Máster en Ingeniería Informática

Aplicaciones Avanzadas de la Inteligencia Artificial 2024-2025
Grupo 1

Práctica 3

"IA en Empresa"

Jorge Ramos Santana – 100451001
Carlos Salinas Gancedo – 100549334
Luis Daniel Casais Mezquida – 100429021

Equipo 1

Profesor
María Paula de Toledo

Índice

Índice de figuras

1. Introducción

En este trabajo se presenta un clasificador de noticias basado en modelos de lenguaje preentrenados. El objetivo es categorizar textos periodísticos en cinco clases diferentes: negocios (*business*), entretenimiento (*entertainment*), política (*politics*), deportes (*sports*) y tecnología (*tech*).

1.1. Dataset

Para la realización de este proyecto se ha utilizado el dataset BBC Full Text Document Classification, una colección que consta de 2225 documentos provenientes del portal web de noticias BBC de los años 2004 y 2005. Estos documentos están etiquetados en las cinco clases mencionadas anteriormente, lo que permite un entrenamiento y evaluación supervisados.

1.2. Motivación

La clasificación automática de textos periodísticos tiene numerosas aplicaciones prácticas, desde la organización de contenido en portales digitales hasta el análisis de tendencias informativas. Este proyecto explora cómo los modelos de lenguaje más modernos pueden abordar esta tarea tradicional de procesamiento del lenguaje natural, comparando diferentes arquitecturas y configuraciones para determinar su eficacia. Uno de los problemas más comunes en la clasificación de textos es la ambigüedad y la superposición temática entre categorías. Por ejemplo, un artículo sobre una nueva tecnología en el ámbito deportivo podría clasificarse tanto como *sports* como *tech*. Por otro lado, un artículo satírico sobre política podría ser difícil de clasificar debido a su tono humorístico. Estos desafíos hacen que la tarea de clasificación sea compleja y requieren modelos que puedan captar matices sutiles en el lenguaje. Además, en el entorno de la empresa, podría ser util tener un clasificador de textos, pero no se tiene un dataset etiquetado para entrenar un modelo, o se tiene un dataset pequeño. En estos casos, los modelos de lenguaje preentrenados pueden ser una solución efectiva, ya que pueden adaptarse a nuevas tareas con poco o ningún entrenamiento adicional.

2. Metodología

2.1. Enfoque basado en modelos de lenguaje preentrenados

Para este proyecto hemos optado por un enfoque más moderno, utilizando LLMs (Large Langue Models). A diferencia de los enfoques clásicos, estos modelos aprovechan el conocimiento adquirido durante su preentrenamiento con enormes cantidades de texto, lo

que les permite comprender el contexto, la semántica y las sutilezas del lenguaje humano sin necesidad de entrenamiento específico para la tarea.

Los modelos evaluados en este trabajo incluyen familias como:

- **Mistral**: Creados por Mistral AI, una startup francesa, estos modelos son conocidos por su arquitectura eficiente y su capacidad para manejar tareas complejas de procesamiento del lenguaje natural. El primer modelo de esta familia, Mistral 7B, supuso un gran avance en la capacidad de los modelos de lenguaje disponibles abiertamente. [1]
- **Llama 3**: Desarollados por Meta, y pioneros en la creación de modelos de lenguaje abiertos. [2]
- **Qwen 2.5**: Desarollados por Alibaba, estos modelos son conocidos por su capacidad para realizar tareas de lenguaje natural con alta precisión [3]
- **Gemma 3**: Desarollados por Google, son el homólogo de los modelos Gemini, pero optimizados para ser más pequeños y eficientes. [4]
- **SMoLLM**: Modelos experimentales desarollados por Hugginface, optimizados para ser más pequeños y eficientes. [5]

Es importante destacar que se ha utilizado la version *instruct* de cada modelo, que ha sido entrenada para seguir instrucciones y responder preguntas de manera más efectiva. Una alternativa sería utilizar el modelo *base*, que con técnicas de *few-shot prompting* podría adaptarse a la tarea de clasificación. Esta opción se considera como posible trabajo futuro.

Todos los modelos evaluados son *open weights*, lo que significa que están disponibles para su uso y modificación por parte de la comunidad. Esto permite una mayor transparencia y colaboración en el desarrollo de modelos de lenguaje, así como la posibilidad de adaptarlos a necesidades específicas. Además, la capacidad de poder desplegar los modelos en un entorno local, sin depender de servicios externos, es una ventaja significativa en términos de privacidad y control sobre los datos para empresas que manejan información sensible.

2.2. Alternativas clásicas

Aunque nuestro enfoque se centra en modelos de lenguaje preentrenados, es importante mencionar que la clasificación de textos ha sido tradicionalmente abordada con métodos como:

- **TF-IDF**: Pondera la importancia de las palabras según su frecuencia en el documento y su rareza en el corpus.

- **Naive Bayes**: Un clasificador probabilístico basado en el teorema de Bayes, asumiendo independencia entre características.
- **SVM**: Máquinas de soporte vectorial que intentan encontrar el hiperplano que mejor separa las clases.

Estos métodos tradicionales suelen requerir una fase de extracción de características más elaborada y un entrenamiento específico para la tarea de clasificación, mientras que los modelos de lenguaje modernos pueden abordar la tarea directamente mediante instrucciones en lenguaje natural.

2.3. Implementación del clasificador

La implementación de nuestro clasificador se basa en un enfoque de *prompt engineering*. A través de la API de OpenAI (configurada para conectarse a un servidor local que ejecuta los modelos), enviamos el texto de la noticia junto con instrucciones específicas que indican al modelo qué tarea debe realizar.

```
def classify_text(text):
    """Classify a single text using the model"""
    messages = [
        {"role": "system", "content": "You are a text classifier. Given a
            news article, classify it into one of the following
            categories: business, entertainment, politics, sports, tech.
            You must return a json containing the classification. "},
        {"role": "user", "content": text}
    ]

    # Define the expected response structure
    character_schema = {
        "type": "json_schema",
        "json_schema": {
            "name": "characters",
            "schema": {
                "type": "object",
                "properties": {
                    "classification": {
                    "type": "string",
                    "description": "The classification of the item",
                    "enum": [
                        "business",
                        "entertainment",
                        "politics",
                        "sports",
```

```
                    "tech"
                ]
                }
            },
            "required": [
                "classification"
            ],
        },
    }
}

try:
    response = client.chat.completions.create(
        model="qwen2.5-7b-instruct-mlx@4bit",
        messages=messages,
        response_format=character_schema,
        temperature=0 # Set temperature to 0 for deterministic outputs
    )
    results = json.loads(response.choices[0].message.content)
    return results["classification"]
except Exception as e:
    print(f"Error classifying text: {e}")
    return None
```

2.4. Uso de JSON Schema para respuestas estructuradas

Uno de los aspectos más importantes de nuestra implementación es el uso de JSON Schema para garantizar que las respuestas del modelo sean estructuradas y válidas. Esta técnica permite:

- **Formato consistente**: Asegurar que todas las respuestas sigan una estructura predefinida.
- **Validación de tipos**: Garantizar que los valores devueltos corresponden a los tipos esperados.
- **Restricción de valores**: Limitar las posibles clasificaciones a las cinco categorías definidas.
- **Parseo automático**: Facilitar el procesamiento posterior de las respuestas.

En nuestro caso, definimos un esquema JSON que especifica un objeto con una propiedad obligatoria llamada classification, que debe ser una cadena de texto cuyo valor está restringido a una de las cinco categorías.

Al utilizar el parámetro `response_format` con este esquema, instruimos al modelo para que genere únicamente respuestas que cumplan con estas restricciones, evitando así respuestas ambiguas, múltiples clasificaciones o categorías no contempladas.

3. Configuración experimental

3.1. Hardware

Para la evaluación de los modelos se utilizó un MacBook Pro de 14 pulgadas equipado con un chip Apple M3 Max y 128GB de memoria unificada. Esta configuración de gama alta permitió ejecutar localmente todos los modelos evaluados, incluso aquellos de mayor tamaño, sin necesidad de recurrir a servicios externos.

3.2. Software

Los modelos fueron ejecutados utilizando LM Studio como servidor local, permitiendo su uso a través de una API compatible con OpenAI. Para el procesamiento y evaluación de los resultados se utilizó Python con bibliotecas como scikit-learn para el cálculo de métricas y matplotlib para la visualización.

3.3. Modelos evaluados

Se evaluaron múltiples modelos de diferentes familias y tamaños:

- **Familia Mistral**: Mistral Large 3 (123B), Mistral Small 3 (24B)
- **Familia Llama**: Llama 3 (1B, 3B, 8B, 70B)
- **Familia Qwen**: Qwen 2.5 (0.5B, 1.5B, 3B, 7B, 32B)
- **Familia Gemma**: Gemma 3 (4B)
- **SMoLLM**: SMoLLM (135M, 360M)

Para cada modelo, se evaluó su precisión en la clasificación, así como el tiempo de inferencia y los requisitos de memoria.

4. Resultados y discusión

4.1. Comparativa de modelos

La siguiente figura muestra una comparativa de la precisión alcanzada por los diferentes modelos evaluados:

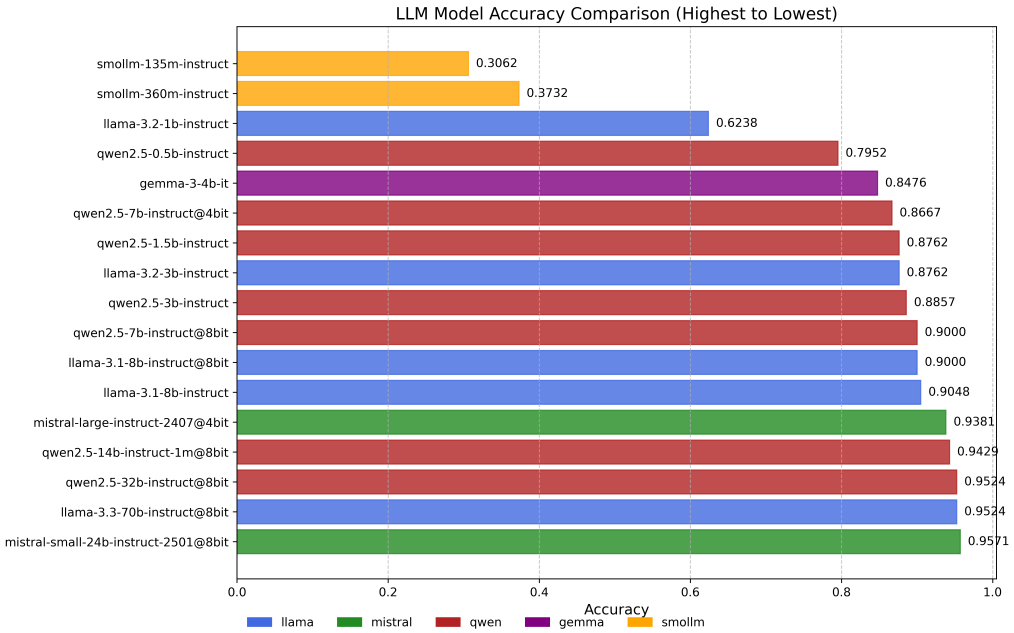

Fig. 1: *Comparativa de precisión entre los diferentes modelos evaluados*

Como se puede observar, Mistral Small 24B alcanzó la mayor precisión (95.71 %), seguido por modelos como Qwen 2.5 32B y Llama 3 70B, lo que sugiere que los modelos de mayor tamaño tienden a ofrecer mejores resultados en esta tarea. Sin embargo, es destacable que algunos modelos pequeños como Qwen2.5 1.5B lograron resultados competitivos, resaltando la importancia de la arquitectura y el entrenamiento del modelo.

4.2. Análisis de relación tamaño-rendimiento

Para las principales familias de modelos, analizamos la relación entre el tamaño del modelo y su precisión:

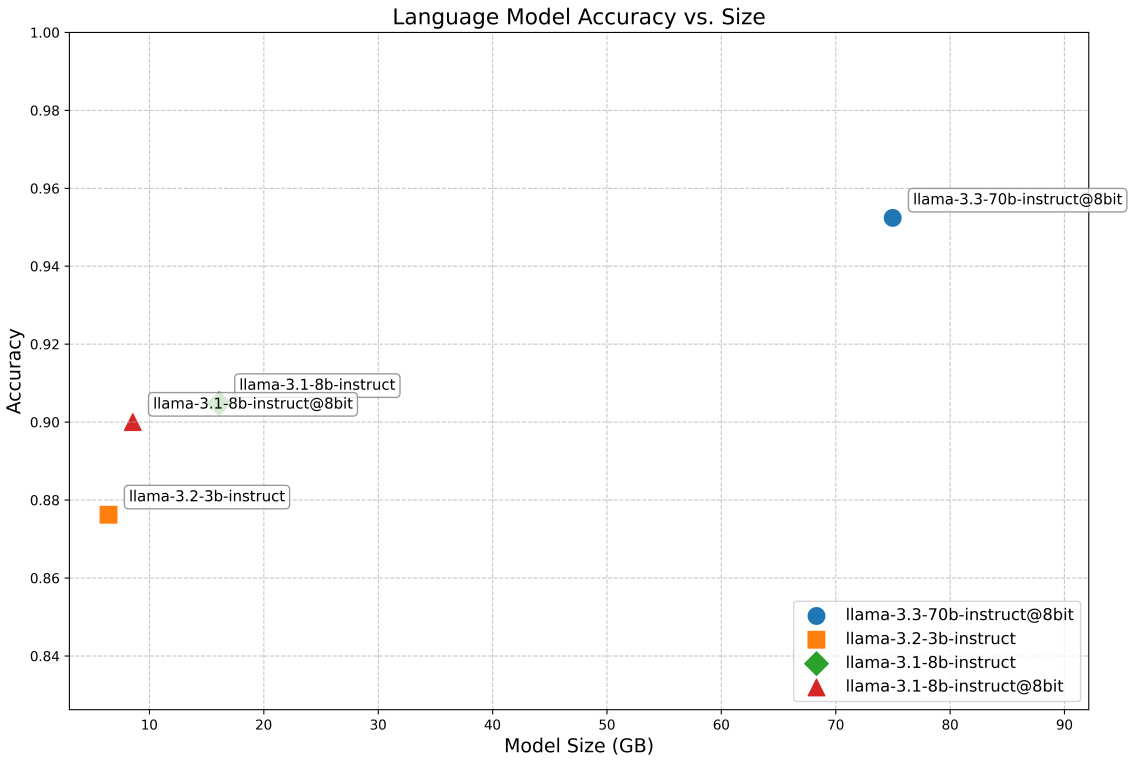

Fig. 2: *Familia Llama: Precisión vs. Tamaño*

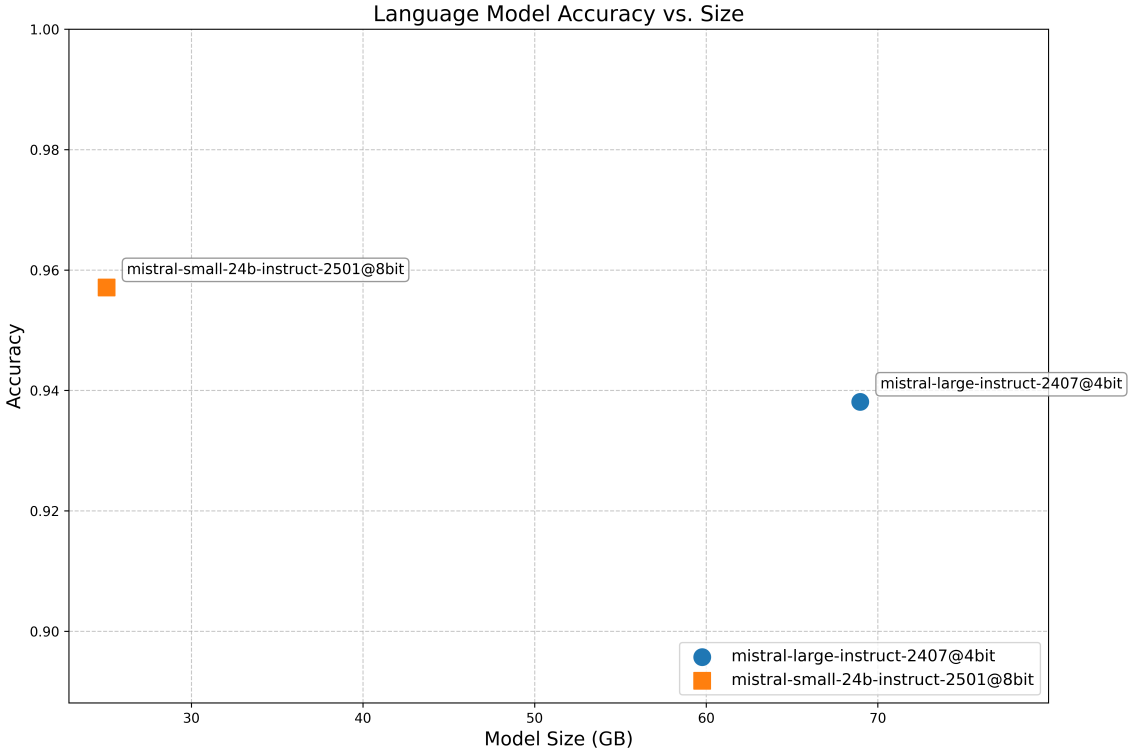

Fig. 3: *Familia Mistral: Precisión vs. Tamaño*

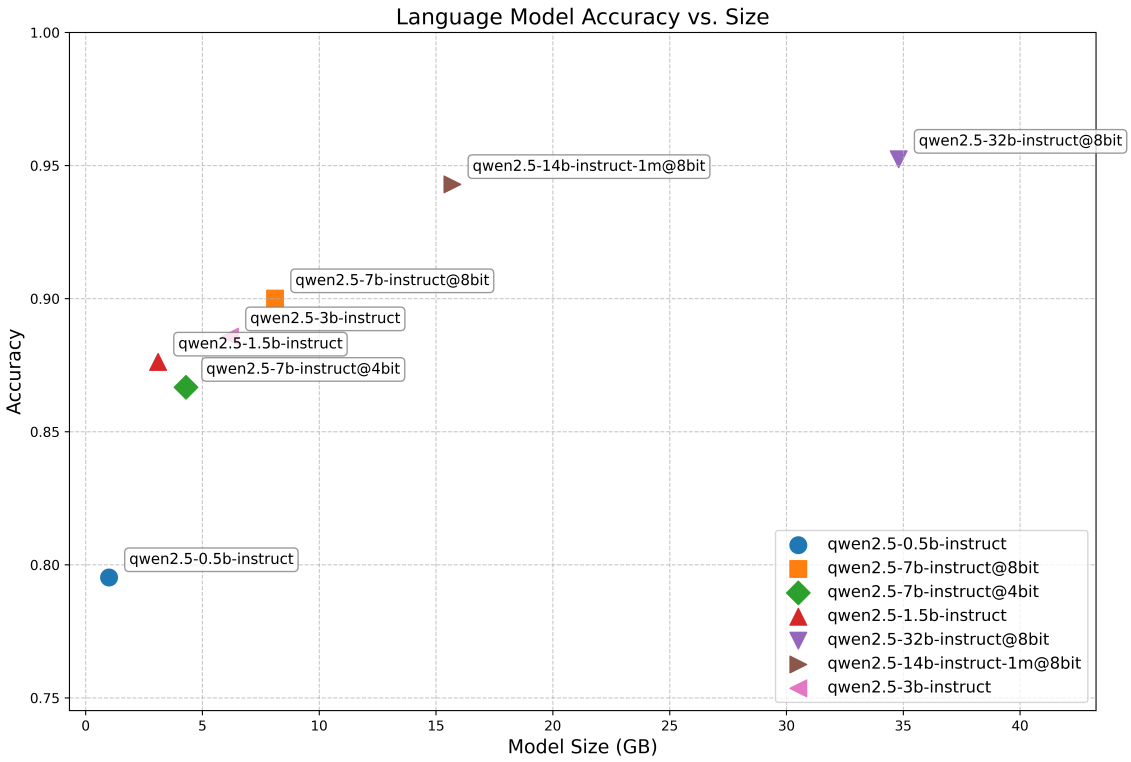

Fig. 4: *Familia Qwen 2.5: Precisión vs. Tamaño*

4.3. Análisis de relación tiempo-rendimiento

También analizamos la relación entre el tiempo de inferencia y la precisión:

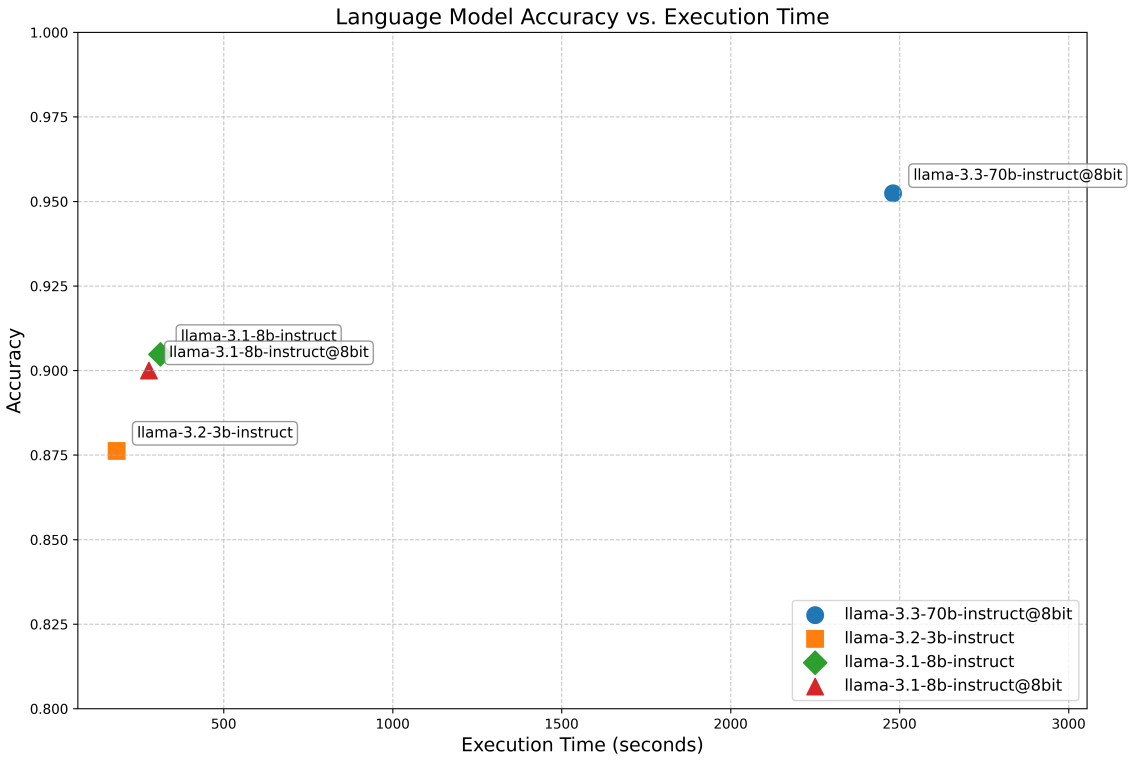

Fig. 5: *Familia Llama: Precisión vs. Tiempo*

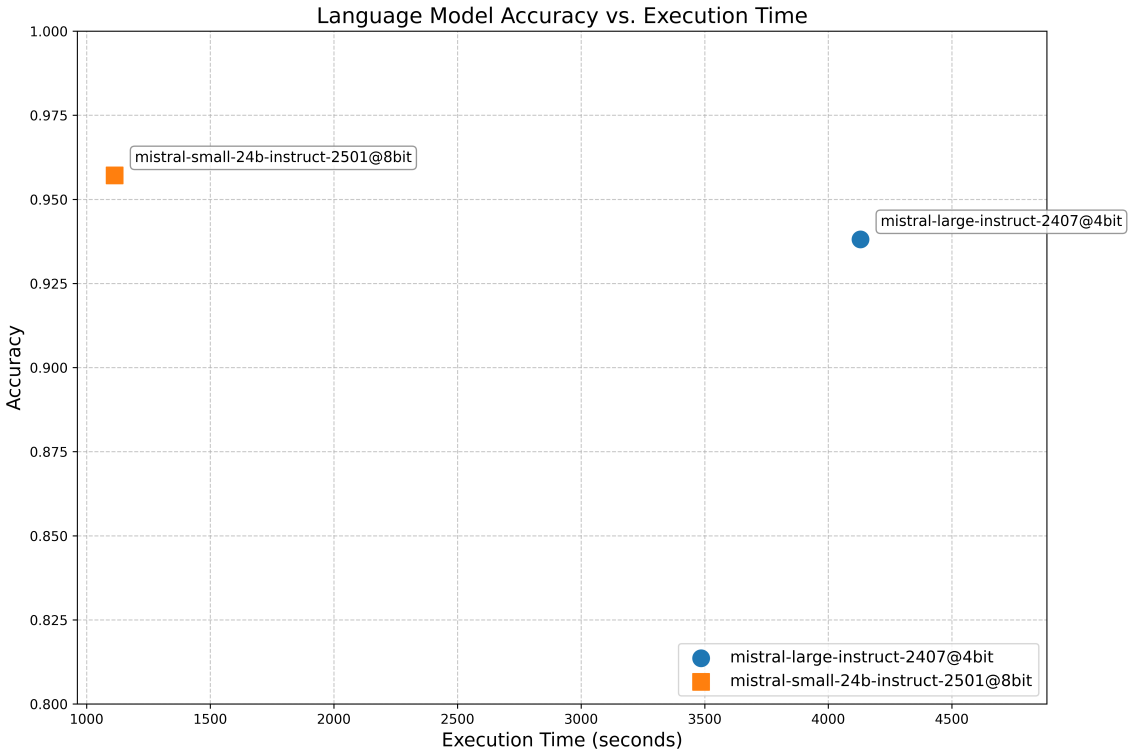

Fig. 6: *Familia Mistral: Precisión vs. Tiempo*

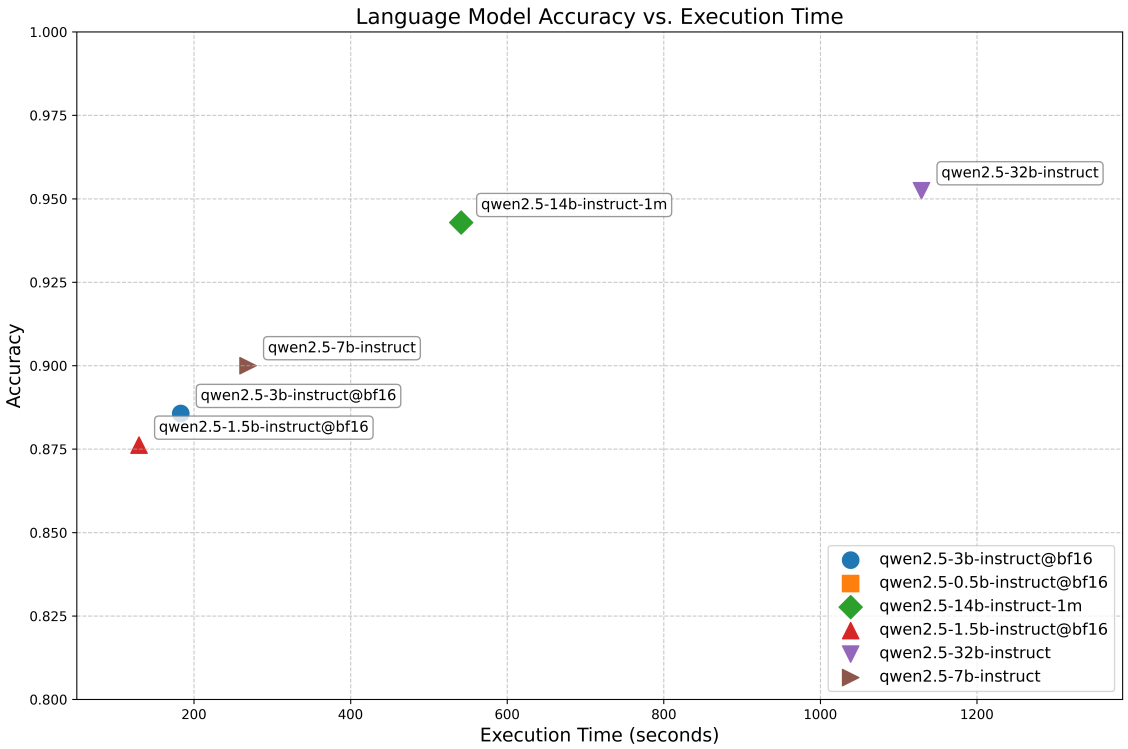

Fig. 7: *Familia Qwen 2.5: Precisión vs. Tiempo*

4.4. Análisis detallado del mejor modelo: Mistral Small

El modelo Mistral Small 24B mostró el mejor rendimiento en nuestras pruebas, con una precisión del 95.71 %. A continuación se presenta un análisis detallado de sus resultados:

Tabla 1: *Matriz de confusión para Mistral Small 24B*

	Business	**Entertainment**	**Politics**	**Sports**	**Tech**
Business	47	0	2	1	0
Entertainment	1	28	1	0	0
Politics	2	0	38	0	0
Sports	0	0	0	50	0
Tech	0	1	0	1	38

Como se observa en la matriz de confusión, el modelo muestra un rendimiento especialmente destacado en la categoría de deportes, donde clasifica correctamente todos los ejemplos. Las mayores confusiones se producen entre las categorías de negocios y política, lo que tiene sentido dada la superposición temática que puede existir entre estas áreas.

Tabla 2: *Métricas por categoría para Mistral Small 24B*

Categoría	Precisión	Exhaustividad	F1-Score	Soporte
Business	0.94	0.94	0.94	50
Entertainment	0.97	0.93	0.95	30
Politics	0.93	0.95	0.94	40
Sports	0.96	1.00	0.98	50
Tech	1.00	0.95	0.97	40
Promedio macro	0.96	0.95	0.96	210
Promedio ponderado	0.96	0.96	0.96	210

Las métricas detalladas por categoría confirman el excelente rendimiento del modelo, con valores de F1-Score superiores a 0.94 en todas las categorías. Destaca especialmente la categoría de deportes con un F1-Score de 0.98, mientras que las categorías de negocios, entretenimiento, política y tecnología también muestran resultados muy buenos.

El tiempo total de ejecución para la evaluación completa con este modelo fue de 1113.41 segundos (aproximadamente 18.5 minutos) para clasificar 210 artículos, lo que resulta en un tiempo medio de aproximadamente 5.3 segundos por artículo.

5. Análisis de clustering

5.1. Enfoque clásico para clustering de documentos

Además del enfoque basado en LLMs para la clasificación supervisada, hemos explorado una metodología más clásica para el análisis no supervisado de los textos mediante técnicas de clustering. A diferencia del enfoque anterior, en este caso no se utilizan modelos de lenguaje preentrenados, sino métodos tradicionales de procesamiento de texto y agrupamiento.

El proceso implementado consta de las siguientes etapas:

- **Preprocesamiento**: Limpieza y normalización del texto, eliminando caracteres especiales y convirtiendo todo a minúsculas.
- **Vectorización TF-IDF**: Transformación de los documentos en vectores numéricos que reflejan la importancia de cada término en el contexto del corpus.
- **Reducción de dimensionalidad**: Aplicación de PCA para visualizar los documentos en un espacio bidimensional.
- **Clustering**: Aplicación del algoritmo K-Means para agrupar los documentos en clusters basados en su similitud semántica.

- **Análisis**: Extracción de los términos más relevantes de cada cluster para caracterizarlos.

Para este análisis, hemos decidido deliberadamente utilizar 5 clusters, coincidiendo con el número de categorías reales de los documentos (business, entertainment, politics, sports, tech). Esto nos permite evaluar en qué medida los clusters obtenidos de forma no supervisada se corresponden con las categorías predefinidas.

5.2. Visualización de clusters

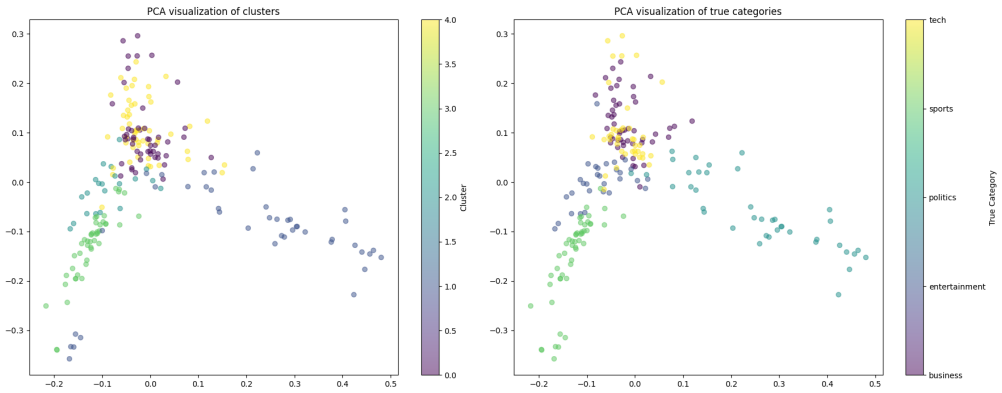

Fig. 8: *Visualización PCA de los clusters obtenidos (izquierda) y categorías reales (derecha)*

En la Figura 8 se muestra la proyección bidimensional de los documentos utilizando PCA. A la izquierda se visualizan los documentos coloreados según el cluster asignado por el algoritmo K-Means, mientras que a la derecha se muestran los mismos documentos coloreados según su categoría real. La comparación visual permite apreciar el grado de correspondencia entre los clusters descubiertos y las categorías predefinidas.

5.3. Análisis de términos por cluster

El análisis de los términos más representativos de cada cluster reveló los siguientes patrones:

- **Cluster 1**: Dominado por términos como “sales”, “economy”, “growth”, “economic”, “bank”, que se corresponde con la categoría de negocios (business).
- **Cluster 2**: Caracterizado por términos como “film”, “open”, “match”, “final”, “australian”, parece combinar elementos de entretenimiento y deportes, especialmente relacionados con torneos y competiciones.

- **Cluster 3**: Definido por términos como “mr”, “labour”, “election”, “party”, “government”, más asociado con la categoría de política.
- **Cluster 4**: Incluye términos como “mobile”, “microsoft”, “software”, “phone”, “users”, claramente alineado con la tecnología.
- **Cluster 5**: Contiene términos como “holmes”, “athens”, “olympic”, “athletics”, “champion”, identificándolo como un subconjunto específico de deportes centrado en atletismo y juegos olímpicos.

Es interesante observar cómo el algoritmo de clustering ha separado la categoría de deportes en dos clusters distintos (2 y 5), uno más orientado a deportes de competición/-torneos y otro más específico de atletismo y eventos olímpicos, mientras que ha agrupado parcialmente entretenimiento dentro del cluster 2.

5.4. Potencial mejora con LLMs

Aunque para este análisis de clustering hemos utilizado un enfoque tradicional, una interesante mejora sería incorporar modelos de lenguaje grandes para la generación automática de etiquetas descriptivas para cada cluster. En lugar de analizar manualmente los términos más frecuentes, se podría utilizar un LLM para generar títulos para cada cluster basado en sus términos más representativos, producir un resumen de los temas principales abordados en cada cluster o identificar subtemas o categorías emergentes dentro de clusters heterogéneos.

Este enfoque híbrido combinaría la eficiencia computacional de los métodos tradicionales de clustering con la capacidad de interpretación y generación de lenguaje natural de los LLMs, facilitando la comprensión y análisis de grandes colecciones de documentos.

6. Conclusiones

Nuestro análisis demuestra que los modelos de lenguaje preentrenados modernos son extremadamente eficaces para la clasificación de textos periodísticos, alcanzando niveles de precisión superiores al 95 % sin necesidad de entrenamiento específico para la tarea. Esto representa una ventaja significativa frente a los enfoques tradicionales que requieren un diseño y un entrenamiento específico.

Las principales conclusiones que podemos extraer son:

- Los modelos más grandes tienden a ofrecer mejor rendimiento, aunque algunos modelos de tamaño medio también muestran resultados muy competitivos.
- Existe un compromiso claro entre precisión y tiempo de inferencia, siendo los modelos más grandes generalmente más lentos pero más precisos.

- El uso de JSON Schema para la estructuración de respuestas resulta fundamental para garantizar salidas consistentes y fáciles de procesar.
- La categoría de deportes parece ser la más fácil de identificar para los modelos, posiblemente debido a su vocabulario más específico y diferenciado.
- En cuanto al análisis de clustering, observamos que los métodos no supervisados son capaces de identificar estructuras que se corresponden bastante bien con las categorías predefinidas, especialmente en temáticas con vocabulario distintivo como tecnología y política.
- El clustering reveló matices interesantes como la separación de la categoría deportes en dos clusters diferentes (competiciones generales y atletismo/olimpismo), lo que sugiere subdivisiones temáticas que podrían ser relevantes para un sistema de clasificación más granular.

Estos resultados sugieren que los modelos de lenguaje preentrenados representan una alternativa muy potente a los métodos tradicionales para tareas de clasificación de texto, especialmente cuando se dispone del hardware adecuado para su ejecución.

6.1. Trabajos futuros

Viendo el rendimiento de los modelos pequeños, como Qwen2.5 1.5B, se podría explorar la posibilidad de realizar un fine tuning específico para la tarea de clasificación de textos periodísticos, lo que podría mejorar aún más la precisión. Además, se podría aprovechar la capacidad de estos modelos para hacer una clasificación multietiqueta, permitiendo que un artículo pertenezca a más de una categoría. Esto podría ser especialmente útil en el contexto de noticias, donde un artículo puede abarcar múltiples temas. Otro área a explorar es la posibilidad de utilizar técnicas de *few-shot learning*, donde se utiliza un modelo base preentrenado y se le proporciona un pequeño número de ejemplos etiquetados para mejorar su rendimiento en una tarea específica.

Respecto al análisis de clustering, sería interesante implementar el enfoque híbrido mencionado, donde los LLMs se utilizarían para generar automáticamente etiquetas descriptivas para cada cluster. Este enfoque podría extenderse a un sistema completo de descubrimiento de temas, donde el clustering identificaría grupos semánticos y los LLMs proporcionarían interpretaciones detalladas de cada grupo. Además, podría explorarse el uso de técnicas de clustering jerárquico para identificar subcategorías dentro de las categorías principales, como se observó en el caso de los clusters de deportes.

Parte I

Bibliografía

[1] A. Q. Jiang et al., *Mistral 7B*, 2023. arXiv: 2310.06825 [cs.CL]. dirección: https://arxiv.org/abs/2310.06825.

[2] A. Grattafiori et al., *The Llama 3 Herd of Models*, 2024. arXiv: 2407.21783 [cs.AI]. dirección: https://arxiv.org/abs/2407.21783.

[3] Qwen et al., *Qwen2.5 Technical Report*, 2025. arXiv: 2412.15115 [cs.CL]. dirección: https://arxiv.org/abs/2412.15115.

[4] G. Team et al., *Gemma 3 Technical Report*, 2025. arXiv: 2503.19786 [cs.CL]. dirección: https://arxiv.org/abs/2503.19786.

[5] L. B. Allal et al., *SmolLM2: When Smol Goes Big – Data-Centric Training of a Small Language Model*, 2025. arXiv: 2502.02737 [cs.CL]. dirección: https://arxiv.org/abs/2502.02737.