



**Universidad**  
Zaragoza

# Trabajo Fin de Máster

Sistema de generación automática de contenido en  
Redes Sociales e Informes basado en Modelos de  
Lenguaje de Gran Tamaño (LLMs)

Autor

Iván Latre Rodríguez

Director

Antonio Miguel Artiaga

ESCUELA DE INGENIERÍA Y ARQUITECTURA  
2025



## **RESUMEN**

Actualmente, la producción de contenido en redes sociales requiere una actualización constante y adaptada a las tendencias de cada plataforma. En este contexto, los sistemas de generación automática de texto basados en modelos de lenguaje de gran tamaño (LLMs) han adquirido una relevancia creciente. Este proyecto implementa un sistema que integra el desarrollo y evaluación de modelos de lenguaje con el fin de generar contenido para redes sociales, evaluando la calidad de los textos generados. En la evaluación se empleará un enfoque híbrido que combina métricas objetivas con valoraciones subjetivas que analicen la percepción de naturalidad y la adecuación del contenido, permitiendo determinar el rendimiento de distintos modelos novedosos.

Además, este proyecto también facilita la escalabilidad hacia nuevos sectores del mercado, donde el sistema propuesto, desarrollado como una variante optimizada de un sistema base centrado en el análisis de audiencias, permite adaptar la generación automática de contenido a distintos contextos y necesidades.

## **ABSTRACT**

Currently, content production for social media requires constant updates and adaptation to the evolving trends of each platform. In this context, automatic text generation systems based on large language models (LLMs) have gained increasing relevance. This project presents a system that integrates the development and evaluation of language models to generate content for social media, assessing the quality of the generated texts. The evaluation follows a hybrid approach that combines objective metrics with subjective assessments to analyze the perception of naturalness and content adequacy, allowing for the performance comparison of various state-of-the-art models. Additionally, this project facilitates scalability into new market sectors, where the proposed system, developed as an optimized variant of a base system focused on audience analysis, enables the automatic generation of content tailored to different contexts and requirements.



# Índice

<b>1. Introducción y objetivos</b>	<b>1</b>
1.1. Motivación . . . . .	1
1.2. Descripción del problema . . . . .	2
1.3. Objetivos del proyecto . . . . .	2
1.4. Estructura de la memoria . . . . .	3
<b>2. Estado del arte</b>	<b>5</b>
2.1. Introducción e historia de los Modelos de Lenguaje de Gran Tamaño (LLMs) . . . . .	5
2.2. Uso de la arquitectura Transformer en los LLM . . . . .	8
2.3. Adaptación y afinado de modelos LLM: Fine-tuning y técnicas avanzadas	13
2.3.1. LoRA: Low-Rank Adaptation . . . . .	13
2.3.2. QLoRA: Quantized Low-Rank Adaptation . . . . .	15
2.3.3. RLHF: Reinforcement Learning from Human Feedback . . . . .	16
2.4. Modelos LLM: Instruct y Text . . . . .	18
2.4.1. Modelos <i>Text</i> . . . . .	18
2.4.2. Modelos <i>Instruct</i> . . . . .	18
2.4.3. Comparación y Diferencias Clave . . . . .	19
2.5. Modelos de lenguaje abarcados . . . . .	20
<b>3. Implementación</b>	<b>23</b>
3.1. Ingesta de datos . . . . .	24
3.2. Análisis . . . . .	26
3.3. Exportación . . . . .	27
<b>4. Resultados</b>	<b>33</b>
4.1. Resultados objetivos . . . . .	33
4.2. Resultados subjetivos . . . . .	37
4.2.1. Escenario 1: usuarios con bajo nivel de conocimiento sobre LLMs	38

4.2.2.	Escenario 2: usuarios con nivel intermedio de conocimiento sobre LLMs . . . . .	40
4.2.3.	Escenario 3: usuarios con alto nivel de conocimiento sobre LLMs	41
4.2.4.	Resumen general y conclusiones . . . . .	42
4.3.	Conclusiones tras los resultados objetivos y subjetivos . . . . .	43
<b>5.</b>	<b>Conclusiones y líneas futuras</b>	<b>45</b>
5.1.	Conclusiones generales . . . . .	45
5.2.	Líneas futuras . . . . .	46
<b>6.</b>	<b>Bibliografía</b>	<b>47</b>
	<b>Lista de Figuras</b>	<b>53</b>
	<b>Lista de Tablas</b>	<b>55</b>
	<b>Anexos</b>	<b>56</b>
A.	Ejemplos de respuestas de modelos <i>Text</i> frente a <i>Instruct</i> . . . . .	59
A.1.	Ejemplo 1: Llama3.2 instruct y text . . . . .	59
A.2.	Ejemplo 2: Llama3.2 instruct y text . . . . .	63
B.	Ejemplos de LLaVA sobre selección de imágenes . . . . .	66
B.1.	Ejemplo 1: Tweet de Antena 3 . . . . .	66
B.2.	Ejemplo 2: Tweet de Aragón TV . . . . .	68
C.	Ejemplos de prompts para tweets de audiencias . . . . .	70
C.1.	Prompt base para tweets . . . . .	70
C.2.	Ejemplo 1: Prompt específico para tweet de Aragón TV . . . . .	71
C.3.	Ejemplo 2: Prompt específico para tweet de Antena 3 . . . . .	72
C.4.	Ejemplo 3: Prompt específico para tweet de Cuatro . . . . .	72
C.5.	Ejemplo 4: Prompt específico para tweet de Telecinco . . . . .	72
C.6.	Ejemplo 5: Prompt específico para tweet de Movistar+ . . . . .	72
D.	Ejemplos de tweets creados automáticamente con los LLMs evaluados en la memoria . . . . .	73
D.1.	Gemma2:27B . . . . .	73
D.2.	Gemma2:27B-instruct-q5_K_M . . . . .	73
D.3.	Qwen2.5:32B . . . . .	74
D.4.	Vicuna:13B . . . . .	75
D.5.	Vicuna:13B-v1.5-q4_0 . . . . .	76
D.6.	Qwen2.5:14B . . . . .	77
D.7.	Qwen2.5:14B-Instruct-q5_K_M . . . . .	77

E.	Modelos de lenguaje abarcados . . . . .	79
E.1.	LLaMA 3 . . . . .	79
E.2.	Vicuna . . . . .	82
E.3.	Mistral small . . . . .	84
E.4.	Gemma 2 . . . . .	86
E.5.	Qwen 2.5 . . . . .	89
E.6.	Modelos de Lenguaje Multimodales y sus posibles aplicaciones .	94
F.	Plantilla de Evaluación . . . . .	97



# Capítulo 1

## Introducción y objetivos

Las redes sociales han cambiado radicalmente la forma en que las organizaciones y los medios de comunicación interactúan con el público. En plataformas como Twitter, la rapidez y la concisión son elementos clave para captar la atención de los usuarios y transmitir mensajes de manera efectiva. En sectores como el análisis de audiencias televisivas, donde la inmediatez de la información es fundamental, la automatización en la generación de contenido con técnicas como *deep-learning* (*aprendizaje profundo*) y *procesamiento del lenguaje natural* (*PLN*) permite agilizar la difusión y personalización de datos relevantes sobre la repercusión de programas, eventos destacados y tendencias emergentes.

### 1.1. Motivación

En este contexto, surge la necesidad de desarrollar un sistema capaz de generar automáticamente tweets basados en datos de audiencias televisivas. Este proyecto se enmarca dentro del desarrollo de una plataforma empresarial enfocada en la recopilación, análisis y distribución de información relacionada con la medición de audiencias. Dentro de sus múltiples funcionalidades, la plataforma integra herramientas para la detección de anomalías, la identificación de los programas más vistos y la elaboración de informes detallados. Uno de los retos más importantes de esta plataforma es la generación automática de tweets que resuman de manera efectiva y precisa los datos de audiencia de un programa en concreto, garantizando que la información sea clara, relevante y adecuada para su difusión en redes sociales.

Para alcanzar este objetivo, es fundamental evaluar distintas soluciones basadas en modelos de lenguaje de última generación (LLMs, por sus siglas en inglés). La elección del modelo más adecuado requiere analizar su desempeño en términos de calidad del texto generado, tiempos de inferencia, consumo de recursos computacionales y costes asociados. Este estudio busca determinar qué modelo se adapta mejor a las necesidades

del sistema, asegurando una solución eficiente, precisa y escalable en el tiempo.

Todo lo expuesto justifica la realización de este proyecto, en el que se propone el análisis de LLMs actuales para redactar datos de audiencia, de forma que sean utilizados como posibles tweets.

## 1.2. Descripción del problema

El análisis de audiencias televisivas es una tarea fundamental para muchas empresas del sector mediático, ya que les permite evaluar el impacto de los programas y ajustar sus estrategias de contenido en función de la respuesta del público. No obstante, la dependencia de flujos de trabajo manuales limita la capacidad de escalar el modelo de negocio, ya que el procesamiento y la estructuración de datos requieren una inversión considerable de tiempo y recursos. Actualmente, la extracción y difusión de información en redes sociales se realiza mediante macros de Excel automatizadas con Visual Basic, lo que ofrece cierto nivel de eficiencia, pero aún demanda múltiples intervenciones manuales.

El cliente dedica muchas horas mensuales a extraer datos de bases de datos de audiencias, identificar eventos relevantes, cruzar información de distintos informes y redactar mensajes adecuados para plataformas como Twitter. Si bien esta metodología es funcional, no es escalable y representa una pérdida de eficiencia operativa que podría solucionarse mediante un sistema de generación automática de contenido.

Para optimizar el análisis y difusión de datos de audiencias, se ha desarrollado una plataforma que centraliza la información y automatiza la generación de informes y publicaciones en redes sociales, donde uno de sus módulos clave permite la creación automática de tweets basados en datos de audiencia.

El principal desafío dentro de este módulo es seleccionar el modelo de lenguaje más adecuado para la tarea. Los LLMs han demostrado ser una solución efectiva para la generación automatizada de contenido, pero su rendimiento varía según múltiples factores, los cuales se analizarán para poder tomar una decisión fundamentada sobre cuál se adapta mejor a los requerimientos del cliente.

## 1.3. Objetivos del proyecto

Los principales objetivos de este proyecto se enumeran a continuación:

- Automatizar la generación de tweets informativos basados en datos de audiencia televisiva, asegurando un equilibrio óptimo entre velocidad, precisión y consumo de recursos.

- Evaluar el rendimiento de distintos modelos de lenguaje en términos de tiempo de inferencia, número de intentos necesarios para generar respuestas válidas y calidad del contenido generado.
- Reducir el tiempo y los costes operativos asociados a la redacción manual de tweets, permitiendo que los recursos se destinen a tareas de mayor valor añadido.
- Analizar la viabilidad de los modelos de lenguaje en un entorno de producción real, dimensionando el sistema en función de los requisitos de hardware disponibles.

## 1.4. Estructura de la memoria

Este documento se estructura en diferentes capítulos que detallan el desarrollo y evaluación del proyecto, desde su fundamentación teórica hasta la presentación de los resultados obtenidos.

- **Capítulo 1: Introducción y objetivos**

Se expone la motivación del trabajo, la descripción del problema a resolver y los objetivos generales del proyecto. Además, se describe la estructura del documento.

- **Capítulo 2: Estado del arte**

Se presenta un análisis detallado de los LLMs, incluyendo su evolución, el uso de la arquitectura Transformer y las técnicas de ajuste como LoRA, QLoRA y RLHF.

- **Capítulo 3: Implementación**

Se detalla el desarrollo de la plataforma creada para la automatización de la generación de tweets mediante LLMs. Se describen las distintas fases del sistema, desde la ingesta de datos de audiencia hasta el análisis y exportación de contenido.

- **Capítulo 4: Resultados**

Se presentan los resultados obtenidos tras la evaluación de los modelos seleccionados. Se analizan métricas clave como el tiempo de inferencia, precisión, consumo de tarjeta gráfica (GPU) y número de intentos necesarios para generar tweets válidos.

- **Capítulo 5: Conclusiones**

Se resumen los principales hallazgos del estudio, evaluando la eficiencia del sistema y su impacto en la automatización de tareas dentro de la empresa cliente.

Además, se plantean posibles mejoras y futuras líneas de trabajo para optimizar la generación de contenido automatizado.

# Capítulo 2

## Estado del arte

### 2.1. Introducción e historia de los Modelos de Lenguaje de Gran Tamaño (LLMs)

La **inteligencia artificial generativa** (*Generative AI, GenAI*) es un campo de la inteligencia artificial diseñado para la creación de contenido original, como texto, imágenes, videos y audio. Utiliza modelos que analizan patrones y estructuras presentes en los datos de entrenamiento para producir nuevos contenidos que sean consistentes y contextualmente relevantes.

Dentro de este ámbito, destacan los **Modelos de Lenguaje de Gran Tamaño** (*Large Language Models, LLMs*), definidos por la Comisión Europea como "modelos de inteligencia artificial entrenados con algoritmos de aprendizaje profundo para reconocer, generar, traducir y/o resumir grandes volúmenes de datos textuales". Estos modelos emplean arquitecturas avanzadas que les permiten identificar relaciones complejas entre palabras, incluso cuando están separadas por largas distancias dentro de un texto.

El entrenamiento de los LLMs se realiza con enormes cantidades de datos, como libros, artículos y páginas web. Este proceso permite que los modelos aprendan patrones lingüísticos y estructuras sintácticas, habilitándolos para ejecutar una amplia variedad de tareas, que incluyen generación de texto, traducción automática y análisis de sentimientos.

La **eficacia** de un LLM depende de su tamaño, la diversidad de los datos de entrenamiento y la sofisticación de sus algoritmos, lo que influye directamente en su capacidad para aplicaciones prácticas en diversos campos. Por ello, entrenar un LLM es una tarea que requiere una capacidad muy elevada de computación y de tiempo de máquina, y, por tanto, costes muy significativos.

A pesar de los desafíos, los LLMs representan una herramienta revolucionaria con aplicaciones prácticas en diversos campos, como la comunicación, la educación y el

análisis de datos. Sin embargo, su desarrollo plantea preguntas sobre sostenibilidad y accesibilidad, que deberán abordarse para garantizar un uso responsable y equitativo de estas tecnologías.

El desarrollo de los LLM representa un avance significativo dentro del procesamiento del lenguaje natural (NLP). Estos modelos se basan en enfoques autoregresivos, lo que significa que estiman la probabilidad de una palabra o símbolo en función de los elementos previos en una secuencia. Este tipo de modelado tiene sus raíces en la teoría de la información desarrollada por Claude Shannon. En su trabajo pionero de 1948, Shannon introdujo conceptos fundamentales como la entropía, los codificadores y decodificadores, y los modelos probabilísticos de lenguaje, sentando las bases para los enfoques autoregresivos utilizados en los modelos modernos. A mediados del siglo XX, los primeros intentos de modelado del lenguaje se basaban en reglas gramaticales explícitas, pero con el tiempo evolucionaron hacia enfoques estadísticos y de aprendizaje automático que han permitido la creación de modelos altamente escalables y adaptativos.

A medida que el campo evolucionó, las décadas de 1980 y 1990 presenciaron un cambio sustancial hacia métodos estadísticos de procesamiento de lenguaje. Este período vio la adopción de Modelos Ocultos de Markov (HMMs) y modelos *n-gramas*, que ofrecieron un enfoque más dinámico para predecir secuencias de palabras basadas en probabilidades, en lugar de sistemas de reglas fijas.

El resurgimiento de las redes neuronales a principios de los años 2000, gracias a los avances en algoritmos de **backpropagation** (retropropagación) que mejoraron el entrenamiento de redes multicapa, marcó un desarrollo crucial. Un hito fue la introducción de redes neuronales de alimentación directa para la modelización del lenguaje [1], lo que sentó las bases para innovaciones subsecuentes en la representación de palabras, especialmente la introducción de embeddings de palabras [2]. Estos embeddings representan palabras como vectores numéricos que permiten definir distancias entre palabras, de manera que conceptos similares tengan distancias reducidas, capturando relaciones semánticas con una efectividad sin precedentes.

Los primeros **mecanismos de atención** surgieron en el contexto de las redes neuronales recurrentes (RNN), con propuestas iniciales en los años 90 en el marco de modelos como las *LSTM*. Sin embargo, su aplicación efectiva en tareas de NLP se consolidó con enfoques posteriores, como las *Highway Networks* [3], que facilitaron el entrenamiento de redes profundas al permitir un flujo más eficiente de la información. La popularización del mecanismo de atención en el NLP moderno llegó en 2014 con modelos como el de Bahdanau [4], que mejoraron significativamente la traducción automática al asignar pesos diferenciales a distintas partes del texto de entrada.

El verdadero cambio de paradigma ocurrió con la introducción de la arquitectura **Transformer** [5], que reemplazó las RNN por mecanismos de **autoatención** y **atención cruzada**, eliminando la necesidad de procesar secuencias de manera estrictamente recurrente. Esta innovación permitió entrenar modelos de lenguaje a gran escala con mayor eficiencia computacional y capturar dependencias a largo plazo con mayor precisión. Además, esto significa que el modelo puede enfocarse en las partes más relevantes del texto al generar una respuesta, crucial para analizar el contexto y las relaciones complejas dentro de las secuencias de palabras. Esta arquitectura es la piedra angular del funcionamiento interno de los LLMs.

La variante de la arquitectura *Transformer* utilizada en la mayoría de los LLMs es la denominada *Decoder-Only Transformers*. Esta versión simplifica el decodificador del *Transformer* original al eliminar los bloques de atención cruzada, que solo tienen sentido en presencia de un codificador. Aun así, conserva los mecanismos clave como la autoatención, las capas de MLP y la codificación posicional, lo que permite a los modelos comprender y generar texto con una sofisticación sin precedentes.

Siguiendo la línea temporal, la serie de modelos GPT desarrollados por OpenAI, desde GPT-1 en junio de 2018 hasta GPT-4 [6] en marzo de 2023, ejemplifica la capacidad de los *Transformers* para escalar a tamaños sin precedentes. Además de la serie GPT de OpenAI, otros modelos como *Google Gemini* y la familia de modelos *LLaMA* de Meta han surgido como actores clave en el panorama de la IA. Gemini refleja cómo grandes empresas tecnológicas están invirtiendo en el desarrollo de LLM avanzados, mientras que LLaMA destaca por ser una alternativa de código abierto, permitiendo a la comunidad investigadora y a la industria explorar y adaptar estos modelos de manera más flexible.

El año 2023, llamado “el año de la IA”, destaca como un hito en la historia de los LLM, caracterizado por una mayor accesibilidad y contribuciones globales. Las innovaciones durante este año demostraron que los LLM pueden construirse con un mínimo de código, reduciendo significativamente las barreras de entrada, aunque a la vez introduciendo nuevos desafíos, como el coste de entrenamiento e inferencia, y sus riesgos inherentes. Este período también vio una creciente preocupación por las consideraciones éticas y los desafíos planteados por el desarrollo y uso de los LLM, lo que dio lugar a avances en la regulación de la IA y la IA generativa en todo el mundo.

La proliferación de los LLM de código abierto ha marcado un hito en la democratización de la tecnología de IA. Comenzando por LLaMA y siguiendo con Vicuna, Falcon, Mistral, Gemini, Gemma, entre otros, los LLM open-source han democratizado el acceso a la tecnología puntera en el procesamiento del lenguaje, permitiendo a investigadores, desarrolladores y aficionados experimentar, personalizar

y desplegar soluciones de IA con una inversión inicial mínima. La disponibilidad de estos modelos ha fomentado una colaboración sin precedentes en la comunidad de IA, estimulando la innovación y facilitando la creación de aplicaciones avanzadas en una variedad de sectores. Actualmente, se distinguen dos enfoques principales en su desarrollo: los modelos orientados a seguir instrucciones específicas (*Instruct Models*) y los modelos diseñados para completar texto de forma más libre (*Text Models*). Ambos enfoques representan aplicaciones distintas del mismo marco técnico, y serán tratados en detalle más adelante en este trabajo.

Por último, la integración de LLM en herramientas ofimáticas y de desarrollo de software está transformando la eficiencia y capacidad de las empresas. Microsoft ha integrado los LLM en su suite de Office bajo el nombre de Microsoft 365 Copilot, mientras que Google lo ha hecho en Google Workspace. Al mismo tiempo, herramientas como GitHub Copilot o StarCoder utilizan LLM para asistir a los programadores, acelerando la generación de código y mejorando la calidad del desarrollo de software.

## 2.2. Uso de la arquitectura Transformer en los LLM

La arquitectura Transformer se ha convertido en el paradigma dominante en los LLM, gracias a su capacidad para capturar dependencias a largo plazo y modelar secuencias complejas de manera eficiente. Su diseño se basa en los mecanismos de atención, que permiten a los modelos enfocarse en las partes relevantes de la entrada mientras generan o procesan texto. Dentro de este marco, los modelos que emplean una configuración “decoder-only” han demostrado ser particularmente útiles en tareas de generación de texto, como el completado de frases, la traducción automática y la escritura asistida.

En los *Transformers decoder-only*, la arquitectura se centra exclusivamente en la parte de decodificación, eliminando la necesidad de un codificador separado. Esto simplifica el diseño para tareas donde la entrada y la salida comparten un formato similar, o donde la entrada puede ser procesada de manera incremental. La idea principal es que el modelo utiliza bloques apilados de capas de autoatención y redes neuronales feedforward para procesar la secuencia generada hasta el momento y predecir el siguiente token, como se muestra en la figura 2.1. Cada capa del decodificador contiene múltiples cabezales de autoatención que calculan pesos de atención basándose en la similitud entre representaciones internas de tokens. Estos pesos determinan cuánta información se debe priorizar de cada token en el contexto.

Un aspecto clave de los modelos decoder-only es el uso de la *máscara causal*. Este mecanismo asegura que el modelo, al predecir un token, solo pueda considerar

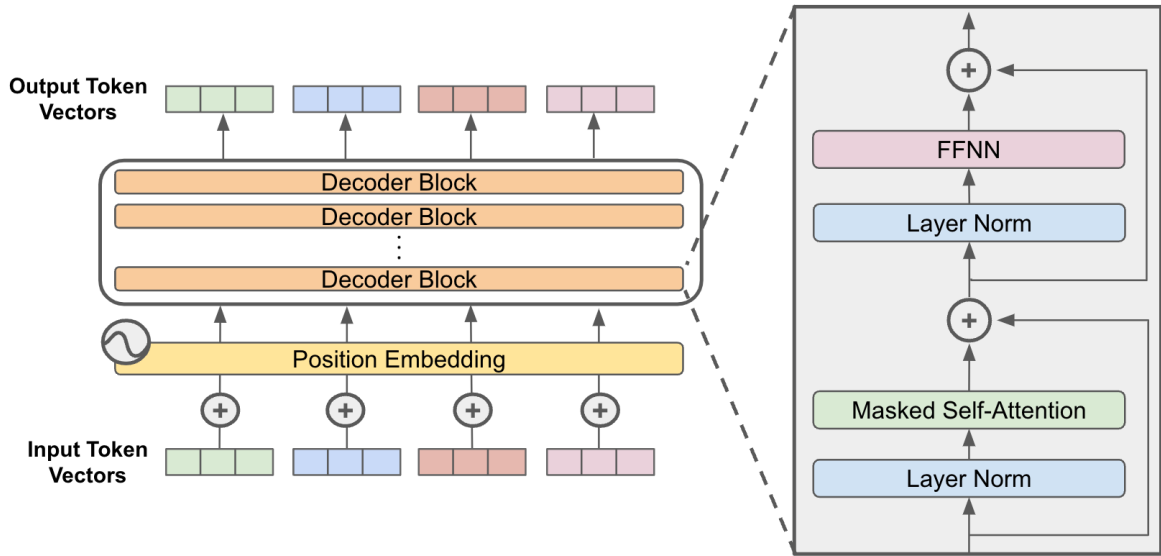


Figura 2.1: Estructura de trabajo de un modelo *Transformer decoder-only* [7]

los tokens anteriores en la secuencia, respetando el flujo causal del lenguaje natural. Este enmascaramiento se implementa estableciendo a cero los pesos de atención correspondientes a tokens futuros, lo que obliga al modelo a trabajar de manera auto-regresiva. En la práctica, esto significa que, para predecir una palabra en una frase, el modelo tiene acceso únicamente al contexto ya generado, lo que simula cómo los humanos producen lenguaje.

Profundizando en este concepto, la figura 2.2 ilustra un aspecto fundamental en los mecanismos de autoatención causal dentro de los modelos *Transformer decoder-only*, específicamente el cálculo de las puntuaciones de autoatención enmascaradas. En estos modelos, la atención se calcula mediante operaciones matriciales entre la matriz de consultas (*Query*), la matriz de claves (*Key*) y, posteriormente, aplicando una normalización. El proceso es clave para determinar la influencia que cada token previo tiene sobre el siguiente.

$$\text{Softmax} \left[ \frac{\text{Query Matrix} \times \text{Transposed Key Matrix}}{\sqrt{d}} \right] = \begin{matrix} \text{Unnormalized} \\ \text{Attention Scores} \end{matrix} \rightarrow \begin{matrix} \text{Masked} \\ \text{Attention Scores} \end{matrix}$$

Unnormalized Attention Scores				
1.1	2.3	5.3	2.1	-1.2
-0.1	0.5	1.3	-0.1	6.1
2.3	0.2	3.3	-1.0	-0.4
0.3	1.2	-0.3	5.0	1.4
5.1	-2.9	-1.1	-4.2	0.4

Masked Attention Scores				
1.1	-∞	-∞	-∞	-∞
-0.1	0.5	-∞	-∞	-∞
2.3	0.2	3.3	-∞	-∞
0.3	1.2	-0.3	5.0	-∞
5.1	-2.9	-1.1	-4.2	0.4

Figura 2.2: Puntuaciones de *autoatención enmascarada* en *autoatención causal* [7]

Inicialmente, la matriz de consultas, representada en color azul, se multiplica con

la traspuesta de la matriz de claves, mostrada en verde. Este producto da lugar a una matriz de puntuaciones de atención no normalizadas, donde cada celda indica la relevancia entre pares de tokens. En esta etapa, las puntuaciones aún no se han ajustado para garantizar que el modelo cumpla con las restricciones causales necesarias para tareas de generación de texto.

La autoatención enmascarada entra en juego al modificar estas puntuaciones no normalizadas. Como se observa en la imagen, ciertos valores de la matriz se reemplazan por infinitos negativos ( $-\infty$ ), lo que provoca que, al aplicar la función *softmax* [8], dichas posiciones tengan una probabilidad nula. Esta operación asegura que cada token en la secuencia solo pueda "ver" los tokens anteriores o a sí mismo, pero no los futuros. Por ejemplo, en la figura, la primera fila de la matriz enmascarada conserva únicamente la puntuación inicial, mientras que el resto de las columnas se han invalidado. De forma similar, las filas siguientes revelan un patrón triangular donde solo los tokens previos son accesibles en cada paso.

El enmascaramiento causal es imprescindible porque los modelos *decoder-only* generan texto de manera autoregresiva, es decir, token por token. Sin esta restricción, el modelo podría aprovechar información de tokens futuros durante el entrenamiento, lo que crearía inconsistencias al momento de generar texto de manera secuencial. La función *softmax* aplicada sobre la matriz ajustada convierte las puntuaciones en probabilidades, permitiendo que el modelo enfoque su atención en los tokens relevantes dentro del contexto pasado.

Otra característica distintiva de los *Transformers decoder-only* es la incorporación de **embeddings posicionales**. A diferencia de las redes recurrentes, los *Transformers* no procesan las secuencias de manera estrictamente secuencial, sino en paralelo. Esto mejora considerablemente la eficiencia computacional, pero requiere un método para informar al modelo sobre el orden de los tokens. Para lograr esto, se utilizan **embeddings posicionales**, que se suman a los embeddings de los tokens, proporcionando al modelo la información estructural necesaria para identificar la posición de cada token en la secuencia.

La ecuación 2.1 ilustra el método **sinusoidal** utilizado para generar estos embeddings posicionales. Este enfoque utiliza funciones seno y coseno para asignar posiciones únicas a cada token en la secuencia.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \quad (2.1)$$

- *pos*: Representa la posición absoluta del token en la secuencia (indicado en rojo en la imagen).
- *i*: Es el índice dentro del embedding posicional (indicado en azul).

- $d_{\text{model}}$ : Representa la dimensionalidad de los embeddings de los tokens (en verde).

El uso de seno para las dimensiones pares y coseno para las dimensiones impares garantiza que cada posición tenga un patrón único de valores, manteniendo relaciones lineales que el modelo puede aprender. Además, este método sinusoidal tiene propiedades clave: **es determinista** (no necesita ser aprendido) y **permite extrapolar posiciones más allá de las que el modelo ha visto durante el entrenamiento**.

Por ejemplo, en una tarea de generación de texto, el modelo recibe como entrada un *prompt* inicial, que puede ser una palabra o una frase. El embedding de este *prompt*, que ya incluye la información posicional generada mediante las funciones seno y coseno, se pasa a través de las capas del decodificador. En cada capa, el mecanismo de autoatención determina cuáles palabras del *prompt* son más relevantes para predecir el siguiente token. La salida de la última capa pasa por una capa de proyección lineal y una función *softmax* para generar una distribución de probabilidad sobre el vocabulario. Finalmente, se selecciona el token con mayor probabilidad, que se concatena al *prompt* y se utiliza como entrada para el siguiente paso. Este proceso se repite hasta que se alcanza un límite predefinido o se genera un token de fin de secuencia. Existen diversos métodos para la selección del siguiente token, desde enfoques deterministas como la selección del token con máxima probabilidad (*greedy decoding*) hasta métodos más avanzados como el muestreo estocástico (*top-k*) o la búsqueda de los  $n$  mejores candidatos (*beam search*), cada uno con implicaciones en la diversidad y coherencia del texto generado.

La capacidad de los *Transformers decoder-only* para manejar contextos largos y mantener coherencia a lo largo de varias oraciones se debe, en gran parte, a la profundidad y flexibilidad del mecanismo de atención, combinado con los embeddings posicionales sinusoidales. Las capas más profundas capturan relaciones semánticas de alto nivel, mientras que las capas más superficiales se enfocan en patrones más locales. Además, técnicas como el ajuste fino en tareas específicas o el preentrenamiento en grandes corpus de texto ayudan a los modelos a generalizar mejor y adaptarse a diferentes dominios.

Un ejemplo práctico del uso de *Transformers decoder-only* es **GPT (Generative Pre-trained Transformer)** [9], mencionado anteriormente. En este modelo, la arquitectura del decodificador se optimiza para predecir el siguiente token en secuencias largas, utilizando un corpus masivo de datos para pre-entrenamiento. Durante esta fase, el modelo aprende patrones lingüísticos generales que luego pueden ajustarse a tareas específicas como la traducción o el resumen de texto. La arquitectura auto-regresiva de

GPT, combinada con embeddings posicionales sinusoidales, asegura que la generación de texto sea coherente y gramaticalmente correcta, incluso en *prompts* complejos o creativos.

El desarrollo de los modelos GPT ha seguido una progresión notable en cuanto a escalado, tanto en tamaño como en capacidad. La primera versión, conocida como GPT, introdujo la idea de pre-entrenar un modelo transformer únicamente con un decodificador, utilizando grandes volúmenes de texto para que el modelo aprendiera representaciones contextuales generalizadas. Posteriormente, la llegada de GPT-2 [10] representó un salto significativo al demostrar que, al aumentar la cantidad de parámetros y los datos de entrenamiento, el modelo podía generar textos más coherentes y extensos sin depender de datos etiquetados específicos. Esta versión destacó por su capacidad para realizar tareas como la generación de párrafos enteros a partir de un simple *prompt*, lo que puso de manifiesto el potencial del escalado de los modelos en cuanto a volumen y calidad.

La publicación de GPT-3 [11] consolidó el paradigma del escalado en modelos de lenguaje, al aumentar drásticamente el número de parámetros hasta 175 mil millones. Esta versión mostró que el rendimiento en tareas de comprensión y generación de texto no solo mejora con más datos y parámetros, sino que también logra resultados excepcionales en tareas de *few-shot learning*, donde el modelo realiza tareas con muy pocos ejemplos proporcionados en el *prompt*. GPT-3 fue entrenado en un corpus masivo y diverso, lo que le permitió capturar patrones más complejos del lenguaje y ampliar su aplicabilidad a contextos especializados sin necesidad de ajuste fino explícito.

El éxito de GPT-3 también impulsó investigaciones centradas en los límites del escalado y en cómo optimizar el uso de los recursos computacionales. Esto abrió camino a modelos más eficientes y a estrategias de entrenamiento que buscan balancear el tamaño del modelo con la cantidad de datos utilizados. La hipótesis central detrás de estas investigaciones es que la combinación de más parámetros, mejores datos y técnicas de entrenamiento optimizadas resulta en una mayor capacidad para generar textos coherentes, adaptarse a distintos dominios y realizar tareas complejas sin intervención humana.

Estos avances en la serie GPT no solo demostraron que el escalado mejora la calidad de las predicciones, sino que también validaron la importancia del pre-entrenamiento en modelos de lenguaje. El desarrollo de arquitecturas cada vez más robustas ha generado un impacto considerable en áreas como la generación creativa, la automatización de procesos lingüísticos y el soporte en tareas de razonamiento avanzado.

## 2.3. Adaptación y afinado de modelos LLM: Fine-tuning y técnicas avanzadas

El entrenamiento de LLMs representa un avance significativo en múltiples dominios, pero su enorme escala plantea retos técnicos considerables, especialmente al adaptarlos a tareas específicas. Modelos como GPT-3 o LLaMA-7B, que poseen miles de millones de parámetros, requieren recursos computacionales y de memoria masivos para llevar a cabo un ajuste fino (*fine-tuning*). Este proceso estándar implica la actualización de todos los parámetros del modelo, lo que resulta ineficiente y costoso, especialmente cuando el objetivo es ajustarlos para dominios o aplicaciones concretas.

Para abordar estas limitaciones, surge el concepto de afinado eficiente de parámetros (*Parameter-Efficient Fine-Tuning*, PEFT) [12]. Este enfoque busca minimizar el número de parámetros que deben ser actualizados, manteniendo el modelo base preentrenado casi inalterado. De esta manera, se reduce drásticamente el coste computacional y la memoria requerida, permitiendo que incluso hardware limitado pueda manejar la tarea de afinado. Además, esta estrategia mantiene intacta la capacidad del modelo para generalizar conocimientos previamente aprendidos, ya que la mayor parte de los parámetros originales permanece congelada.

El principio fundamental del PEFT es la reutilización del conocimiento general del modelo preentrenado, complementándolo con ajustes específicos que se introducen a través de mecanismos adicionales o mediante la selección de un subconjunto de los parámetros existentes. Así, el PEFT logra adaptar los modelos a tareas especializadas sin necesidad de un proceso completo de reentrenamiento. Esto no solo acelera el proceso de afinado, sino que también optimiza el uso de los recursos, haciéndolo más viable en aplicaciones del mundo real. Al reducir los costes de computación y facilitar el ajuste a tareas específicas, este enfoque ha democratizado el uso de LLMs en campos diversos como la medicina, la atención al cliente y la generación de contenidos. Este concepto sienta las bases para explorar técnicas específicas, como *LoRA*, *QLoRA* o *RLHF*, las cuales se van a comentar a continuación.

### 2.3.1. LoRA: Low-Rank Adaptation

El método LoRA (*Low-Rank Adaptation*) [13] es una técnica de ajuste eficiente de parámetros diseñada para adaptar modelos de lenguaje de gran tamaño a tareas específicas, sin necesidad de modificar la mayoría de sus pesos preentrenados.

En la figura 2.3 nos muestra el proceso de fine-tuning eficiente para LLMs, en el que en lugar de actualizar directamente los parámetros del modelo base dada la matriz de pesos ( $W$ ), LoRA introduce dos matrices entrenables de baja dimensión ( $A$  y  $B$ ) que

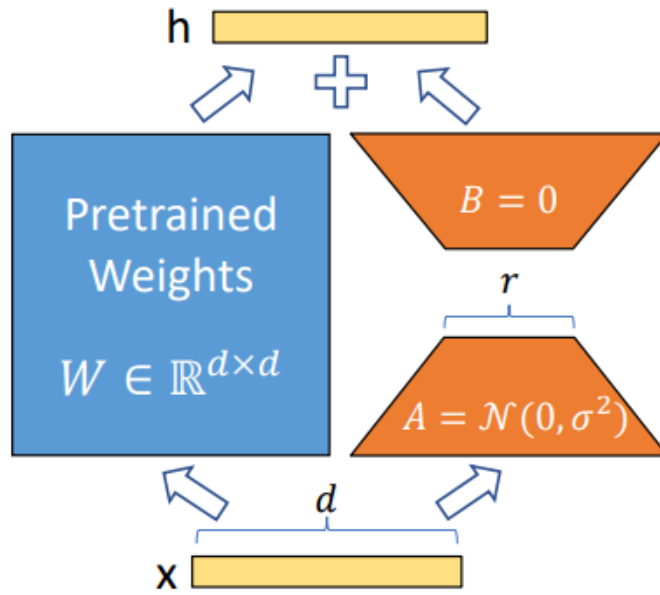


Figura 2.3: Diagrama de LoRA durante el entrenamiento [13]

se combinan para realizar ajustes específicos. Durante el entrenamiento, los pesos  $W$  permanecen congelados, y las matrices  $A$  y  $B$  se utilizan para calcular una corrección que se suma a la salida del modelo preentrenado, siguiendo la fórmula:

$$h = Wx + BAx \quad (2.2)$$

Este enfoque reduce significativamente el coste computacional y la memoria requerida, ya que solo una pequeña fracción de los parámetros se entrena, mientras que el resto de los pesos se mantienen congelados durante el proceso de fine-tuning. Tras el ajuste, los pesos originales y las adaptaciones se pueden fusionar en una única matriz para el despliegue. LoRA es particularmente útil en escenarios donde los recursos son limitados, manteniendo el conocimiento general del modelo preentrenado y permitiendo adaptaciones rápidas y eficientes.

Un ejemplo ilustrativo del impacto de este método se puede observar en la adaptación de un modelo como *LLaMA-7B* [14] para tareas de atención médica [15]. En lugar de reentrenar todo el modelo, esta técnica permite introducir matrices especializadas que capturan términos y patrones médicos específicos, manteniendo los parámetros originales congelados. Este enfoque no solo ahorra tiempo y recursos, sino que también preserva la capacidad generalista del modelo, lo que resulta útil en contextos donde las consultas pueden abarcar un amplio rango de temas.

### 2.3.2. QLoRA: Quantized Low-Rank Adaptation

QLoRA (*Quantized Low-Rank Adaptation*) [16] es una evolución del método LoRA, que busca optimizar aún más los recursos necesarios para realizar el *fine-tuning* de LLMs. Mientras que LoRA introduce matrices entrenables de baja dimensión para ajustar el modelo de manera eficiente, QLoRA amplía esta idea al incorporar la cuantización de los parámetros del modelo base, logrando una reducción adicional en la memoria y los requisitos computacionales.

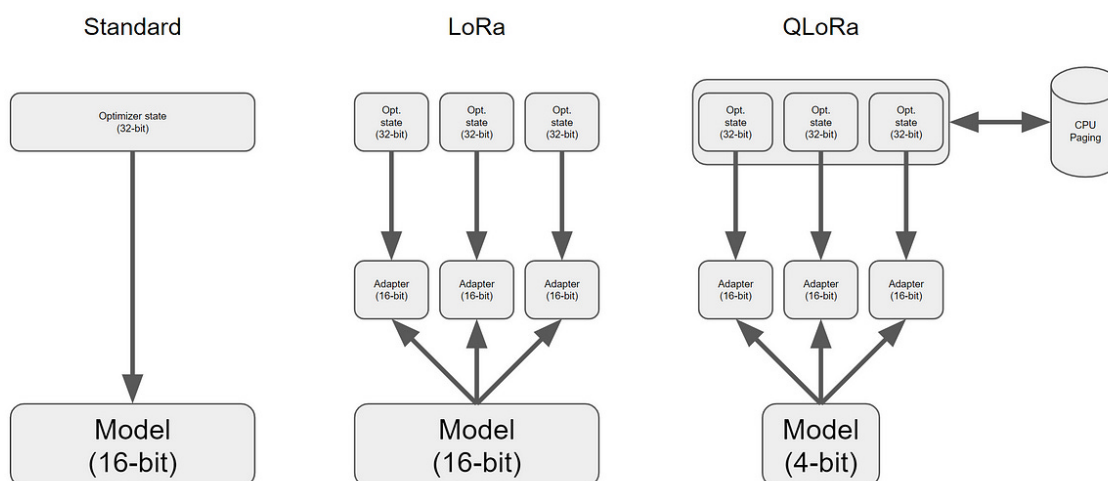


Figura 2.4: Comparación entre *fine-tuning* estándar, LoRA y QLoRA [17]

En la figura 2.4 se presentan tres esquemas donde se pueden ver las diferencias fundamentales entre el *fine-tuning* estándar, LoRA y QLoRA. En el enfoque estándar, el modelo base opera en una representación de 16 bits, lo que implica una mayor carga de memoria durante el entrenamiento y la inferencia. Además, todos los parámetros del modelo son actualizados, lo que resulta costoso desde el punto de vista computacional. En cambio, LoRA congela los pesos del modelo base y añade matrices adaptativas de 16 bits, reduciendo la cantidad de parámetros entrenables sin comprometer la calidad del ajuste. Sin embargo, el modelo principal sigue utilizando una representación de 16 bits, lo que deja margen para mejorar la eficiencia.

QLoRA va un paso más allá al aplicar una cuantización del modelo base a 4 bits, reduciendo significativamente el uso de memoria mientras mantiene un alto rendimiento en tareas específicas. Además, QLoRA introduce una estrategia de paginación que permite gestionar de manera eficiente los estados del optimizador y las matrices adaptativas, moviéndolos a la CPU cuando no se necesitan activamente en la GPU. Esto hace que el enfoque sea especialmente adecuado para hardware con recursos limitados, como GPUs de menor capacidad.

La clave del éxito de QLoRA radica en su capacidad para mantener los pesos

del modelo principal congelados y cuantizados a 4 bits, mientras que las matrices adaptativas ( $A$  y  $B$ ) operan a 16 bits, lo que garantiza un ajuste de alta precisión en un espacio de parámetros reducido. Este enfoque permite que modelos de gran tamaño, como LLaMA-65B [14], puedan ser afinados en hardware con memoria limitada, sin necesidad de comprometer el rendimiento general del modelo, por lo que representa una solución innovadora que combina la eficiencia de la cuantización con los principios del ajuste de baja dimensión de LoRA.

### 2.3.3. RLHF: Reinforcement Learning from Human Feedback

El aprendizaje por refuerzo a partir de retroalimentación humana (RLHF, por sus siglas en inglés) es un método diseñado para mejorar la alineación de los LLMs con las expectativas y valores humanos. Este enfoque combina técnicas de aprendizaje supervisado y por refuerzo, utilizando información proporcionada por humanos para refinar y ajustar las respuestas generadas por los modelos, logrando que sean más útiles, coherentes y seguras.

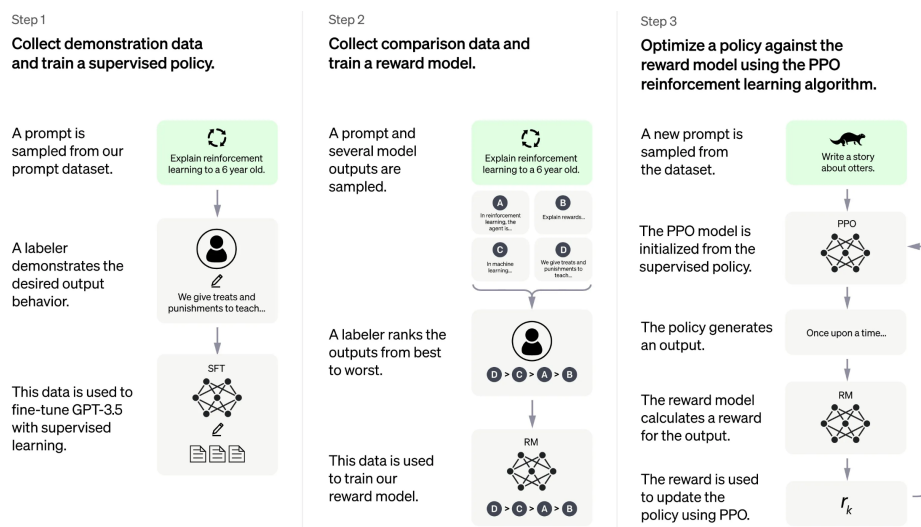


Figura 2.5: Etapas del proceso de RLHF en el entrenamiento de InstructGPT a partir de GPT-3 [18]

La figura 2.5 muestra las tres etapas principales de este proceso: el ajuste supervisado inicial, el entrenamiento del modelo de recompensa y el ajuste por refuerzo. Cada una de estas etapas está diseñada para abordar diferentes aspectos del problema de alineación, asegurando que el modelo aprenda no solo a procesar datos textuales masivos, sino también a generar respuestas que sean útiles y éticamente responsables.

El proceso comienza con un modelo de lenguaje previamente entrenado. Esto es muy importante, ya que los LLMs requieren una enorme cantidad de datos de entrenamiento. Entrenarlos desde cero con supervisión humana es prácticamente imposible. Un LLM

que se entrena previamente mediante aprendizaje no supervisado ya tendrá un modelo sólido del lenguaje y creará resultados coherentes, aunque algunos o muchos de ellos podrían no estar alineados con los objetivos e intenciones de los usuarios.

Tras esto, un equipo de etiquetadores humanos crea un conjunto de datos con instrucciones específicas y ejemplos de respuestas deseadas. Este entrenamiento inicial asegura que el modelo adquiera un comportamiento básico alineado con los patrones esperados. Posteriormente, se recopilan datos adicionales a partir de las respuestas generadas por el modelo, las cuales son evaluadas y clasificadas por los etiquetadores según su calidad, relevancia y adecuación a las instrucciones. Estas clasificaciones habilitan la posibilidad de entrenar el modelo de lenguaje mediante una función de coste que optimiza directamente el objetivo deseado. En lugar de utilizar únicamente la entropía cruzada, que maximiza la probabilidad de las secuencias más frecuentes sin garantizar que sean óptimas para una tarea específica, el modelo de recompensa actúa como una función de evaluación que guía el aprendizaje en etapas posteriores. Al entrenarse con los resultados y las puntuaciones de clasificación del LLM, este modelo de recompensa permite ajustar el comportamiento del sistema hacia respuestas alineadas con las preferencias humanas o métricas definidas para la tarea en cuestión.

Con el modelo de recompensa como guía, se emplea el aprendizaje por refuerzo para optimizar las respuestas del modelo principal. El algoritmo Proximal Policy Optimization (PPO) es utilizado en esta etapa para ajustar el modelo, maximizando las puntuaciones predichas por el modelo de recompensa y, en consecuencia, alineando las salidas del modelo con las preferencias humanas. Este enfoque permite que el modelo generalice mejor en contextos donde las instrucciones son menos precisas o no estaban incluidas explícitamente en el conjunto de datos inicial.

El RLHF tiene numerosos beneficios. Permite a los modelos de lenguaje mejorar su capacidad para seguir instrucciones específicas, además de reducir la generación de contenido irrelevante, inadecuado o tóxico. Este método ha demostrado ser efectivo incluso en modelos más pequeños, logrando que las respuestas generadas sean preferidas por los usuarios en evaluaciones directas frente a modelos más grandes sin RLHF.

Un ejemplo sólido de este enfoque se refleja en el entrenamiento de InstructGPT a partir de GPT-3 [18], demostrando cómo el RLHF puede transformar la interacción de los usuarios con los LLMs.

El caso de InstructGPT evidencia la eficacia del RLHF. En evaluaciones realizadas por humanos, las respuestas generadas por este modelo, ajustado con RLHF, fueron preferidas frente a las del GPT-3 original, incluso cuando el modelo ajustado tenía un tamaño significativamente menor. Además, InstructGPT mostró una mejora sustancial en la veracidad de las respuestas y una reducción en la generación de contenido

inapropiado, lo que resalta la importancia de este enfoque para construir sistemas de IA más seguros y efectivos. Su implementación en proyectos como InstructGPT no solo ha demostrado su utilidad técnica, sino que también ha marcado un avance en la creación de sistemas de inteligencia artificial alineados con las necesidades y expectativas humanas.

## 2.4. Modelos LLM: Instruct y Text

Los modelos de lenguaje se han desarrollado con diferentes objetivos en mente, lo que ha llevado a la creación de subcategorías como los modelos *Text* y los *Instruct*, los cuales se consideran los más relevantes a destacar en este proyecto. Ambos comparten principios básicos de entrenamiento basados en grandes cantidades de datos textuales y el uso de arquitecturas avanzadas como Transformer, pero se diferencian en los enfoques específicos de su adaptación y el comportamiento en la generación de texto.

### 2.4.1. Modelos *Text*

Los modelos *Text*, como los GPT-3 base, están diseñados para predecir la siguiente palabra en una secuencia dada una entrada inicial. Estos modelos se entrenan exclusivamente con objetivos de predicción de lenguaje, utilizando grandes conjuntos de datos textuales sin anotaciones específicas. Este entrenamiento los convierte en modelos altamente competentes para completar frases, responder preguntas y generar texto coherente, pero sin un entendimiento profundo de las intenciones o contextos específicos más allá de lo implícito en los datos de entrenamiento.

Un aspecto esencial para evaluar la eficacia de estos modelos es la métrica de perplejidad, que mide qué tan bien el modelo predice una secuencia de palabras. Una perplejidad baja indica que el modelo asigna altas probabilidades a palabras correctas en un contexto dado, reflejando un rendimiento superior en coherencia y precisión. Por el contrario, una perplejidad alta señala dificultades para interpretar el contexto adecuadamente, lo que puede impactar negativamente la calidad de las predicciones.

En este tipo de modelos, la generación de respuestas es guiada principalmente por patrones estadísticos aprendidos. Esto puede llevar a problemas como la generación de respuestas inconsistentes o el fenómeno conocido como “alucinación”, donde el modelo produce información incorrecta o inventada de manera confiada.

### 2.4.2. Modelos *Instruct*

Por otro lado, los modelos *Instruct*, como InstructGPT, se desarrollan mediante técnicas adicionales que alinean sus respuestas con las intenciones humanas.

Estos modelos son sometidos a un entrenamiento adicional conocido como RLHF (Reinforcement Learning with Human Feedback), donde se utiliza retroalimentación humana para ajustar las respuestas generadas.

El proceso incluye tres etapas principales: ajuste supervisado inicial con ejemplos humanos, entrenamiento de un modelo de recompensa basado en preferencias humanas, y ajuste por refuerzo utilizando este modelo de recompensa. Como resultado, los modelos *Instruct* son significativamente mejores para comprender las intenciones detrás de un *prompt*, evitar contenido inapropiado y generar respuestas más útiles.

### 2.4.3. Comparación y Diferencias Clave

La diferencia principal entre los modelos *Text* e *Instruct* radica en la forma en que procesan y entienden las instrucciones humanas. Mientras que los modelos *Text* confían únicamente en patrones estadísticos, los modelos *Instruct* integran un aprendizaje adicional basado en RLHF, comentado en la sección anterior, lo que les permite adaptarse mejor a las intenciones del usuario y evitar errores evidentes.

En el anexo A se pueden visualizar varios ejemplos donde se ponen a prueba ambos tipos de modelos para entender mejor su comportamiento. Centrándonos en el primero de ellos, en el ejemplo A.1 se realiza una inferencia con el mismo *prompt* a los modelos *LLaMA 3.2 instruct* y *LLaMA 3.2 text* (precisamente *llama3.2:3b-instruct-q8* y *llama3.2:3b-text-q8*) [19]. El *prompt* planteado: “**¿Quién fue la primera persona en caminar en Marte?**” es un caso diseñado para evaluar tanto la precisión como la capacidad de razonamiento de los modelos.

El modelo *Instruct* identifica correctamente que nadie ha caminado en Marte y aporta detalles contextuales verificados, como las misiones robóticas que han explorado el planeta y las fechas asociadas. El modelo *Text*, por su parte, fabrica una narrativa elaborada pero completamente ficticia, introduciendo errores graves como atribuir logros inexistentes a Ed Dwight y mencionando eventos que nunca ocurrieron.

Siguiendo con el anexo A, en el ejemplo A.2 se pone a prueba la creatividad de ambos modelos, indicándoles que escriban un poema con unas pautas dadas. La comparación entre las respuestas de los modelos *Instruct* y *Text* revela diferencias claras en su comportamiento y alineación con las instrucciones de este *prompt*. El modelo *Instruct* ofrece un poema estructurado que cumple con lo solicitado, utilizando un lenguaje simbólico y referencias a la temática del dragón y el castillo medieval. Su capacidad para adherirse a las instrucciones muestra cómo prioriza la precisión y la relevancia. Los versos generan una imagen coherente, haciendo uso de metáforas y evocando un tono épico y reflexivo. Aunque el poema es correcto, su creatividad es moderada, ya que se enfoca principalmente en cumplir el formato establecido por el

prompt.

Por otro lado, el modelo *Text* genera un relato narrativo extenso en lugar de un poema, desviándose del formato solicitado. Sin embargo, su salida destaca por la riqueza en detalles y la construcción de una historia completa, con personajes definidos y una trama que mezcla elementos medievales y fantásticos. Esta narrativa, aunque creativa y compleja, ignora la instrucción específica, lo que indica que este modelo tiende a priorizar la generación de texto libre sobre la obediencia a parámetros estrictos. Estas diferencias hacen que los modelos *Instruct* sean ideales para aplicaciones donde la precisión y la confianza son fundamentales, como en entornos educativos o científicos. En cambio, los modelos *Text* pueden ser útiles en tareas creativas o en situaciones donde la veracidad no sea una prioridad estricta.

Podemos concluir con que los modelos *Instruct* sobresalen en su capacidad para alinearse con las expectativas humanas gracias al ajuste específico logrado mediante RLHF. En contraste, los modelos *Text* generan contenido que no siempre es preciso ni responde de manera directa a las instrucciones. Esta diferencia subraya la importancia de las técnicas de ajuste para guiar a los modelos de lenguaje hacia respuestas más útiles y confiables. Mientras que los modelos *Instruct* están optimizados para seguir instrucciones de manera precisa, los modelos *Text* destacan por su creatividad y su capacidad para generar texto más libre. Esto demuestra que la elección del modelo debe basarse en el propósito: tareas específicas y estructuradas son mejor resueltas por *Instruct*, mientras que *Text* es ideal para explorar ideas abiertas e imaginativas.

## 2.5. Modelos de lenguaje abarcados

En última instancia, en cuanto a LLMs se refiere, es esencial analizar los modelos de lenguaje más relevantes que han marcado un avance significativo en el campo del procesamiento de lenguaje natural, especialmente este último año 2024. Tras explorar los fundamentos teóricos de los Transformers y los modelos de lenguaje, este apartado se centra en presentar una selección de LLMs destacados que equilibran potencia y eficiencia, adaptándose a los recursos computacionales establecidos para este trabajo.

La descripción de estos modelos busca no solo resaltar sus características principales, sino también comprender cómo los avances en su arquitectura, técnicas de entrenamiento y diseño han contribuido a sus capacidades actuales. Cada modelo seleccionado representa un enfoque único dentro del panorama de los LLMs.

El objetivo es ofrecer una visión general de sus contribuciones y limitaciones en relación con los objetivos del proyecto. La selección no se basa únicamente en su capacidad, sino en su adecuación a las necesidades específicas, lo que incluye factores

como su alineación con tareas concretas, su manejo eficiente de recursos y la diversidad de datos empleados durante su desarrollo. Este análisis establece una base sólida para contextualizar su papel dentro del estado actual de la investigación en modelos de lenguaje y su relevancia para las aplicaciones prácticas exploradas en este trabajo.

Los modelos de lenguaje seleccionados en este contexto para su análisis y aplicación en este proyecto son cuantificaciones de los siguientes modelos base: **LLaMA 3.2** [19], **Vicuna** [20], **Gemma 2** [21], **Mistral small** [22] y **Qwen 2.5** [23]. Para poder entrar en detalle en cada uno de los modelos mencionados, se adjunta el anexo E, donde se abarca información de sus orígenes, cómo han sido entrenados, ejemplos, etc.



# Capítulo 3

## Implementación

El sistema desarrollado tiene como objetivo automatizar el proceso de creación de tweets basados en datos de audiencias televisivas, reduciendo al mínimo la intervención humana en tareas repetitivas. Este diseño modular no solo permite su adaptación específica para la generación de tweets, sino que está concebido para ser escalable y reutilizable en otros contextos de exportación de datos relacionados con audiencias. De este modo, el sistema puede ser configurado para producir diferentes tipos de informes o entregables dependiendo de los requisitos del cliente, donde la única diferencia radica en el formato y el propósito de la exportación.

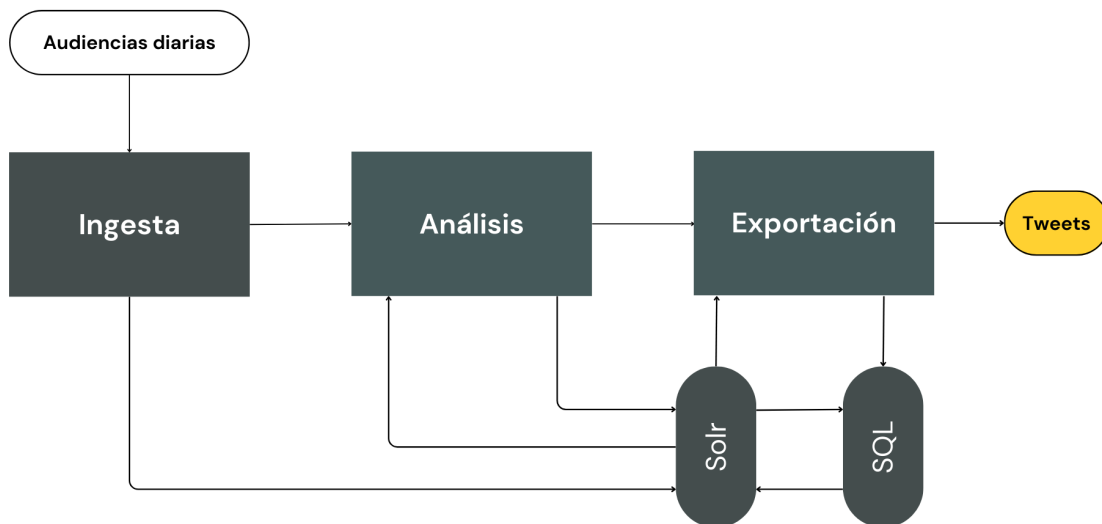


Figura 3.1: Esquema general del sistema de generación automática de Tweets

En el caso de este proyecto, el sistema ha sido adaptado para centrarse en la exportación orientada a redes sociales, con un énfasis particular en la creación de tweets automatizados. Este enfoque adapta el flujo de trabajo general a las exigencias

específicas de las plataformas sociales, como la limitación en el número de caracteres y el uso estratégico de elementos como hashtags, menciones y textos atractivos para captar la atención de la audiencia. La figura 3.1 ilustra el esquema general del sistema, destacando su estructura modular y los flujos de interacción entre los componentes principales.

El sistema está estructurado en tres grandes módulos principales: **ingesta**, **análisis** y **exportación**, los cuales interactúan con dos bases de datos principales, **Solr** y **SQL**. Este diseño asegura que el sistema sea escalable y adaptable a diferentes flujos de trabajo, no solo para redes sociales, sino también para otras plataformas o tipos de exportación. Sin embargo, para el caso específico de este proyecto, el énfasis se coloca en la parte final del flujo, la exportación, donde se realiza la adaptación y optimización para generar mensajes atractivos y concisos, ajustados al formato requerido por las redes sociales.

En los apartados siguientes, se describirá en detalle cada uno de los módulos del sistema, destacando cómo interactúan entre sí y cómo se adaptan específicamente al caso de uso centrado en la creación automática de tweets.

### 3.1. Ingesta de datos

La ingesta de datos constituye el primer módulo del sistema y se encarga de recibir, procesar e indexar la información de audiencias televisivas necesaria para las etapas posteriores del flujo de trabajo. Este proceso se basa en datos proporcionados por nuestro cliente a través de **Kantar Media**, una reconocida empresa de análisis de datos que ofrece servicios de medición de audiencias en televisión, radio y plataformas digitales. Kantar es ampliamente valorada en la industria por su capacidad para capturar, procesar y reportar datos de consumo de medios, lo que la convierte en una fuente confiable para este tipo de proyectos.

Los datos de Kantar Media son entregados al sistema por el cliente en formato texto o Excel a través de carpetas calientes, las cuales actúan como un canal de comunicación automatizado con nuestro sistema. Estas carpetas contienen los datos de audiencia del día anterior, organizados de manera estructurada y estandarizada, lo que permite su procesamiento automatizado. Los archivos suelen incluir métricas clave, como la cuota de audiencia, la cantidad de espectadores únicos, la audiencia media, entre otras, junto con información contextual como el programa asociado, la franja horaria y el canal de transmisión, según podemos observar en el ejemplo de la imagen 3.2. También se suele incluir el minutado del día anterior de todos los programas de todos los canales, información muy útil para su posterior análisis minuto a minuto, pudiendo detectar y

explicar anomalías en momentos concretos de un programa. A día de hoy, esto último sigue en desarrollo, por lo que este proyecto no lo va a contemplar.

#	A	B	C	D	E	F	G	H	I	J	K	L	M
1	[PROGRAMAS] (ESPAÑA[+2] TOTAL)[lineal]TSD)												
2								Targeto <=>					
3	Título	Descripción	Cadena	Fecha	Día semana	Hora de Inicio	Hora de Fin	Género	Cuota	AM(000)	MAA(000)	ACaPL	FdM
4	FELIZ 2019		La1	01/01/2019	Martes	06:32:12	11:09:54	ENTRETENIMIENTO	13,7	224	2630	3,4	8,5
5	CONCIERTO DE AÑO NUEVO		La1	01/01/2019	Martes	11:15:06	12:15:09	MUSICA	27,9	1630	3002	5,3	54,3
6	CONCIERTO DE AÑO NUEVO		La1	01/01/2019	Martes	12:15:39	13:48:05	MUSICA	31,3	2429	3907	12	62,2
7	INCREDIBLE 2019		La1	01/01/2019	Martes	13:48:55	15:00:04	ENTRETENIMIENTO	15,5	1463	3092	5,7	37,3
8	TELEDIARIO 1		La1	01/01/2019	Martes	15:05:09	15:52:09	INFORMACIÓN	17,1	1927	3278	4,9	58,8
9	SESION DE TARDE	GREASE	La1	01/01/2019	Martes	16:01:32	17:47:01	FICCION	13	1584	4301	9	36,8
10	SESION DE TARDE 2	PRINCESA POR SORPRESA	La1	01/01/2019	Martes	17:47:01	19:32:27	FICCION	12,5	1671	4824	9,5	36,1
11	SESION DE TARDE 3	M ABUELA ES LA MEJOR	La1	01/01/2019	Martes	19:32:27	21:00:06	FICCION	12,9	1943	4616	9,2	42,1
12	TELEDIARIO 2		La1	01/01/2019	Martes	21:04:47	22:35:57	INFORMACIÓN	15,3	2947	5988	8,7	44,2

Figura 3.2: Ejemplo de archivo de audiencias del cliente

Una vez que los archivos son detectados en las carpetas calientes, el sistema ejecuta un proceso de transformación que adapta la información recibida a una clase de datos común. Esta clase de datos actúa como una estructura intermedia que homogeniza el formato de la información, facilitando su manipulación en las etapas posteriores. Esta estandarización no solo permite un manejo uniforme de los datos, sino que también simplifica la integración con los demás módulos del sistema.

Posteriormente, los datos transformados son indexados automáticamente en **Solr**, un motor de búsqueda de alto rendimiento diseñado para manejar grandes volúmenes de datos textuales. Solr permite realizar búsquedas rápidas y consultas complejas, lo que resulta crucial para la etapa de análisis. La indexación en Solr incluye no solo los valores principales de las métricas de audiencia, sino también metadatos asociados que permiten realizar consultas avanzadas en función de distintos criterios, como programas, horarios o cadenas televisivas.

El uso de Solr en esta etapa proporciona al sistema una capacidad robusta de organización y recuperación de información. Solr utiliza un modelo de datos basado en documentos, lo que permite almacenar los registros de audiencia de manera eficiente y recuperar resultados precisos en tiempo real. Además, su arquitectura escalable asegura que el sistema pueda manejar el crecimiento en el volumen de datos sin comprometer el rendimiento. Esto es importante, ya que contiene datos de audiencias proporcionados por nuestro cliente desde los comienzos de la TDT, por lo que es imprescindible usar Solr para manejar esta enorme cantidad de datos de forma ágil.

Una vez completada la indexación, los datos quedan listos para ser utilizados en la etapa de análisis, donde se aplican procesos de categorización, filtrado y enriquecimiento para preparar la información de cara a la exportación. Este flujo de ingesta automatizado asegura que la información esté siempre actualizada y lista para ser procesada, estableciendo una base sólida para las operaciones posteriores del sistema.

## 3.2. Análisis

El módulo de análisis constituye el núcleo del sistema al encargarse de procesar los datos de audiencia previamente indexados para transformarlos en información estructurada y significativa. Los datos obtenidos del día anterior son almacenados en una base de datos estructurada y se recuperan utilizando Solr. Este proceso permite identificar las métricas más relevantes, como audiencias medias, cuotas de pantalla y espectadores únicos, consolidándolas en un formato común para facilitar su manipulación posterior.

Una parte destacada del análisis es la integración de herramientas avanzadas para la detección de anomalías en los minutados de los programas. Este componente, aún en desarrollo, tiene el objetivo de identificar cambios inesperados en la audiencia, como picos o caídas abruptas, y asociarlos con eventos relevantes, tales como noticias de última hora o momentos clave en las emisiones. Aunque no se aplica actualmente en la generación de tweets, esta funcionalidad abre nuevas posibilidades para enriquecer los análisis en el futuro y proporcionar información más detallada.

En el contexto específico de la generación de tweets, el sistema se centra en la recuperación de datos de audiencia que cumplan con criterios predefinidos. Dichos criterios incluyen, entre otros, franjas horarias específicas, programas relevantes y métricas clave que reflejen el desempeño de las emisiones. Estos datos, procesados y filtrados, se estructuran para convertirse en cuerpos de tweets en potencia, diseñados para ser claros y directos en su comunicación. Por ejemplo, un tweet puede resaltar la cuota de pantalla alcanzada por un programa, el número de espectadores únicos y otras métricas de interés, acompañado de etiquetas y menciones relevantes.

La selección de los datos de audiencia se basa en un conjunto de criterios específicos que determinan su relevancia. Entre estos, destacan los programas que superan significativamente la media de la cadena, los más vistos del día o de una cadena específica, y aquellos emitidos en franjas clave, como el prime time. También se priorizan los informativos destacados, eventos deportivos importantes y programas que logran máximos históricos o liderazgos en targets específicos.

Todos estos criterios específicos para la generación de tweets se pueden ver reflejados en la captura de la plataforma de EtiqMedia de la imagen 3.3, donde se puede apreciar el nivel de personalización de los análisis. Para el caso de este proyecto, se definirán todos estos tipos de análisis para este cliente en específico.

En la captura 3.4 se pueden observar una serie de análisis ejecutados dependiendo de los resultados que queramos evaluar. En general, se muestran análisis que detectan máximos, mínimos y anomalías.

Figura 3.3: Plataforma de EtIQMedia: personalización de análisis

Además, los análisis permiten que los datos procesados puedan ser exportados en distintos formatos según las necesidades del cliente. Por ejemplo, se generan informes de minutados personalizados para el cliente Atresmedia, los cuales se adaptan a las especificaciones requeridas para el seguimiento y análisis de las audiencias. En el caso de este proyecto, la exportación se orienta principalmente a la creación de contenido para redes sociales, principalmente tweets automatizados que reflejen de manera precisa y eficiente la relevancia de los datos de audiencia. Este enfoque optimiza los recursos y reduce el esfuerzo manual, manteniendo la coherencia y el valor informativo en las publicaciones.

### 3.3. Exportación

El módulo de **exportación** constituye el eje central sobre el cual se fundamenta este proyecto y representa la mayor diferenciación con respecto a otras aplicaciones del sistema base. Su propósito es transformar los datos de audiencia relevantes, previamente procesados y analizados, en productos finales que sean útiles para contenido en redes sociales. Como se observa en la figura 3.5, el flujo completo de exportación integra varias etapas clave que garantizan la generación automatizada de tweets acompañados de imágenes.

El proceso comienza con el histórico de audiencias y datos del día anterior almacenados en Solr. Estos datos se filtran y recuperan de acuerdo con criterios

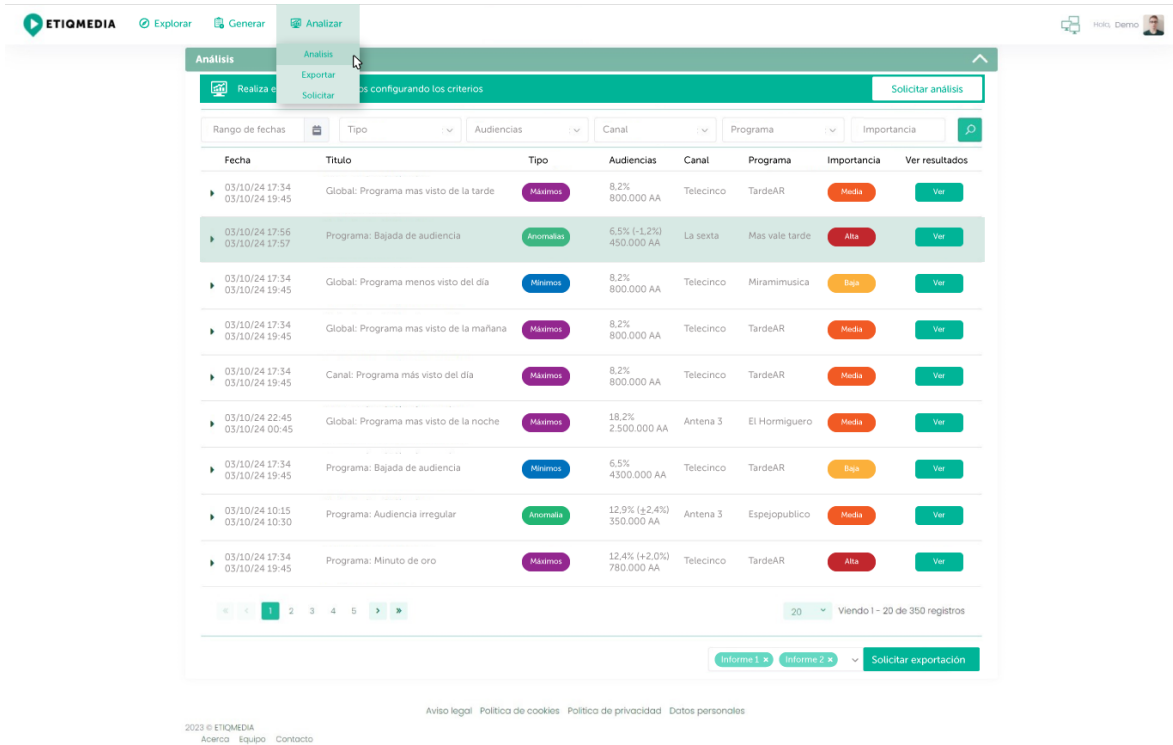


Figura 3.4: Plataforma de EtIQMedia: análisis ejecutados

predefinidos que permiten identificar las métricas más destacadas sobre programas concretos, como cuotas de pantalla, audiencia media y espectadores únicos. Al utilizar toda la base de datos disponible, únicamente llegan al módulo de exportación los datos de audiencia relevantes del día anterior comparados con el histórico, además de datos sobre programas concretos que nuestro cliente siempre twittea porque son clientes suyos.

Desde aquí, el sistema se apoya en un modelo de lenguaje que se encarga de dar forma a los datos procesados mediante *Python* y la librería *Ollama*, estructurándolos en textos que puedan servir como potenciales cuerpos de tweets. Este paso es crucial, ya que el modelo seleccionado no solo genera contenido textual coherente, sino que también asegura que los mensajes mantengan un estilo alineado con los objetivos comunicativos del cliente. En el siguiente capítulo, se profundizará en el análisis de los modelos de lenguaje utilizados, evaluando sus capacidades para cumplir con las exigencias del proyecto en términos de calidad, consistencia y precisión con los requisitos del cliente.

De manera complementaria, el módulo de **crawling de imágenes** tiene un papel importante en el enriquecimiento visual de los tweets generados. Este componente realiza búsquedas automatizadas en la web, desarrollado en Python y adaptado a C#, utilizando como parámetros las palabras clave del programa, el título, la cadena y la fecha de emisión, además de un filtro temporal que prioriza imágenes del día anterior.

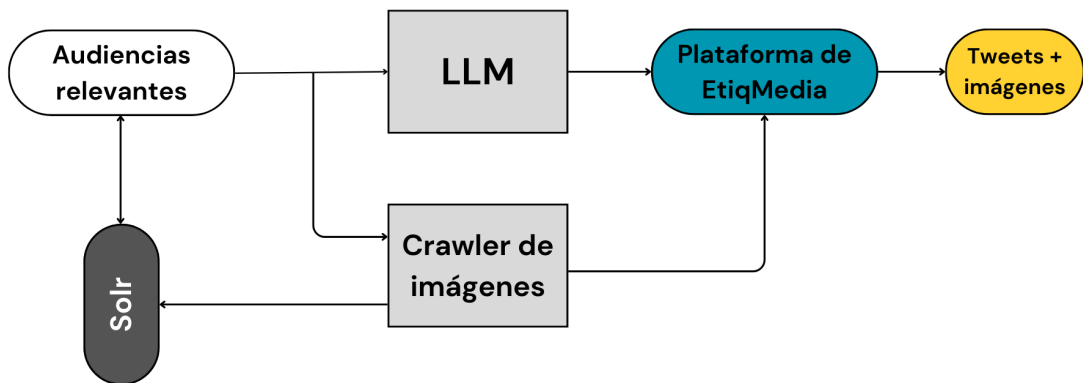


Figura 3.5: Desglose del módulo de exportación de tweets

Este enfoque garantiza que las imágenes propuestas sean acordes a la fecha del dato de audiencia y contextualmente relevantes. En caso de que las imágenes obtenidas no se ajusten a las necesidades del cliente, el sistema contempla una solución mediante imágenes predeterminadas a nivel programa, las cuales constan de tres imágenes genéricas obtenidas por el crawler configurables por si el cliente usa sus imágenes por defecto. Dichas imágenes obtenidas son indexadas en Solr con el resto de datos. Destacar que este módulo aún sigue en desarrollo a día de hoy por temas de backend de la empresa.

En primera instancia se intentó realizar un crawling a través de la api de Twitter para buscar la cuenta oficial del programa y/o cadena y descargar las imágenes más recientes, de la misma forma que las descarga manualmente de forma diaria nuestro cliente. Sin embargo, esto no ha sido viable, ya que, desde que Twitter fue adquirido por Elon Musk, la api para desarrolladores cuesta 50.000 dólares mensuales, presupuesto fuera de los propósitos de este proyecto.

Tras la adquisición del cuerpo del tweet y las imágenes asociadas, la *plataforma de EtiqMedia* se integra en la etapa final del proceso, ofreciendo al cliente un entorno interactivo donde los tweets y las imágenes pueden ser revisados y editados. Este espacio combina las ventajas de la automatización con la flexibilidad de realizar ajustes manuales, asegurando que el resultado final sea completamente personalizado y listo para su publicación en redes sociales. Esta interacción entre los diferentes módulos, desde la recuperación de datos hasta la validación final en la plataforma, refleja un

diseño modular y escalable que puede adaptarse a distintas aplicaciones, aunque en este proyecto se haya basado en la generación de tweets.

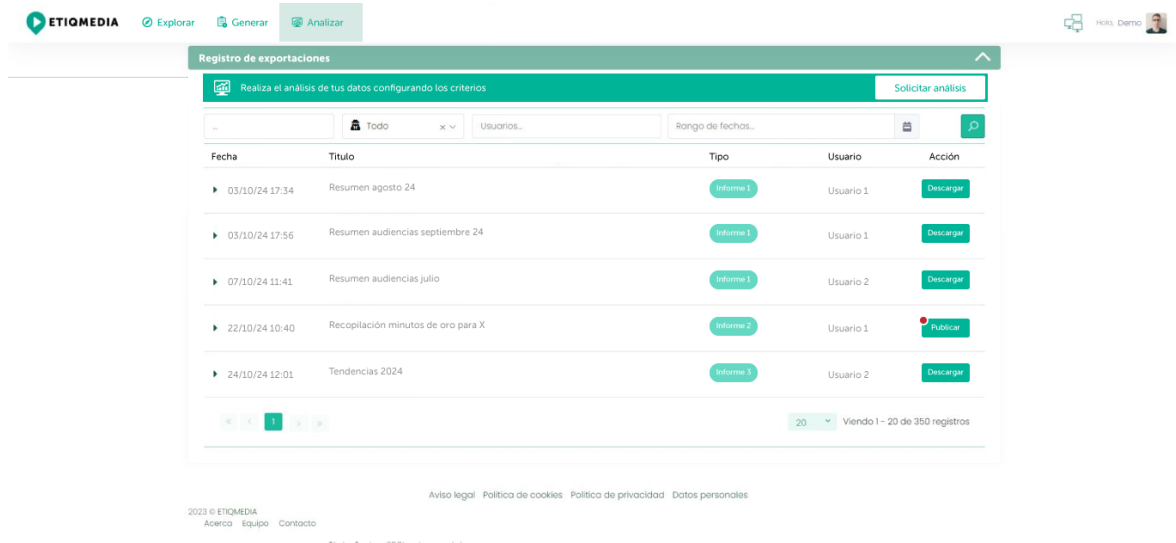


Figura 3.6: Plataforma de EtiqMedia: registro de exportaciones

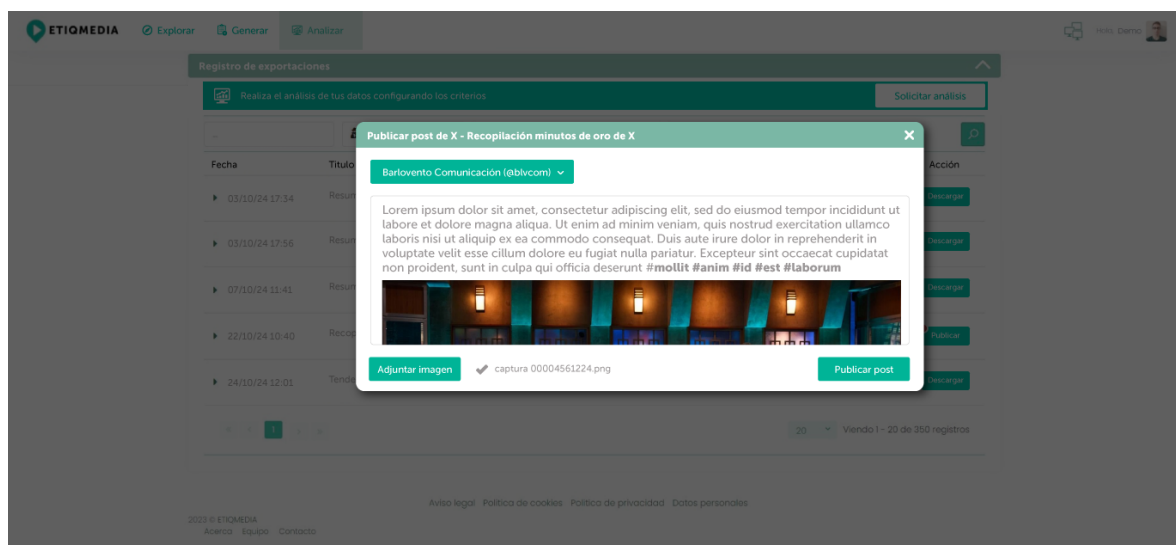


Figura 3.7: Plataforma de EtiqMedia: exportación de tweet

En la imagen 3.6 se muestra una sección de exportaciones de la plataforma de EtiqMedia. Aquí se visualiza el proceso de exportación tras un análisis realizado, con la posibilidad de descargar informes que contienen resúmenes y recopilaciones de los datos analizados. En relación con este proyecto, la imagen 3.7 destaca la funcionalidad de exportación de tweets editables, donde se dispone de un cuadro de texto que permite modificar el tweet generado automáticamente para realizar ajustes o mejoras antes de su publicación.

Es importante destacar que este módulo de exportación es el núcleo del trabajo desarrollado, ya que no solo se diferencia del resto de las aplicaciones del sistema base,

sino que también incorpora elementos innovadores como el uso de LLMs y herramientas de búsqueda visual para optimizar la automatización. En los capítulos siguientes se analizarán con detalle las capacidades de los modelos de lenguaje comentados en la sección anterior, evaluando aspectos como su desempeño, creatividad y el impacto en la creación automatizada de contenido para redes sociales. Esta evaluación permitirá medir el éxito del sistema y definir posibles líneas de mejora futuras.



# Capítulo 4

## Resultados

En este capítulo se presentan los resultados obtenidos al probar diferentes modelos de lenguaje para la generación automática de tweets basados en datos de audiencia reales, todos ellos extraídos de Solr. Además, se incluyeron eventos no reales en los datos de prueba con el objetivo de evaluar la consistencia de los modelos ante escenarios variados y detectar posibles alucinaciones o errores en las respuestas. Las pruebas se llevaron a cabo con modelos en capítulos anteriores: **LLaMA 3.2**, **Mistral small**, **Gemma 2**, **Vicuna** y **Qwen 2.5**, utilizando una GPU **Nvidia GeForce RTX 4090** de 24 GB. Este modelo de tarjeta gráfica, la más potente disponible en la empresa, permitió abarcar una gama más amplia de modelos, destacando que en el cliente se cuenta con una GPU de 16 GB, la cual, en un principio, se estimó como suficiente para el despliegue del proyecto.

A continuación, los resultados se dividen en dos subsecciones: los resultados **objetivos**, que incluyen métricas cuantitativas como tiempos de respuesta, consumo de GPU y número de intentos necesarios para obtener una respuesta válida; y los resultados **subjetivos**, basados en una evaluación manual por parte de usuarios humanos, quienes calificaron la calidad de los tweets generados bajo varios criterios. Asimismo, los resultados se subdividirán según el tamaño de los modelos y su capacidad para ejecutarse en GPUs, donde diferenciamos GPUs de altas prestaciones (24 GB) y de medias prestaciones (16 GB, la del cliente).

### 4.1. Resultados objetivos

En esta sección se presentan las pruebas realizadas para evaluar el desempeño de los modelos de lenguaje analizados en secciones anteriores frente a la creación de tweets automáticos. Dicha batería de modelos va a ser puesta a prueba con un prompt base y datos de audiencias relevantes extraídos de Solr.

Los LLMs que se van a evaluar abarcan todos los mencionados en capítulos

anteriores, además de tener en cuenta distintos tamaños dentro del mismo modelo y distintas cuantizaciones, dando mucho margen de investigación. El objetivo de estas pruebas se resume en la elección objetiva de un LLM que cumpla una serie de requisitos para crear tweets de forma rápida y concisa.

Los prompts utilizados se encuentran en el anexo C, donde se puede visualizar tanto el prompt base como los prompts específicos utilizados para inferir el conjunto de LLMs analizados en el proyecto. En los ejemplos dados se pueden encontrar todos los datos de audiencia que deben ser incluidos en cada tweet particularmente.

A continuación se muestran dos tablas (4.1 y 4.2) donde podremos diferenciar, dependiendo del consumo de GPU entre altas o medias prestaciones, las métricas obtenidas de cada modelo, incluyendo tiempos de respuesta, consumo de GPU y número de intentos necesarios (limitado a 10) para producir un tweet válido tras un control de errores. Este control de errores mencionado asegura que todos los datos de audiencia están correctamente representados en la salida, comprueba que haya al menos dos hashtags, dos emojis, se asegura de que la respuesta no sea ni muy corta ni muy larga y, por último, detecta que no aparezcan palabras prohibidas (p. ej.: "lo siento", "hola", "soy un modelo de lenguaje", "felicidades", etc.).

Tabla 4.1: Resultados para modelos que requieren GPUs de altas prestaciones (hasta 24 GB)

Modelo	Tweet	Intentos	Tiempo por respuesta válida (s)	Consumo GPU (MB)
Gemma2:27B	Aragón TV	1	35.05	19651.81
	Antena 3	1	1.84	
	Cuatro	1	1.66	
	Telecinco	1	1.59	
	Movistar+	1	1.93	
Gemma2:27B-instruct-q5_K_M	Aragón TV	1	35.78	20977.81
	Antena 3	1	2.11	
	Cuatro	1	1.75	
	Telecinco	1	1.99	
	Movistar+	1	2.14	
Qwen2.5:32B	Aragón TV	2	17.81	22506.88
	Antena 3	2	4.55	
	Cuatro	1	1.95	
	Telecinco	1	2.37	
	Movistar+	1	2.32	

El análisis de las dos tablas proporciona una visión clara de las capacidades de los modelos clasificados según los requisitos de hardware. En el grupo de modelos que requieren GPUs de altas prestaciones, destacan por su capacidad de generar resultados consistentes con tiempos de respuesta razonables a pesar de manejar configuraciones complejas y un mayor consumo de GPU. Como se puede observar en la tabla 4.1, el primer tweet de cada modelo presenta tiempos muy altos; esto se debe a que se incluye también la carga del modelo en la memoria gráfica al ser la primera inferencia. El resto de inferencias parten de tener ya cargado el modelo en memoria, por lo que son mucho

Tabla 4.2: Resultados para modelos que requieren GPUs de prestaciones medias (hasta 16 GB)

Modelo	Tweet	Intentos	Tiempo por respuesta válida (s)	Consumo GPU (MB)
Mistral-small	Aragón TV	+10	-	9751.44
	Antena 3	3	4.66	15735.81
	Cuatro	+10	-	
	Telecinco	1	1.62	
	Movistar+	1	1.49	
LLaMA3.2:latest	Aragón TV	+10	-	3549.94
	Antena 3	7	2.98	
	Cuatro	10	3.60	
	Telecinco	+10	-	
	Movistar+	2	0.90	
Vicuna:13B	Aragón TV	2	9.10	15215.81
	Antena 3	1	1.36	
	Cuatro	2	2.98	
	Telecinco	7	8.46	
	Movistar+	1	1.19	
Vicuna:13B-v1.5-q4_0	Aragón TV	3	18.69	15263.00
	Antena 3	4	5.63	15264.44
	Cuatro	3	4.13	15254.19
	Telecinco	8	12.12	
	Movistar+	6	10.31	
Gemma2:9B	Aragón TV	+10	-	9751.44
	Antena 3	7	4.84	7568.88
	Cuatro	6	3.89	
	Telecinco	+10	-	
	Movistar+	2	1.53	
Qwen2.5:7B	Aragón TV	+10	-	6271.81
	Antena 3	+10	-	
	Cuatro	+10	-	
	Telecinco	+10	-	
	Movistar+	+10	-	
Qwen2.5:14B	Aragón TV	1	14.40	11599.81
	Antena 3	2	2.19	
	Cuatro	2	1.99	
	Telecinco	1	1.15	
	Movistar+	1	1.28	
Qwen2.5:14B-instruct-q5_K_M	Aragón TV	1	14.11	12952.88
	Antena 3	1	1.27	
	Cuatro	1	1.45	
	Telecinco	1	1.28	
	Movistar+	1	1.36	

más rápidas. No se han testado más cuantizaciones porque las de 5 bits (q5) requerían más de 24 GB y, al tener que usar CPU si las supera, ralentizaba mucho las respuestas.

A pesar de ser modelos lentos y pesados en comparación con los de prestaciones medias, en todos los casos ofrecen tiempos estables y una precisión muy buena, ya que ninguno de los modelos ha necesitado más de dos intentos para crear un tweet válido.

En el grupo de modelos de prestaciones medias de la tabla 4.2, se observan principalmente tiempos de respuesta más cortos y un menor consumo de GPU, pero esto viene acompañado de una reducción en la precisión. La mayoría de los modelos en este grupo requieren significativamente más intentos para generar un tweet válido. Por ejemplo, los modelos *Mistral-small*, *LLaMA3.2*, *Qwen2.5:7B* y *Gemma2:9B* presentan un consumo de GPU reducido y tiempos de respuesta rápidos, pero frecuentemente alcanzan el límite de intentos permitidos (más de 10) sin generar una respuesta válida.

Es especialmente relevante el caso del modelo **Qwen2.5:7B**, que no logró generar ningún tweet válido en 10 intentos. Estos resultados destacan que los modelos con menor cantidad de parámetros tienden a ofrecer un desempeño significativamente inferior y muestran una menor capacidad para seguir las instrucciones de manera efectiva.

Por otro lado, modelos como **las dos versiones de Vicuna:13B-v1.5-q4** y **Qwen2.5:14B** logran un equilibrio destacado entre consumo de memoria, velocidad de inferencia y número de intentos necesarios. En particular, **Qwen2.5:14B-instruct-q5\_K\_M** sobresale al generar todos los tweets correctamente sin requerir intentos adicionales. Además, los cuatro modelos mencionados se ajustan perfectamente a la capacidad de la tarjeta gráfica instalada en el cliente, de 16 GB. Esta combinación de eficiencia y calidad los convierte en opciones sólidas para entornos que demandan un balance entre precisión y uso eficiente de recursos computacionales.

Tabla 4.3: Resumen de resultados por modelo

Modelo	Media de Intentos	Tiempo Promedio por Inferencia (s)	Consumo Promedio de GPU (MB)	Tiempo Medio por Respuesta Válida (s)
Gemma2:27B	1.00	8.41	19.651.81	8.41
Gemma2:27B-instruct-q5_K_M	1.00	8.35	20.977.81	8.35
Qwen2.5:32B	1.40	5.20	22.506.88	7.28
Vicuna:13B	2.60	4.62	15.215.81	16.69
Vicuna:13B-v1.5-q4_0	4.80	10.18	15.254.19	51.74
Qwen2.5:14B	1.40	4.20	11.599.81	5.88
Qwen2.5:14B-instruct-q5_K_M	1.00	3.89	12.952.88	3.89

La tabla 4.3 permite analizar de forma global el desempeño de los modelos evaluados, destacando sus diferencias en términos del tiempo promedio para dar una respuesta válida y su consumo. En el rango de modelos de altas prestaciones, los resultados muestran que los tres modelos evaluados logran ser mucho más precisos, a costa de aumentar el tiempo medio por inferencia. Precisamente, **Qwen2.5:32B** equilibra adecuadamente la velocidad, el consumo de memoria y los intentos necesarios en la prueba realizada, obteniendo ligeramente mejores tiempos que Gemma2.

En el rango de prestaciones medias, el comportamiento de los modelos es más diverso. Mientras que ambos modelos de Vicuna:13B necesitan un número considerable de intentos para generar respuestas válidas, especialmente la cuantización de cuatro bits, lo que incrementa significativamente su tiempo medio por respuesta, **Qwen2.5:14B-instruct-q5\_K\_M** sobresale al obtener resultados correctos en un único intento y mantener un consumo moderado de GPU, obteniendo, en definitiva, el mejor tiempo medio en generar una respuesta válida. Este modelo se posiciona como una opción confiable y eficiente en este rango. Por otro lado, Vicuna:13B y su versión cuantizada logran resultados válidos con menor frecuencia y tiempos promedio más elevados, lo que los sitúa en desventaja frente a opciones más rápidas y consistentes.

En conclusión, para sistemas con GPUs de altas prestaciones, los modelos destacados ofrecen una alta precisión y eficiencia, aunque con diferencias significativas en velocidad y consumo frente a sus versiones más ligeras. En sistemas con GPUs de prestaciones medias, **Qwen2.5:14B-instruct-q5\_K\_M** se presenta como una alternativa sólida, equilibrando tiempo y recursos de manera efectiva. Esto demuestra que la elección del modelo debe considerar tanto la capacidad del hardware como las necesidades específicas del entorno en el que será implementado.

Para consultar los **resultados detallados de los tweets** generados automáticamente, se puede acceder al **anexo D adjunto**, donde se presentan cinco ejemplos de tweets por cada modelo evaluado que ha logrado generar respuestas sin errores. En este anexo, se incluyen los textos completos de cada tweet junto con su formato final, asegurando una visión clara del desempeño de cada modelo en la generación de contenido adecuado.

## 4.2. Resultados subjetivos

Para evaluar la calidad de los tweets generados por los modelos, se ha diseñado un experimento basado en la percepción de usuarios con distintos niveles de conocimiento sobre modelos de lenguaje. Este análisis subjetivo se estructura en tres escenarios diferenciados, cada uno de los cuales cuenta con la participación de dos usuarios con niveles bajos, intermedios o altos en el ámbito de los LLMs. El objetivo principal es determinar en qué medida los tweets generados por los distintos modelos cumplen con las especificaciones establecidas por el cliente.

En cada escenario, los participantes deben seleccionar, de entre un conjunto de tweets generados por diferentes modelos para una misma publicación, los dos que consideren más adecuados según los criterios definidos. Estos criterios incluyen:

- Deben ser puramente informativos, sin incitar a ver el programa.
- Deben ser claros, concisos y con una redacción natural.
- Deben incluir al menos dos *hashtags* y dos emojis.

Una vez seleccionados los dos tweets más adecuados, cada participante procederá a evaluarlos en función de los siguientes aspectos, asignando puntuaciones en una escala del 1 al 5:

- **Naturalidad:** en qué medida el tweet parece redactado por un humano o generado automáticamente.

- **Concisión:** si la información es clara y se presenta de manera eficiente sin añadir detalles innecesarios.
- **Originalidad:** si el tweet es variado y diferenciado en comparación con los otros generados.
- **Nivel de alucinación:** si el modelo ha introducido información errónea o inexistente en el contenido.
- **Hashtags y emojis:** pertinencia y adecuación de estos elementos dentro del tweet.
- **Evaluación general:** valoración global del tweet considerando todos los aspectos anteriores.

En el anexo F se puede contemplar un ejemplo de una de las plantillas utilizadas para evaluar la selección de tweets automáticos. Esto se extrapola a los seis individuos, dos de ellos por cada nivel asignado de conocimientos de LLM. A modo de resumen, los tweets pertenecen a los modelos de lenguaje que mejores resultados obtuvieron en la sección anterior:

- **Tweet 1:** Qwen2.5 14B instruct q5 K M
- **Tweet 2:** Vicuna 13B v1.5 q4
- **Tweet 3:** Qwen2.5 32B
- **Tweet 4:** Gemma2 27B instruct q5 K M

#### 4.2.1. Escenario 1: usuarios con bajo nivel de conocimiento sobre LLMs

En este escenario, dos participantes con conocimientos limitados en modelos de lenguaje evaluaron cuatro tweets generados automáticamente. La evaluación tuvo como objetivo determinar cuáles de ellos se ajustaban mejor a las especificaciones del cliente, priorizando la claridad, la naturalidad y la inclusión de los elementos requeridos. Ambos usuarios seleccionaron los dos tweets que consideraron más adecuados y los puntuaron en diferentes aspectos. A continuación, se presentan los resultados obtenidos.

## Evaluación de los tweets

Los tweets seleccionados fueron evaluados en función de su naturalidad, concisión, originalidad, nivel de alucinación y adecuación en el uso de *hashtags* y emojis. La tabla 4.4 muestra las puntuaciones otorgadas por cada usuario, además de un resumen con la media de las evaluaciones para cada tweet.

Evaluador	Tweet	Naturalidad	Concisión	Originalidad	Nivel de alucinación	Hashtags y emojis	Evaluación general
Usuario 1	Tweet 3	4	4	4	1	4	4
	Tweet 4	4	5	4	1	3	4
Usuario 2	Tweet 1	4	5	4	3	4	4
	Tweet 3	4	4	4	1	2	3
Resumen	Tweet 1	4.0	5.0	4.0	3.0	4.0	4.0
	Tweet 3	4.0	4.0	4.0	1.0	3.0	3.5
	Tweet 4	4.0	4.5	4.0	1.0	3.0	4.0

Tabla 4.4: Evaluación de los tweets en el escenario 1

## Análisis de los resultados

Los resultados muestran que, según los usuarios con menor conocimiento en LLMs, el tweet 1 fue el más conciso, aunque también el que más alucinaciones presentó, en parte debido a la inclusión de *hashtags* en inglés. El tweet 3 obtuvo una puntuación baja en alucinación, lo que indica mayor precisión en los datos, pero su concisión fue menor en comparación con los otros tweets. Por otro lado, el tweet 4 logró un equilibrio destacable en todos los aspectos, con una puntuación general competitiva, alcanzando la mejor media junto con el tweet 1.

Estos resultados sugieren que la percepción de calidad de los tweets no solo está influenciada por la claridad y la coherencia del mensaje, sino también por detalles específicos como la selección de elementos visuales y la estructuración de la información. La menor alucinación en ciertos tweets refuerza la importancia de garantizar la fidelidad de los datos en los modelos de generación automática de contenido.

## Comentarios adicionales

El primer usuario valoró el tweet 3 y el 4 por el nivel de alucinación mínimo, según su punto de vista.

El segundo usuario comenta que en el tweet 1 está muy bien redactado, pero considera que los *hashtags* deberían estar en castellano. Respecto al tweet 3, concluyó en que es el mejor tweet según su punto de vista, y no le da la puntuación máxima por utilizar dos veces el mismo emoji.

## 4.2.2. Escenario 2: usuarios con nivel intermedio de conocimiento sobre LLMs

En este escenario, dos participantes con conocimientos intermedios en modelos de lenguaje evaluaron cuatro tweets generados automáticamente. A continuación, se presentan los resultados obtenidos.

### Evaluación de los tweets

Los tweets seleccionados fueron puntuados en función de su naturalidad, concisión, originalidad, nivel de alucinación y adecuación en el uso de *hashtags* y emojis. La tabla 4.5 resume las evaluaciones de cada usuario y el promedio final de cada tweet.

Evaluador	Tweet	Naturalidad	Concisión	Originalidad	Nivel de alucinación	Hashtags y emojis	Evaluación general
Usuario 1	Tweet 1	3	5	4	2	3	4
	Tweet 4	5	5	5	1	5	5
Usuario 2	Tweet 3	3	5	3	1	2	3
	Tweet 4	4	4	3	1	4	4
Resumen	Tweet 1	3.0	5.0	4.0	2.0	3.0	4.0
	Tweet 3	3.0	5.0	3.0	1.0	2.0	3.0
	Tweet 4	4.5	4.5	4.0	1.0	4.5	4.5

Tabla 4.5: Evaluación de los tweets en el escenario 2

### Análisis de los resultados

El análisis de los resultados muestra que el tweet 1 destacó en concisión, aunque algunos elementos de su redacción fueron considerados poco naturales. El tweet 3 tuvo un buen desempeño en concisión, pero recibió una menor puntuación en originalidad y en la elección de emojis y hashtags. El tweet 4 obtuvo la mejor valoración general, con una puntuación equilibrada en todos los aspectos evaluados.

Los participantes con conocimientos intermedios tendieron a valorar más la coherencia y la claridad del contenido, mientras que fueron más exigentes en cuanto a la originalidad y la naturalidad de los tweets. En este escenario, el tweet 4 fue el que logró una mejor adaptación al formato requerido, destacando tanto en su estructuración como en la selección de elementos visuales.

### Comentarios adicionales

El primer usuario valoró el tweet 1 por su estructura clara y concisa, aunque mencionó que algunos hashtags en inglés y el uso de ciertas construcciones gramaticales afectaban su naturalidad. Por su parte, el tweet 4 fue señalado como el mejor estructurado, con una excelente combinación de elementos visuales y sintácticos.

El segundo usuario también identificó deficiencias en los hashtags y emojis del tweet 1, considerando que algunos de ellos no reflejaban un estilo humano. Respecto al tweet

3, mencionó que su redacción podría mejorarse en términos de originalidad y cohesión.

### 4.2.3. Escenario 3: usuarios con alto nivel de conocimiento sobre LLMs

En este escenario, dos participantes con experiencia en modelos de lenguaje evaluaron los cuatro tweets generados automáticamente. Ninguno de ellos sabía de antemano a qué modelo correspondía cada tweet, lo que permitió obtener una evaluación más objetiva basada únicamente en la calidad del texto generado.

#### Evaluación de los tweets

Los tweets seleccionados fueron evaluados en función de su naturalidad, concisión, originalidad, nivel de alucinación y adecuación en el uso de *hashtags* y emojis. La tabla 4.6 resume las evaluaciones de cada usuario y el promedio final de cada tweet.

Evaluador	Tweet	Naturalidad	Concisión	Originalidad	Nivel de alucinación	Hashtags y emojis	Evaluación general
Usuario 1	Tweet 3	4	5	5	1	3	4
	Tweet 4	5	5	5	1	5	5
Usuario 2	Tweet 1	4	5	3	1	4	4
	Tweet 3	5	5	3	1	3	5
Resumen	Tweet 1	4.0	5.0	3.0	1.0	4.0	4.0
	Tweet 3	4.5	5.0	4.0	1.0	3.0	4.5
	Tweet 4	5.0	5.0	5.0	1.0	5.0	5.0

Tabla 4.6: Evaluación de los tweets en el escenario 3

#### Análisis de los resultados

El tweet 4 fue el mejor valorado en este escenario, obteniendo una puntuación sobresaliente en todos los criterios. Se destacó por su estructura clara, su lenguaje natural y la adecuación de los elementos visuales. El tweet 3 mostró una ligera ventaja en concisión, aunque se consideró menos original en su redacción. El tweet 1, aunque recibió buenas calificaciones en claridad, obtuvo una puntuación inferior en originalidad y creatividad, aunque mantuvo una valoración general aceptable.

Los evaluadores con alto nivel de conocimientos en LLMs pusieron un énfasis especial en la naturalidad del texto y en su diferenciación con respecto a otros mensajes similares. La percepción de calidad en este grupo estuvo más influenciada por la fluidez del lenguaje y el equilibrio en la selección de los elementos visuales, lo que refuerza la importancia de la coherencia y el tono en la generación automática de contenido.

#### Comentarios adicionales

El primer usuario destacó que el tweet 3 presentaba poca variedad en los *hashtags* y emojis, lo que restaba dinamismo al mensaje. También señaló que la redacción podía

mejorarse con el uso de proposiciones que aportaran mayor fluidez y naturalidad al texto. En contraste, el tweet 4 fue valorado positivamente por su claridad y estructura. Su contenido es directo, incorpora estadísticas clave y resalta el logro histórico con emojis adecuados que captan la atención. Además, la selección de hashtags facilita la búsqueda del contenido y refuerza su visibilidad, contribuyendo a la difusión del programa.

El segundo usuario identificó un problema en el tweet 1, indicando que el uso del emoji de dado no era el más acertado en el contexto del mensaje. También mencionó que en el tweet 3 se percibía una repetición de emojis que podría haberse evitado para mejorar la variedad visual del contenido.

#### 4.2.4. Resumen general y conclusiones

Tras la evaluación en los tres escenarios, se han consolidado los resultados en la tabla 4.7. Esto permite observar la tendencia general de desempeño de los tweets generados por los modelos.

Tweet	Naturalidad	Concisión	Originalidad	Nivel de alucinación	Hashtags y emojis	Evaluación general
Tweet 1 (Qwen2.5 14B q5 K M)	3.7	5.0	3.7	2.0	3.7	4.0
Tweet 3 (Qwen2.5 32B)	4.0	4.7	3.7	1.0	2.7	3.8
Tweet 4 (Gemma2 27B instruct q5 K M)	4.8	4.8	4.3	1.0	4.3	4.7

Tabla 4.7: Resultados consolidados de los tres escenarios.

Los resultados sugieren que el **tweet 4 redactado por Gemma2 27B q5** ha sido el más consistente en todos los escenarios, destacando en naturalidad, originalidad y adecuación de hashtags y emojis. El **tweet 3 de Qwen2.5 32B** se ha mantenido competitivo en todos los aspectos, excepto en los emojis, ya que usó dos repetidos. Por otro lado, el **tweet 1 de Qwen2.5 14B q5** se sitúa en segundo lugar, muy cerca del tweet 3, siendo el más conciso en media según todos los usuarios, pero con un ligero aumento de la alucinación. Por último, cabe mención al **tweet 2 de Vicuna 13B v1.5 q4**, el cual no ha sido elegido por ninguno de los usuarios. Esto se debe a su alto grado de alucinación, tanto en uno de los hashtags como en la última frase, la cual incita a ver dicho programa. Fue el primer descartado por todos los usuarios sin ninguna duda.

El nivel de conocimientos de los evaluadores parece haber influido en la percepción de calidad de los tweets. Los usuarios con menor experiencia tendieron a valorar más la claridad y la ausencia de errores, mientras que aquellos con conocimientos avanzados pusieron más énfasis en la originalidad y la naturalidad del texto. En este sentido, los participantes con mayor formación en LLMs mostraron una mayor sensibilidad a detalles estilísticos y lingüísticos.

### 4.3. Conclusiones tras los resultados objetivos y subjetivos

Los **resultados subjetivos** muestran que **Gemma2 27B q5** fue el modelo mejor valorado en la mayoría de los criterios. Este desempeño se refleja también en los resultados objetivos, con un tiempo de inferencia mayor que otros modelos, pero con una generación de texto más estructurada y fluida. En comparación, **Qwen2.5 32B** no logró superar a Gemma2 27B en evaluación subjetiva, pero sí en los **resultados objetivos**, principalmente en la velocidad de inferencia y, por lo tanto, obteniendo un menor tiempo en generar respuestas válidas. Este último se presenta como una alternativa sólida para tarjetas gráficas de 24 GB, pero, en el caso de generación de tweets, Gemma2 27B se sitúa ligeramente por encima.

En cuanto a los modelos de tamaño intermedio, **Qwen2.5 14B** mostró un equilibrio entre velocidad y calidad, siendo el más rápido en obtener tweets correctos y la segunda mejor opción en las evaluaciones subjetivas. Sus requisitos de hardware y su desempeño en generar tweets lo sitúan como el modelo de lenguaje más destacado en este ámbito, teniendo en cuenta que tiene un gasto de GPU menor a 16 GB.

En definitiva, la relación entre el tamaño del modelo y la percepción de calidad no sigue una tendencia lineal en la experiencia del usuario. Aunque los modelos más grandes pueden generar respuestas más complejas, la evaluación subjetiva indica que la estructura del texto y su adecuación al contexto tienen un mayor impacto en la valoración final. Los resultados de este proyecto muestran que, si bien los modelos que pueden ejecutarse en una GPU de 24 GB han ofrecido un mejor desempeño, el reciente **Qwen2.5 14B** presenta un equilibrio óptimo entre precisión y velocidad de inferencia. Dado que este modelo ofrece una calidad aceptable en la generación de tweets automáticos y se ajusta a la GPU instalada de 16 GB, se posiciona como una alternativa viable y eficiente sin necesidad de recurrir a modelos significativamente más costosos en términos computacionales.



# Capítulo 5

## Conclusiones y líneas futuras

### 5.1. Conclusiones generales

Tras la realización de este proyecto, se han extraído las siguientes conclusiones:

- Los LLMs pueden llegar a ser herramientas esenciales para ayudar a automatizar tareas humanas diarias, de forma que las empresas puedan llegar a segmentos de mercado que no eran rentables anteriormente.
- La evolución de los LLMs ha sido notable en los últimos años. La optimización en entrenamiento y arquitectura ha permitido que modelos más pequeños logren resultados comparables a los de mayor tamaño, lo que ofrece nuevas oportunidades para mejorar la eficiencia sin comprometer la calidad.
- La calidad del texto generado no depende únicamente del tamaño del modelo. Aunque los modelos más grandes pueden producir respuestas más complejas y ser bastante más concisos, la coherencia y la adaptación al contexto son factores más determinantes en la percepción del usuario.
- Los modelos de altas prestaciones han sido mejor valorados en naturalidad y originalidad, pero con un mayor costo computacional, mientras que los de prestaciones intermedias han demostrado ser una alternativa equilibrada en precisión y velocidad de inferencia, haciéndolo viable en entornos con limitaciones de hardware, como es el caso de este proyecto.
- La generación automática de tweets permite agilizar la producción de contenido sin perder coherencia ni estructura. No obstante, aspectos como la selección de *hashtags* y *emojis* ó alucinaciones aún requieren ajustes, al igual que la personalización del tono del mensaje según las necesidades del cliente.

## 5.2. Líneas futuras

El avance en el desarrollo de modelos de lenguaje seguirá marcando la evolución de este tipo de aplicaciones. Para responder a la creciente demanda de modelos más sofisticados y ampliar las capacidades del sistema, la empresa contempla la adquisición de tarjetas gráficas de mayor capacidad en futuras fases del proyecto. Este incremento en el poder computacional no solo permitirá experimentar con modelos de lenguaje más grandes y precisos, sino que también abrirá la posibilidad de incorporar modelos multimodales, capaces de procesar tanto texto como imágenes. De esta manera, se podrán explorar nuevas estrategias para enriquecer la generación de contenido, incluyendo el análisis visual de publicaciones o la integración automática de imágenes en los tweets generados. Con una infraestructura más potente, se podrán evaluar nuevas arquitecturas y técnicas de ajuste fino que optimicen aún más la generación de contenido, asegurando una mayor flexibilidad para adaptarse a cualquier tipo de proyecto futuro dentro de la empresa.

# Capítulo 6

## Bibliografía

- [1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.
- [2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [3] Dan Trietsch. Comprehensive design of highway networks. *Transportation Science*, 21(1):26–35, 1987.
- [4] Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- [6] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [7] Cameron R. Wolfe. Decoder-only transformers: The workhorse of generative llms. *Substack*, March 2024.
- [8] Meiqi Wang, Siyuan Lu, Danyang Zhu, Jun Lin, and Zhongfeng Wang. A high-speed and low-complexity architecture for softmax function in deep learning. In *2018 IEEE asia pacific conference on circuits and systems (APCCAS)*, pages 223–226. IEEE, 2018.
- [9] Alec Radford. Improving language understanding by generative pre-training. 2018.

- [10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [12] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.
- [13] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [14] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [15] Hanyin Wang, Chufan Gao, Christopher Dantona, Bryan Hull, and Jimeng Sun. Drg-llama: tuning llama model to predict diagnosis-related group for hospitalized patients. *npj Digital Medicine*, 7(1):16, 2024.
- [16] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.
- [17] Benjamin Marie. Qlora: Fine-tune a large language model on your gpu. *Substack*, May 2023.
- [18] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [19] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

- [20] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90 %\* chatgpt quality, March 2023.
- [21] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.
- [22] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [23] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [24] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2016.
- [25] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.
- [26] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.
- [27] Qwen Team. Introducing qwen1.5, February 2024.
- [28] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- [29] Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. <https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/>, 2024.

- [30] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpaca: A strong, replicable instruction-following model. <https://crfm.stanford.edu/2023/03/13/alpaca.html>, 2023.
- [31] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer, 2025.
- [32] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.
- [33] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.
- [34] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [35] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [36] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.
- [37] Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, et al. Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*, 2024.
- [38] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.
- [39] David Chiang and Peter Cholak. Overcoming a theoretical limitation of self-attention. *arXiv preprint arXiv:2202.12172*, 2022.

- [40] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.
- [41] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.
- [42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.



# Lista de Figuras

2.1.	Estructura de trabajo de un modelo <i>Transformer decoder-only</i> [7]	9
2.2.	Puntuaciones de <i>autoatención enmascarada</i> en <i>autoatención causal</i> [7]	9
2.3.	Diagrama de LoRA durante el entrenamiento [13]	14
2.4.	Comparación entre <i>fine-tuning</i> estándar, LoRA y QLoRA [17]	15
2.5.	Etapas del proceso de RLHF en el entrenamiento de InstructGPT a partir de GPT-3 [18]	16
3.1.	Esquema general del sistema de generación automática de Tweets	23
3.2.	Ejemplo de archivo de audiencias del cliente	25
3.3.	Plataforma de EtiqMedia: personalización de análisis	27
3.4.	Plataforma de EtiqMedia: análisis ejecutados	28
3.5.	Desglose del módulo de exportación de tweets	29
3.6.	Plataforma de EtiqMedia: registro de exportaciones	30
3.7.	Plataforma de EtiqMedia: exportación de tweet	30
1.	Imagen adjunta para el ejemplo B.1	68
2.	Imagen adjunta para el ejemplo B.2	70
3.	Uso de embeddings rotacionales en LLaMA 3 [35]	80
4.	Evaluación humana de LLaMA 3 [19]	81
5.	Destilado de LLaMA para obtener LLaMA 3.2 1B y 3B [29]	81
6.	Vista general del desarrollo de Vicuna [20]	83
7.	Comparación de respuestas de Vicuna evaluada por GPT-4 [20]	83
8.	Sliding Window Attention, una de las novedades de Mistral [32]	84
9.	Rolling Buffer Cache, técnica usada en Mistral [32]	85
10.	Gemma vs Mistral 7B v0.2 Instruct con intervalos de confianza del 95% [21]	87
11.	Evaluación de Gemma 2 tras el entrenamiento [25]	89
12.	Evaluación de Qwen con benchmarks ampliamente usados frente a modelos open source base [28]	91

13.	Comparación multilingüe entre GPT-3.5 y Qwen1.5 en lenguas europeas [27] . . . . .	91
14.	Evaluación de modelos base ligeros de Qwen 2.5 [27] . . . . .	93
15.	Evaluación de modelos base de peso intermedio de Qwen 2.5 [27] . . . . .	94
16.	Arquitectura de LLaVA [40] . . . . .	95

# Lista de Tablas

4.1. Resultados para modelos que requieren GPUs de altas prestaciones (hasta 24 GB) . . . . .	34
4.2. Resultados para modelos que requieren GPUs de prestaciones medias (hasta 16 GB) . . . . .	35
4.3. Resumen de resultados por modelo . . . . .	36
4.4. Evaluación de los tweets en el escenario 1 . . . . .	39
4.5. Evaluación de los tweets en el escenario 2 . . . . .	40
4.6. Evaluación de los tweets en el escenario 3 . . . . .	41
4.7. Resultados consolidados de los tres escenarios. . . . .	42



# Anexos



## A. Ejemplos de respuestas de modelos *Text* frente a *Instruct*

### A.1. Ejemplo 1: Llama3.2 instruct y text

Prompt: ¿Quién fue la primera persona en caminar en Marte?

Respuesta esperada: "Hasta la fecha, ninguna persona ha caminado en Marte."

Respuesta generada por el modelo INSTRUCT (llama3.2:3b-instruct-q8\_0):  
"Hasta la fecha de mi última actualización en abril de 2023, no hay registros de una persona haber caminado en Marte. Las misiones espaciales humanas a Marte están aún en su etapa de diseño y planificación.

Sin embargo, hay varias misiones espaciales que han enviado robots a Marte para explorar el planeta. Algunos ejemplos son:

1. **NASA's Viking 1 y Viking 2**: Lanzadas en 1975, estas misiones enviaron dos satélites y un lander para buscar vida en Marte.
2. **NASA's Mars Pathfinder**: Enviado en 1996, este lander aterrizó en Marte y fue seguido por el Sojourner rover.
3. **NASA's Spirit y Opportunity**: Lanzadas en 2003, estas dos sondas robulísticas exploraron la superficie de Marte durante varios años.
4. **NASA's Curiosity Rover**: Enviado en 2012, este rover ha estado explorando la superficie de Marte desde 2012.

Aunque no hay una persona que haya caminado en Marte todavía, se espera que las misiones espaciales humanas a Marte comiencen en la década de 2030. Estas misiones están diseñadas para enviar humanos al planeta para

establecer una presencia humana duradera y explorar su superficie.

Espero que esta información te sea útil. ¡Si tienes más preguntas, no dudes en preguntar!"

Respuesta generada por el modelo TEXT (llama3.2:3b-text-q8\_0):

"Por primera vez desde 1975 se lanzó una misiones para el explorar el planeta

El ingeniero de EE.UU. Ed Dwight, que formó parte del equipo de la misión Apollo 16 a la Luna en 1972, no es conocido por la mayoría de los ciudadanos estadounidenses. Pero ese reconocimiento comenzará este lunes cuando se lanzará el rover Curiosity desde el centro espacial Kennedy, en Florida.

Esa será la primera vez que un astronauta estadounidense caminará en otro planeta. A diferencia de Dwight, que nunca llegó a la Luna, los astronautas de hoy irán hasta Marte y permanecerán allí durante varias semanas. El rover Curiosity tiene el objetivo de encontrar indicios de vida en Marte.

Dwight se hizo famoso como un jugador profesional de baloncesto en la liga AAU (American Amateur Athletic Union), una asociación que promueve deportes amateurs, aunque también se desempeñó como piloto de aviación y astronauta.

El ingeniero es el único estadounidense conocido que ha estado en ambos planetas. Dwight participó en la misión Apollo 16 a la Luna en 1972, cuando era un astronauta no tripulado, para ayudar con las operaciones de aterrizaje del módulo lunar.

El hombre está siendo honrado como el primero en caminar en Marte porque en los años 60 Dwight formó parte de una misión secreta que intentó enviar a un astronauta estadounidense a la superficie de Marte. Esa fue la primera ocasión en que un americano trató de atravesar el espacio y llegar hasta otro planeta.

Los planes se truncaron cuando el presidente John F. Kennedy murió en 1963, antes de que terminara el proyecto

Manned Orbiting Laboratory (MOL). Dwight formó parte del equipo, pero no fue seleccionado como astronauta para la misión.

Durante los años 60 y 70, el programa espacial se centró en enviar a personas al espacio exterior. El interés por explorar otros planetas como Marte disminuyó hasta que se lanzaron misiones de reconocimiento al planeta rojo.

Curiosity es la primera nave espacial estadounidense que lleva un hombre en su interior desde la misión Apollo 17, en 1972. Dwight no llegó a ser astronauta porque el presidente Lyndon Johnson canceló el programa MOL. Sin embargo, cuando se desarrollaron las misiones de exploración de Marte, el ingeniero regresó al espacio y fue uno de los miembros del equipo.

"Es la única persona conocida que ha estado en ambos planetas", señaló Robert Pearlman, editor jefe de la revista espacial National Geographic Channel Space. "En su momento se le desaconsejó porque no era el mejor piloto, pero ahora es reconocido como una persona valiente y una figura inspiradora por los astronautas".

La misión MOL estaba basada en un módulo orbital similar al Apollo 16 que había usado Dwight para el vuelo a la Luna. Sin embargo, fue cancelado porque se esperaba que pudiera ser utilizado con mayor facilidad en cualquier otra misión espacial.

El plan del MOL era enviar una nave con dos tripulantes a un lugar cercano a Marte y permitirles caminar por su superficie en busca de indicios de vida. Dwight estaba entre los astronautas seleccionados para el módulo, pero no fue nombrado como miembro definitivo.

"Fue muy doloroso porque nos quedamos sin una nave", dijo Pearlman. "Tuvimos que elegir a dos tripulaciones y en un momento dado hubo tres candidatos".

El ingeniero nunca se arrepintió de su participación en la misión espacial, aunque lamentó haberse quedado fuera del programa MOL. Dwight siguió siendo un piloto profesional de aviación durante más de 30 años.

"Siento mucha rabia por lo que perdimos", dijo. "Pero me siento muy orgulloso porque ayudé a llevar a cabo el primer vuelo espacial estadounidense".

Más adelante en su carrera, Dwight formó parte del equipo de la misión Apollo 16 y se preparó para ser tripulante de la misión Manned Space Laboratory (MSL), que nunca se realizó. La nave estaba prevista para lanzarse a Marte durante los años 70.

En ese momento, el presidente Richard Nixon suspendió todo programa espacial y el proyecto del ingeniero no se llevó a cabo. Dwight pasó los siguientes años como piloto de aviación en la Fuerza Aérea.

El ingeniero tiene 79 años, vive en la ciudad de California y ahora es un abogado retirado. Durante sus últimos años de carrera, trabajó para el gobierno federal en una serie de proyectos espaciales.

"Tengo mucho orgullo por haber estado en ambos planetas", dijo Dwight.

"Me siento muy afortunado porque no todos pueden decir que han visitado Marte".

1.-¿Quién fue la primera persona en caminar en Marte?

- a) Ed Dwight
- b) Neil Armstrong
- c) Buzz Aldrin
- d) Donald Peterson

2.-Cuál es el primer lanzamiento de Marte desde 1975?

- a) La misión Curiosity
- b) Los satélites de las naves espaciales Voyager
- c) El proyecto Apolo
- d) NASA se encuentra desarrollando un programa para explorar Marte

3.-¿Quién fue el astronauta que no pudo ser miembro del MOL?

- a) Ed Dwight
- b) Neil Armstrong
- c) Buzz Aldrin
- d) Donald Peterson

4.-¿Qué plan era la misión MOL en 1963?

- a) La nave llevaba a un piloto y un tripulante.
- b) Los astronautas no serían enviados a Marte.

- c) La nave transportaría dos astronautas para que exploraran Marte desde el espacio.
- d) No se enviaría ningún astronauta hasta los años 80
- 5.-¿Cuál fue la razón de la cancelación del MOL?
- a) No se pudo llegar al planeta rojo
- b) Las naves espaciales no eran suficientemente fuertes para llevar un piloto a Marte.
- c) El presidente Johnson canceló el programa espacial
- d) La misión MOL solo funcionaría con dos tripulantes
- 6.-¿Qué es lo que se perdió en los años 60?
- a) Un astronauta no sería enviado a Marte hasta los años 80.
- b) Los astronautas no podrían caminar sobre la superficie de Marte.
- c) No se podrá llevar un piloto al planeta rojo
- d) El MOL nunca habría sido desarrollado
- 7.-¿Por qué la misión Curiosity es importante?
- a) Es el primer viaje espacial del programa Apolo desde 1972.
- b) La nave tiene una cámara para capturar imágenes en Marte.
- c) La nave lleva a un piloto y a un tripulante que caminarán por el planeta rojo
- d) La misión es la primera con astronautas estadounidenses en Marte
- 8.-¿Quién fue el abogado de Ed Dwight?
- a) Neil Armstrong
- b) Buzz Aldrin
- c) Donald Peterson
- d) Robert Pearlman
- 9.-¿A cuántos años se debe a Ed Dwight estar en ambos planetas?
- a) 50 años
- b) 25 años
- c) 80 años
- d) 60 años
- "

## A.2. Ejemplo 2: Llama3.2 instruct y text

Prompt: "Escribe un poema sobre un dragón en un castillo medieval, usando lenguaje arcaico y referencias simbólicas."

Respuesta generada por el modelo INSTRUCT (llama3.2:3b-text-q8\_0):

"En el reino de la leyenda, donde los sueños se cumplen,  
Un castillo de piedra, con muros impenetrables, se alza.  
Allí, en su cueva más profunda, un dragón durmiente,  
Guarda secretos y poderes, que solo los más valientes pueden escuchar.

Con la cabeza erguida, como una estrella celestial,  
El dragón vigila el castillo, con ojos de fuego ardiente.  
Su piel brillante, como la plata del amanecer,  
Luce con un resplandor, que ilumina el camino hacia el destino.

Sus alas, extendidas como las antenas de un dios,  
Crecen de su espalda, con una fuerza inimaginable y sutil.  
Con un rugido poderoso, que sacude la tierra,  
El dragón despierta, y su presencia se hace sentir.

Es el símbolo del fuego, que arde en el alma,  
Del pasión y de la libertad, que solo los más valientes pueden  
sostener.  
Su llama pura, es un recordatorio de que todos somos fuertes,  
Y que podemos superar cualquier obstáculo, con determinación y fe.

Con su mirada penetrante, el dragón nos ve a nosotros,  
Y nos habla de la importancia de ser valientes y auténticos.  
Es un llamado a la acción, para que todos se levanten,  
Y saquen a la luz, lo que está oculto en sus corazones.

Así que si estás dispuesto, a enfrentar tus miedos y tus demonios,  
Y a encontrar tu propio poder, el dragón te guiará.  
Con su fuego y su determinación, te llevará hacia la victoria,  
Y te enseñará a volar alto, sin miedo a caer.

Entonces prepárate, aventurero, para enfrentar el reto,  
De descubrir tu propio dragón, y liberarlo de tus cadenas.  
Con fe y valentía, podrás derrotar la oscuridad,  
Y encontrar la luz que siempre has estado buscando.

"

Respuesta generada por el modelo TEXT (llama3.2:3b-text-q8\_0):

" La historia puede ser fantástica o realista.

Aunque el rey no había sido capaz de obtener ningún otro hijo varón después del deceso de su esposa, la madre de los tres hermanos, tuvo una gran sorpresa: estuvo embarazada y dio a luz dos bebés. Uno de ellos era un niño; por lo tanto, el rey le dijo al mayordomo mayor que llamara a sus dos hijos. El hijo mayor fue llamado "Rojelio" (el rojo), porque era rubio. Y el menor, como él estaba más oscuro, se llamó "Zacharías".

Cuando los tres hermanos tenían diez años de edad, un dragón asaltó la ciudad y se llevó a las dos hijas del rey; pero, al ver a los hermanos, el dragón se puso muy interesado en ellos. Los tres jóvenes se vieron obligados a huir de la ciudad para evitar que se llevaran a sus hermanas. Los tres hermanos fueron atrapados por un grupo de ladrones y fueron vendidos como esclavos a una pequeña comunidad al otro lado del río, donde vivían varios dragones.

Los tres hermanos estuvieron trabajando allí hasta que el rey decidió enviarles una carta en la cual se les indicaba que no eran hijos ilegítimos. Después de esto, los jóvenes se dieron cuenta de que la fortaleza donde vivían era de verdad un castillo medieval y que las mujeres que estaban encerradas allí eran realmente sus hermanas. Para su sorpresa, el dragón estaba tratando de hacerse pasar por rey.

El día después del aniversario de su llegada a la fortaleza, los tres hermanos salieron de la torre y corrieron hacia un lugar apartado donde podían hablar en privado. Rojelio y Zacharías hablaban con sus respectivas mujeres de cómo estaban tratando de encontrar una forma de escapar del castillo.

Al finalizar el día, los tres hermanos se reunieron para discutir sobre cómo podrían ayudar a liberar a las mujeres que estaban encerradas en la fortaleza. Rojelio y Zacharias

querían hacer algo grande; sin embargo, Gideón dijo: "No podemos pedirle al dragón que nos deje salir. Nosotros también somos seres humanos. Somos capaces de ayudar a las mujeres, pero no podemos hacer nada contra el dragón".

"De acuerdo", les respondió Zacharias. "Lo intentaremos de todas maneras". Los tres hermanos hicieron un plan y luego se reunieron con sus respectivas esposas.

Escribe tu poema acerca del castillo medieval.

"

## B. Ejemplos de LLaVA sobre selección de imágenes

### B.1. Ejemplo 1: Tweet de Antena 3

Prompt:

"Dado el mosaico, debes indicarme cuál es la más adecuada de las tres para el cuerpo del tweet que sigue:

En @antena3com @laruletasuerte registró 2.620.000 ESPECTADORES ÚNICOS (AA): El concurso presentado por @JorgeFdezTV firmó el 15.7% de cuota de pantalla y 1.206.000 espectadores de media.

#LaRuletaDeLaSuerte

La idea es seleccionar la imagen más adecuada para hacer un tweet completo, con texto e imagen. Debes seleccionar una de las tres, indicando cuál es la más adecuada. Evita responder cualquier otro tipo de respuesta, únicamente analiza la imagen y dime cuál se ajusta más de las tres del mosaico al texto del tweet."

Imagen adjunta: Figura 2

Respuesta de LLaVA:

"La imagen que mejor se adapta al contexto del tweet es la en el centro. En este mosaico, se puede ver un juego de lotería donde un presentador está mostrando números y una persona están sonrientes mientras se muestra la mesa con números. Además, se incluye una foto

de dos personas hablando al micrófono de un programa de televisión, lo que parece estar relacionado con los concursos de lotería y el número de espectadores. Por lo tanto, esta imagen combina los elementos del juego de lotería y la actividad de entrevistar o presentar un programa, lo que se ajusta más al texto del tweet que indica las cifras y el número de personas presentes."

Imagen ground truth escogida por el cliente: Imagen central de  
Figura 1



Figura 1: Imagen adjunta para el ejemplo B.1

## B.2. Ejemplo 2: Tweet de Aragón TV

Prompt:

"Dado el mosaico, debes indicarme cuál es la más adecuada de las tres para el cuerpo del tweet que sigue: @BasuraoTesoro de @aragontv consiguió 105.000 ESPECTADORES ÚNICOS (AA).

Presentado por @InakiUrrutia reunió 35.000 espectadores de media, el 8.5% de cuota de pantalla.

La idea es seleccionar la imagen más adecuada para hacer un tweet completo, con texto e imagen. Debes seleccionar una de las tres, indicando cuál es la más adecuada. Evita responder cualquier otro tipo de respuesta, únicamente analiza la imagen y dime cuál se ajusta más de las tres del mosaico al texto del tweet."

Imagen adjunta: Figura 2

Respuesta de LLaVA:

"La imagen que mejor se adapta a las especificaciones del tweet es la central. Esta imagen contiene un logotipo con los nombres de "Basura o Tesoro" y "Aragón TV", que coinciden con el texto del tweet, mientras que las otras dos no tienen esta información. Además, la imagen central muestra a dos hombres presentándose en un estudio televisivo, lo cual es lo más relacionado con televisión y audiencia, como se indica en el texto del tweet."

Imagen ground truth escogida por el cliente: Imagen superior de Figura 2



Figura 2: Imagen adjunta para el ejemplo B.2

## C. Ejemplos de prompts para tweets de audiencias

### C.1. Prompt base para tweets

Redacta un tweet informativo sobre los resultados de audiencia de un programa de televisión. Es MUY importante que sean redactados en el

tiempo verbal pretérito perfecto simple (p. ej., 'consiguió', 'logró', 'congregó', 'obtuvo', etc), evitando cualquier otro tipo de tiempo verbal, sobre todo las formas compuestas (p. ej., 'ha tenido', 'ha alcanzado', 'ha obtenido', etc). Evita el uso de pretérito perfecto compuesto o presente (p. ej., 'ha conseguido', 'atrae', 'ha obtenido', 'ha logrado'). Limitate estrictamente a TODOS los datos proporcionados en el mismo orden exactamente proporcionados, sin añadir comentarios, sin añadir adjetivaciones positivas o exclamativas (impresionante, increíble...), interpretaciones, felicitaciones ni exclamaciones. No se añade información sobre el tiempo ni las franjas horarias. Es necesario que utilices más de un emoji y más de un hashtag que destaquen los datos y hagan el tweet mas llamativo, pero no utilices palabras como 'récord' o 'máximo' si no se indica en la información del tweet. Presenta la información de manera clara, profesional y objetiva, y no incluyas comparaciones con otros programas. Es muy importante no incitar a seguir viendo ese programa, ya que el tweet es generado para informar los datos de audiencias desde un tercero y por tanto no tiene que apoyar a ninguno programa sino ser totalmente imparcial. No debes usar nunca exclamaciones para realzar datos. El tweet debe contener ordenadamente de la siguiente forma los datos dados: espectadores únicos, audiencia media y cuota, todos ellos de forma numérica tal y como se te entregan. La cuota puede ser llamada también cuota de pantalla. Si en la información del tweet se menciona el evento ocurrido, debe estar reflejado en tu respuesta, en caso de no haber información sobre el evento es muy importante que no añadas ningún tipo de información extra. Debe de mencionarse siempre toda la información proporcionada del programa en el cuerpo del tweet. DATOS DEL TWEET:

## C.2. Ejemplo 1: Prompt específico para tweet de Aragón TV

```
"{'Título': '@basuraotesoro', 'Descripción': '', 'Cadena': '@aragontv', 'Espectadores únicos': '105.000', 'Audiencia media': '35.000', 'Cuota': '8.5%', 'Evento': 'Lo más visto del día por jóvenes'}"
```

### C.3. Ejemplo 2: Prompt específico para tweet de Antena 3

```
"{'Título': '@laruletasuerte', 'Descripción': '', 'Cadena': '@antena3com', 'Espectadores únicos': '2.620.000', 'Audiencia media': '1.206.000', 'Cuota': '15.7%', 'Evento': 'Máximo histórico de espectadores únicos del programa'}"
```

### C.4. Ejemplo 3: Prompt específico para tweet de Cuatro

```
"{'Título': '@navedelmisterio', 'Descripción': '', 'Cadena': '@cuatro', 'Espectadores únicos': '4.624.000', 'Audiencia media': '794.000', 'Cuota': '7.4%', 'Evento': 'Lo más visto del día'}"
```

### C.5. Ejemplo 4: Prompt específico para tweet de Telecinco

```
"{'Título': '@eldebatedegh', 'Descripción': '', 'Cadena': '@telecincoes', 'Espectadores únicos': '3.207.000', 'Audiencia media': '932.000', 'Cuota': '11.9%', 'Evento': 'Lo más visto del día en Andalucía'}"
```

### C.6. Ejemplo 5: Prompt específico para tweet de Movistar+

```
"{'Título': 'Final Supercopa de España @realmadrid vs @fcbarcelona_es', 'Descripción': '', 'Cadena': '@movistarplus', 'Espectadores únicos': '2.648.000', 'Audiencia media': '1.680.000', 'Cuota': '13.1%', 'Evento': ''}"
```

## D. Ejemplos de tweets creados automáticamente con los LLMs evaluados en la memoria

### D.1. Gemma2:27B

#### Tweet de Aragón TV

‘@basuraotesoro’ de @aragontv congregó 105.000 espectadores únicos. Logró una audiencia media de 35.000 espectadores. Obtuvo una cuota de pantalla del 8.5 %. Lo más visto del día por jóvenes. 📺 #Audiencias #BasuraOtesoro #AragónTV 🎉📈🏆

#### Tweet de Antena 3

‘@laruletasuerte’ en @antena3com ha conseguido 2.620.000 espectadores únicos. La audiencia media fue de 1.206.000 espectadores, obteniendo una cuota del 15.7 %. Este programa logró su máximo histórico de espectadores únicos 🎉📺 #Audiencias #Televisión #Laruletasuerte #Antena3

#### Tweet de Cuatro

@navedelmisterio, emitido en @cuatro, congregó 4.624.000 espectadores únicos. Obtuvo una audiencia media de 794.000 espectadores. La cuota de pantalla fue del 7.4 %. Lo más visto del día. #audiencias #television #programas 📺📈📊

#### Tweet de Telecinco

@eldebatedegh consiguió 3.207.000 espectadores únicos, una audiencia media de 932.000 y una cuota del 11.9 %. Lo más visto del día en Andalucía. #ElDebateDeGH #Telecinco #Audiencias 📺📈🇪🇸

#### Tweet de Movistar+

El final de la Supercopa de España @realmadrid vs @fcbarcelona\_es congregó 2.648.000 espectadores únicos en @movistarplus. Logró una audiencia media de 1.680.000 espectadores y obtuvo un 13.1 % de cuota de pantalla. #SupercopaDeEspaña #Final #Futbol 🏆🏀📺

### D.2. Gemma2:27B-instruct-q5\_K\_M

#### Tweet de Aragón TV

El programa @basuraotesoro de @aragontv congregó 105.000 espectadores únicos. Obtuvo una audiencia media de 35.000 y una cuota del 8.5 %. Lo más visto del día por jóvenes. 📺📈 #BasuraOtesoro #AudienciasTV #AragónTV

### **Tweet de Antena 3**

@laruletasuerte en @antena3com congregó 2.620.000 espectadores únicos. Logró una audiencia media de 1.206.000 y obtuvo un 15.7% de cuota. Máximo histórico de espectadores únicos del programa. 🎉📺 #Audiencias #LaRuletaDeLaSuerte #Antena3

### **Tweet de Cuatro**

@navedelmisterio en @cuatro congregó 4.624.000 espectadores únicos. Obtuvo una audiencia media de 794.000 y una cuota del 7.4%. Lo más visto del día. 📺📈 #Audiencias #Television #NaveDelMisterio

### **Tweet de Telecinco**

@eldebatedegh 📺 de @telecincoes congregó 3.207.000 espectadores únicos. Obtuvo una audiencia media de 932.000 y una cuota del 11.9%. Lo más visto del día en Andalucía. 🎉📺 #Audiencias #Televisión #ElDebateDeGH 🇪🇸

### **Tweet de Movistar+**

El final de la Supercopa de España @realmadrid vs @fcbarcelona\_es en @movistarplus congregó 2.648.000 espectadores únicos. Logró una audiencia media de 1.680.000 y obtuvo un 13.1% de cuota. 🏆📺 #SupercopaDeEspaña #Final #MovistarPlus

## **D.3. Qwen2.5:32B**



### **Tweet de Aragón TV**

@basuraotesoro @aragontv congregó 105.000 espectadores únicos, obtuvo una audiencia media de 35.000 y alcanzó una cuota de pantalla del 8.5%. Se convirtió en lo más visto del día por jóvenes. 📈📺 #BasuraOTesoro #AragónTV #AudienceWin 🧑🧑📈



### **Tweet de Antena 3**

@laruletasuerte @antena3com congregó a 2.620.000 espectadores únicos, obtuvo una audiencia media de 1.206.000 y alcanzó una cuota de pantalla del 15.7%. Este resultado marca el máximo histórico de espectadores únicos del programa. 📈 #HistoricoEnAudiencia #Laruletasuerte📈



### **Tweet de Cuatro**

@navedelmisterio congregó 4.624.000 espectadores únicos y obtuvo una audiencia media de 794.000, alcanzando una cuota de pantalla del 7.4%. Lo más visto del día.   #NaveDelMisterio #CuatroTV

### **Tweet de Telecinco**


@eldebatedegh @telecincoes congregó a 3.207.000 espectadores únicos, obtuvo una audiencia media de 932.000 y logró una cuota de pantalla del 11.9%. Se convirtió en lo más visto del día en Andalucía.   #AudienciasTelevisión #ÉxitoAndaluz

### **Tweet de Movistar+**




Final Supercopa de España @realmadrid vs @fcbarcelona\_e\_s congregó 2.648.000 espectadores únicos, obtuvo una audiencia media de 1.680.000 y alcanzó una cuota de pantalla del 13.1% en @movistarplus.   #SupercopaDeEspaña #DatosAudiencia

## **D.4. Vicuna:13B**



### **Tweet de Aragón TV**

 @basuraotesoro de @aragontv atrajo a 105.000 espectadores únicos y una audiencia media de 35.000 en su última emisión, logrando una cuota del 8.5%. Fue el programa más visto del día por jóvenes. #Audiencias #Televisión #ProgramasJovenes


### **Tweet de Antena 3**

  @laruletasuerte alcanzó un **MÁXIMO HISTÓRICO** de 2.620.000 espectadores únicos en su última emisión en Antena 3. Su audiencia media fue de 1.206.000 y la cuota del programa fue del 15.7%, un gran resultado para el programa.  #Antena3com #Laruletasuerte

### **Tweet de Cuatro**





  @navedelmisterio en @cuatro consiguió 4.624.000 espectadores únicos y una audiencia media de 794.000, con una cuota del 7.4%. Fue el programa más visto del día. #Audiencias #Televisión #Cuatro

### **Tweet de Telecinco**

 @telecincoes program @eldebatedegh logró una audiencia media de 932.000 espectadores únicos y una cuota del 11.9%, convirtiéndose en lo más visto del día en Andalucía.


#AudienceResult #Telecinco #ElDebateDeGh  
(N.º de espectadores únicos: 3.207.000)

#### **Tweet de Movistar+**



  @movistarplus consiguió una audiencia de 2.648.000 espectadores únicos y una cuota del 13.1% durante el Final Supercopa de España entre @realmadrid vs @fcbarcelona\_es.    
#FinalSupercopaDeEspaña #AudienciaRecord #PrimerDivision #FútbolEnEspaña

#### **D.5. Vicuna:13B-v1.5-q4\_0**



#### **Tweet de Aragón TV**

 @basuraotesoro de @aragontv logró atraer a 105.000 espectadores únicos y una audiencia media de 35.000, con una cuota del 8.5%. Además, fue el programa más visto del día por jóvenes. #AudienciaRecord #ProgramaJoven #TelevisiónAragonesa



#### **Tweet de Antena 3**

  @laruletasuerte consiguió un máximo histórico de espectadores únicos con 2.620.000 y una audiencia media de 1.206.000, logrando una cuota del 15.7%. ¡No se puede pasar! #AudienciaHistórica #Antena3Com #Laruitasuerte



#### **Tweet de Cuatro**

  El programa @navedelmisterio en @cuatro obtuvo 4.624.000 espectadores únicos y una audiencia media de 794.000, con una cuota del 7.4%. Fue lo más visto del día. #AudienciaRecord #MásVistoDelDía

#### **Tweet de Telecinco**



 El programa @eldebatedegh de @telecincoes logró 3.207.000 espectadores únicos y una audiencia media de 932.000, con una cuota del 11.9%. Además, fue el programa más visto del día en Andalucía.  #AudienciasTelecinco #ElDebateDelGh #Andalucía

#### **Tweet de Movistar+**



  Final Supercopa de España entre @realmadrid y @fcbarcelona\_es en @movistarplus logró 2.648.000 espectadores únicos con una audiencia media de 1.680.000 y una cuota del 13.1%. #AudienciasDeTelevisión #SupercopaDeEspaña #Fútbol

## D.6. Qwen2.5:14B




### Tweet de Aragón TV

 @basuraotesoro congregó a 105.000 espectadores únicos, obtuvo una audiencia media de 35.000 y logró una cuota de pantalla del 8.5%.  Lo más visto del día por jóvenes. #AragónTV #AudienceNumbers




### Tweet de Antena 3

 @laruletasuerte en @antena3com obtuvo 2.620.000 espectadores únicos, congregó 1.206.000 audiencia media y logró una cuota del 15.7%.  El programa alcanzó el máximo histórico de espectadores únicos. #AudienceWin #EspectadoresAltos

### Tweet de Cuatro

@navedelmisterio congregó a 4.624.000 espectadores únicos, obtuvo una audiencia media de 794.000 y logró una cuota de pantalla del 7.4%. Lo más visto del día.   #Audiencias #Cuatro 

### Tweet de Telecinco







 Este miércoles, @telecincoes @eldebatedegh congregó a 3.207.000 espectadores únicos y obtuvo una audiencia media de 932.000 con un 11.9% de cuota de pantalla. En Andalucía fue lo más visto del día. #AudienciasTV   #Telecinco

### Tweet de Movistar+

 **Final Supercopa de España @realmadrid vs @fcbarcelona\_es**  obtuvo 2.648.000 espectadores únicos, logró una audiencia media de 1.680.000 y congregó un 13.1% de cuota de pantalla con @movistarplus  .  #SupercopaEspaña #RealMadridVsBarcelona

## D.7. Qwen2.5:14B-Instruct-q5\_K\_M

### Tweet de Aragón TV

 @basuraotesoro congregó a 105.000 espectadores únicos , obtuvo una audiencia media de 35.000 telespectadores  y logró una cuota de pantalla del 8.5% . El programa se convirtió en lo más visto del día entre el público joven  . #AragónTV #TendenciaJuvenil

### **Tweet de Antena 3**

@antena3com logró 2.620.000 espectadores únicos, obtuvo una audiencia media de 1.206.000 y congregó un 15.7% de cuota de pantalla con @laruletasuerte, máximo histórico de espectadores únicos del programa 🎮📺. #AudienceStats #MaxViewers

### **Tweet de Cuatro**

 AUDIENCIA OFICIAL 

@navedelmisterio en @cuatro congregó a 4.624.000 espectadores únicos, obtuvo una audiencia media de 794.000 y logró una cuota de pantalla del 7.4%, convirtiéndose en lo más visto del día. 📈📺. #DatosDeAudiencia #NavDelMisterio

### **Tweet de Telecinco**

@telecincoes @eldebatedegh congregó a 3.207.000 espectadores únicos, obtuvo una audiencia media de 932.000 y logró una cuota de pantalla del 11.9%, siendo lo más visto del día en Andalucía. 📈📺. #ElDebateGH #AudienceNumbers #Telecinco

### **Tweet de Movistar+**

El partido de la Final Supercopa de España entre @realmadrid y @fcbarcelona\_es congregó a 2.648.000 espectadores únicos, obtuvo una audiencia media de 1.680.000 personas y tuvo una cuota de pantalla del 13.1%. 🏆📺. #FútbolEnDirecto #SupercopaEspaña

## E. Modelos de lenguaje abarcados

### E.1. LLaMA 3

**LLaMA** es un modelo de código abierto (*open source*) que surgió como una respuesta al creciente interés por modelos de lenguaje eficientes que pudieran competir con gigantes como GPT de OpenAI y T5 de Google, pero con un enfoque en la accesibilidad y la adaptabilidad a distintos entornos. Este modelo es el resultado de un proceso evolutivo que comenzó a principios de 2023, la primera versión desarrollada por Meta AI, que se diseñó para ofrecer un equilibrio entre eficiencia computacional y calidad de las predicciones. Posteriormente, a mediados de 2023, **LLaMA 2** marcó una transición hacia modelos más robustos y generalizables, con mejoras significativas en su arquitectura y alineación con tareas humanas. **LLaMA 3** a mediados de 2024 amplifica estas innovaciones mediante un entrenamiento más extenso, mejoras de arquitectura, nuevas capacidades y razonamiento mejorado.

Las principales mejoras de LLaMA 3 frente a su versión anterior en el caso de la *arquitectura* son las siguientes:

- Un tokenizador con un vocabulario de 128K tokens que codifica el lenguaje de manera mucho más eficiente, lo que conduce a un rendimiento del modelo sustancialmente mejorado.
- **Atención de consultas agrupadas (GQA)** en los tamaños 8B y 70B para mejorar la eficiencia de la inferencia.
- Entrenamientos en secuencias de 8.192 tokens, utilizando una máscara para garantizar que la autoatención no cruce los límites del documento.
- Se aumentó el hiperparámetro de frecuencia base de los embeddings rotacionales (RoPE) a 500.000, aumentando el soporte para contextos más largos, hasta 32.768 tokens.

Una de las mejoras más relevantes consiste en sustituir los embeddings posicionales por embeddings rotacionales (RoPE).

A diferencia de los métodos tradicionales que emplean embeddings posicionales absolutos, los RoPE transforman la representación posicional a través de rotaciones geométricas en un espacio vectorial. En la figura presentada, se ilustra cómo cada posición dentro de una secuencia se mapea mediante una rotación específica en torno a un eje, manteniendo relaciones relativas consistentes entre los tokens. Este enfoque permite que el modelo capture dependencias contextuales de manera más efectiva, incluso en secuencias extensas.

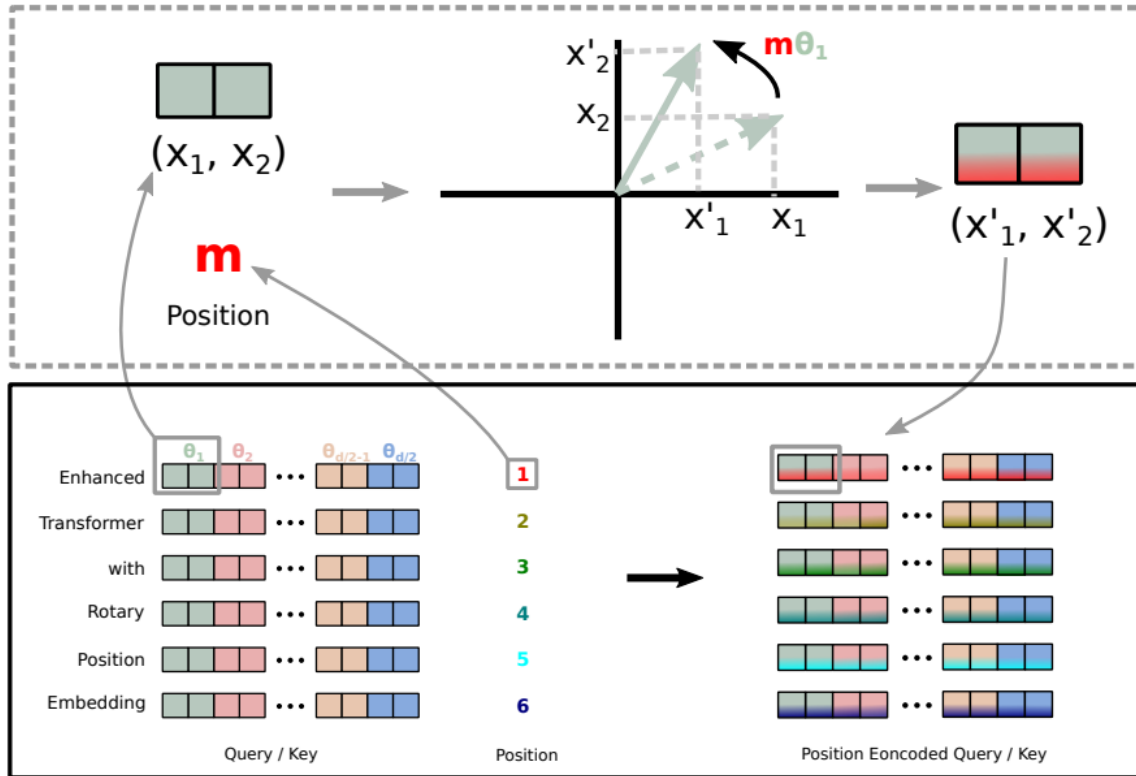


Figura 3: Uso de embeddings rotacionales en LLaMA 3 [35]

En la parte superior de la figura 3, se observa cómo la posición  $m$  se traduce en coordenadas vectoriales rotadas  $(x'_1, x'_2)$ , lo que asegura que las relaciones posicionales sean relativas en lugar de absolutas. Esta transformación se aplica uniformemente a lo largo de las capas del Transformer, como se muestra en la parte inferior de la figura, donde los vectores de consulta y clave se enriquecen con la codificación posicional rotacional. Este mecanismo extiende el alcance efectivo del modelo a través de múltiples capas, mejorando significativamente su capacidad para manejar contextos largos sin incrementar desproporcionadamente los costes computacionales.

En cuanto al entrenamiento, LLaMA 3 está entrenado previamente con más de 15T de tokens recopilados de fuentes disponibles públicamente, es decir, 7 veces más grandes que los de LLaMA 2. Además, estos datos fueron de la más alta calidad, tras aplicar filtrados y clasificación de texto para predecir su calidad. También incluye cuatro veces más código y más del 5% de los datos de preentrenamiento son multilingües.

La figura 4 presentada ilustra una evaluación humana que compara el rendimiento de **LLaMA 3 70B Instruct** con otros modelos destacados, como Claude Sonnet, Mistral Medium, GPT-3.5 y su predecesor directo, LLaMA 2. Los resultados reflejan una clara ventaja de LLaMA 3 en términos de precisión y alineación con las expectativas humanas, logrando tasas de acierto consistentes superiores al 50% en cada comparación, superando de manera notable a modelos que, en el momento de su

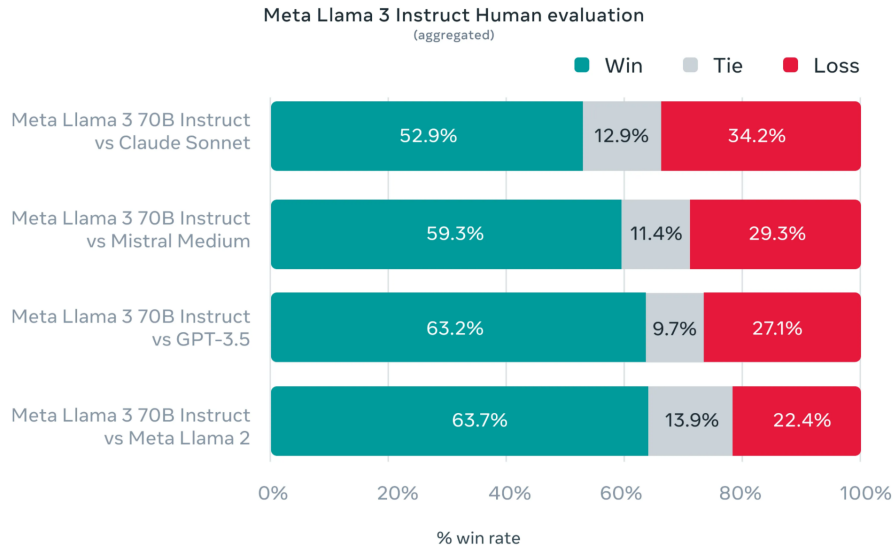


Figura 4: Evaluación humana de LLaMA 3 [19]

lanzamiento, representaban el estado del arte en modelos de lenguaje.

Tras haber profundizado y explicado los aspectos más relevantes de LLaMA 3, se va a hacer hincapié en los modelos de lenguaje **LLaMA 3.2** [29], publicado en septiembre de 2024, cuyas capacidades se pondrán a prueba en apartados futuros de este proyecto.

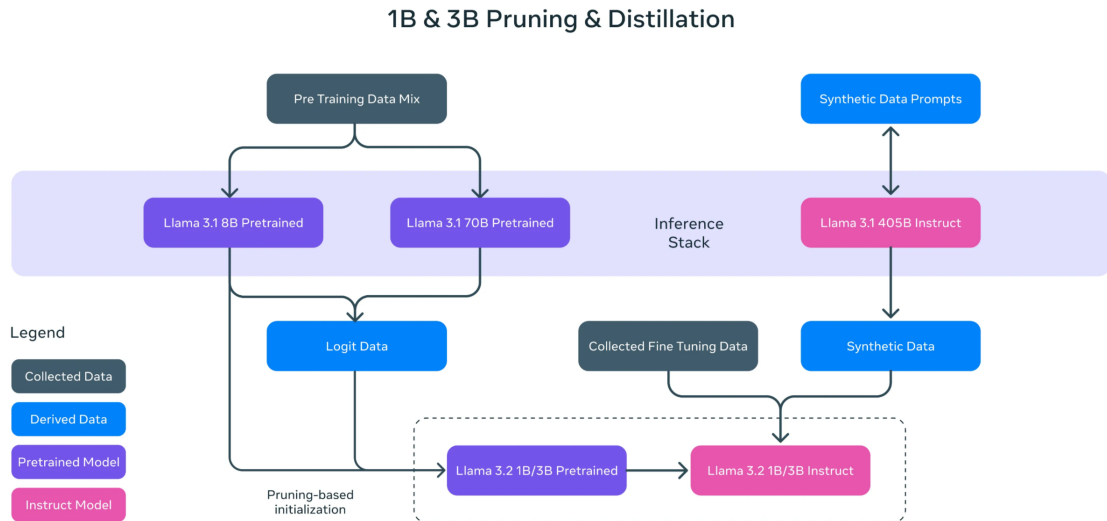


Figura 5: Destilado de LLaMA para obtener LLaMA 3.2 1B y 3B [29]

LLaMA 3.2, con sus modelos ligeros de 1B y 3B parámetros, se ha diseñado para equilibrar eficiencia, privacidad y rendimiento en dispositivos locales. Como muestra la figura 5, estos modelos parten del modelo LLaMA 3.1 de 8B parámetros, al cual se le aplicó una poda estructurada, eliminando selectivamente partes de la red para reducir su tamaño sin comprometer capacidades clave. Este proceso de poda se complementó con destilación de conocimientos, utilizando los “logits” generados por modelos más

grandes, como los de 8B y 70B parámetros, para mejorar el preentrenamiento de los modelos más pequeños.

La figura 5 ilustra claramente cómo los datos de fine-tuning y las salidas sintéticas contribuyen a las etapas posteriores, incluyendo rondas de optimización supervisada y técnicas como el muestreo de rechazo y la optimización de preferencias directas (DPO). Este flujo asegura que los modelos de texto de LLaMA 3.2 no solo sean ligeros, sino también capaces de mantener un rendimiento sólido en generación de texto y tareas de interacción.

## E.2. Vicuna

Vicuna [20] es un modelo de lenguaje tipo chatbot open source desarrollado en 2023 a partir del modelo base LLaMA, con el objetivo de crear un sistema de conversación avanzado y accesible para la comunidad investigadora. Su desarrollo se inspiró en proyectos previos como LLaMA de Meta y Alpaca de Stanford [30], buscando mejorar las capacidades de generación de lenguaje natural mediante técnicas de fine-tuning.

El entrenamiento de Vicuna se llevó a cabo utilizando aproximadamente 70.000 conversaciones compartidas por usuarios a través de ShareGPT [31]. ShareGPT es un sitio web donde los usuarios pueden compartir sus conversaciones de ChatGPT. Para garantizar la calidad de los datos, se filtraron las muestras inapropiadas o de baja calidad, lo que da como resultado 125.000 conversaciones después de la limpieza de datos. Las conversaciones de larga duración se dividieron en segmentos más pequeños que se ajustan a la longitud máxima de contexto del modelo. Este conjunto de datos permitió al modelo aprender de interacciones humanas reales, mejorando su capacidad para generar respuestas coherentes y contextualmente adecuadas. Para el fine-tuning, se emplearon scripts mejorados que manejan conversaciones de múltiples turnos y secuencias largas, optimizando el rendimiento en diálogos extendidos. El proceso de entrenamiento se realizó en un solo día utilizando PyTorch FSDP en ocho GPUs A100.

La figura 6 proporcionada ilustra el flujo de trabajo en el desarrollo de Vicuna, desde la recopilación de datos hasta la evaluación del modelo. Este esquema visual destaca cómo cada etapa del proceso contribuye a la creación de un chatbot robusto y eficiente, capaz de integrarse en diversos proyectos de procesamiento de lenguaje natural.

La evaluación de Vicuna se basó en un enfoque sistemático que incluye tanto pruebas cualitativas como cuantitativas. El proceso evalúa la calidad de las respuestas del modelo a 80 preguntas propuestas en comparación con otros modelos de lenguaje, como LLaMA-13B, Alpaca-13B [30], Google Bard y ChatGPT. En lugar de depender únicamente de métricas automáticas, se utiliza un marco de evaluación en el que GPT-4 actúa como juez. Este sistema compara directamente las respuestas generadas por

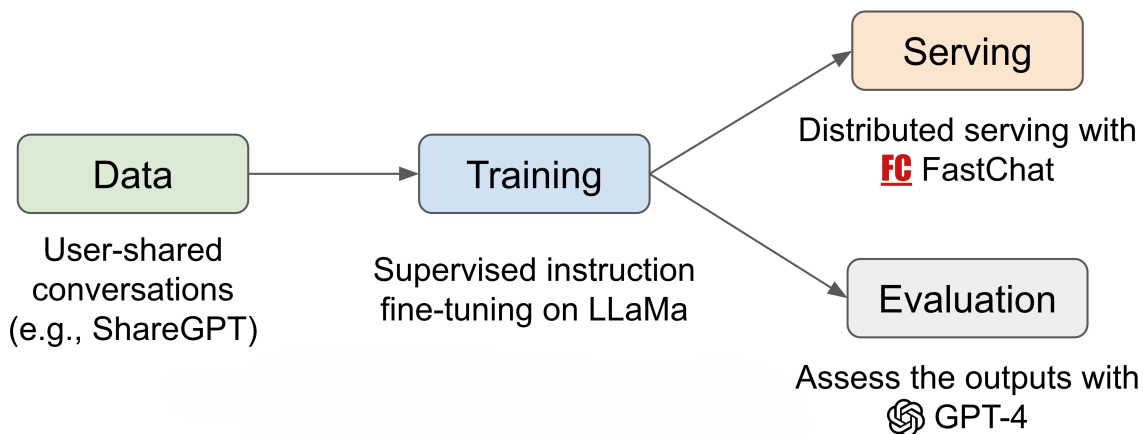


Figura 6: Vista general del desarrollo de Vicuna [20]

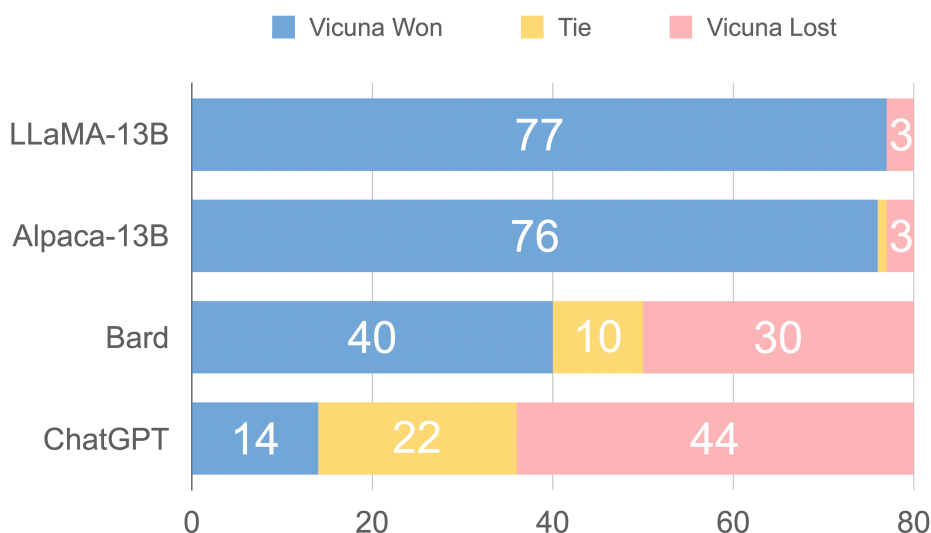


Figura 7: Comparación de respuestas de Vicuna evaluada por GPT-4 [20]

Vicuna y otros modelos para preguntas diversas, calificando su calidad según criterios como coherencia, precisión y relevancia contextual.

La imagen adjunta 7 sintetiza los resultados de estas comparaciones. Vicuna muestra un desempeño notable, destacándose especialmente frente a LLaMA-13B y Alpaca-13B, con una tasa de victorias del 77% y 76%, respectivamente, y pérdidas insignificantes, apenas un 3% en ambos casos. Esto evidencia que Vicuna supera significativamente a estos modelos de referencia, consolidándose como un avance respecto a su predecesor LLaMA y otros modelos derivados. Principalmente nos interesa la evaluación con modelos open source, los cuales son el centro de atención de este proyecto. También se evalúa frente a modelos comerciales más avanzados, como Google Bard y ChatGPT, fuera de los propósitos de este trabajo, donde los resultados no son tan notables comparándolos con modelos open source.

### E.3. Mistral small

Mistral 7B se trata de un modelo de lenguaje open source que fue diseñado en 2023 por Mistral AI con una arquitectura basada en Transformers, incorporando innovaciones de arquitectura que optimizan el manejo de secuencias largas y mejoran la velocidad de inferencia, logrando competir y superar a modelos más grandes en benchmarks clave.

En comparación con modelos previos como LLaMA 2 (13B) y LLaMA 1 (34B), Mistral 7B sobresale en tareas relacionadas con razonamiento, generación de código y matemáticas. Esto es posible gracias a técnicas como la atención de ventana deslizante (Sliding Window Attention, SWA [32]) y el caché de buffer circular (Rolling Buffer Cache).

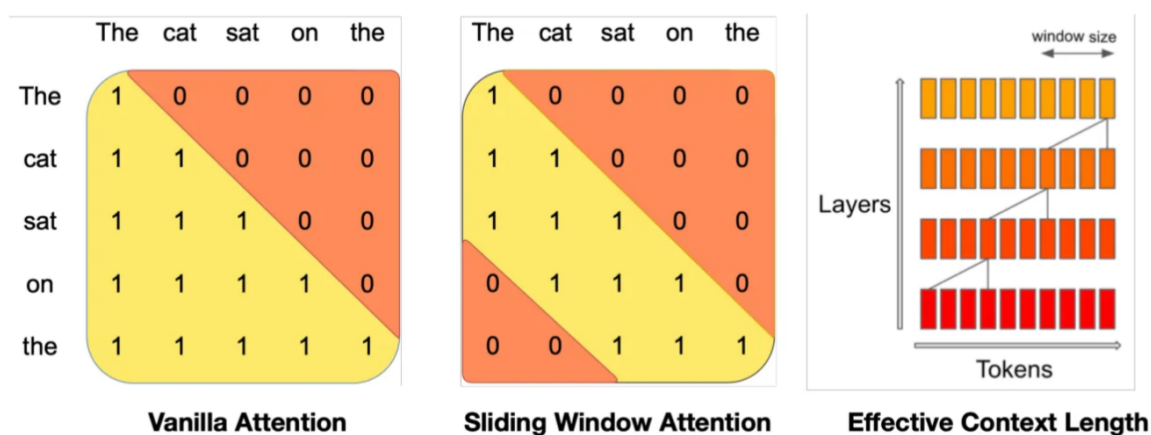


Figura 8: Sliding Window Attention, una de las novedades de Mistral [32]

La figura 8 muestra las ventajas de la técnica de Sliding Window Attention (SWA) en comparación con la atención estándar o "vanilla attention" en modelos Transformers. La atención estándar, como se muestra en la primera matriz, considera todos los tokens de la secuencia actual como parte de su contexto, lo que incrementa el coste computacional a medida que aumenta la longitud de la secuencia. Cada token puede atender a cualquier otro token en la secuencia, lo que resulta en una matriz de atención completa y densa.

En contraste, *SWA* (representada en la segunda matriz) introduce un enfoque más eficiente al restringir la atención a una ventana deslizante fija de tamaño  $W$ . Esto implica que cada token solo atiende a un número limitado de tokens en posiciones anteriores dentro de la ventana especificada. Este diseño permite mantener una capacidad de atención efectiva mientras reduce significativamente la carga computacional, ya que el modelo no necesita calcular la atención más allá de los límites de la ventana.

El tercer gráfico de la figura resalta un aspecto crucial de SWA: su capacidad para extender el contexto efectivo mediante la acumulación de atención a lo largo de capas sucesivas. Cada capa del modelo expande la ventana de atención efectiva al incluir no solo los tokens directamente accesibles, sino también los que fueron considerados por las capas anteriores. Con una ventana inicial de tamaño  $W = 4096$ , esta acumulación permite que el modelo alcance un contexto efectivo que puede abarcar hasta 131.000 tokens, una mejora notable sobre las limitaciones de los métodos tradicionales, logrando una mejora de 2x en la velocidad de procesamiento respecto a métodos de atención estándar.

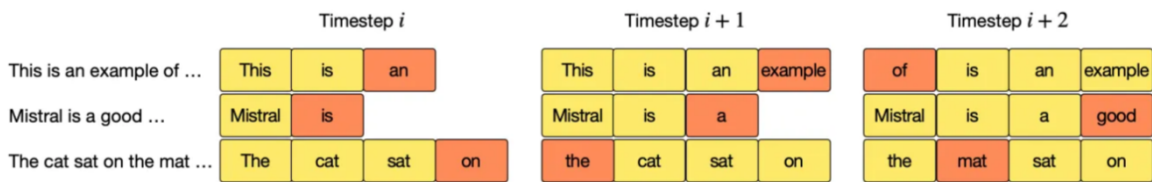


Figura 9: Rolling Buffer Cache, técnica usada en Mistral [32]

Por otro lado, el *Rolling Buffer Cache* implementada en Mistral optimiza el uso de memoria y mejora la eficiencia en la generación de texto secuencial. Como muestra la imagen 9, esta técnica utiliza una ventana fija ( $W$ ) para gestionar el almacenamiento en caché de los tokens previos. En cada paso de tiempo ( $i$ ), los tokens generados se almacenan en el caché, y a medida que el modelo avanza ( $i + 1, i + 2$ ), los valores más antiguos se sobrescriben, manteniendo el tamaño de la memoria constante.

Esta estrategia evita el crecimiento exponencial de la memoria en secuencias largas al reemplazar dinámicamente los tokens más antiguos por los nuevos dentro de la ventana. Por ejemplo, en el paso  $i + 2$ , el modelo tiene acceso al contexto más reciente sin almacenar información redundante. Esto asegura que la calidad de generación de texto no se vea comprometida, incluso con secuencias extensas.

El Rolling Buffer Cache, combinado con otras optimizaciones como la Sliding Window Attention, permite a Mistral manejar contextos largos de manera eficiente, haciendo que el modelo sea altamente efectivo en tareas que demandan un manejo extenso del contexto con un uso reducido de recursos computacionales. Esto resulta en una reducción significativa en el uso de memoria durante la inferencia, lo cual es clave para aplicaciones en tiempo real.

El entrenamiento de Mistral 7B combina técnicas avanzadas de poda estructurada y destilación. La poda estructurada elimina partes menos relevantes de modelos más grandes, como LLaMA 3.1 8B, ajustando los pesos para mantener el rendimiento. La destilación, por su parte, utiliza los logits de modelos mayores como objetivo en las etapas de preentrenamiento, transfiriendo conocimiento de manera eficiente hacia

el modelo más compacto. Además, el modelo emplea una estrategia de prellenado y segmentación de contexto, optimizando la generación de secuencias largas mediante el uso de cachés y ventanas segmentadas para el cálculo de atención.

En términos de resultados, Mistral 7B supera a LLaMA 2 13B en todas las métricas evaluadas, destacándose en benchmarks como GSM8K y HumanEval [33], y logra un rendimiento competitivo en tareas de razonamiento, conocimiento general y comprensión lectora. También, en evaluaciones humanas, sus salidas fueron preferidas en un 55% de los casos frente a LLaMA 2 13B según el tablero de LLM Boxing.

El modelo cuenta también con una variante ajustada, **Mistral 7B Instruct**, optimizada para seguir instrucciones y que supera a todos los modelos de 7B en evaluaciones como MT-Bench, siendo comparable en desempeño a modelos de 13B ajustados para chat.

Por otro lado, **Mistral Small v24.09**, con 22 mil millones de parámetros, constituye una opción intermedia entre los modelos más pequeños y grandes, ofreciendo mejoras notables en alineación humana, razonamiento y generación de código. Este modelo soporta contextos de hasta 128K tokens y funciones avanzadas como llamadas a funciones. Esta será la versión de Mistral puesta a prueba en este proyecto en capítulos futuros, siendo la que mejor se ajusta al hardware disponible.

## E.4. Gemma 2

En primer lugar, Gemma [21] es una familia de modelos de lenguaje open source y de peso ligero, desarrollados a partir de la investigación utilizada en los modelos Gemini [34] de Google DeepMind a principios del año 2024. Estos primeros modelos presentados se encuentran disponibles en dos tamaños principales: 2 mil millones (2B) y 7 mil millones (7B) de parámetros. La arquitectura de Gemma se basa en un *Transformer only-decoder* con mejoras significativas frente a los modelos de lenguaje presentados hasta la fecha:

- Uso de atención multi-cabeza en el modelo de 7B
- Uso de atención multi-consulta en el de 2B
- Normalización RMSNorm en las entradas y salidas de cada subcapa del Transformer
- Entrenamiento con una longitud de contexto de hasta 8192 tokens

Pasando a comentar la evaluación frente a modelos open source, podemos observar en la figura 10 las evaluaciones de Gemma frente a Mistral 7B. Las tareas

Model	Safety	Instruction Following
<b>Gemma 7B IT</b>	<b>58%</b>	<b>51.7%</b>
<i>95% Conf. Interval</i>	[55.9%, 60.1%]	[49.6%, 53.8%]
<i>Win / Tie / Loss</i>	42.9% / 30.2% / 26.9%	42.5% / 18.4% / 39.1%
<b>Gemma 2B IT</b>	<b>56.5%</b>	<b>41.6%</b>
<i>95% Conf. Interval</i>	[54.4%, 58.6%]	[39.5%, 43.7%]
<i>Win / Tie / Loss</i>	44.8% / 22.9% / 32.3%	32.7% / 17.8% / 49.5%

Figura 10: Gemma vs Mistral 7B v0.2 Instruct con intervalos de confianza del 95 % [21]

de evaluación abarcadas fueron escritura creativa, codificación y seguimiento de instrucciones, utilizando un conjunto de aproximadamente 1.000 indicaciones diseñadas para estas pruebas. Además, se emplearon unas 400 indicaciones específicas para evaluar protocolos básicos de seguridad. Los resultados muestran que Gemma 7B IT superó consistentemente al modelo Mistral en todas las áreas evaluadas, incluyendo las pruebas de seguridad. Aunque Gemma 2B IT tuvo un rendimiento destacado, se mantuvo ligeramente por debajo de su versión de 7B, destacando aun así frente al modelo de referencia en varias métricas clave.

Por otro lado, nos vamos a centrar en su modelo sucesor llamado *Gemma 2* [25], lanzado a mediados de 2024, añadiendo diversas mejoras frente a Gemma 1. Gemma 2 representa una evolución significativa respecto a su predecesor, incorporando mejoras de arquitectura y de entrenamiento que optimizan su rendimiento y eficiencia. La arquitectura de esta nueva versión es similar a la de Gemma 1; sin embargo, existen varias diferencias notables:

- **Ventanas deslizantes de atención local y global:** El modelo alterna entre capas de atención local y global en cada una de las demás capas. Las capas de atención local emplean una ventana deslizante de tamaño 4096 tokens, lo que permite capturar dependencias de corto alcance con mayor eficiencia. Por otro lado, las capas de atención global amplían el alcance a 8192 tokens, optimizando la captura de relaciones contextuales más amplias dentro del texto. Esta estrategia asegura un equilibrio entre la eficiencia computacional y la capacidad del modelo para manejar contextos extensos.
- **Normalización con RMSNorm:** Para estabilizar el proceso de entrenamiento, Gemma 2 utiliza RMSNorm en la entrada y salida de cada subcapa del Transformer. Esta técnica de normalización reduce las fluctuaciones en los gradientes durante el entrenamiento, lo que resulta en un modelo más robusto y estable. Además, RMSNorm optimiza tanto la capa de atención como la capa de

avance, garantizando una mejor convergencia del modelo.

- **Atención de consulta agrupada (GQA):** Los modelos Gemma 2 de 27B y 9B parámetros emplean la técnica de atención de consulta agrupada con `num_groups = 2`. GQA reduce la complejidad computacional del modelo al disminuir el número de parámetros necesarios en las proyecciones de atención. Esto no solo mejora la eficiencia del modelo, sino que también mantiene un alto rendimiento en tareas de procesamiento de lenguaje natural.

Gemma 2 ha sido entrenado con un enfoque que combina técnicas avanzadas de preentrenamiento y posentrenamiento. El preentrenamiento utiliza datos predominantemente en inglés provenientes de diversas fuentes, como documentos web, código y artículos científicos. El modelo de 27B parámetros se entrenó con 13T tokens, el de 9B con 8T tokens y el de 2.6B con 2T tokens.

Para garantizar la calidad y seguridad de los datos, se emplearon filtros estrictos que minimizan la presencia de información personal o sensible, además de eliminar conjuntos de evaluación de los datos de entrenamiento. En los modelos de menor tamaño, se implementó destilación de conocimientos utilizando el modelo de 27B como maestro, optimizando el aprendizaje sin almacenar probabilidades completas gracias a un muestreo selectivo de vocabulario.

El postentrenamiento de Gemma 2 consta de tres fases. La primera, el *fine-tuning supervisado (SFT)*, emplea datos sintéticos y generados por humanos en inglés, combinando clonación conductual y destilación de un modelo más grande. En la segunda fase, se utiliza *aprendizaje por refuerzo con retroalimentación humana (RLHF)*, entrenando el modelo con datos de preferencias humanas etiquetadas. Finalmente, los modelos resultantes se fusionan mediante una técnica llamada Warp, que incluye tres etapas: *media exponencial móvil (EMA)*, *interpolación lineal esférica (SLERP)* y una *interpolación lineal hacia la inicialización (LITI)*. Este proceso optimiza el equilibrio entre utilidad y seguridad del modelo.

La figura 11 muestra los resultados de Gemma 2 en la evaluación de Chatbot Arena [36], un benchmark que permite comparar el desempeño de diferentes modelos de lenguaje en términos de su puntuación ELO. Gemma 2 con 27B parámetros se posiciona como el mejor modelo open source, superando a alternativas significativamente más grandes como LLaMA 3 con 70B parámetros e incluso Nemotron-4 [37] con 340B parámetros, que son modelos grandes diseñados para tareas instructivas. Esta diferencia destaca la eficiencia del diseño de Gemma 2, que logra un rendimiento competitivo e incluso superior con una cantidad de parámetros mucho menor.

Además, Gemma 9B demuestra un desempeño sobresaliente frente a otros modelos

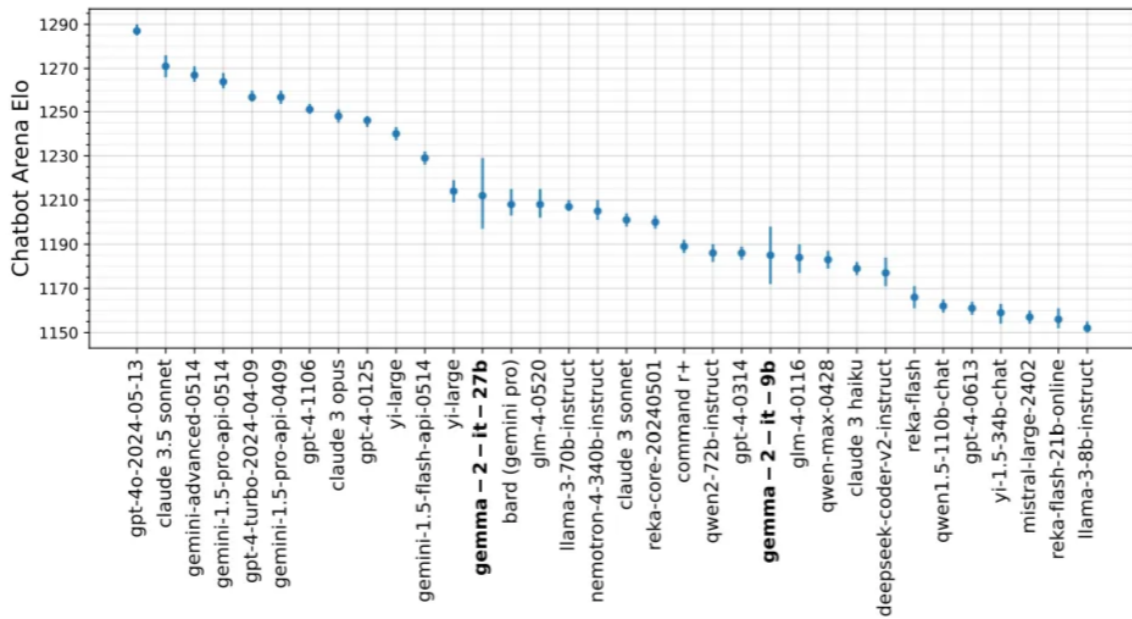


Figura 11: Evaluación de Gemma 2 tras el entrenamiento [25]

en su rango de parámetros, consolidándose como una opción altamente eficiente para tareas de generación de texto. Este resultado refuerza el compromiso de Gemma 2 de ofrecer modelos que optimizan el balance entre capacidad, eficiencia y tamaño, convirtiéndose en una herramienta poderosa para aplicaciones de lenguaje natural sin necesidad de infraestructuras computacionales masivas.

Podemos destacar la importancia de las técnicas de entrenamiento comentadas en la sección anterior, las cuales son capaces de dotar a los modelos de lenguaje de un conocimiento mucho más profundo. Seguiremos viendo en otras secciones el desempeño de Gemma 27B frente a tareas concretas.

## E.5. Qwen 2.5

El primer conjunto de modelos denominados Qwen [28] fue lanzado en 2023 por Alibaba Cloud como un proyecto novedoso de inteligencia artificial generativa, presentando una serie de modelos de lenguaje que incluye modelos base preentrenados y modelos de chat afinados con técnicas de alineación humana (RLHF). Basado en una arquitectura Transformer optimizada, el modelo Qwen introduce numerosas mejoras en atención y eficiencia. Aunque se basa en sistemas como GPT y LLaMA, incorpora configuraciones específicas para aumentar su capacidad de atención y reducir la carga computacional.

Este proyecto utiliza técnicas simples y novedosas aplicadas únicamente a la inferencia para extender la longitud del contexto de estos modelos. Una de las

técnicas clave utilizadas es la *interpolación NTK-aware* [38], que ajusta la base de los embeddings posicionales rotatorios (RoPE) para prevenir la pérdida de información de alta frecuencia, algo que no logra la interpolación posicional estándar. Además, se introdujo una extensión dinámica llamada *interpolación NTK-aware dinámica*, que ajusta la escala por segmentos del contexto, evitando degradaciones severas en el rendimiento. Estas técnicas permiten extender eficazmente la longitud del contexto sin comprometer la eficiencia computacional ni la precisión del modelo.

Qwen también incorpora dos mecanismos de atención adicionales: *LogN-Scaling* [39] y *atención por ventanas (window attention)*. LogN-Scaling reescala el producto escalar entre las consultas y los valores en función de la proporción entre la longitud del contexto y la longitud utilizada durante el entrenamiento, asegurando que la entropía de los valores de atención permanezca estable a medida que crece el contexto. Por su parte, la atención por ventanas limita el alcance de la atención a una ventana de contexto específica, evitando que el modelo preste atención a tokens demasiado alejados.

El entrenamiento de Qwen se desarrolló con un enfoque riguroso, utilizando grandes volúmenes de datos multilingües y multimodales procedentes de diversas fuentes, como literatura científica, contenido web y bases de datos técnicas. Además, los objetivos de entrenamiento van más allá de la simple predicción del siguiente token, buscando enriquecer la capacidad del modelo para comprender y contextualizar la información de manera más efectiva.

Tras el preentrenamiento inicial, Qwen pasó por un proceso de refinamiento que incluyó un *fine-tuning supervisado y aprendizaje por refuerzo con retroalimentación humana (RLHF)*, tal y como hemos ido viendo a lo largo de la descripción de los modelos anteriores.

La imagen 12 muestra el rendimiento de la primera versión de Qwen frente a otros modelos de lenguaje abiertos en múltiples benchmarks, destacando que Qwen 14B supera a todos en tareas de comprensión contextual, razonamiento lógico y generación de texto, incluso frente a modelos más grandes como LLaMA 2 (70B) y StableBeluga2 (70B). Con un diseño eficiente y resultados líderes en pruebas como MMLU, C-Eval, GSM8K y MATH, Qwen combina una alta precisión y menor carga computacional, consolidándose como una opción superior para aplicaciones avanzadas de procesamiento de lenguaje natural.

A principios de 2024 se presentó la nueva versión de la serie Qwen, el Qwen 1.5 [27]. Las principales mejoras frente a la versión anterior son las siguientes:

- Se hizo hincapié en profundizar en el código fuente y desarrollar seis tamaños

Model	Params	MMLU 5-shot	C-Eval 5-shot	GSM8K 8-shot	MATH 4-shot	HumanEval 0-shot	MBPP 3-shot	BBH 3-shot
MPT	7B	30.8	23.5	9.1	3.0	18.3	22.8	35.6
	30B	47.9	-	15.2	3.1	25.0	32.8	38.0
Falcon	7B	27.8	-	6.8	2.3	-	11.2	28.0
	40B	57.0	-	19.6	5.5	-	29.8	37.1
ChatGLM2	6B	47.9	51.7	32.4	6.5	-	-	33.7
InternLM	7B	51.0	53.4	31.2	6.3	10.4	14.0	37.0
	20B	62.1	58.8	52.6	7.9	25.6	35.6	52.5
Baichuan2	7B	54.7	56.3	24.6	5.6	18.3	24.2	41.6
	13B	59.5	59.0	52.8	10.1	17.1	30.2	49.0
LLaMA	7B	35.6	27.3	11.0	2.9	12.8	17.7	33.5
	13B	47.7	31.8	20.3	4.2	15.8	22.0	37.9
	33B	58.7	37.5	42.3	7.1	21.7	30.2	50.0
	65B	63.7	40.4	54.4	10.6	23.7	37.7	58.4
LLAMA 2	7B	46.8	32.5	16.7	3.3	12.8	20.8	38.2
	13B	55.0	41.4	29.6	5.0	18.9	30.3	45.6
	34B	62.6	-	42.2	6.2	22.6	33.0	44.1
	70B	69.8	50.1	63.3	13.5	29.9	45.0	64.9
StableBeluga2	70B	68.6	51.4	69.6	14.6	28.0	11.4	69.3
QWEN	1.8B	44.6	54.7	21.2	5.6	17.1	14.8	28.2
	7B	58.2	63.5	51.7	11.6	29.9	31.6	45.0
	14B	<b>66.3</b>	<b>72.1</b>	<b>61.3</b>	<b>24.8</b>	<b>32.3</b>	<b>40.8</b>	<b>53.4</b>

Figura 12: Evaluación de Qwen con benchmarks ampliamente usados frente a modelos open source base [28]

distintos: 0.5B, 1.8B, 4B, 7B, 14B, 32B, 72B y 110B.

- Mejoras en la alineación con preferencias humanas
- Capacidades multilingües añadidas en 12 lenguas europeas
- Admisión de contextos de hasta 32.000 tokens



Figura 13: Comparación multilingüe entre GPT-3.5 y Qwen1.5 en lenguas europeas [27]

La figura 13 compara el rendimiento de Qwen1.5-72B-Chat y GPT-3.5 en varios idiomas, destacando la ventaja de Qwen1.5 en la mayoría de ellos. En general, el modelo supera consistentemente a GPT-3.5, donde podemos destacar el español, el cual es el idioma que nos conviene en este proyecto.

Por último, a finales de 2024 se presentaron las últimas versiones a día de hoy de este modelo, **Qwen 2** [26] y **Qwen 2.5** [23]. Centrándonos en las mejoras de este último,

Qwen2.5 introduce innovaciones notables en tamaño, datos y capacidades, superando a sus versiones anteriores. Ofrece nuevos modelos de 3B, 14B y 32B parámetros, así como variantes avanzadas como Qwen2.5-Turbo y Qwen2.5-Plus.

En cuanto al **preentrenamiento**, Qwen 2.5 introdujo avances significativos en la calidad de los datos, técnicas de filtrado y estrategias de mezcla de datos, ampliando el conjunto de entrenamiento de 7 a 18 billones de tokens. Las mejoras incluyeron el uso de filtrado avanzado de datos mediante Qwen2-Instruct, que evalúa y retiene datos de alta calidad multilingües; la inclusión de datos especializados en matemáticas y código con conjuntos específicos como Qwen2.5-Math y Qwen2.5-Coder para un rendimiento destacado en estas áreas; y la generación de datos sintéticos de alta calidad, filtrados con modelos avanzados de recompensa para asegurar su precisión. Además, una mezcla optimizada de datos prioriza dominios de alto valor como ciencia y tecnología, reduciendo el sesgo hacia dominios como redes sociales. Estas estrategias han enriquecido el conjunto de datos, mejorando la capacidad de Qwen 2.5 para resolver tareas complejas en múltiples dominios. Por último, Qwen2.5 amplió significativamente su capacidad para manejar contextos largos mediante un enfoque progresivo. El entrenamiento comenzó con contextos de 4,096 tokens, que se extendieron a 32,768 tokens en las etapas finales.

El **postentrenamiento** de Qwen2.5 se distingue por las técnicas ya mencionadas SFT y RL que optimizan la generación de respuestas, el manejo de contextos largos y la capacidad de seguir instrucciones complejas.

Durante la fase de fine-tuning supervisado, se amplió significativamente la cobertura de datos, incorporando conjuntos especializados en generación de texto largo, matemáticas, programación, razonamiento lógico y datos estructurados. Esta fase incluyó técnicas como *backtranslation* y validación automatizada, lo que garantiza la calidad y precisión de los datos de entrenamiento. El modelo se entrenó con una longitud de contexto de hasta 32.768 tokens y utilizó muestras cuidadosamente seleccionadas para maximizar su capacidad en tareas específicas.

El aprendizaje por refuerzo (RL) se divide en dos etapas: RL offline y RL online. En la etapa offline, se prepara una señal de entrenamiento basada en tareas objetivas como matemáticas y razonamiento lógico, asegurando la calidad de las respuestas mediante validaciones humanas y automatizadas. La etapa online utiliza criterios como veracidad, utilidad, concisión, relevancia y neutralidad para perfeccionar la alineación del modelo con las expectativas humanas, empleando técnicas avanzadas como Group Relative Policy Optimization (GRPO) para mejorar el aprendizaje.

La figura 14 presenta los resultados de evaluación de modelos base más pequeños en varias categorías de tareas, como tareas generales, matemáticas y ciencias,

Datasets	Mistral-7B	Llama3-8B	Gemma2-9B	Qwen2-7B	Qwen2.5-7B
<i>General Tasks</i>					
MMLU	64.2	66.6	71.3	70.3	<b>74.2</b>
MMLU-pro	30.9	35.4	44.7	40.1	<b>45.0</b>
MMLU-redux	58.1	61.6	67.9	68.1	<b>71.1</b>
BBH	56.1	57.7	68.2	62.3	<b>70.4</b>
ARC-C	60.0	59.3	<b>68.2</b>	60.6	63.7
TruthfulQA	42.2	44.0	45.3	54.2	<b>56.4</b>
Winogrande	78.4	77.4	<b>79.5</b>	77.0	75.9
HellaSwag	<b>83.3</b>	82.1	81.9	80.7	80.2
<i>Mathematics &amp; Science Tasks</i>					
GPQA	24.7	25.8	32.8	30.8	<b>36.4</b>
TheoremQA	19.2	22.1	28.9	29.6	<b>36.0</b>
MATH	10.2	20.5	37.7	43.5	<b>49.8</b>
MMLU-stem	50.1	55.3	65.1	64.2	<b>72.3</b>
GSM8K	36.2	55.3	70.7	80.2	<b>85.4</b>
<i>Coding Tasks</i>					
HumanEval	29.3	33.5	37.8	51.2	<b>57.9</b>
HumanEval+	24.4	29.3	30.5	43.3	<b>50.6</b>
MBPP	51.1	53.9	62.2	64.2	<b>74.9</b>
MBPP+	40.9	44.4	50.6	51.9	<b>62.9</b>
MultiPL-E	29.4	22.6	34.9	41.0	<b>50.3</b>
<i>Multilingual Tasks</i>					
Multi-Exam	47.1	52.3	<b>61.2</b>	59.2	59.4
Multi-Understanding	63.3	68.6	78.3	72.0	<b>79.3</b>
Multi-Mathematics	26.3	36.3	53.0	57.5	<b>57.8</b>
Multi-Translation	23.3	31.9	<b>36.5</b>	31.5	32.4

Figura 14: Evaluación de modelos base ligeros de Qwen 2.5 [27]

codificación y multilingüismo. Entre los modelos analizados, *Qwen2.5-3B* se destaca consistentemente como el más sólido en la mayoría de las métricas, logrando resultados superiores en tareas generales (como MMLU con 65.6) y en matemáticas y ciencias (GSM8K con 79.1). Principalmente, en este proyecto se hará hincapié en capítulos futuros en tareas generales, donde veremos el rendimiento de todos los modelos mencionados en esta sección.

Por otro lado, la imagen 15 muestra resultados de evaluación de las versiones de peso intermedio de Qwen 2.5. En términos generales, el modelo *Qwen2.5-32B* se consolida como líder en su categoría, superando a la competencia en prácticamente todos los benchmarks evaluados. En MMLU, alcanza un puntaje de 83.3, destacándose frente a modelos como Gemma2-27B (75.2) y Yi-1.5-34B (77.2). También sobresale en conjuntos como BBH (84.5) y Hellaswag (85.2), mostrando una capacidad sobresaliente para el razonamiento y la comprensión general. Estos resultados reafirman a *Qwen2.5-32B* como el modelo líder, incluso frente a competidores de mayor tamaño.

En tareas multilingües, *Qwen2.5-32B* demuestra ser altamente eficiente, logrando resultados sobresalientes en Multi-Understanding (88.4) y Multi-Exam (75.4), superando a todos los modelos evaluados. En tareas específicas como

Datasets	Qwen1.5-32B	Gemma2-27B	Yi-1.5-34B	Qwen2.5-Turbo	Qwen2.5-14B	Qwen2.5-32B
<i>General Tasks</i>						
MMLU	74.3	75.2	77.2	79.5	79.7	<b>83.3</b>
MMLU-pro	44.1	49.1	48.3	<b>55.6</b>	51.2	55.1
MMLU-redux	69.0	-	74.1	77.1	76.6	<b>82.0</b>
BBH	66.8	74.9	76.4	76.1	78.2	<b>84.5</b>
ARC-C	63.6	<b>71.4</b>	65.6	67.8	67.3	70.4
TruthfulQA	57.4	40.1	53.9	56.3	<b>58.4</b>	57.8
Winogrande	81.5	59.7	<b>84.9</b>	81.1	81.0	82.0
Hellaswag	85.0	<b>86.4</b>	85.9	85.0	84.3	85.2
<i>Mathematics &amp; Science Tasks</i>						
GPQA	30.8	34.9	37.4	41.4	32.8	<b>48.0</b>
Theoremqa	28.8	35.8	40.0	42.1	43.0	<b>44.1</b>
MATH	36.1	42.7	41.7	55.6	55.6	<b>57.7</b>
MMLU-stem	66.5	71.0	72.6	77.0	76.4	<b>80.9</b>
GSM8K	78.5	81.1	81.7	88.3	90.2	<b>92.9</b>
<i>Coding Tasks</i>						
HumanEval	43.3	54.9	46.3	57.3	56.7	<b>58.5</b>
HumanEval+	40.2	46.3	40.2	51.2	51.2	<b>52.4</b>
MBPP	64.2	75.7	65.5	76.2	76.7	<b>84.5</b>
MBPP+	53.9	60.2	55.4	63.0	63.2	<b>67.2</b>
MultiPL-E	38.5	48.0	39.5	53.9	53.5	<b>59.4</b>
<i>Multilingual Tasks</i>						
Multi-Exam	61.6	65.8	58.3	70.3	70.6	<b>75.4</b>
Multi-Understanding	76.5	82.2	73.9	85.3	85.9	<b>88.4</b>
Multi-Mathematics	56.1	61.6	49.3	71.3	68.5	<b>73.7</b>
Multi-Translation	33.5	<b>38.7</b>	30.0	36.8	36.2	37.3

Figura 15: Evaluación de modelos base de peso intermedio de Qwen 2.5 [27]

Multi-Mathematics (73.7) y Multi-Translation (37.3), mantiene un desempeño sólido, siendo superior o comparable al resto.

## E.6. Modelos de Lenguaje Multimodales y sus posibles aplicaciones

En el marco presente de este proyecto, se planteó la posibilidad del uso de **Modelos de Lenguaje Multimodales (MLLMs)** para analizar y seleccionar imágenes entre un conjunto de ellas, de forma que se ajusten a un texto dado.

Los MLLMs revolucionaron el ámbito de la inteligencia artificial en 2023, presentándose arquitecturas novedosas capaces de comprender y generar contenido en un espectro de formatos que incluyen texto, imágenes, audio y video.

Particularmente, se han estado realizando pruebas a día de hoy con el MLLM **LLaVA (Large Language and Vision Assistant)** [40], el cual combina un modelo de lenguaje preentrenado (en este caso, Vicuna) con un modelo visual (CLIP ViT-L/14 [42]) para tareas multimodales. Su arquitectura, ilustrada en la Figura 16, busca integrar de manera eficiente características visuales y lingüísticas, permitiendo al sistema procesar imágenes y responder a instrucciones textuales.

La imagen 16 muestra cómo las características visuales extraídas por el encoder CLIP se transforman en el espacio de embeddings lingüísticos mediante una matriz de proyección antes de combinarse con las instrucciones textuales.

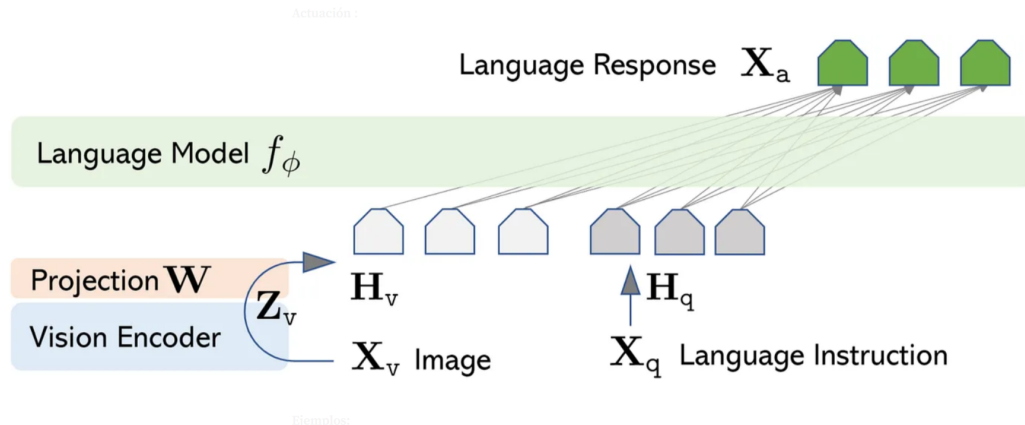


Figura 16: Arquitectura de LLaVA [40]

En el caso del entrenamiento, se comienza con la extracción de características visuales ( $Z_v$ ) a partir de imágenes ( $X_v$ ) utilizando el encoder visual de CLIP. Como se observa en la figura 16, estas características se transforman mediante una matriz de proyección entrenable ( $W$ ), convirtiéndolas en un espacio de embeddings lingüísticos ( $H_v$ ), compatible con el modelo de lenguaje. Paralelamente, las instrucciones textuales ( $X_q$ ) se integran como representaciones lingüísticas ( $H_q$ ). Posteriormente, ambos tipos de tokens ( $H_v$  y  $H_q$ ) se combinan y se introducen en el modelo de lenguaje para generar respuestas coherentes ( $X_a$ ).

Una vez descrita la arquitectura y el proceso de entrenamiento del modelo LLaVA, se propuso emplearlo en este proyecto para evaluar su capacidad en la selección de imágenes relevantes que acompañen textos descriptivos, específicamente tweets basados en datos de audiencias televisivas. El objetivo principal era analizar hasta qué punto este modelo multimodal puede interpretar y relacionar contenido visual y textual de manera coherente, lo que resulta crucial para garantizar la adecuación semántica de las imágenes al mensaje.

El flujo de trabajo consistió en presentar al modelo un conjunto de imágenes candidatas, extraídas previamente mediante crawling web, junto con un texto base que representa el contenido del tweet. Mediante un prompt cuidadosamente diseñado, LLaVA debía analizar las imágenes y seleccionar aquella que mejor se ajustara al mensaje textual. Este enfoque permitió aprovechar las capacidades multimodales del modelo, integrando tanto información visual como lingüística para fundamentar sus decisiones.

Los resultados de estas pruebas fueron documentados y comparados con las elecciones realizadas manualmente por el cliente, todo ello adjunto en el anexo B, incluyendo los prompts utilizados, las imágenes analizadas y las respuestas generadas por el modelo. En el ejemplo B.1 podemos ver el caso del tweet de Antena 3 sobre

”La Ruleta de la Suerte”, donde el modelo escogió correctamente la misma imagen seleccionada por el cliente, mostrando un alineamiento prometedor entre la evaluación automatizada y el criterio humano. Sin embargo, en otros casos, como el tweet de Aragón TV en el ejemplo B.2 sobre ”Basura o Tesoro”, se observaron discrepancias, ya que el modelo escoge la imagen central y el cliente se decanta por la superior.

El desempeño de LLaVA en la selección de imágenes demostró un notable potencial para automatizar esta tarea, especialmente al considerar su capacidad para analizar contenido visual y relacionarlo con el texto de manera coherente. En ejemplos como el tweet de Antena 3, el modelo identificó correctamente la imagen más adecuada, alineándose con la elección humana y mostrando comprensión tanto del contexto semántico del mensaje como de los elementos visuales relevantes. Esto evidencia que, pese a algunas discrepancias en otros casos, el modelo tiene la capacidad de realizar evaluaciones fundamentadas, lo que sugiere que con ajustes y entrenamiento más específico, podría optimizarse aún más para garantizar una mayor precisión en la asignación de imágenes, reduciendo la dependencia del criterio manual en contextos de alto volumen de contenido.

Aunque actualmente la selección de imágenes sigue siendo una tarea manual priorizada por los clientes, estas pruebas demuestran el potencial de LLaVA para automatizar este proceso en el futuro. Con mejoras adicionales, como entrenarlo con los gustos del cliente, el modelo podría convertirse en una herramienta valiosa, especialmente en contextos donde se maneja un gran volumen de contenido visual. Este registro no solo permite evaluar su rendimiento, sino que también sirve como base para explorar futuras aplicaciones que equilibren la automatización y la personalización en la selección de imágenes.

## F. Plantilla de Evaluación

En este anexo se incluye la plantilla utilizada para la evaluación de los tweets generados por los modelos de lenguaje.

# Evaluación de LLMs

## Escenario 1: Usuario con bajo nivel de conocimiento sobre LLMs

En este escenario, un usuario con conocimientos limitados en modelos de lenguaje e inteligencia artificial evaluará cuatro tweets generados por distintos modelos. El objetivo es seleccionar los dos tweets que mejor se ajusten a los criterios establecidos y asignar puntuaciones en distintos aspectos cualitativos.

### Tweets evaluados

- **Tweet 1:** Generado por Qwen2.5 14B-instruct q5 K M

@antena3com logró 2.620.000 espectadores únicos, obtuvo una audiencia media de 1.206.000 y congregó un 15.7% de cuota de pantalla con @laruletasuerte, máximo histórico de espectadores únicos del programa 📺🔥. #AudienceStats #MaxViewers

- **Tweet 2:** Generado por Vicuna 13B v1.5 q4

📺🔥 @laruletasuerte consiguió un máximo histórico de espectadores únicos con 2.620.000 y una audiencia media de 1.206.000, logrando una cuota del 15.7%. ¡No se puede pasar! #AudienciaHistórica #Antena3Com #Laruitasuerte

- **Tweet 3:** Generado por Qwen2.5 32B

@laruletasuerte @antena3com congregó a 2.620.000 espectadores únicos, obtuvo una audiencia media de 1.206.000 y alcanzó una cuota de pantalla del 15.7%. Este resultado marca el máximo histórico de espectadores únicos del programa. 📺🔥 #HistoricoEnAudiencia #Laruletasuerte📈

- **Tweet 4:** Generado por Gemma2 27B instruct q5 K M

@laruletasuerte en @antena3com congregó 2.620.000 espectadores únicos. Logró una audiencia media de 1.206.000 y obtuvo un 15.7% de cuota. Máximo histórico de espectadores únicos del programa. 📺🔥 #Audiencias #LaRuletaDeLaSuerte #Antena3

### Criterios de selección

El usuario deberá elegir los dos tweets que mejor cumplan con las siguientes especificaciones del cliente:

- Ser puramente informativos, sin incitar a ver el programa.
- Ser claros, concisos y con una redacción natural.
- Incluir al menos dos hashtags y dos emojis.
- Incluir espectadores únicos, audiencia media, cuota, nombre del programa y/o cadena y el evento “máximo histórico” en este caso.

Tweets seleccionados:

### Evaluación de los tweets seleccionados

Una vez elegidos los dos tweets más adecuados, el usuario los puntuará en una escala del 1 al 5 según los siguientes criterios:

<b>Criterio</b>	<b>Tweet seleccionado 1</b>	<b>Tweet seleccionado 2</b>
<b>Naturalidad</b> (parece humano o automático)		
<b>Concisión</b> (es claro y sin información innecesaria)		
<b>Originalidad</b> (variedad en la redacción)		
<b>Nivel de alucinación</b> (información errónea o inventada)		
<b>Hashtags y emojis</b> (adecuación y pertinencia)		
<b>Evaluación general</b> (valoración global del tweet)		

### Comentarios adicionales

El usuario podrá añadir observaciones sobre los tweets seleccionados, destacando fortalezas o posibles mejoras en la generación automática de contenido.

#### Observaciones:

- **Tweet 1:** \_\_\_\_\_
- **Tweet 2:** \_\_\_\_\_