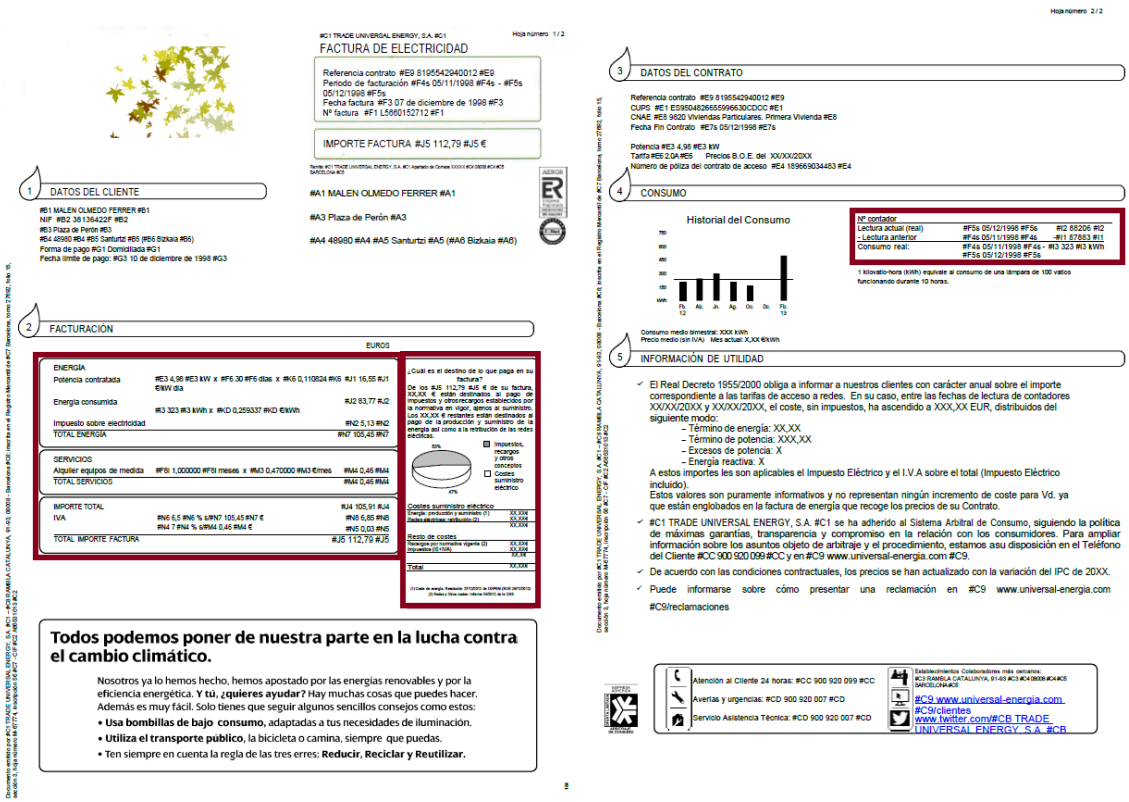


Figura 9.6: Disposición visual de la factura T5.

Este análisis de caso de estudio sobre el rendimiento *zero-shot* proporciona una evidencia empírica para una de las conclusiones principales de este trabajo. La drástica diferencia en la precisión de extracción entre T1 y T5 se explica por la interacción de dos factores:

1. **Degradación Estructural:** La maquetación de T1 sobrevive bien a la conversión a un formato textual lineal, preservando las relaciones semánticas. La estructura de T5, en cambio, se degrada significativamente en este proceso, generando un texto ruidoso y ambiguo para el modelo.

2. **Composición del Contenido:** La factura T5 no solo es estructuralmente más compleja, sino que también presenta un conjunto de etiquetas con mayor ambigüedad

inherente y menor frecuencia en el corpus general, como “M3” o “E8”. Sin ejemplos que lo guíen, el modelo tiene más dificultades para interpretar correctamente estos campos menos comunes.

En definitiva, la capacidad de un LLM para “razonar” sobre un documento en un escenario *zero-shot* está directamente condicionada por la calidad y la falta de ambigüedad de la representación textual que se le presenta. Una estructura visual clara y la presencia de campos estándar y bien definidos son los predictores más fiables del éxito en la extracción.

Capítulo 10

Conclusiones y Trabajo Futuro

Llegados al capítulo final de esta memoria, es el momento de cerrar el círculo y retornar al punto de partida, pero enriquecidos con la evidencia empírica obtenida a través del proceso de investigación realizado. Este Trabajo de Fin de Grado nació de una pregunta fundamental que resuena con fuerza en la industria y la academia contemporáneas: más allá del potencial teórico y la expectación mediática, ¿pueden los Modelos Extensos de Lenguaje (LLM) de propósito general cumplir la promesa de automatizar tareas de extracción de información en dominios tan estructurados, complejos y exigentes como el de los documentos empresariales? La misión, por tanto, no era simplemente aplicar una tecnología novedosa, sino llevar a cabo una validación sistemática y rigurosa de su viabilidad, robustez y límites en un caso de uso real y de alto valor: el procesamiento de facturas del sector eléctrico español.

Para responder a esta cuestión, se emprendió un viaje metodológico deliberado y meticuloso, anclado en el estándar industrial CRISP-DM. Este camino no estuvo exento de los desafíos inherentes a la investigación aplicada: desde la superación de obstáculos técnicos en la conversión de formatos de documento y la gestión de APIs comerciales, hasta el refinamiento iterativo del propio corpus de datos para garantizar una evaluación justa y precisa. El corazón de esta investigación ha sido el diseño y la ejecución de un marco experimental a gran escala, que ha implicado la realización de cientos de miles de llamadas a las APIs de los modelos. Se ha explorado sistemáticamente el impacto de los parámetros de inferencia y, de manera crucial, se ha elevado la **ingeniería de instrucciones** de una mera técnica a una variable experimental central, diseñando y contrastando un amplio espectro de estrategias de *prompting*, desde las más simples hasta las más sofisticadas. Las conclusiones que se presentan a continuación, por tanto, no son fruto de la intuición, sino el resultado destilado de esta forja de evidencia empírica, controlada y cuantitativa.

Al cruzar el umbral de los descubrimientos, los resultados obtenidos han demostrado ser profundamente esclarecedores. Han arrojado luz sobre las verdaderas palancas que gobiernan el éxito en la aplicación de LLM a esta tarea, desafiando algunas de las suposiciones más comunes y, a su vez, magnificando la importancia de otros factores a menudo subestimados. Los hallazgos que se discutirán revelan una clara jerarquía de prioridades, donde la obsesión por el microajuste de los parámetros de generación cede su protagonismo a la calidad fundamental del formato de los datos de entrada y, de manera predominante, al arte y la ciencia de establecer una comunicación precisa, contextualizada y estratégica con el modelo.

Este capítulo final se ha estructurado para guiar al lector a través de la síntesis de este viaje. En primer lugar, se presentarán de manera articulada las conclusiones fundamen-

tales extraídas de los datos, condensando los aprendizajes clave sobre el comportamiento de los modelos, la eficacia de las distintas estrategias y los factores críticos de éxito. A continuación, y reconociendo que toda investigación abre nuevas puertas, se trazará una hoja de ruta para el trabajo futuro. Partiendo de los hallazgos y las limitaciones identificadas en este estudio, se propondrán líneas de investigación y desarrollo que prometen no solo expandir los límites del conocimiento actual, sino también construir sobre los sólidos cimientos aquí establecidos para llevar el campo hacia nuevas y prometedoras fronteras.

Este Trabajo de Fin de Grado se propuso evaluar de manera sistemática la capacidad de los modelos extensos de lenguaje (LLM) para la tarea de extracción de información de documentos complejos y semi-estructurados, utilizando como caso de estudio un corpus de facturas del sector eléctrico español [Sánchez et al., 2022]. A través de una metodología experimental rigurosa, se han analizado los efectos de los parámetros de generación y de diversas estrategias de ingeniería de instrucciones. Los resultados obtenidos permiten extraer una serie de conclusiones fundamentales sobre la aplicación práctica de los modelos extensos de lenguaje a la extracción de datos de documentos.

La estructura del documento domina sobre los parámetros de generación. El hallazgo más contundente de la investigación es que la precisión en la extracción de datos está determinada de manera predominante por las características inherentes a la plantilla del documento (su maquetación, claridad y contenido) y no por el ajuste de los hiperparámetros de inferencia del modelo (temperatura, `Top-p`, `Top-k`). Si bien se ha identificado que tanto el determinismo absoluto (temperatura 0) como la aleatoriedad excesiva son subóptimos, la variabilidad en el rendimiento global entre las 19 configuraciones de parámetros es marginal. Este resultado sugiere que, para tareas de extracción de información, la optimización de los parámetros de muestreo ofrece un retorno de la inversión bajo en comparación con la adaptación al formato del documento y el desarrollo de instrucciones y estrategias de *prompting*.

La calidad del formato de entrada es un factor crítico para el rendimiento. Como se ha detallado en el desarrollo del proyecto, el preprocesamiento de los datos es una fase de igual o mayor importancia que el diseño de los *prompts*. La capacidad del LLM para procesar y analizar la información depende directamente de que los datos de entrada sean accesibles y estén bien estructurados. En este trabajo, tras una investigación de alternativas, se concluyó que el formato Markdown era la opción más adecuada. Esta decisión se fundamenta en la evidencia de que los LLM procesan este formato con mayor eficacia que el texto plano, y en estudios como el de [Braun et al., 2025], que demuestran que proporcionar una estructura textual explícita puede mejorar significativamente la precisión en tareas de extracción. En este trabajo, con el uso de Markdown, se logró hasta un 95 % de precisión en la extracción, lo que nos hace concluir que este formato permite al LLM procesar prácticamente todos los datos de la factura de manera precisa.

La calidad de la instrucción (prompt) es el factor clave para el éxito. Se ha demostrado una clara y consistente jerarquía en el rendimiento de las estrategias de prompting. El enfoque zero-shot, que depende únicamente del conocimiento pre-entrenado del modelo, establece una línea base de rendimiento pobre, validando su insuficiencia para tareas de extracción de documentos de esta complejidad. La introducción de ejemplos mediante el aprendizaje en contexto (few-shot) produce un salto cualitativo en la precisión, y esta mejora escala con la sofisticación de los ejemplos. Las estrategias que emplean múltiples ejemplos diversos a través de la validación cruzada y, especialmente, la estrategia de extracción iterativa que descompone la tarea, alcanzan los mejores resultados, superando el 95 % de F1_Score en los casos óptimos. Además, los resultados presentan un claro

aumento de la precisión cuando se explica de manera detallada y profunda el significado de cada etiqueta. El *prompt* iterativo, a pesar de no presentar ejemplos de extracción como lo hacen las técnicas de *few-shot*, muestra resultados de hasta un 94 % para Gemini y de un 92 % para Mistral. Esto sugiere la importancia de granularizar bien las solicitudes, además de crear instrucciones que contengan descripciones muy detalladas sobre la tarea que debe realizar.

La capacidad de generalización es alta, pero sensible a la representatividad de los ejemplos. Los experimentos de validación cruzada, en particular aquellos que evaluaron el rendimiento sobre plantillas estructuralmente diferentes a las de los ejemplos, demostraron una notable capacidad de generalización por parte de los modelos. Esto indica que los LLM, debido a sus altas capacidades de comprensión y procesamiento del lenguaje natural son capaces de aprender la tarea de extracción de forma abstracta, más allá de la maquetación específica para este conjunto de datos, guiándose por el significado semántico de la información presente en las facturas. Sin embargo, el análisis por etiqueta revela que el rendimiento decae significativamente en aquellos campos con una baja frecuencia de aparición en el conjunto de datos, y concretamente, de los ejemplos proporcionados en los *prompt*. Esto, subraya una limitación del aprendizaje en contexto: su efectividad con las instrucciones evaluadas en este trabajo es dependiente de que los ejemplos proporcionados sean representativos del espectro completo de datos a extraer.

El modelo Mistral-small se consolida como una alternativa viable y eficiente. Aunque Gemini 1.5 Pro obtiene una precisión superior en la mayoría de las configuraciones, el rendimiento de Mistral-small es notablemente competitivo, especialmente en las configuraciones few-shot. A pesar de ser un modelo considerablemente más pequeño, su eficacia lo posiciona como una alternativa altamente viable para tareas de extracción de información en entornos donde los recursos computacionales o los costes de las APIs son una limitación, ofreciendo un excelente equilibrio entre rendimiento y eficiencia.

En síntesis, este trabajo valida de forma empírica que la aplicación exitosa de los LLM a tareas de extracción de información de negocio depende menos de un ajuste fino de los parámetros de generación y mucho más de una ingeniería de instrucciones sofisticada y de una comprensión y calidad de los datos de entrada. La capacidad de descomponer problemas complejos y de proporcionar ejemplos contextuales de alta calidad emerge como la estrategia más efectiva para maximizar la precisión y la fiabilidad de estos sistemas.

A partir de los hallazgos y las limitaciones identificadas en este estudio, a continuación se proponen líneas de investigación y desarrollo para trabajos futuros en este sector.

Dado que el análisis exhaustivo de los parámetros se realizó exclusivamente sobre Gemini 1.5 Pro por limitaciones de recursos, una línea de trabajo futuro natural sería replicar la búsqueda sistemática de 19 experimentos en el modelo Mistral-small. Esto permitiría verificar si la conclusión sobre el impacto marginal de los parámetros se mantiene en arquitecturas de menor tamaño, o si es inversamente proporcional a la cantidad de parámetros del modelo.

Además, el dataset IDSEM, a pesar de su realismo, es sintético. Un paso crucial para validar la robustez de los hallazgos sería aplicar los *prompts* de mejor rendimiento (validación cruzada e iterativo) a un corpus de facturas reales, para evaluar cómo los modelos se comportan ante el ruido y la variabilidad no controlada del mundo real.

En cuanto a las estrategias de prompting, prompt iterativo demostró ser el más preciso pero también el más costoso. Una línea de investigación prometedora sería el desarrollo de una estrategia híbrida. Se podría diseñar un primer "prompt enrutador" que, en una única llamada, identifique qué campos están probablemente presentes en la factura. Posterior-

mente, se ejecutarían las llamadas iterativas y detalladas únicamente para ese subconjunto de campos, optimizando drásticamente el número de llamadas a la API sin sacrificar la precisión. Además, debido a la precisión prácticamente a la par de los resultados obtenidos por las instrucciones *few-shot*, sería interesante considerar una fusión de la técnica de instrucciones iterativas y la estrategia *few-shot* con varios ejemplos, creando una instrucción detallada, granularizada al nivel del prompting iterativo, y con ejemplos que ayuden al modelo a aprender aún más sobre la tarea a realizar.

La evaluación del rendimiento de los LLM en este trabajo se ha centrado en métricas cuantitativas. Un futuro análisis cualitativo de los errores permitiría diferenciar entre fallos de formato (ej. una coma extra), errores semánticos (ej. confundir la dirección del cliente con la del comercializador) o alucinaciones completas. Este análisis más profundo podría informar sobre el diseño de *prompts* aún más robustos.

Este trabajo se basó en la conversión de PDFs a formato Markdown, perdiendo inherentemente la información visual. Una línea de investigación de vanguardia sería replicar los experimentos utilizando las capacidades multimodales de modelos como Gemini 1.5 Pro, enviando directamente la imagen del PDF. Una comparación entre los resultados del enfoque basado en texto y el multimodal podría cuantificar de manera precisa el valor de la información visual en este tipo de tareas. Esta estrategia podría considerarse diviendo la tarea en dos fases, realizando la identificación de datos presentes en gráficas o imágenes en una, y la extracción de los datos presentes en el contenido textual de la factura en otra.

El análisis reveló un conjunto de etiquetas universalmente difíciles de extraer, a menudo por su baja frecuencia. Un enfoque de futuro podría ser realizar un ajuste fino específico (utilizando técnicas como LoRA) para estas etiquetas “raras”, entrenando al modelo para que se especialice en su identificación, y combinando este modelo ajustado con estrategias de prompting generales para el resto de los campos. Además, debido a una estructura relativamente común de las facturas de la luz en el sector español, el proceso de adaptación y entrenamiento de un modelo extenso de lenguaje a esta tarea es una línea de investigación que, basándonos en las investigaciones y conclusiones obtenidas en este trabajo, podría producir resultados muy interesantes.

Apéndice A

Resultados detallados

En este apéndice se detalla el contenido exacto de los *prompts* que constituyeron el instrumento principal de interacción con los Modelos Extensos de Lenguaje en este trabajo. Su inclusión busca garantizar la máxima transparencia metodológica y facilitar la replicabilidad de los hallazgos presentados en los capítulos centrales, permitiendo a otros investigadores comprender y, si lo desean, reconstruir el marco experimental aquí descrito.

A.1. Prompt del Sistema

Este *prompt* es la instrucción base que se proporcionó al modelo en todos y cada uno de los experimentos. Su objetivo es establecer un rol de experto, condicionando al LLM para que aborde la tarea desde una perspectiva de especialista en extracción de datos. La estrategia de role-play prompting ha demostrado ser efectiva para mejorar el razonamiento y la coherencia del modelo [Kong et al., 2023].

```
Eres un modelo avanzado de IA especializado en la extracción de datos
estructurados de facturas, recibos de la luz y documentos similares.
Tu tarea consiste en extraer con precisión la información basándote
en etiquetas predefinidas y respetando estrictamente las reglas de
formato e integridad. Debes responder en formato JSON según las
instrucciones del usuario.
```

A.2. Prompt Zero-Shot

Este *prompt* implementa la estrategia de aprendizaje *zero-shot*. Su objetivo era evaluar la capacidad del modelo para realizar la extracción basándose únicamente en su conocimiento obtenido en el entrenamiento, sin ningún ejemplo contextual. Su diseño se caracteriza por proporcionar una descripción de la tarea, un conjunto de reglas de formato y el esquema de salida JSON completo.

```
A continuación se muestra el texto extraído de una factura de
electricidad.
Su tarea consiste en extraer información específica de la factura y
```

```
organizarla en un formato JSON estructurado de acuerdo con las 107
categorías predefinidas.
Siga atentamente estas instrucciones:

### Category Matching:
Extraiga los datos de cada una de las 107 etiquetas de la sección
correspondiente de la factura. Si falta un valor, compruebe el texto
circundante dentro de su sección para obtener información relevante.
### Uso de Unidades:
No incluya las unidades de los datos extraídos. Si extrae €, kW o
cualquier otra unidad, extraiga sólo el valor sin la unidad.
### Formatos de Fecha:
Si una etiqueta es una fecha, las etiquetas terminadas en:
- "s": Formato de barra oblicua (dd/mm/aaaa).
- "p": Formato punto (dd.mm.aaaa).
- l'**: Formato largo (por ejemplo, "03 de Febrero de 2025").
- u'**: Formato guión (dd-mm-aaaa).


### Tratamiento de saltos de línea:
Procesa la información de manera que ningún valor extraído para las
etiquetas contenga el carácter de salto de línea (\n). Si un dato en
el documento original se extiende por varias líneas, concatena estas
líneas en una sola en el valor extraído, utilizando un espacio como
separador.
Etiqueta incorrecta sin tratar saltos de línea: "C1": "PONTS
COMERCIALITZADORA MUNICIPAL D'ENERGIA ELÈCTRICA,\nS.L.U"
Etiqueta correctamente formateada sin saltos de línea: "C1": "PONTS
COMERCIALITZADORA MUNICIPAL D'ENERGIA ELÈCTRICA, S.L.U"

### Formato de salida:
La información extraída debe salir en formato JSON válido. Se
estructurará de la siguiente manera, siendo el valor de cada etiqueta
la información que debe extraerse:

{{
    'A1': 'Nombre del cliente',
    'A2': 'Número de identificación del cliente',
    'A3': 'Dirección del cliente',
    'A4': 'Código postal',
    'A5': 'Ciudad del cliente',
    ...
}}

### Entrada:
El texto extraído de la factura de la luz se proporcionará a
continuacion.
```

```
### Salida:
Devuelve solamente el objeto JSON estructurado siguiento las
instrucciones declaradas previamente. La factura a extraer es la
siguiente:
```

A.3. Prompts Few-Shot

Este *prompt* implementa la estrategia de aprendizaje en contexto o *few-shot*, proporcionando un único ejemplo de la tarea resuelta. El objetivo era evaluar cómo mejoraba la precisión de la extracción con respecto al *prompt zero-shot*. La característica principal de este *prompt* es que el texto de la factura de ejemplo está "anotado", es decir, contiene marcadores conceptuales (como "#A1 Texto #A1") que señalan explícitamente qué fragmento de texto corresponde a cada etiqueta, con la hipótesis de que esto facilitaría al modelo el aprendizaje de la correspondencia.

```
{{Instrucciones Generales y Esquema de Salida (Idéntico al Zero-Shot)}}

### Ejemplo de Referencia (Factura Anotada y Extraccion Correcta)
Para ayudarte a entender la tarea, aqui tienes un ejemplo de
una factura que HA SIDO ANOTADA (...) y su extraccion
JSON correcta.

[
#C1 ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L. #C1
Cif: #C2 B47770656 #C2
...
(Resto de la factura anotada)
...
]

### Extraccion JSON Correcta para la factura de Ejemplo:
[
{
  "A1": "Ulises Vasquez Lagunal",
  "C1": "ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L.",
  ...
  (Resto del JSON)
  ...
}
]

### Tarea de Extraccion (Nueva Factura - NO ANOTADA)
Ahora, aplicarás lo aprendido a una nueva factura.
...
```

Esta es una variante del *prompt few-shot* anterior. Su objetivo era comparar directamente el efecto de las anotaciones. Para ello, la estructura es idéntica a la del *prompt*

anterior, pero el texto de la factura de ejemplo se presenta sin los marcadores conceptuales. Este diseño permite evaluar si el modelo es capaz de inferir la relación entre el texto y el JSON por sí mismo, y si esta capacidad de "auto-aprendizaje" es tan efectiva como la guía explícita de las anotaciones.

```
[Instrucciones Generales y Esquema de Salida (Idéntico al Zero-Shot)]

### Ejemplo de Referencia (Factura y Extraccion Correcta)
Para ayudarte a entender la tarea, aqui tienes un ejemplo de una
factura y su extraccion JSON correcta.

[
K


ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L.

Cif: B47770656

C/ MONASTERIO SANTO DOMINGO DE SILOS, 11 1oB 47015 - VALLADOLID
...
(Resto de la factura sin anotar)
...
]

### Extraccion JSON Correcta para la factura de Ejemplo:
[
{
  "A1": "Ulises Vasquez Lagunal",
  "C1": "ENERGIA Y COMERCIALIZACION EMPRESARIAL, S.L.",
  ...
  (Resto del JSON)
  ...
}
]

### Tarea de Extraccion: Nueva Factura
...
```

A.4. Prompts de Validación Cruzada

Estos *prompts* implementan una estrategia *few-shot* más avanzada, utilizando tres ejemplos distintos en cada instrucción. El objetivo era evaluar la capacidad de generalización del modelo al ser expuesto a una mayor diversidad de plantillas. Se diseñaron dos tipos de agrupación para los ejemplos (secuencial y por similitud estructural), y la evaluación se realizaba de forma cruzada, probando el rendimiento sobre plantillas que no estaban presentes en las demostraciones del *prompt*.

```
[Instrucciones Generales y Esquema de Salida (Idéntico al Zero-Shot)]

### Ejemplo de Referencia (Factura y Extraccion Correcta)
Para ayudarte a entender la tarea, aqui tienes 3 ejemplos de facturas
junto a su extraccion correcta en formato JSON.

### Factura de Ejemplo 1:
[ Texto de la factura de la Plantilla 3 ]
### Extraccion JSON Correcta para la factura de Ejemplo 1:
[ JSON correspondiente a la factura de la Plantilla 3 ]

### Factura de Ejemplo 2:
[ Texto de la factura de la Plantilla 5 ]
### Extraccion JSON Correcta para la factura de Ejemplo 2:
[ JSON correspondiente a la factura de la Plantilla 5 ]

### Factura de Ejemplo 3:
[ Texto de la factura de la Plantilla 6 ]
### Extraccion JSON Correcta para la factura de Ejemplo 3:
[ JSON correspondiente a la factura de la Plantilla 6 ]

### Tarea de Extraccion: Nueva Factura.
...
```

A.5. Estrategia de Extracción Iterativa

Este enfoque, inspirado en la descomposición de tareas como en la técnica de Cadena de Pensamiento (CoT), busca maximizar la precisión al reducir la complejidad de cada petición. En lugar de una única llamada a la API para todas las etiquetas, se realizan múltiples llamadas, cada una enfocada en un único campo o en un pequeño grupo de campos semánticamente relacionados. El sistema se basa en dos componentes: una plantilla de *prompt* base y un fichero de configuración JSON con las instrucciones detalladas para cada etiqueta. A continuación, se presenta primero la plantilla base sobre la que se completa toda la información a extraer. Después, se muestra un fragmento del archivo que contiene todas las etiquetas con la información para rellenar en el prompt base.

```
Analiza el siguiente texto de una factura y extrae la
informacion solicitada.

**Dato a extraer**: {{descripcion_humana}}
**Instruccion para el LLM**: {{instruccion_especifica_llm}}
**Formato esperado DEL DATO EXTRAIDO**: {{formato_esperado}}
**Ejemplo de extraccion DEL DATO**: {{ejemplo_extraccion}}

Contexto de la factura:
---
```

```
{texto_factura}
---

IMPORTANTE: Responde UNICAMENTE con un objeto JSON que contenga
una sola clave llamada "{{etiqueta}}", y cuyo valor sea el dato
extraido para "{{descripcion_humana}}".
```

```
[
  {
    "id_etiqueta": "A1",
    "descripcion_humana": "Nombre del cliente",
    "instruccion_especifica_llm": "Localiza y extrae el nombre...",
    "formato_esperado": "[Nombre] [Primer Apellido]...",
    "ejemplo_extraccion": "De la linea... extrae '...'"
  },
  {
    "id_etiqueta": "A2",
    "descripcion_humana": "Numero de identificacion del cliente",
    ...
  },
  ...
]
```

Apéndice B

Competencias Específicas Cubiertas

El presente trabajo constituye una síntesis aplicada de múltiples competencias adquiridas a lo largo del Grado en Ingeniería Informática. Más allá de la simple aplicación técnica, este proyecto pone de manifiesto la capacidad para integrar conocimientos teóricos avanzados con metodologías prácticas en el contexto de los sistemas inteligentes. En esta sección se destacan aquellas competencias específicas que han sido fundamentales tanto para el planteamiento como para el desarrollo de la solución propuesta.

Cada una de ellas se ha visto reflejada en las diferentes fases del proyecto: desde el análisis de modelos existentes y la exploración del estado del arte hasta el diseño, implementación y evaluación de los sistemas funcionales desarrollados. Esta aplicación transversal de competencias demuestra no solo la madurez técnica alcanzada, sino también la capacidad para abordar desafíos abiertos en ciencia de datos con una perspectiva crítica, rigurosa y orientada a la innovación.

A continuación, se exponen las competencias específicas de la titulación que han sido cubiertas, junto a su justificación en el marco de este proyecto:

CI6: Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.

Esta competencia se ha materializado en el diseño de las distintas estrategias de *prompting*. Se han desarrollado y evaluado procedimientos algorítmicos, como la estrategia de extracción iterativa, para resolver el problema de la extracción de información. Además, se ha analizado la idoneidad de cada enfoque (zero-shot, few-shot, validación cruzada) en función de la precisión y la eficiencia, lo que constituye un análisis de la complejidad y adecuación de las soluciones propuestas.

CI15: Conocimiento y aplicación de los principios fundamentales y técnicas básicas de los sistemas inteligentes y su aplicación práctica.

El núcleo del trabajo reside en la aplicación práctica de sistemas inteligentes. Se han utilizado los principios de los Modelos Extensos de Lenguaje (LLM) para una tarea de Procesamiento del Lenguaje Natural (PLN). La aplicación de técnicas como el aprendizaje en contexto (*few-shot learning*), la ingeniería de instrucciones y el ajuste de hiperparámetros (temperatura, `Top-p`, `Top-k`) para modular el comportamiento del modelo son una demostración directa de esta competencia.

Apéndice C

Objetivos de Desarrollo Sostenible

Aunque el enfoque principal de este trabajo es técnico, centrado en la aplicación de la inteligencia artificial para la extracción de información, las implicaciones de la solución desarrollada se vinculan directamente con varios Objetivos de Desarrollo Sostenible (ODS) promovidos por las Naciones Unidas. Estas relaciones no solo derivan del potencial de la tecnología para optimizar procesos y aumentar la eficiencia, sino también del enfoque metodológico que favorece soluciones accesibles. A continuación, se describen los ODS relevantes y se justifica el grado de contribución de este proyecto a cada uno de ellos.

- **ODS 9: Industria, Innovación e Infraestructura** (Grado de relación: 3 – Alto) El proyecto contribuye directamente a este objetivo al aplicar tecnologías de vanguardia como los Modelos Extensos de Lenguaje (LLM) para modernizar un proceso industrial clave: la gestión y extracción de información documental. A diferencia de los sistemas de aprendizaje automático tradicionales que requieren un costoso reentrenamiento para cada tipo de documento, este enfoque basado en la ingeniería de instrucciones sobre modelos pre-entrenados promueve una innovación más ágil y accesible. De este modo, se facilita la creación de infraestructuras de datos inteligentes, flexibles y escalables que pueden ser adaptadas a diversos sectores industriales para optimizar sus flujos de trabajo.

- **ODS 8: Trabajo Decente y Crecimiento Económico** (Grado de relación: 2 – Medio) La automatización de la extracción de datos de documentos como las facturas aborda directamente la optimización de procesos empresariales. Al delegar esta tarea repetitiva y propensa a errores a un sistema inteligente, se libera capital humano que puede ser reasignado a funciones de mayor valor añadido, como el análisis de datos, la toma de decisiones estratégicas o la gestión de relaciones con clientes. Este cambio no solo incrementa la productividad y eficiencia operativa, sino que también enriquece la calidad del trabajo, favoreciendo un crecimiento económico sostenible basado en la valorización del talento humano.

- **ODS 16: Paz, Justicia e Instituciones Sólidas** (Grado de relación: 2 – Bajo) De forma indirecta, el proyecto contribuye a este objetivo al proponer un sistema que refuerza la integridad y fiabilidad de la información documental. La alta precisión alcanzada en la extracción de datos minimiza los errores humanos en la transcripción de documentos clave, un factor fundamental para la solidez de los procesos institucionales y empresariales. Un sistema automatizado y auditable para procesar facturas, y por extensión, otros documentos oficiales, mejora la trazabili-

ODS	**Nivel de relación**			
	0	**1**	**2**	**3**
1. Fin de la Pobreza	**X**			
2. Hambre cero	**X**			
3. Salud y Bienestar	**X**			
4. Educación de calidad	**X**			
5. Igualdad de género	**X**			
6. Agua limpia y saneamiento	**X**			
7. Energía asequible y no contaminante	**X**			
8. Trabajo decente y crecimiento económico			**X**	
9. Industria, Innovación e Infraestructuras				**X**
10. Reducción de las desigualdades	**X**			
11. Ciudades y comunidades sostenibles	**X**			
12. Producción y consumo sostenibles	**X**			
13. Acción por el clima	**X**			
14. Vida submarina	**X**			
15. Vida de ecosistemas terrestres	**X**			
16. Paz, justicia e instituciones sólidas		**X**		
17. Alianzas para lograr objetivos	**X**			

Tabla C.1: Tabla con la valoración del grado de relación del TFG con los ODS, según una escala de 0 a 3 (no procede, bajo, medio, alto).

dad y la transparencia de la información. Esto es un pilar para la construcción de instituciones más eficientes, responsables y justas.

Acrónimos

API Application Programming Interface. v, vii, 3, 11, 16, 28, 29, 31, 33–36, 57–59, 61, 63, 64, 76–79, 95, 97, 98

BERT Bidirectional Encoder Representations from Transformers. v, 48, 50, 51, 56

BPE Byte-Pair Encoding. 48

CNMC Comisión Nacional de los Mercados y la Competencia. 23, 39, 45

CRF Conditional Random Field. 2

CRISP-DM CRoss-Industry Standard Process for Data Mining. 6, 21, 22, 24, 26, 30, 95

DAG Directed Acyclic Graphs. 37

DIT Document Image Transformer. 14, 16

DLA Disposición de los Elementos dentro de un Documento. 14, 15

DUE Document Understanding Evaluation. 16, 18

ERP Enterprise Resource Planning. 1

FFN Feed Forward Net. 54, 55

GPT Generative Pre-trained Transformer. v, 48, 50–53, 70, 71

GPU Graphics Processing Unit. 3, 10, 12, 31, 32, 56, 62

GQA Grouped-Query Attention. 50, 51, 56

HITL Human-in-the-Loop. 17

IDP Intelligent Document Processing. 13–18

IDSEM Invoices database of the Spanish electricity market. v, 5, 6, 22, 23, 27, 39–46, 57, 58, 64, 78, 97

JSON JavaScript Object Notation. v, vi, 15, 23, 27, 36, 39, 40, 43, 70, 72, 76–78

LLM Large Language Models. ix, x, 2–5, 9–13, 15, 17–19, 24, 26–29, 33, 34, 36, 37, 43, 47, 50, 55, 58, 59, 61, 63–65, 67, 69–71, 74, 75, 93, 95–98, 105, 107

LoRA Low-Rank Adaptation. v, 11, 12, 98

MLM Masked Language Model. 50

MLOps Machine Learning Operations. 36, 37

MoE Mezcla de Expertos. v, 10, 11, 18, 47, 50–56

NLU Natural Language Understanding. 50

OCR Optical Character Recognition. 1, 2, 5, 14, 15, 59

ODS Objetivos de Desarrollo Sostenible. vii, 107, 108

PaaS Platform as Service. 16

PDF Portable Document Format. 5, 14, 23, 27, 29, 34, 37, 40, 42, 43, 46, 57–60, 98

PEFT Parameter-efficient fine-tuning. 10, 11

PLN Procesamiento de Lenguaje Natural. 3, 14, 15, 47, 48, 105

PYMES Pequeñas y Medianas Empresas. 3, 4

QLoRA Quantized Low-Rank Adaptation. 11, 12, 18

RAG Retrieval-augmented generation. 13, 19

RNN Recurrent Neural Network. 10

SaaS Software as Service. 16

SSM State Space Model. 10

SVM Support Vector Machine. 2, 15

TFG Trabajo de Fin de Grado. vii, ix, 18, 19, 28, 36, 37, 108

TTMM Test-Time Model Merging. 11

VRDU Visually-rich Document Understanding. 16, 18

Bibliografía

[ABBYY, 2025] ABBYY (2025). The intelligent automation company. `https://www.abbyy.com/`.

[Abolghasemi et al., 2023] Abolghasemi, M. et al. (2023). Transformer-based models for long-form document matching: Challenges and empirical analysis. *arXiv preprint arXiv:2302.03765*.

[Ackley et al., 1985] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169.

[Adevinta, 2025] Adevinta (2025). Deep dive in paddleocr inference. `https://adevinta.com/techblog/deep-dive-in-paddleocr-inference/`.

[Ali and Fromm, 2024] Ali, M. and Fromm, M. (2024). Tokenizer choice for LLM training: Negligible or crucial? In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924, Mexico City, Mexico. Association for Computational Linguistics.

[Anaconda, 2022] Anaconda (2022). Anaconda study: State of data science 2021.

[Ba et al., 2016] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization.

[Barchard and Pace, 2011] Barchard, K. and Pace, L. (2011). Preventing human error: The impact of data entry methods on data accuracy and statistical results. *Computers in Human Behavior*, 27:1834–1839.

[Boiangiu et al., 2021] Boiangiu, C.-A. et al. (2021). Due: End-to-end document understanding benchmark. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

[Bostrom and Durrett, 2020] Bostrom, K. and Durrett, G. (2020). Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

[Braun et al., 2025] Braun, C., Lilienbeck, A., and Mentjukov, D. (2025). The hidden structure – improving legal document understanding through explicit text formatting.

[Brown and Mann, 2020] Brown, T. B. and Mann, B. (2020). Language models are few-shot learners. *CoRR*, abs/2005.14165.

[Caliskan et al., 2024] Caliskan, A. et al. (2024). Explicitly unbiased large language models still form biased associations. *Proceedings of the National Academy of Sciences (PNAS)*, 122.

[Chase, Harrison, 2025] Chase, Harrison (2025). Langchain: Building applications with llms through composability. https://github.com/langchain-ai/langchain.

[Chen et al., 2025] Chen, X. et al. (2025). SELF-MOE: Towards compositional large language models with self-specialized experts. *OpenReview*.

[Cloud, 2025a] Cloud, G. (2025a). Multimodal ai. https://cloud.google.com/use-cases/multimodal-ai.

[Cloud, 2025b] Cloud, G. (2025b). Nivel gratuito de google cloud. https://cloud.google.com/free?hl=es.

[Cloud, 2025c] Cloud, G. (2025c). Prompt engineering. https://www.kaggle.com/whitepaper-prompt-engineering.

[CloudThat, 2025] CloudThat (2025). Ai-based text extraction services. https://www.cloudthat.com/resources/blog/comparison-of-ai-based-text-extraction-services.

[Cobbe et al., 2021] Cobbe, K. et al. (2021). Gsm8k dataset. Papers With Code.

[Code, 2025] Code, P. W. (2025). Document layout analysis. https://paperswithcode.com/task/document-layout-analysis.

[Data, 2025] Data, L. Y. (2025). Ocr data extraction: Definition, features, and methods. https://labelyourdata.com/articles/ocr-data-extraction-methods.

[Dettmers et al., 2023a] Dettmers, T. et al. (2023a). Qlora: Efficient finetuning of quantized llms. OpenReview.

[Dettmers et al., 2023b] Dettmers, T. et al. (2023b). Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

[Devoteam, 2025] Devoteam (2025). Human-in-the-loop: What, how and why. https://www.devoteam.com/expert-view/human-in-the-loop-what-how-and-why/.

[Docker, Inc., 2025] Docker, Inc. (2025). Docker: Build, share, and run any app, anywhere. https://www.docker.com/.

[DocuWare, 2025] DocuWare (2025). Ocr vs. idp: Choosing the right solution. https://start.docuware.com/blog/document-management/idp-vs-ocr.

[Du et al., 2021] Du, N., Huang, Y., Dai, A., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A., Firat, O., Zoph, B., Fedus, L., Bosma, M., Zhou, Z., Wang, T., Wang, Y., Webster, K., Pellat, M., Robinson, K., and Cui, C. (2021). Glam: Efficient scaling of language models with mixture-of-experts.

[Elementl, Inc., 2025] Elementl, Inc. (2025). Dagster: The data orchestrator. https://dagster.io/.

[Face, 2025] Face, H. (2025). Accelerating document ai. `https://huggingface.co/blog/document-ai`.

[Fan et al., 2018] Fan, A., Lewis, M., and Dauphin, Y. (2018). Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

[Fan et al., 2024] Fan, Y. et al. (2024). Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

[Fedus et al., 2021] Fedus, W., Zoph, B., and Shazeer, N. (2021). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.

[Foundation, 2025a] Foundation, P. S. (2025a). Documentación del módulo argparse. `https://docs.python.org/3/library/argparse.html`.

[Foundation, 2025b] Foundation, P. S. (2025b). Documentación del módulo os. `https://docs.python.org/3/library/os.html`.

[Gali and Agarwal, 2025] Gali, V. K. and Agarwal, R. (2025). Zero-shot learning and few-shot learning with generative ai: Bridging the data gap for real-world applications. *Integrated Journal for Research in Arts and Humanities*, 5:193–200.

[Gao et al., 2025] Gao, L. et al. (2025). Grag: Graph retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*.

[Gao et al., 2023] Gao, Y., Xiong, Y., Gao, X., Zeng, A., Lai, H., Li, Y., Liu, Y., Wang, A. Z., Knop, J., Tan, P., and et al. (2023). Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997.

[GeeksforGeeks, 2025] GeeksforGeeks (2025). Information extraction in nlp. `https://www.geeksforgeeks.org/nlp/information-extraction-in-nlp/`.

[Google, 2025] Google (2025). Documentación de la api de python para google generative ai. `https://ai.google.dev/api/python/google/generativeai`.

[Google Cloud, 2025a] Google Cloud (2025a). Ajusta los valores de los parámetros del modelo. `https://cloud.google.com/vertex-ai/generative-ai/docs/learn/prompts/adjust-parameter-values?hl=es-419`.

[Google Cloud, 2025b] Google Cloud (2025b). Descripción general del modelo gemini 1.5. `https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/1-5-pro?hl=es-419`.

[Graves, 2013] Graves, A. (2013). Generating sequences with recurrent neural networks. *ArXiv*, abs/1308.0850.

[Gu and Dao, 2023] Gu, A. and Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752.

[Guide, 2024a] Guide, P. (2024a). Técnicas de few-shot prompting. `[https://www.promptingguide.ai/es/techniques/fewshot](https://www.promptingguide.ai/es/techniques/fewshot)`.

[Guide, 2024b] Guide, P. (2024b). Técnicas de zero-shot prompting. `https://www.promptingguide.ai/es/techniques/zeroshot`.

[Han et al., 2024] Han, C. et al. (2024). Cobra: Extending mamba to multi-modal large language model for efficient inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[Ho-Thanh-Tam and Tamine-Lechani, 2022] Ho-Thanh-Tam, L. and Tamine-Lechani, L. (2022). Challenges and advances in information extraction from scientific literature: A review. *ACM Computing Surveys*, 55(2).

[Holtzman et al., 2019] Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2019). The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

[Hong et al., 2024] Hong, D. et al. (2024). Layoutllm: Large language model instruction tuning for visually rich document understanding. *arXiv preprint arXiv:2403.14252*.

[Hu et al., 2021] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

[Hu et al., 2025] Hu, J. et al. (2025). Parameter-efficient fine-tuning of large language models via deconvolution in subspace. In *Proceedings of the 2025 Conference on Computational Linguistics*.

[IBM, 2025] IBM (2025). What is optical character recognition (ocr)? `https://www.ibm.com/think/topics/optical-character-recognition`.

[Ilharco et al., 2025] Ilharco, G. et al. (2025). Local mixtures of experts: Essentially free test-time training via model merging. *arXiv preprint arXiv:2505.14136*.

[Infrrd, 2025] Infrrd (2025). Human-in-the-loop intelligent document processing. `https://www.infrrd.ai/blog/does-ai-need-a-human-in-the-loop`.

[IPython, 2024] IPython (2024). Ipython.display — ipython 8.25.0 documentation. `https://ipython.readthedocs.io/en/stable/api/generated/IPython.display.html`.

[Jacobs et al., 1991] Jacobs, R., Jordan, M., Nowlan, S., and Hinton, G. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3:79–87.

[Ji et al., 2025] Ji, Z. et al. (2025). Mitigating llm hallucination with smoothed knowledge distillation. *arXiv preprint arXiv:2502.11306*.

[Jiang et al., 2023] Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de la Rosa, D., Figain, F., Bressand, F., Malartic, G., Lample, G., Introducing, L., a, M., an, L., and, O., and and, S. (2023). Mistral 7b. *arXiv preprint arXiv:2310.06825*.

[Koch, 2019] Koch, B. (2019). The e-invoicing journey 2019-2025. Technical report, Billentis.

[Kong et al., 2023] Kong, A., Zhao, S., Chen, H., Li, Q., Qin, Y., Sun, R., Zhou, X., Wang, E., and Dong, X. (2023). Better zero-shot reasoning with role-play prompting. *arXiv preprint arXiv:2308.07702*.

[Kudo and Richardson, 2018] Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

[Lepikhin et al., 2020] Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. (2020). Gshard: Scaling giant models with conditional computation and automatic sharding.

[Li et al., 2025] Li, Y. et al. (2025). Decoder-only multimodal state space model via quadratic to linear distillation. *arXiv preprint arXiv:2502.13145*.

[Li et al., 2024] Li, Z. et al. (2024). Unveiling the deficiencies of pre-trained text-and-layout models in real-world visually-rich document information extraction. *arXiv preprint arXiv:2402.02379*.

[Lialin et al., 2023] Lialin, K. et al. (2023). Transformer models: an introduction and catalog. *arXiv preprint arXiv:2302.07730*.

[Lialina et al., 2025] Lialina, V. et al. (2025). Revisiting fine-tuning: A survey of parameter-efficient techniques. `https://www.preprints.org/manuscript/202504.0743/v1`.

[Lin et al., 2021] Lin, S., Hilton, J., and Evans, O. (2021). Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

[Liu et al., 2021] Liu, P., Yuan, W., Fu, J., Zhang, Z., Luan, H., Geng, Y., et al. (2021). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

[Liu, Jerry, 2025] Liu, Jerry (2025). Llamaindex: A data framework for llm applications. `https://www.llamaindex.ai/`.

[Masry et al., 2022] Masry, A., Long, D. X., Tan, J. Q., Joty, S., and Hoque, E. (2022). ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2263–2279, Dublin, Ireland. Association for Computational Linguistics.

[Mathew et al., 2022] Mathew, M., Bagal, V., Tito, R., Karatzas, D., Valveny, E., and Jawahar, C. V. (2022). Infographicvqa. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2582–2591.

[Mathew et al., 2021] Mathew, M., Karatzas, D., and Jawahar, C. V. (2021). Docvqa: A dataset for vqa on document images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2199–2208.

[Matplotlib, 2024] Matplotlib (2024). Matplotlib: Visualization with python. https://matplotlib.org/stable/contents.html.

[Microsoft, 2025a] Microsoft (2025a). Documentación de visual studio code. https://code.visualstudio.com/docs.

[Microsoft, 2025b] Microsoft (2025b). Read model ocr data extraction - document intelligence. https://learn.microsoft.com/en-us/azure/ai-services/document-intelligence/prebuilt/read?view=doc-intel-4.0.0.

[Microsoft, 2025c] Microsoft (2025c). What is azure ai document intelligence? https://learn.microsoft.com/en-us/azure/ai-services/document-intelligence/overview?view=doc-intel-4.0.0.

[Microsoft, 2025d] Microsoft (2025d). What is intelligent document processing (idp)? https://www.microsoft.com/en-us/power-platform/products/power-automate/topics/business-process/intelligent-document-processing.

[Milvus, 2025] Milvus (2025). How does multimodal ai combine different types of data. https://milvus.io/ai-quick-reference/how-does-multimodal-ai-combine-different-types-of-data.

[Mistral, 2025a] Mistral (2025a). Documentación del cliente de python de mistral ai. https://docs.mistral.ai/getting-started/client/.

[Mistral, 2025b] Mistral (2025b). What are the limits of the free-models? https://docs.mistral.ai/#free-models.

[Mistral AI, 2025] Mistral AI (2025). Mistral small-2402. https://mistral.ai/news/mistral-small-3-1.

[NumPy, 2024] NumPy (2024). Numpy. https://numpy.org/doc/.

[Ollama, 2025] Ollama (2025). Ollama. https://ollama.com/.

[OpenAI, 2025] OpenAI (2025). Text generation models. https://platform.openai.com/docs/guides/text?api-mode=responses.

[OpenRouter, 2025] OpenRouter (2025). Openrouter: Build with any model. https://openrouter.ai/.

[PaddlePaddle, 2025] PaddlePaddle (2025). Paddleocr/readme_en.md. https://github.com/PaddlePaddle/PaddleOCR/blob/main/README_en.md.

[Pandas, 2024] Pandas (2024). pandas-dev/pandas: Pandas. https://pandas.pydata.org/.

[PaperOffice, 2025] PaperOffice (2025). Document data extraction with llm technology: Revolutionizing... https://start.paperoffice.com/en/data-extraction-documents-llm-vs-machine-learning.

[Prefect Technologies, Inc., 2025] Prefect Technologies, Inc. (2025). Prefect: The code-as-workflows orchestration platform. https://www.prefect.io/.

[Project, 2025] Project, T. J. (2025). Documentación de jupyter notebook. `https://jupyter-notebook.readthedocs.io/`.

[PyPI, 2025] PyPI (2025). pymupdf4llm: Documentación de pymupdf4llm. `https://pypi.org/project/pymupdf4llm/`.

[Python, 2025a] Python (2025a). Documentación del módulo itertools. `https://docs.python.org/3/library/itertools.html`.

[Python, 2025b] Python (2025b). Documentación del módulo json. `https://docs.python.org/3/library/json.html`.

[Python, 2025c] Python (2025c). Documentación del módulo pathlib. `https://docs.python.org/3/library/pathlib.html`.

[Python, 2025d] Python (2025d). Documentación del módulo re (expresiones regulares). `https://docs.python.org/3/library/re.html`.

[Python, 2025e] Python (2025e). Documentación del módulo sys. `https://docs.python.org/3/library/sys.html`.

[Python, 2025f] Python (2025f). Documentación del módulo time. `https://docs.python.org/3/library/time.html`.

[Python, 2025g] Python (2025g). Documentación del módulo types. `https://docs.python.org/3/library/types.html`.

[Qwen Team, 2024] Qwen Team (2024). Qwen3. `https://qwen3.org/`.

[Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *OpenAI*.

[Reid et al., 2024] Reid, M., Savinov, N., and et al. (2024). Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *ArXiv*, abs/2403.05530.

[Renze and Guven, 2024] Renze, M. and Guven, E. (2024). The effect of sampling temperature on problem solving in large language models. *arXiv preprint arXiv:2402.05201*.

[Research, 2025] Research, G. (2025). Advances in document understanding. `https://research.google/blog/advances-in-document-understanding/`.

[Riquelme et al., 2021] Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Pinto, A., Keysers, D., and Houlsby, N. (2021). Scaling vision with sparse mixture of experts.

[Rossum, 2025] Rossum (2025). Ingest & understand transactional documents at scale. `https://rossum.ai/platform/receive-documents/`.

[Science, 2024] Science, A. (2024). A quick guide to amazon's papers at cvpr 2024. `https://www.amazon.science/blog/a-quick-guide-to-amazons-papers-at-cvpr-2024`.

[SciPy, 2024] SciPy (2024). Scipy documentation. `https://docs.scipy.org/doc/scipy/`.

[SeaBorn, 2024] SeaBorn (2024). seaborn: statistical data visualization. https://seaborn.pydata.org/.

[Sennrich et al., 2016] Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

[Services, 2025] Services, A. W. (2025). What is intelligent document processing? - idp explained. https://aws.amazon.com/what-is/intelligent-document-processing/.

[Shang et al., 2025] Shang, J. et al. (2025). Flex: A benchmark for evaluating robustness of fairness in large language models. *arXiv preprint arXiv:2503.19540*.

[Shazeer et al., 2017] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. (2017). Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer.

[Shearer, 2000] Shearer, C. (2000). The CRISP-DM model: the new blueprint for data mining. *Journal of data warehousing*, 5(4):13–22.

[Siino et al., 2023] Siino, M., Tinnirello, I., and La Cascia, M. (2023). Is text preprocessing still worth the time? a comparative survey on the influence of popular preprocessing methods on transformers and traditional classifiers. *Information Systems*.

[Slashdot, 2025] Slashdot (2025). Compare amazon textract vs. google cloud document ai in 2025. https://slashdot.org/software/comparison/Amazon-Textract-vs-Google-Cloud-Document-AI/.

[Sokolova and Lapalme, 2009] Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45:427–437.

[Sánchez and Cuervo Londoño, 2024] Sánchez, J. and Cuervo Londoño, G. (2024). A bag-of-words approach for information extraction from electricity invoices. *AI*, 5:1837–1857.

[Sánchez et al., 2022] Sánchez, J., Salgado, A., García, A., and Monzón, N. (2022). IDSEM, an invoices database of the Spanish electricity market. *Scientific Data*, 9:786.

[The MLflow Project, 2025] The MLflow Project (2025). Mlflow: An open source platform for the mlops lifecycle. https://mlflow.org/.

[Udupa, 2024] Udupa, S. (2024). Explicitly unbiased large language models still form biased associations. Knowledge UChicago.

[Unstract, 2025] Unstract (2025). Llmwhisperer: Extract text and layout from any pdf. https://unstract.com/llmwhisperer/.

[V. A., 2024] V. A., T. (2024). *Towards Grounded Multimodal Enterprise Document Understanding*. PhD thesis, Carnegie Mellon University.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[Wagner and Fischer, 1974] Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

[Wei et al., 2022] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., and Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

[Weights & Biases, 2025] Weights & Biases (2025). W&b: The ai developer platform. `https://wandb.ai/site/`.

[Wikipedia, 2024] Wikipedia (2024). Cross industry standard process for data mining — Wikipedia, la enciclopedia libre.

[Wikipedia, 2025] Wikipedia (2025). Document layout analysis. `https://en.wikipedia.org/wiki/Document_layout_analysis`.

[Wu et al., 2025] Wu, Z. et al. (2025). A survey of large language model empowered agents for recommendation and search: Towards next-generation information retrieval. *arXiv preprint arXiv:2503.05659*.

[Xu et al., 2019] Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., and Zhou, M. (2019). Layoutlm: Pre-training of text and layout for document image understanding. *arXiv preprint arXiv:1912.13318*.

[Zhang et al., 2025] Zhang, H. et al. (2025). Flex: A benchmark for evaluating robustness of fairness in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*.

[Zhang, 2025] Zhang, M. (2025). Mamba: Linear-time sequence modeling with selective state spaces. `https://minjiazhang.github.io/courses/sp24-resource/Mamba-pre.pdf`.

[Zhao et al., 2023] Zhao, W. X. et al. (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*.

[Zhao et al., 2025] 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., Liu, P., Nie, J.-Y., and Wen, J.-R. (2025). A survey of large language models.

[Zoph et al., 2022] Zoph, B., Bello, I., Kumar, S., Du, N., Huang, Y., Dean, J., Shazeer, N., and Fedus, W. (2022). Designing effective sparse expert models.

ULPGC
Universidad de
Las Palmas de
Gran Canaria

ESCUELA DE INGENIERÍA
INFORMÁTICA