

# Colección Facticia de Emilio Roig de Leuchsenring

Daniel Toledo Martínez  
Osvaldo Roberto Moreno Prieto  
Claudia Puentes Hernández  
Ángel Daniel Alonso Guevara  
Alejandro Camacho Pérez

## 1 Introducción

Este proyecto surge como colaboración de la Universidad de la Habana con el departamento de digitalización de edificio Santo Domingo, comúnmente conocido como San Geronimo por la Universidad que allí se encuentra. El objetivo de este proyecto es extraer las imágenes de los documentos de la colección facticia, legada por Emilio Roig de Leuchsenring, y almacenarlas en un formato digital junto con su descripción, para agilizar el proceso de digitalización de los documentos.

## 2 Estado del Arte

La gran parte de las investigaciones en el campo de la extracción de imágenes y texto se ha centrado en el uso de modelos de aprendizaje profundo (*deep learning*) para la detección de objetos en imágenes y la extracción de texto en documentos. Sin embargo, ambas tareas han sido complicadas por la heterogeneidad de las imágenes, las tipografías de los textos, la orientación de los textos, el flujo y la maquetación de los documentos.

En el caso general de la segmentación de imágenes, se han utilizado modelos de aprendizaje profundo como YOLO v8, debido a los grandes resultados obtenidos, su extensa documentación y el soporte por parte de la comunidad de código abierto (*open source*). Entre los problemas que se han resuelto con YOLO se encuentran la detección de matrículas vehiculares y el análisis de publicidad (Wang et al., 2024) [1]. YOLO, además, posee una gran variedad de arquitecturas según el problema a resolver, con diferentes niveles de parámetros para ajustar la precisión y la velocidad del modelo. En paralelo, la extracción de texto se ha mejorado con el uso de motores OCR (*Optical Character Recognition*, en español, Reconocimiento Óptico de Caracteres) como EasyOCR, EfficientOCR, Calamari y Tesseract, que permiten transformar información visual en datos legibles y estructurados (Skelbye & Danellés, 2021) [2], proporcionando *benchmarks* confiables para medir su desempeño.

El problema de la similitud entre imágenes y texto ha sido abordado por modelos como CLIP (*Contrastive Language-Image Pre-training*), que ha demostrado ser altamente efectivo. CLIP utiliza un *encoder* de imágenes basado en ViT (*Vision Transformer*) y un *encoder* de texto basado en *transformers* para generar representaciones vectoriales que permiten medir la similitud entre ambas modalidades mediante aprendizaje contrastivo. Además, estudios recientes han optimizado su desempeño en el emparejamiento de imágenes con noticias mediante técnicas de selección de texto clave y procesamiento de imágenes a resoluciones específicas (Vu et al., 2022) [3]. Otros modelos, como MKL-VisITA, han integrado *Multi-Kernel Learning* y *Vision Transformers* para mejorar la representación compartida de imágenes y texto, alcanzando niveles de precisión superiores en conjuntos de datos como MSCOCO y Flickr30K (Wang et al., 2024 [4]. Estas metodologías han permitido detectar patrones semánticos complejos, lo que resulta crucial para la recuperación de información en archivos históricos y otros contextos.

Un aspecto fundamental dentro de estos procesos es el preprocesamiento de datos, que influye directamente en la calidad de los resultados obtenidos. En el caso de las imágenes, se han implementado técnicas como la binarización con *Otsu's thresholding*, la eliminación de ruido y la normalización del color (escala de grises) [5].

Centrándonos en las investigaciones realizadas, podemos determinar que existen diferentes enfoques para resolver el problema de reconocimiento y extracción de textos e imágenes, y asociar los datos extraídos. Sin embargo, no existe un modelo único que resuelva todo el problema, por lo que la solución debe ser una combinación de varios modelos, cada uno especializado en una tarea. Otro punto clave es la calidad de los datos, ya que la mayoría de las investigaciones se han realizado con datos de buena calidad y no están públicos para su uso, siendo escasas las investigaciones con datos de colecciones antiguas y deterioradas.

Para más detalles, se puede acceder a la tabulación del estado del arte en el siguiente enlace: Tabulación del estado del arte.

### 3 Conjunto de Datos

El conjunto de datos consiste en 220 carpetas que contienen entre 30 y 400 fotografías cada una. Estas fotografías están en un orden específico, preservando la estructura de los documentos originales, por lo cual es útil conservar esta información en la extracción de datos sobre ellas para encontrar otras características de interés de los historiadores. Las fotografías están en formato *jpg* con una resolución mayor a *1080px* en ambas dimensiones.

Pueden contener textos horizontales, verticales o una combinación de ambos en una misma imagen. De igual forma, una imagen puede contener textos en diferentes fuentes, tamaños, interlineados y grosores.



Figura: Diferentes fuentes y tamaños

Figure 1: Ejemplo de imagen con diferentes fuentes, tamaños e interlineados.

El color del papel o el color de fondo de las fotografías puede variar entre blanco y tonalidades de amarillo, incluso rojizas en algunos casos. Los textos, en su mayoría, son mecanografiados, aunque existen tachaduras y escritos a mano en algunos casos, como suelen ser fechas, firmas y anotaciones marginales, números internos de la página, entre otros. El texto está en español, aunque en algunos casos puede contener palabras en otros idiomas, pero del alfabeto latino.

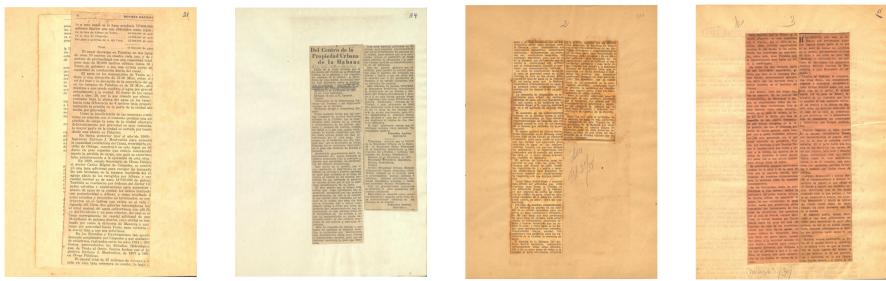


Figura: Diferentes tonalidades

Figure 2: Ejemplo de imagen con diferentes tonalidades de fondo y tipos de texto.

Las fotografías pueden ser imágenes de una página completa, fragmentos de una página, estar en una columna de texto o dispuestas horizontalmente a lo largo de la página. Las imágenes pueden ser dibujos, gráficos, mapas o fotografías de personas, y pueden estar rodeadas de algún contorno o marco que las delimita. En su mayoría, las imágenes tienen forma rectangular, aunque existen casos de imágenes con formas irregulares.



Figura: Variedad de imágenes

Figure 3: Ejemplo de imágenes con distintas disposiciones dentro del documento.

Debido a que estas fotografías son de documentos antiguos, pueden contener manchas, dobleces, arrugas, desgaste del papel y otros daños que afectan la calidad de la imagen. En algunos casos, las fotografías pueden estar rotadas en un ángulo, pero siempre con una perspectiva frontal. Es un requisito del problema recuperar las imágenes en el ángulo correcto para evitar que el investigador tenga que rotar la imagen o recortarla para la digitalización.



Figura: Imágenes deterioradas

Figure 4: Ejemplo de imágenes deterioradas con manchas y dobleces.

El conjunto de datos pesa alrededor de 60GB en total, es de acceso público y puede ser descargado desde las páginas oficiales de la Oficina del Historiador de La Habana: Repositorio Digital OHC.

## 4 Propuesta de solución

El problema de correspondencia entre imágenes y texto en documentos históricos presenta desafíos en la localización de contenido dentro de las fotografías y su posterior asociación con la información correcta. Para abordar esta problemática, hemos dividido nuestro enfoque en dos subproblemas principales:

- (1) Detección y extracción de texto y
- (2) Asociación de imágenes con texto.

## 4.1 Detención de Imágenes y Textos en las fotografías

Para identificar imágenes y textos en las fotografías, decidimos utilizar el modelo de *Deep Learning* YOLO, en su versión más reciente (YOLOv11). YOLO posee la variante OBB (*Oriented Bounding Boxes* - cuadros delimitadores orientados), que permite cumplir con el requisito de identificar las imágenes según el ángulo en que ha sido tomada la fotografía. Además, rotar el texto a su alineación original facilita la integración con los OCR. Permite identificar varias clases diferentes en un mismo análisis, lo que nos evita tener dos modelos separados para identificar texto e imágenes.

El modelo puede refinarse en el futuro añadiendo nuevas fotografías para mejorar las métricas en los escenarios que sean de interés. De igual manera, nos permite usar modelos preentrenados y mejorarlos con nuestros datos. YOLO, al ser un modelo de *Deep Learning*, tiene la capacidad de reconocer patrones complejos en las fotografías, como textos deformados, fondos borrosos o imágenes movidas.

YOLO utiliza técnicas de multi-escala (por ejemplo, FPN - *Feature Pyramid Networks*) para detectar objetos pequeños, lo cual puede ser útil para columnas estrechas y pies de foto. Además, YOLO tiene varios modelos que permiten ajustar la precisión y la velocidad del modelo. En nuestro caso, usamos el modelo *Nano*, que permite procesar grandes volúmenes de imágenes en un tiempo relativamente rápido sin necesidad de usar GPU costosas, lo cual es ideal para el desarrollo de un software con los pesos exportados.

Para el entrenamiento, etiquetamos cuatro clases principales:

- **Texto:** Todo tipo de textos mecanografiados y encabezados, siempre manteniendo su dirección original según el flujo de la página.
- **Imagen:** Todo tipo de imágenes, dibujos, gráficos, mapas, fotografías, entre otros.
- **Escrito a mano:** Anotaciones marginales, firmas, fechas, números internos de la página, entre otros.
- **Caption:** Textos que acompañan a las imágenes, como títulos, subtítulos y pies de foto. Esta clase la agregamos como extra para poder asociar imágenes con sus descripciones. Los *captions* usualmente son textos cortos y están cerca de las imágenes, en negrita o en cursiva. Para mejorar la selección de *captions*, decidimos extender el *bounding box* desde el texto, tomando un fragmento pequeño de la imagen.

La primera iteración la realizamos con 893 fotografías, teniendo etiquetados 2,903 textos, 341 imágenes, 223 *captions* y 980 escritos a mano. Tras realizar ajustes de hiperparámetros y posteriormente entrenar el modelo, obtuvimos las siguientes métricas:

#### 4.1.1 Primer Entrenamiento - Métricas

##### Matriz de Confusión

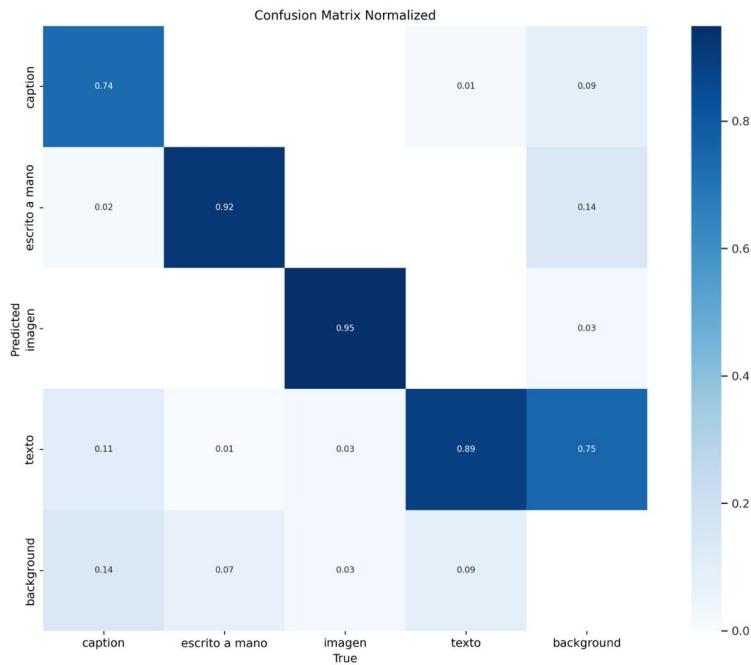


Figure 5: Matriz de confusión del modelo.

Como se puede observar en la matriz de confusión, se detectaron partes del fondo de las fotografías como texto. Esto se debe a que existen fotografías en las que los textos del otro lado de la página se marcan y son visibles.

##### Precisión-Confianza Curve

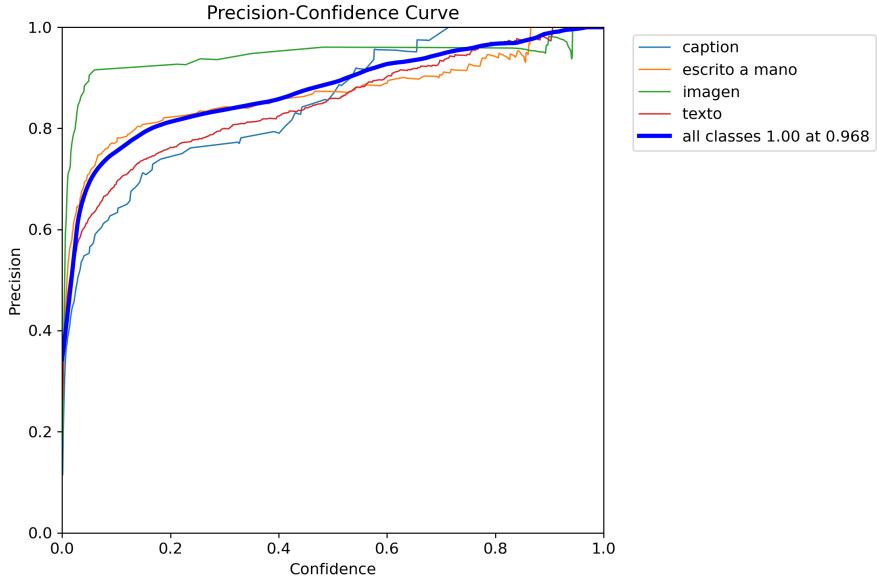


Figure 6: Curva de Precisión-Confianza del modelo.

La *Precision-Confidence Curve* muestra cómo la precisión del modelo varía en función de los niveles de confianza. Se observa que la clase "imagen" alcanza una precisión alta incluso en niveles bajos de confianza, mientras que la clase "caption" presenta un menor rendimiento inicial, lo que indica una mayor presencia de falsos positivos. La clase "texto" no logra mantener una precisión alta en niveles de confianza bajos, lo que sugiere que el modelo tiene dificultades para identificar correctamente los textos en las fotografías. En general, se nota falta de consistencia en las clases "caption" y "escrito a mano", lo que puede deberse al desbalance de los datos etiquetados.

### Curva de Recall

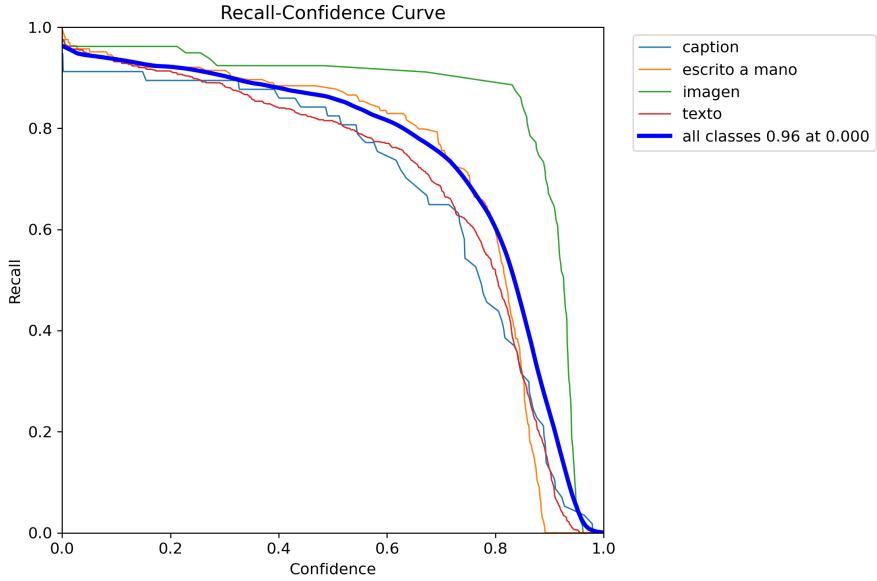


Figure 7: Curva de Recall del modelo.

En la curva de *recall*, se puede observar que la clase "imagen" es la que tiene el mejor *recall*. A pesar de ser una clase con pocos datos etiquetados, logra mantener un buen *recall*, lo que puede sugerir que existen patrones muy notables y consistentes que identifican a las imágenes, o que este alto *recall* se debe a la escasez de datos en el entrenamiento. Por otro lado, la clase "texto", que tiene la mayor cantidad de datos etiquetados, no logra mantener un *recall* alto, lo que puede sugerir que la calidad de los datos o la variabilidad de los textos afecta el rendimiento del modelo.

### Curva de Precisión-Recall

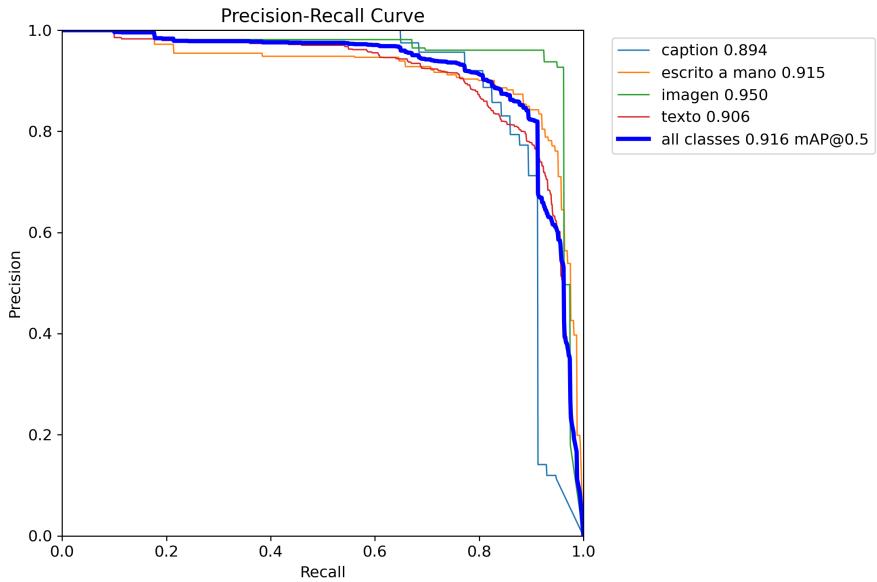


Figure 8: Curva de Precisión-Recall del modelo.

Finalmente, la tabla de *Precision-Recall* muestra que la clase "imagen" es la que tiene el mejor rendimiento, seguida por la clase "escrito a mano", mientras que las clases "texto" y "caption" presentan un rendimiento inferior. Se nota que la clase "caption" tiene un rendimiento significativamente más bajo en comparación con las otras clases, lo que sugiere que el modelo tiene dificultades para identificar correctamente los *captions* en las fotografías.

## Curva F1

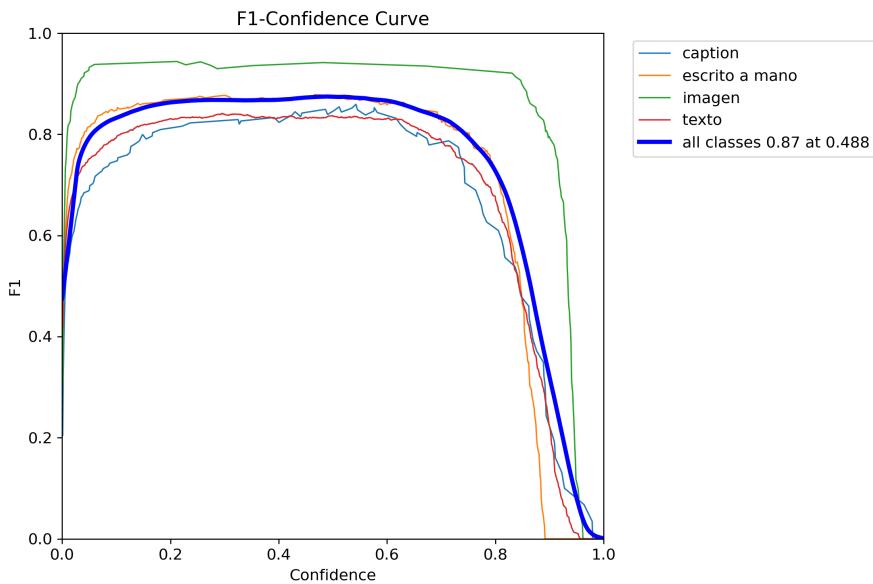


Figure 9: Curva F1 del modelo.

La gráfica F1 para cada una de las clases muestra el balance entre la precisión y el *recall*. Se observa que la clase "imagen" es la que tiene el mejor rendimiento, seguida por la clase "escrito a mano", mientras que las clases "texto" y "caption" presentan un rendimiento inferior.

### 4.1.2 Segundo Entrenamiento - Aumento de datos - Métricas

Para mejorar la cantidad de datos etiquetados, decidimos hacer un segundo etiquetado de las fotografías que el modelo no identificó, aquellas con bajo nivel de confianza o con resultados erróneos respecto al ángulo y tamaño, obteniendo un total de 1,221 fotografías etiquetadas.

De las 1,221 fotografías finalmente etiquetadas, obtuvimos: 3,492 textos, 794 imágenes, 483 *captions* y 981 escritos a mano. Realizamos un segundo ajuste de hiperparámetros y entrenamos el modelo nuevamente, obteniendo las siguientes métricas:

#### Matriz de Confusión

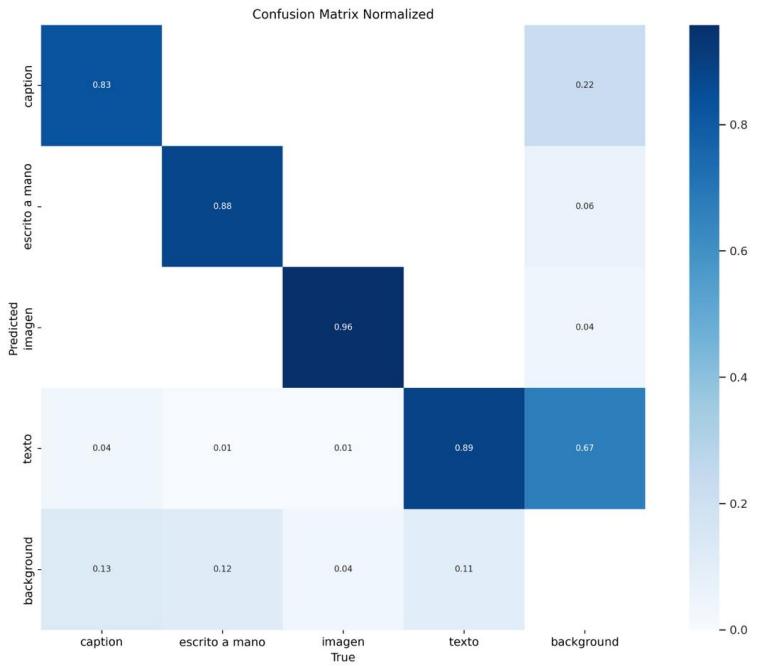


Figure 10: Matriz de confusión del modelo después del aumento de datos.

Tras los cambios realizados, obtuvimos una mejora en los verdaderos positivos en todas las clases. En la clase "texto", logramos disminuir la clasificación errónea de fondo como texto.

### Curva de Precisión-Confianza

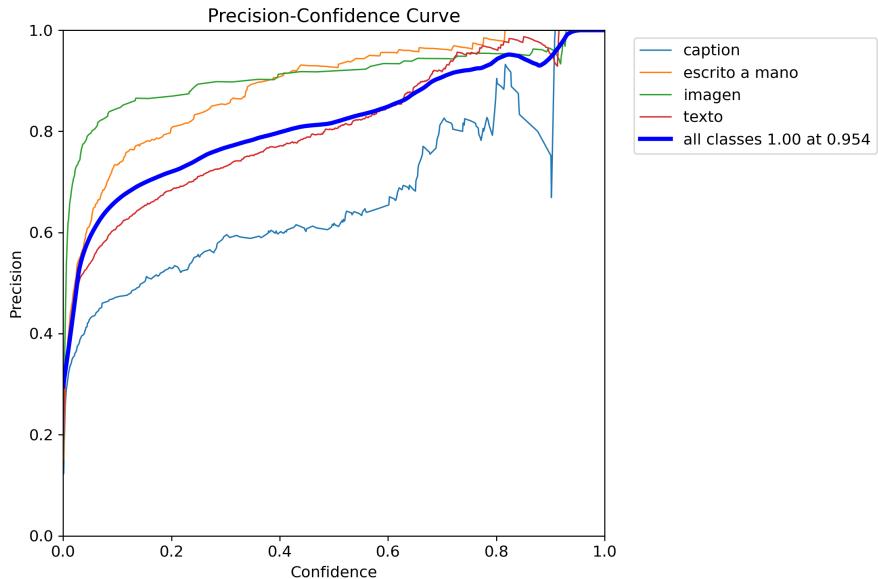


Figure 11: Curva de Precisión-Confianza después del aumento de datos.

Con el aumento de los datos etiquetados en cada clase, obtenemos un poco más de heterogeneidad en los resultados, permitiendo analizar mejor las gráficas. Sin

embargo, salen a relucir errores que tenían menor visibilidad, ya que estaban relacionados con la deficiencia de datos entre las clases.

## Curva de Recall

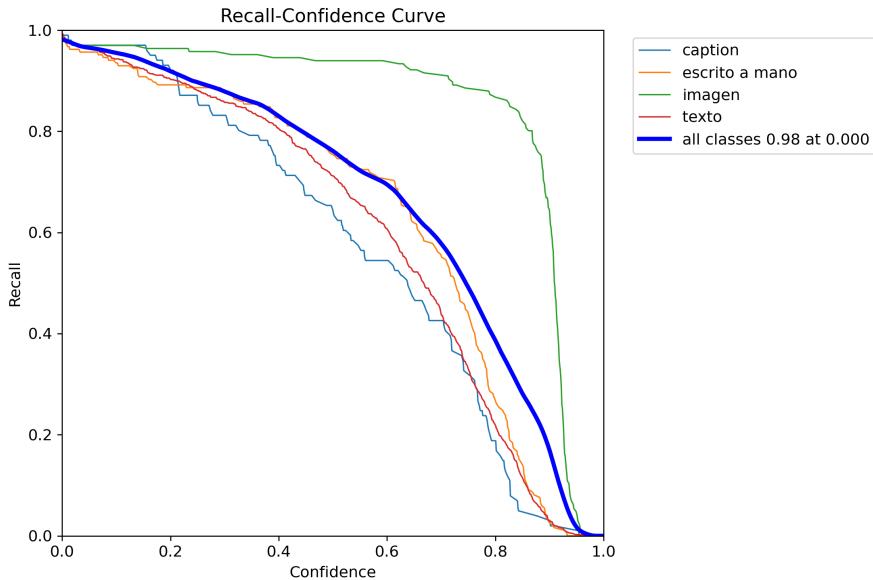


Figure 12: Curva de Recall después del aumento de datos.

En la gráfica de *recall*, se nota cómo disminuye ligeramente la inclinación de las clases. Sin embargo, la clase "imagen", para altos niveles de confianza, mantiene un *recall* elevado, lo que implica robustez en el modelo.

## Curva de Precisión-Recall

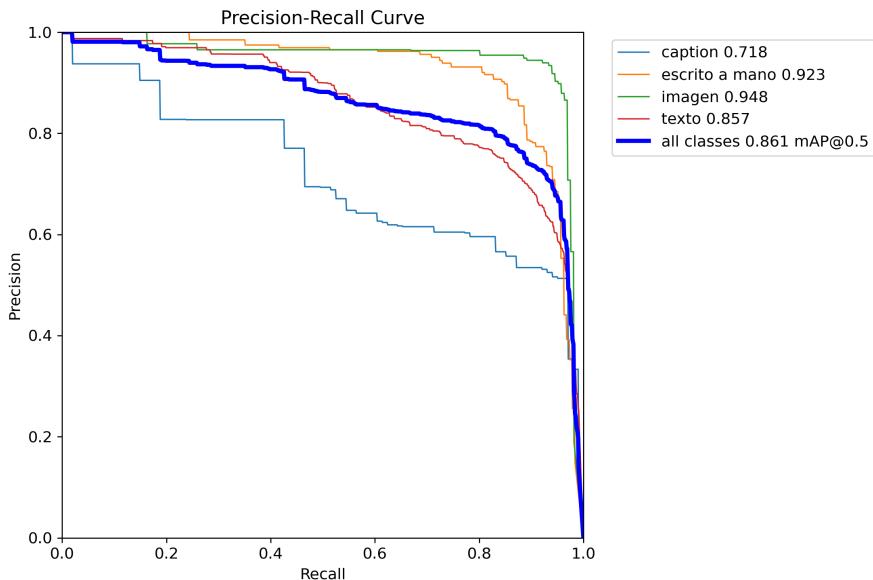


Figure 13: Curva de Precisión-Recall después del aumento de datos.

En la gráfica de *Precision-Recall*, se muestra un mejor análisis de los datos luego del aumento del *dataset*, permitiendo observar caídas bruscas a medida que aumenta el *recall*, lo que representa un aspecto negativo.

## Curva F1

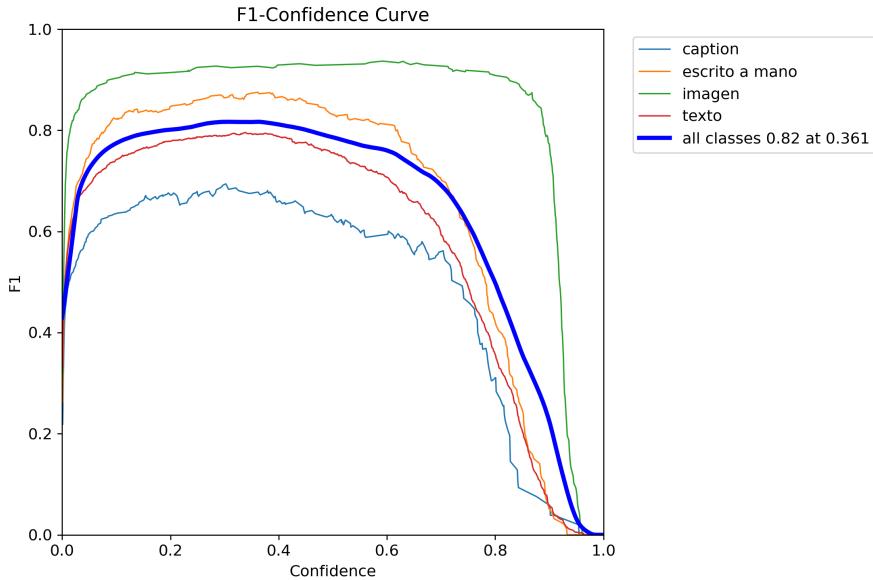


Figure 14: Curva F1 después del aumento de datos.

Al igual que en los gráficos anteriores, se puede observar la separación entre las diferentes clases luego del aumento del *dataset*, lo que evidencia una mejora en la evaluación del modelo.

### 4.1.3 Experimento preprocesando la imágenes antes del entrenamiento

Como mejora al modelo, durante el primer entrenamiento con 1,221 fotografías, se probaron técnicas de preprocesamiento de imágenes. Los resultados más significativos los obtuvimos aplicando escala de grises, lo cual es un aspecto a favor, ya que disminuye la dimensionalidad de los datos, además de no ser relevante para las clases que debemos clasificar en el modelo.

Se aplicó CLAHE (*Contrast Limited Adaptive Histogram Equalization*) para resaltar el texto y los bordes, así como para disminuir los efectos de los cambios de iluminación en las distintas fotografías. También se probó la reducción de ruido con *GaussianBlur* y el suavizado con *BilateralFilter* para preservar los bordes y mejorar la nitidez de la imagen.

### Ejemplo de Preprocesamiento



Figure 15: Ejemplo de preprocesamiento aplicado a una imagen.

### Cambio de Contraste y Escala de Grises



Figure 16: Ejemplo de imagen procesada con cambio de contraste y escala de grises.

Para el ajuste de los hiperparámetros, solo se pudieron realizar 10 iteraciones con diferentes configuraciones. Los mejores resultados usando este preprocesamiento fueron:

### Matriz de Confusión

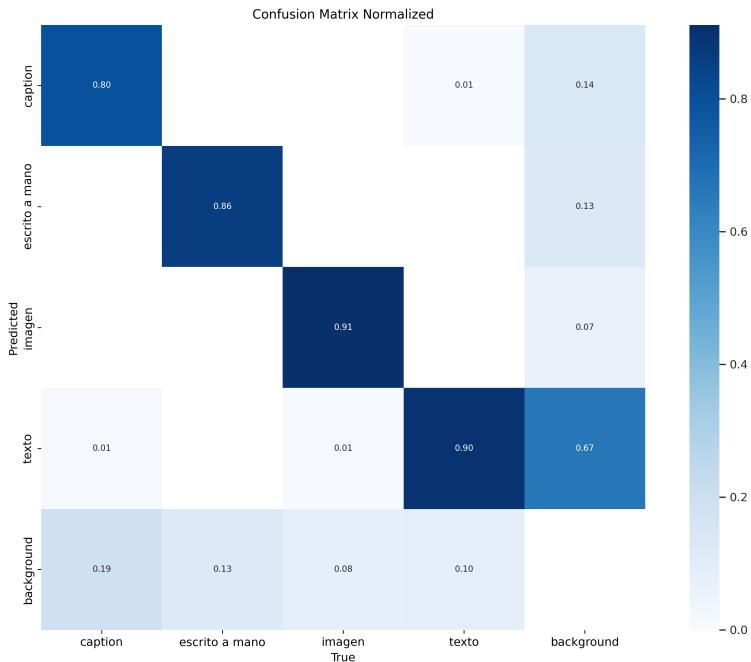


Figure 17: Matriz de confusión con preprocesamiento aplicado.

En general, no hubo cambios significativos en los datos de la matriz de confusión.

## Curva de Precisión

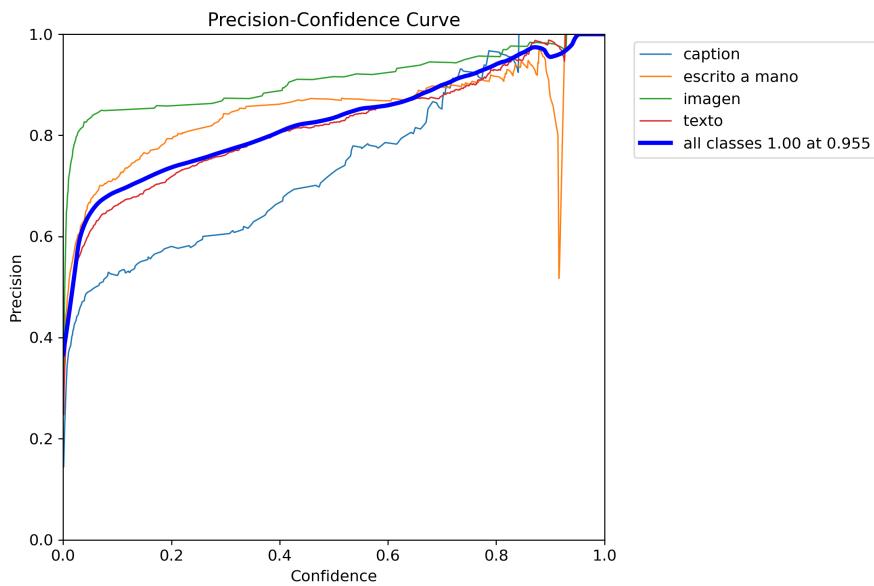


Figure 18: Curva de Precisión con preprocesamiento aplicado.

En la curva de precisión, para mayores niveles de confianza, los resultados convergen mejor que en los experimentos anteriores.

## Curva de Recall

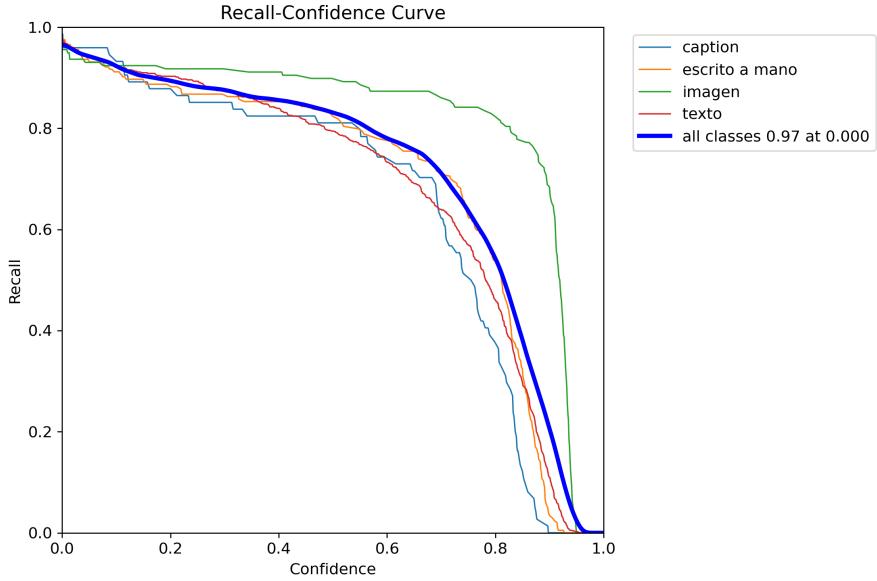


Figure 19: Curva de Recall con preprocesamiento aplicado.

Este es el mejor resultado obtenido en comparación con los experimentos anteriores. Logramos mejorar las métricas de las clases en general, sobre todo en la clase "texto", que es una de las más importantes en nuestro problema.

### Curva de Precisión-Recall

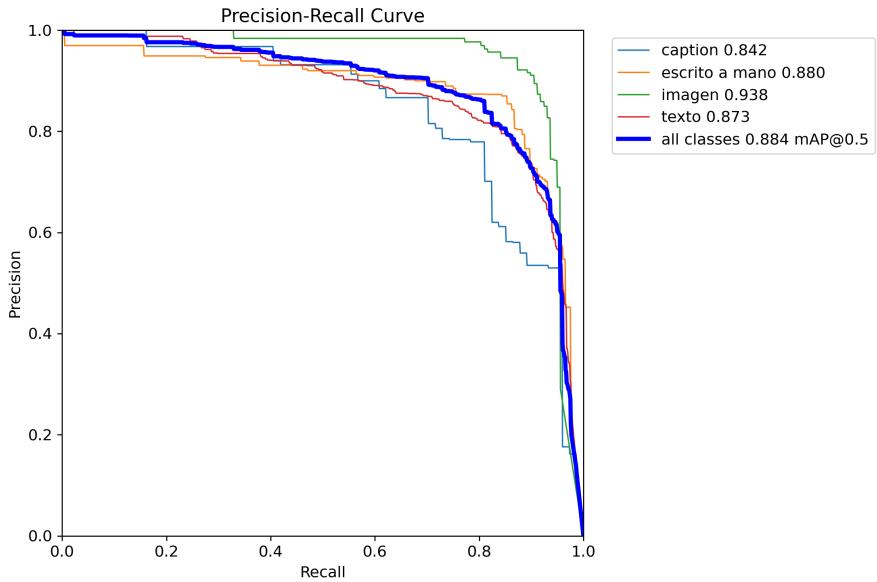


Figure 20: Curva de Precisión-Recall con preprocesamiento aplicado.

Al igual que en la precisión, se nota una mejor convergencia en las clasificaciones de las clases.

### Curva F1

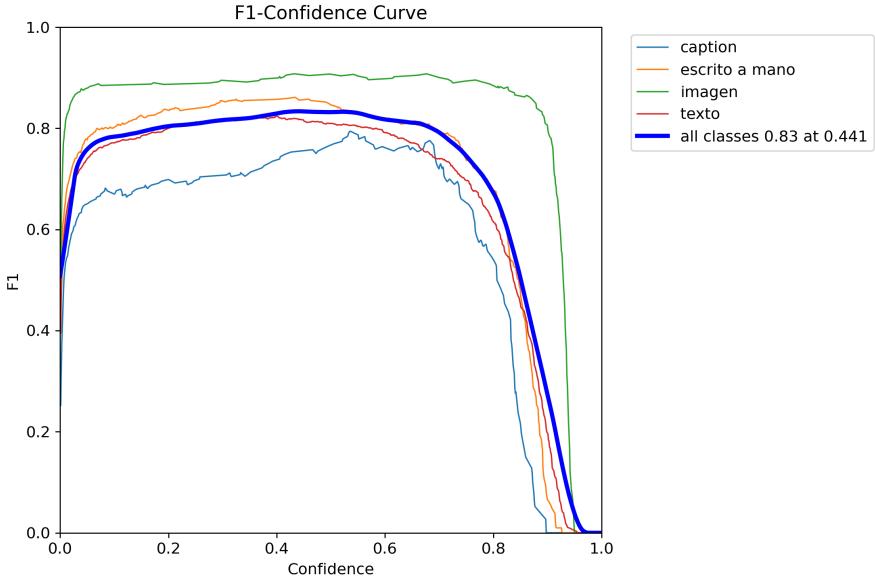


Figure 21: Curva F1 con preprocesamiento aplicado.

Los resultados mejoraron con respecto a los experimentos anteriores a medida que aumenta la confiabilidad del modelo.

## 4.2 Asociación de imágenes y texto

### 4.2.1 Tesseract

Tesseract es un modelo que implementa técnicas actuales en el campo de la extracción de texto a partir de imágenes. Este modelo se encuentra preentrenado para datos en español, por lo que puede ser utilizado en el contexto de nuestro proyecto sin la necesidad de realizar ningún entrenamiento adicional.

Entre sus potencialidades, se encuentra que es un proyecto **open source**, por lo que obtiene grandes beneficios por parte de la comunidad, como contribuciones y mejoras periódicas. Además, incorpora técnicas avanzadas para el **Optical Character Recognition** (OCR), como los enfoques basados en **deep learning**.

Otros modelos que resuelven el mismo problema fueron considerados. Tal es el caso de OCropus, un modelo gratuito que se decidió no utilizar debido a que no contaba con preentrenamientos con datos en español y su entrenamiento sería muy costoso en tiempo y recursos.

Existen otras alternativas como Google Cloud Vision o Amazon Textract, que tienen un muy buen rendimiento, pero, por desgracia, son de pago. Otro candidato fue Kraken, un modelo más robusto que, con suficientes datos de entrenamiento, podría ser una buena opción. Sin embargo, su modelo preentrenado se enfoca más en textos históricos manuscritos. Además, requiere mayor poder de cómputo, lo que implica un esfuerzo adicional sin ninguna ventaja significativa.

#### 4.2.2 Preprocesamiento

Para la selección de las técnicas de preprocesamiento a utilizar, realizamos una evaluación con una pequeña cantidad de datos para determinar qué conjunto de técnicas arrojaba mejores resultados con respecto a datos pertenecientes a nuestro *dataset*.

Para determinar qué combinación era más eficiente, comparamos el caso promedio en una serie de imágenes que contenían textos previamente etiquetados (para cada imagen se tiene su texto correspondiente) y utilizamos las siguientes métricas:

##### Similitud de Jaccard

Mide la proporción de elementos comunes con respecto al total de elementos únicos en ambos conjuntos. Se calcula como:

$$\text{Similitud de Jaccard} = \frac{|A \cap B|}{|A \cup B|}$$

Los resultados obtenidos para esta métrica fueron los siguientes:

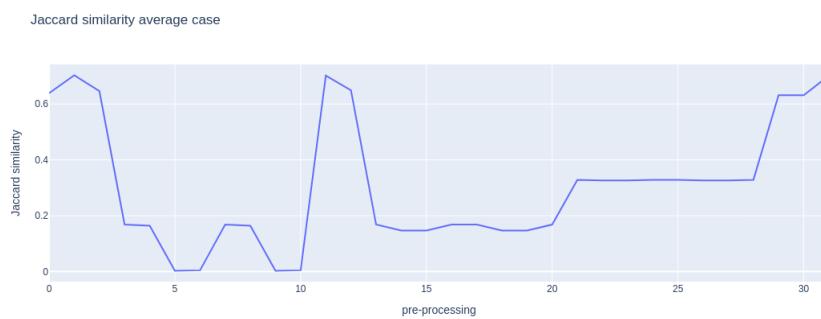


Figure 22: Gráfico de Similitud de Jaccard en el caso promedio.

##### Character Error Rate (CER)

Mide la precisión de un sistema de reconocimiento de texto comparando con un texto de referencia, por lo que es muy útil para evaluar la calidad de un OCR. Se calcula como:

$$\text{CER} = \frac{\text{Edit Distance (Levenshtein)}}{\text{Length of Reference}}$$

Los resultados obtenidos para esta métrica fueron los siguientes:

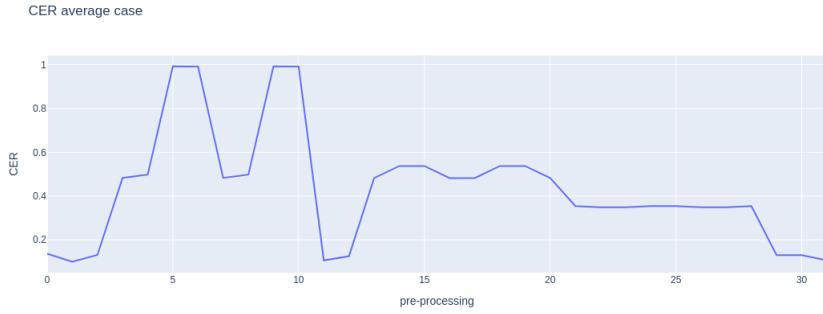


Figure 23: Gráfico de Character Error Rate (CER) en el caso promedio.

En ambos casos, el preprocesamiento que obtuvo mejor resultado consiste en invertir los colores de la imagen y luego llevarla a escala de grises. Por tanto, esta técnica será la que utilizaremos para mejorar el funcionamiento de Tesseract.

La combinación de estas técnicas es particularmente útil cuando existen fondos complejos y textos degradados. En nuestro caso, dada la antigüedad de los documentos, estos se vuelven propensos a poseer dichas características.

#### 4.2.3 Postprocesamiento

Por la naturaleza de los datos, son comunes los casos en que los modelos de reconocimiento de texto suelen tener problemas, además de los casos en que la información extraída está incompleta, lo que podría causar problemas más adelante al utilizarla como contexto.

Por este motivo, para solucionar este problema, decidimos utilizar un **Large Language Model** (LLM). Para una primera evaluación, empleamos las métricas anteriormente utilizadas para determinar la proximidad del resultado obtenido con el LLM respecto al texto extraído, obteniendo los siguientes resultados:

#### Comparación de LLM vs. Solo Tesseract

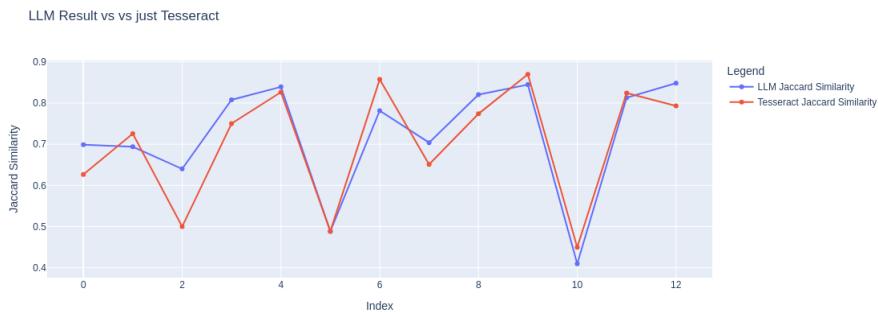


Figure 24: LLM vs. solo Tesseract utilizando similitud de Jaccard.

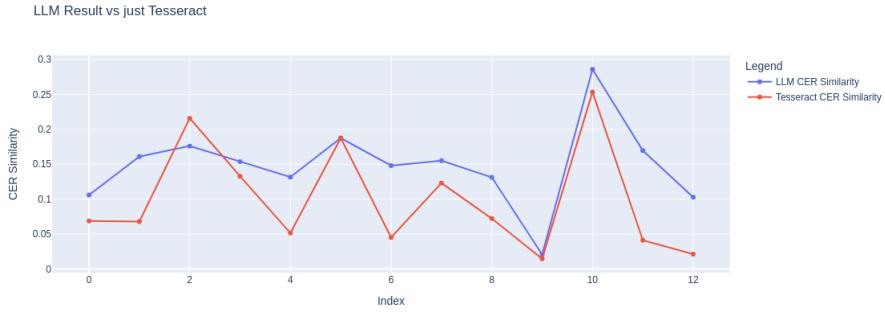


Figure 25: LLM vs. solo Tesseract utilizando CER.

Como se puede apreciar, en la mayoría de los casos la diferencia entre los resultados es poca. A pesar de que los textos extraídos se encuentran a menor distancia del texto original, hay que tener en cuenta que el modelo de lenguaje mejora el texto manteniendo cercana la similitud, por lo que se considera una buena opción para resolver el problema en cuestión.

#### 4.2.4 CLIP

Una vez extraídos tanto textos como imágenes, hay que enfrentarse a la tarea principal, que es asociar las imágenes en cuestión con los textos. Para ello, utilizamos CLIP, un modelo entrenado sobre un gran conjunto de pares de imagenTexto.

En lugar de predecir etiquetas específicas, CLIP aprende un espacio donde las imágenes se encuentran cercanas a sus descripciones de texto y alejadas de descripciones de texto no relacionadas. Esto permite al modelo generalizar para datos que no ha visto nunca sin necesidad de entrenamiento específico para la tarea (**Zero-Shot**).

El modelo utiliza redes neuronales, en particular *transformers*, para extraer las *\*características\** visuales y textuales y asignar a cada entrada una ubicación en un espacio de **embeddings** con la misma dimensionalidad.

El uso de este modelo reduce considerablemente el tiempo de desarrollo al no requerir entrenamiento previo para la obtención de resultados relevantes. Además, establece un puente entre imágenes y texto, lo que lo convierte en un muy buen candidato para la tarea en cuestión.

### 4.3 Pipeline

- Cargar el modelo con sus respectivos pesos (entrenar en caso de ser necesario).
- Utilizar el modelo para extraer y clasificar las cuatro clases.
- Preprocesar las imágenes que contienen texto.
- Utilizar Tesseract sobre las imágenes que contienen texto.

- Posprocesar el texto extraído utilizando el LLM.
- Realizar la asociación por cercanía entre *ímágenes* y *captions*.
- En los casos en que no se encontraron *captions* cercanos, utilizar el modelo de CLIP con todos los textos extraídos en la fotografía para determinar similitud en cuanto a temática.

#### 4.4 Conclusiones y Propuestas a seguir

Concluimos que el mejor modelo entrenado fue el que utilizó preprocesamiento de las fotografías, por lo cual recomendamos realizar un ajuste de hiperparámetros con más iteraciones para lograr una mejor configuración.

Además, recomendamos probar con otras técnicas de preprocesamiento como las descritas en el siguiente enlace: Tolstoy AI - Transcripción de Periódicos.

También proponemos probar con otros modelos de YOLO que requieren mayor capacidad de cómputo, como YOLOv11 OBB M, y analizar en qué medida mejoran los resultados.

Finalmente, sugerimos mejorar y regularizar el *dataset* con un mayor equilibrio de clases para evitar sesgos en el modelo y alcanzar mejores resultados.

#### 4.5 Referencias

### References

- [1] Li, H., & Zhang, N. (2024). Computer Vision Models for Image Analysis in Advertising Research. *Journal of Advertising*, 53(5), 771–790. <https://doi.org/10.1080/00913367.2024.2407644>.
- [2] Skelbye, M. B., & Dannélls, D. (2021). OCR Processing of Swedish Historical Newspapers Using Deep Hybrid CNN-LSTM Networks. [https://doi.org/10.26615/978-954-452-072-4\\_023](https://doi.org/10.26615/978-954-452-072-4_023).
- [3] Vu, H.-N., Nguyen, H.-D., & Tran, M.-T. (2022). Re-matching Images and News Using CLIP Pretrained Model. *CEUR Workshop Proceedings*. Recuperado de <https://arxiv.org/abs/2103.00020>.
- [4] Yindumathi, K. M., Chaudhari, S. S., & Aparna, R. (2020). Analysis of Image Classification for Text Extraction from Bills and Invoices. *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kharagpur, India, pp. 1-6. <https://doi.org/10.1109/ICCCNT49239.2020.9225564>.
- [5] Tolstoy AI. Transcripción de Periódicos. <https://tolstoy.ai/tolstoy-wall-st-journal-transcribe-newspapers/>.