

Colección Factice de Emilio Roig de Leuchsenring

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

1 Introducción

La digitalización de documentos históricos es una tarea fundamental para la preservación del patrimonio cultural y el acceso al conocimiento. Sin embargo, este proceso enfrenta grandes desafíos debido a la heterogeneidad de los documentos, el deterioro del papel con el paso del tiempo y la variedad de estructuras y formatos en los textos e imágenes.

Actualmente, la digitalización se realiza en su mayoría de manera manual, un proceso lento, propenso a errores y costoso. Esto hace necesario desarrollar un sistema automatizado que facilite la identificación y extracción del contenido con mayor precisión y eficiencia, optimizando así el trabajo de preservación documental.

En este contexto, surge este proyecto como una colaboración entre la Universidad de La Habana y el departamento de digitalización del edificio Santo Domingo, conocido como San Gerónimo. Su objetivo principal es desarrollar una solución que permita la detección, extracción y clasificación de imágenes y textos en documentos antiguos pertenecientes a la Colección Factice de Emilio Roig de Leuchsenring.

Más allá de la digitalización de archivos históricos, este proyecto busca optimizar el proceso de catalogación y facilitar la búsqueda de información. Al estructurar los documentos de manera más accesible, se mejora su preservación y se garantiza su consulta de forma eficiente.

Además, la implementación de técnicas avanzadas de detección y extracción de texto permite una mayor precisión en la identificación de elementos dentro de los documentos, algo especialmente útil para investigaciones académicas y gestión de archivos históricos. Con esta propuesta, se pretende desarrollar una solución escalable y adaptable, capaz de integrarse en futuros procesos de digitalización, aprovechando el potencial del aprendizaje profundo y el procesamiento avanzado de imágenes y texto.

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 & Daniells, 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 utilizado en este proyecto, como mencionamos anteriormente, proviene de documentos históricos pertenecientes a la Colección Factivia de Emilio Roig de Leuchsenring, abarcando materiales desde 1940 en adelante. Estos documentos han sido digitalizados y organizados en 220 carpetas, cada una conteniendo entre 30 y 400 fotografías, manteniendo el orden original de los documentos para facilitar su análisis y conservación.

Las imágenes están en formato JPG, con una resolución mínima de 1080 px en ambas dimensiones, lo que permite preservar detalles importantes tanto en los textos como en las ilustraciones. Conservar esta información estructurada es fundamental para facilitar su procesamiento y extraer características de interés para la investigación histórica.

El conjunto de datos presenta una gran diversidad en la estructura y contenido de los documentos, lo que representa un desafío significativo para su procesamiento automatizado. En las imágenes pueden encontrarse textos dispuestos en diferentes orientaciones, ya sean horizontales, verticales o combinaciones de ambas dentro de una misma página. Además, los textos presentan variaciones en las fuentes, tamaños, interlineados y grosores, lo que añade complejidad a su detección y extracción.



Figura: Diferentes fuentes y tamaños

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

Otro aspecto importante es la variabilidad en el color del papel y del fondo de los documentos. Mientras que algunas páginas mantienen un fondo blanco, otras tienen tonalidades amarillentas o incluso rojizas debido al envejecimiento del material. Esto puede afectar el contraste entre el texto y el fondo, dificultando su segmentación

automática. Los textos, en su mayoría mecanografiados, también pueden presentar anotaciones a mano, firmas, fechas y numeraciones internas. Aunque el idioma predominante es el español, en algunos casos pueden aparecer palabras en otros idiomas que utilizan el alfabeto latino.

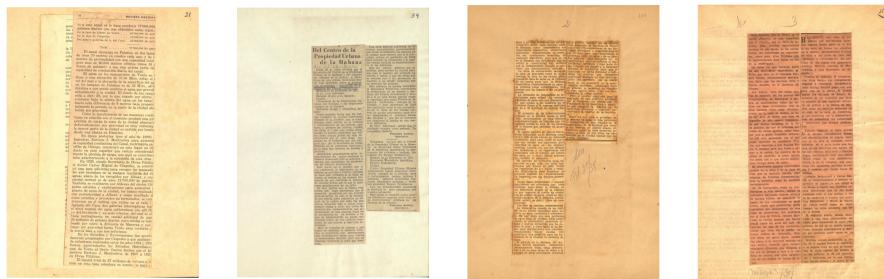


Figura: Diferentes tonalidades

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

Las fotografías del conjunto de datos incluyen una variedad de disposiciones. Algunas imágenes capturan páginas completas, mientras que otras son fragmentos de documentos o secciones específicas dentro de columnas de texto. También pueden encontrarse gráficos, mapas, ilustraciones y recortes de periódicos, algunos de ellos rodeados por marcos o con formas irregulares. Esta diversidad en la estructura de las imágenes añade otro nivel de complejidad a la detección automática de los elementos de interés.



Figura: Variedad de imágenes

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

Dado que se trata de documentos antiguos, es común encontrar signos de deterioro como manchas, dobleces, arrugas y desgaste en el papel. En ciertos casos, la tinta está desvanecida o existen zonas ilegibles debido al paso del tiempo, daños que afectan la calidad de la imagen. Además, algunas imágenes pueden haber sido digitalizadas con ligeras rotaciones o inclinaciones, lo que puede afectar la precisión de los modelos de detección de texto e imágenes. Para garantizar una correcta digitalización, es importante recuperar las imágenes en su orientación adecuada y corregir cualquier distorsión que pueda interferir con el reconocimiento de los elementos.



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

En la mayoría de los estudios revisados en el estado del arte, la detección de imágenes y textos, así como su asociación, se han tratado como problemas separados. Sin embargo, en nuestro caso, ambos procesos están estrechamente relacionados y deben abordarse de manera conjunta para lograr una digitalización más efectiva.

Para ello, es necesario desarrollar un sistema que permita:

- Manejar la variabilidad en la disposición del contenido, considerando textos en diferentes orientaciones, tamaños y tipografías.
- Detectar y extraer automáticamente textos e imágenes, diferenciando correctamente los distintos elementos dentro de cada documento.
- Asociar los textos extraídos con las imágenes correspondientes, asegurando que se mantenga la relación entre ambos dentro del documento digitalizado.

Para abordar estos desafíos de manera integral, se propone un sistema basado en visión por computadora y aprendizaje profundo, estructurado en dos etapas principales.

1. Detección y extracción de texto e imágenes: Se utiliza un modelo de detección de objetos que permite identificar y clasificar los distintos elementos dentro de cada imagen, diferenciando entre texto mecanografiado, manuscrito, imágenes y captions.

2. Asociación de imágenes con texto: Una vez extraídos los textos e imágenes, se emplean técnicas de reconocimiento óptico de caracteres (OCR) para convertir el texto en formato digital y se utilizan modelos de aprendizaje profundo para establecer relaciones entre los textos y las imágenes que los acompañan.

4.1 Detención de Imágenes y Textos en las fotografías

Para lograr una detección precisa, es fundamental identificar correctamente textos mecanografiados, imágenes, textos manuscritos y captions, considerando sus variaciones en tamaño, orientación y formato. Como primer paso, se realizó un proceso de etiquetado de los datos, esencial para el entrenamiento del modelo de detección, ya que le permite aprender a diferenciar y clasificar cada uno de estos elementos dentro de las imágenes.

4.1.1 Etiquetado de los Datos

Para entrenar el modelo de detección, fue necesario crear un conjunto de datos anotado con las clases relevantes. Utilizamos Label Studio para marcar manualmente los distintos elementos dentro de cada imagen y asignarles su categoría correspondiente.

Se etiquetaron las siguientes clases:

- **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 imágenes, generalmente en negrita o cursiva, incluyendo títulos, subtítulos y pies de foto.

Cada objeto fue delimitado con un bounding box, generando coordenadas (x , y , w , h , r), que indican su posición, tamaño y rotación dentro de la imagen. Posteriormente, estos datos fueron exportados en un formato compatible con YOLOv11 OBB, asegurando que el modelo pudiera utilizar esta información para aprender a detectar y clasificar correctamente los elementos en nuevas imágenes.

Uno de los principales retos fue la anotación de los captions, ya que su ubicación y formato variaban dentro de los documentos. Para mejorar su detección, se decidió extender los bounding boxes incluyendo una pequeña parte de la imagen adyacente, lo que permitió al modelo diferenciarlos mejor de los bloques de texto principales.

4.1.2 Preprocesamiento de las Imágenes

El preprocesamiento es una fase fundamental en la detección de objetos, ya que permite mejorar la calidad de las imágenes y reducir el impacto de factores que pueden afectar el rendimiento del modelo, como el ruido, la variabilidad en el contraste y la iluminación desigual. Además, ayuda a eliminar información irrelevante, permitiendo que el modelo se enfoque en los elementos importantes para la detección, en lugar de aprender patrones no deseados que podrían generar errores. Una imagen

bien procesada facilita que el modelo generalice mejor y logre una detección más precisa.

Existen varias técnicas de preprocesamiento utilizadas en visión por computadora, entre ellas:

- **Conversión a escala de grises:** Reduce la dimensionalidad de los datos al eliminar la información de color, manteniendo únicamente los niveles de intensidad.
- **Ecualización de histograma:** Mejora el contraste general de la imagen distribuyendo de manera uniforme los valores de brillo.
- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Una variante de la ecualización de histograma que ajusta localmente el contraste sin sobreexponer áreas brillantes.
- **Filtros de suavizado:** Métodos como GaussianBlur y BilateralFilter reducen el ruido sin perder detalles importantes.
- **Binarización de Otsu:** Convierte la imagen a blanco y negro aplicando un umbral óptimo, útil en OCR pero no siempre en detección de objetos.

Dado que trabajamos con documentos históricos con diferentes niveles de degradación, probamos varias de estas técnicas y seleccionamos aquellas que ofrecieron mejores resultados en la extracción de texto e imágenes, sin afectar la precisión del modelo.

Las técnicas aplicadas fueron:

- **Escala de grises:** Eliminó la información de color, reduciendo la cantidad de datos irrelevantes y mejorando la detección al resaltar los contrastes entre el texto y el fondo. Esto evitó que el modelo aprendiera patrones basados en el color, que no son útiles para la tarea de detección.
- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Se utilizó para mejorar el contraste en regiones específicas de la imagen, resaltando los textos y los bordes sin sobreexponer las áreas más claras. Esto fue especialmente útil para documentos con tonos de fondo variables.
- **GaussianBlur::** Aplicado para suavizar el ruido sin afectar los bordes del texto, evitando que el modelo confundiera imperfecciones del papel con caracteres.
- **BilateralFilter::** Se empleó para reducir el ruido manteniendo los bordes definidos, lo que ayudó a mejorar la nitidez de los textos y evitó la distorsión en zonas de alto contraste.

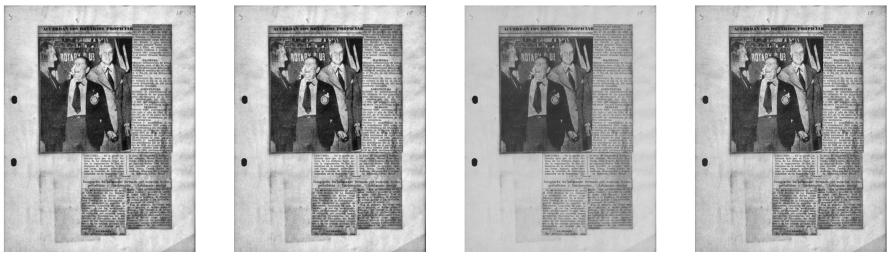


Figura: Diferentes preprocesados

Figure 5: Ejemplo de imágenes con diferentes preprocesamientos.

Después de probar distintas combinaciones, se determinó que la escala de grises combinada con CLAHE era la mejor opción, ya que aumentaba la claridad del texto y facilitaba su segmentación sin afectar la precisión.

La elección de estas técnicas se basó en la necesidad de mejorar la calidad general de las imágenes, asegurando que los elementos dentro de ellas fueran más fáciles de detectar por el modelo. Métodos como la binarización de Otsu, aunque útiles en OCR, no fueron adecuados aquí porque podían eliminar detalles importantes en imágenes degradadas o con variaciones de color. En cambio, el preprocesamiento aplicado permitió resaltar los textos y reducir el ruido.

4.1.3 Modelo Utilizado

Para la detección de textos e imágenes en los documentos, se evaluaron distintas opciones y se decidió utilizar YOLOv11 con OBB (Oriented Bounding Boxes), una variante de YOLO diseñada para detectar objetos considerando su orientación. Esta característica es clave en nuestro caso, ya que muchos documentos contienen textos en distintos ángulos y formatos, lo que hace que un modelo de detección tradicional basado en cajas alineadas al eje XY no sea suficiente.

YOLOv11 OBB permite identificar tanto imágenes como textos en un mismo análisis, evitando la necesidad de entrenar modelos separados para cada tipo de objeto. Además, su capacidad para detectar elementos inclinados facilita la rotación del texto a su alineación original para una mejor integración con sistemas OCR, lo que mejora la extracción de información en documentos con textos rotados o mal escaneados.

Otra ventaja clave es que YOLO es un modelo de una sola etapa (one-stage detector), lo que optimiza la velocidad de inferencia sin comprometer demasiado la precisión. Esto nos permite procesar grandes volúmenes de imágenes de manera eficiente, algo esencial en este proyecto.

Por otro lado, YOLO nos brinda la posibilidad de refinar el modelo en el futuro agregando nuevas imágenes etiquetadas, lo que ayudaría a mejorar las métricas en escenarios más complejos. Además, su compatibilidad con pesos preentrenados fa-

cilita la adaptación del modelo a nuestros datos, permitiendo que aprenda mejor a reconocer patrones complejos como textos deformados, fondos borrosos o imágenes con baja calidad.

Antes de tomar la decisión, analizamos otras arquitecturas de detección de objetos que podrían haber sido útiles para nuestro caso, pero cada una presentaba limitaciones que hacían que YOLOv11 OBB fuera la mejor opción:

- **Faster R-CNN:** Aunque tiene una alta precisión, es más lento porque opera en dos etapas (two-stage detector). Su procesamiento más detallado es útil en algunos casos, pero en nuestro contexto, donde hay un gran volumen de imágenes, el tiempo de inferencia es un factor crítico.
- **SSD (Single Shot MultiBox Detector):** Similar a YOLO en estructura, pero con menor rendimiento en la detección de objetos pequeños y textos orientados en distintos ángulos.
- **RetinaNet:** Presenta mejoras en la detección de clases poco representadas gracias a su función de pérdida focal (Focal Loss), pero su tiempo de inferencia es mayor. Además, no maneja bounding boxes orientados, lo que lo hacía menos efectivo para textos en diferentes ángulos.

Dado que nuestra prioridad era un modelo rápido, eficiente y capaz de manejar textos en múltiples orientaciones, YOLOv11 OBB fue la mejor opción.

El modelo utilizado puede seguir refinándose en el futuro, permitiendo mejorar su desempeño mediante la incorporación de nuevas fotografías y el ajuste de sus parámetros en función de los escenarios más complejos. Una de sus ventajas es la posibilidad de utilizar modelos preentrenados y adaptarlos a nuestros datos, lo que facilita su optimización sin necesidad de entrenarlo desde cero.

YOLO, al ser un modelo de Deep Learning, tiene la capacidad de reconocer patrones complejos en las imágenes, como textos deformados, fondos borrosos o documentos con baja calidad de escaneo. Además, emplea técnicas de multi-escala, como Feature Pyramid Networks (FPN), que permiten mejorar la detección de objetos de distintos tamaños dentro de una imagen. FPN funciona combinando características extraídas en diferentes niveles de la red neuronal, lo que ayuda a identificar tanto elementos grandes como detalles pequeños con mayor precisión. Esto resulta especialmente útil en nuestro caso, donde es necesario detectar captions, anotaciones marginales y otros textos de tamaño reducido que podrían perderse con métodos de detección convencionales.

Para garantizar un equilibrio entre precisión y eficiencia, se optó por la versión YOLOv11 Nano, que permite procesar grandes volúmenes de imágenes de forma rápida y sin necesidad de hardware de alto rendimiento.

4.2 Métricas de evaluación en la detección de objetos

Para evaluar el desempeño de los modelos de detección de objetos, se emplean diversas métricas que permiten medir su precisión y capacidad de generalización. Estas métricas proporcionan información clave sobre la efectividad del modelo en la identificación y localización de objetos dentro de una imagen. Algunas de estas métricas son clásicas, heredadas de la evaluación de modelos de aprendizaje automático y adaptadas a modelos de detección de objetos.

4.2.1 Matriz de Confusión

La matriz de confusión proporciona un desglose detallado de las predicciones:

- **Verdaderos Positivos (TP):** El modelo predijo correctamente una etiqueta que coincide con el cuadro delimitador real (ground truth).
- **Verdaderos Negativos (TN):** El modelo no predijo una etiqueta y esta tampoco estaba en el ground truth.
- **Falsos Positivos (FP):** El modelo predijo una etiqueta incorrectamente cuando no estaba presente en el ground truth.
- **Falsos Negativos (FN):** El modelo no predijo una etiqueta que sí estaba en el ground truth.

4.2.2 Precisión y Recall

Las métricas de precisión y recall son esenciales; permiten cuantificar su capacidad para identificar correctamente los objetos sin generar excesivos falsos positivos ni omitir detecciones relevantes.

- **Precisión:** Representa la proporción de verdaderos positivos con respecto al total de predicciones positivas realizadas por el modelo. Se expresa mediante la siguiente ecuación:

$$\text{Precisión} = \frac{TP}{TP + FP} \quad (1)$$

donde TP (True Positives) indica el número de detecciones correctas y FP (False Positives) representa los casos en los que el modelo predijo erróneamente la presencia de un objeto.

- **Recall:** Mide la capacidad del modelo para identificar correctamente todas las instancias de una clase, asegurando que no se omitan objetos relevantes. Se define como:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2)$$

donde FN (False Negatives) representa la cantidad de objetos que el modelo no detectó correctamente.

4.2.3 Intersección sobre Unión (IoU)

La métrica *Intersection over Union* (IoU) mide el grado de superposición entre un cuadro delimitador predicho y el cuadro delimitador real. Se define como la razón entre el área de intersección y el área de unión de ambas regiones:

$$IoU = \frac{\text{Área de intersección}}{\text{Área de unión}} \quad (3)$$

Un valor de IoU más alto indica una mejor precisión en la localización del objeto detectado.

4.2.4 Precisión Promedio (AP) y Precisión Promedio Media (mAP)

La Precisión Promedio (AP) se calcula como el área bajo la curva de precisión-recall, proporcionando un valor único que resume el rendimiento del modelo en términos de precisión y cobertura. Este valor es particularmente útil para evaluar modelos en tareas donde la detección de cada instancia es crítica.

La Mean Average Precision (mAP) extiende el concepto de AP al considerar múltiples clases. Se obtiene como el promedio de los valores de AP para todas las clases presentes en el conjunto de datos, proporcionando una evaluación integral del modelo en escenarios de detección de múltiples clases.

Dos de las variantes más importantes de esta métrica son:

- **mAP@50:** Calcula la precisión promedio considerando un umbral de IoU de 0.5, lo que significa que una predicción se considera correcta si su IoU con la verdad de terreno supera este valor.
- **mAP@50:95:** Promedia los valores de AP considerando umbrales de IoU que van desde 0.5 hasta 0.95 en incrementos de 0.05, ofreciendo una evaluación más robusta del desempeño del modelo.

Las ecuaciones para calcular mAP se expresan como:

$$AP = \int_0^1 P(R) dR \quad (4)$$

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (5)$$

donde $P(R)$ es la precisión en función del recall, y N es el número total de clases.

En la detección de textos, un mAP@50:95 alto garantiza que se capturen todas las regiones de texto sin omitir partes críticas, facilitando el posterior reconocimiento óptico de caracteres (OCR). Un alto mAP@50:95 asegura que las imágenes sean detectadas y extraídas con precisión, minimizando la necesidad de intervención manual para recortes o ajustes posteriores.

Un bajo valor de mAP puede indicar la necesidad de refinamientos en el modelo. Un IoU bajo sugiere dificultades en la localización precisa de los objetos, lo que podría mejorarse con diferentes métodos de delimitación. Un bajo valor de precisión

implica un alto número de falsas detecciones, mientras que un bajo recall indica que el modelo está omitiendo objetos relevantes. Evaluar estos valores permite ajustar parámetros como los umbrales de confianza y mejorar la calidad de los datos de entrenamiento.

4.3 Experimentos

Para el proceso de test y evaluación de nuestro modelo se etiquetaron nuevas imágenes, quedando un total de 197 documentos, teniendo 984 cuadros de texto 267 de imágenes, 187 captions y 60 escritos a mano. Donde el valor de mAP lo escogemos como representativo para medir la calidad de nuestro modelo.

4.4 Ajuste de hiperparametros

YOLO cuenta con una interfaz que facilita la configuración de los hiperparámetros del modelo. Este proceso resulta esencial, ya que YOLO incluye aproximadamente 30 hiperparámetros, de los cuales algunos están relacionados con técnicas de *data augmentation* que buscan optimizar el rendimiento del modelo, y otros mas conocidos como el learning rate y el numero de épocas(epotchs). Según la documentación oficial de Ultralytics, el ajuste de estos hiperparámetros puede mejorar significativamente la precisión y la capacidad de generalización del modelo, permitiendo adaptarse mejor a diferentes conjuntos de datos y condiciones de entrenamiento (Ultralytics, 2025). El ajuste de hiperparametros de YOLO es proceso iterativo usando algoritmos genéticos, por lo cual es un proceso iterativo en el cual se exploran un espacio de hiperparametros luego se generan nuevos y se sigue probando hasta que el modelo converge. Este proceso demora horas, por lo que nosotros probamos distintas configuraciones (cantidad de iteraciones) para cada uno de los escenarios que se describen a continuación. Luego de obtener un conjunto de parámetros con métricas aceptables realizamos el entrenamiento de nuestro modelo y su proceso de test.

4.4.1 Primer Entrenamiento

En los primeros entrenamientos, utilizamos un conjunto de 893 fotografías, con un total de 2,903 textos, 341 imágenes, 223 captions y 980 escritos a mano etiquetados. YOLO, por defecto, genera métricas durante el proceso de entrenamiento utilizando los datos de validación. Sin embargo, nos enfocaremos en aquellas métricas que resultan más relevantes para nuestro análisis, evaluando el modelo con el conjunto de prueba definido previamente. En particular, analizaremos las clases imagen, texto y caption, ya que son fundamentales para la resolución de nuestro problema.

| Clase | Imágenes | Instancias | Box(P) | R | mAP50 | mAP50-95 |
|---------|----------|------------|--------|-------|-------|----------|
| all | 197 | 1438 | 0.816 | 0.719 | 0.758 | 0.61 |
| caption | 134 | 187 | 0.834 | 0.572 | 0.717 | 0.452 |
| imagen | 193 | 267 | 0.926 | 0.705 | 0.796 | 0.767 |
| texto | 197 | 984 | 0.687 | 0.881 | 0.762 | 0.612 |

Table 1: Métricas del modelo

Como se puede observar, aún presenta oportunidades de mejora el mPA de las clases imagen y texto, en las que obtuvo un mAP50-90 de 0.76 y 0.612 respectivamente. Por ello, hemos decidido aumentar la cantidad de datos en estas categorías con el objetivo de optimizar las métricas obtenidas y mejorar su desempeño.

4.4.2 Segundo Entrenamiento - Aumento de datos

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.

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

| Clase | Imágenes | Instancias | Box(P) | R | mAP50 | mAP50-95 |
|---------|----------|------------|--------|-------|-------|----------|
| all | 197 | 1438 | 0.790 | 0.838 | 0.836 | 0.645 |
| caption | 134 | 187 | 0.623 | 0.646 | 0.616 | 0.231 |
| imagen | 193 | 267 | 0.960 | 0.985 | 0.984 | 0.955 |
| texto | 197 | 984 | 0.786 | 0.883 | 0.907 | 0.748 |

Table 2: Métricas del modelo en distintas clases

Como se puede observar, el modelo logró mejores resultados en las evaluaciones. Sin embargo, la eficiencia en la clase *caption* disminuyó ligeramente. Esto no representa una preocupación significativa, ya que la clase era experimental y constituye un subconjunto de la clase texto.

4.4.3 Experimento preprocesando la imágenes antes del entrenamiento

Como mejora al modelo, de los entrenamientos 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.

| Class | Images | Instances | Box(P) | R | mAP50 | mAP50-95 |
|---------|--------|-----------|--------|-------|-------|----------|
| all | 197 | 1438 | 0.794 | 0.834 | 0.862 | 0.655 |
| caption | 134 | 187 | 0.677 | 0.652 | 0.703 | 0.289 |
| imagen | 193 | 267 | 0.947 | 0.981 | 0.985 | 0.961 |
| texto | 197 | 984 | 0.759 | 0.869 | 0.897 | 0.714 |

Table 3: Resultados de la evaluación de clases

A pesar de las limitaciones de recursos que redujeron la cantidad de epochs en el entrenamiento, el modelo logró mejoras significativas en las métricas de mAP50 para la clase caption. No obstante, se observó una disminución en su mAP50-95 en comparación con entrenamientos previos.

Concluimos que se lograron mejoras en el mAP50-95 para las clases imagen y texto, alcanzando valores de **0.961** y **0.714**, respectivamente. Esto refleja una mayor precisión y capacidad de generalización en la detección de estos elementos.

En consecuencia, este modelo representa el mejor desempeño obtenido hasta el momento, consolidando su eficacia en la tarea de reconocimiento. Además, demuestra ser una solución efectiva y robusta, capaz de ejecutarse y desplegarse con facilidad, sin requerir un alto consumo de recursos computacionales.

4.5 Asociación de imágenes y texto

Una vez detectados los textos y las imágenes dentro de cada fotografía, el siguiente paso fue su correcta extracción y preparación para la fase de reconocimiento de texto (OCR). Para esto, se procesaron los resultados generados por YOLOv11 OBB, extrayendo los fragmentos de interés y aplicando preprocesamiento a las regiones de texto antes de enviarlas a Tesseract OCR.

4.5.1 Procesamiento de los Resultados de YOLO

Después de ejecutar YOLO sobre cada imagen, el modelo nos devolvió un conjunto de bounding boxes, cada uno con las coordenadas (x_1, x_2, x_3, x_4) que definen la posición y tamaño de cada objeto detectado. Con esta información, generamos un archivo JSON por cada imagen. Este formato nos permitió almacenar las detecciones de cada imagen de manera estructurada, facilitando los siguientes pasos del procesamiento.

Con estas coordenadas, recortamos los elementos detectados (crops), generando imágenes individuales de cada fragmento de texto o imagen dentro del documento.

4.5.2 Preprocesamiento Antes del OCR

Antes de enviar los textos recortados a Tesseract OCR, realizamos un preprocesamiento adicional para optimizar su legibilidad y mejorar los resultados de la extracción. Dado que los documentos analizados presentan variaciones en calidad, iluminación y degradación del papel, sabíamos que el rendimiento de Tesseract dependía en gran medida de la calidad del texto en la imagen, probamos distintas combinaciones de preprocesamiento para determinar cuáles ofrecían mejores resultados en nuestro caso específico.

Para seleccionar la mejor combinación de preprocesamiento, realizamos un experimento utilizando un conjunto reducido de crops de texto y probamos diferentes combinaciones de técnicas. Luego, extraímos los textos con Tesseract y los comparamos con el texto real. Para determinar qué combinación era más eficiente, comparamos el caso promedio de este conjunto utilizando 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:

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

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:

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

Para obtener el texto de referencia, utilizamos una funcionalidad de macOS que permite copiar texto directamente desde imágenes con alta precisión. Esto nos permitió comparar los resultados de Tesseract sin necesidad de transcribir manualmente los textos.

Tras probar distintas combinaciones de preprocesamiento, encontramos que la más efectiva fue:

- **Inversión de colores** (negativo de la imagen)
- **Conversión a escala de grises**

Esta combinación mejoró la extracción de texto en documentos con fondos complejos y degradados, un problema común en nuestro dataset debido a la antigüedad de los documentos.

4.5.3 Uso de Tesseract OCR

Tesseract OCR es un modelo de reconocimiento óptico de caracteres ampliamente utilizado para la extracción de texto a partir de imágenes. Se encuentra preentrenado para datos en español, lo que permitió su uso directo en este proyecto sin necesidad de realizar un entrenamiento adicional.

Al ser un proyecto open source, cuenta con el respaldo de una comunidad activa que contribuye con mejoras continuas y optimizaciones periódicas. Además, incorpora técnicas avanzadas de **Optical Character Recognition** (OCR) basadas en deep learning, lo que lo hace eficiente en la extracción de texto, incluso en imágenes con variabilidad en calidad.

Su accesibilidad y facilidad de implementación fueron factores clave en su elección, ya que permitió integrar un sistema de reconocimiento robusto sin costos adicionales ni requerimientos computacionales elevados.

Antes de elegir Tesseract, evaluamos otras alternativas de OCR, pero cada una presentaba limitaciones que la hacían menos viable para este proyecto. OCROpus fue una de las opciones consideradas, pero no cuenta con modelos preentrenados en español, lo que habría requerido un entrenamiento adicional con un alto costo en tiempo y recursos.

También analizamos servicios como Google Cloud Vision y Amazon Textract, que ofrecen muy buen rendimiento, pero son de pago, lo que los hacía inviables dentro de las restricciones del proyecto. Kraken OCR fue otro candidato, ya que es un modelo más robusto, pero su versión preentrenada está enfocada en textos manuscritos históricos, lo que no se ajustaba del todo a nuestras necesidades. Además, su implementación requiere mayor capacidad de cómputo, lo que implicaba un esfuerzo adicional sin una ventaja significativa frente a otras opciones.

Finalmente, realizamos algunas pruebas y los resultados fueron inferiores en precisión y consistencia en comparación con Tesseract. Este ofreció el mejor balance entre calidad de extracción y viabilidad técnica, permitiéndonos procesar grandes volúmenes de documentos sin necesidad de entrenamientos adicionales ni costos elevados.

4.5.4 Evaluación de Resultados del Preprocesamiento

Para respaldar la selección del mejor preprocesamiento, analizamos los resultados obtenidos utilizando las métricas anteriormente mencionadas de Similitud de Jaccard y Character Error Rate (CER). Los resultados fueron los siguientes:

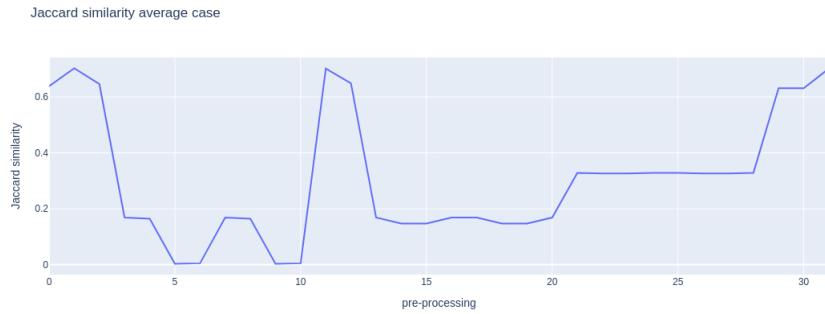


Figure 6: Gráfico de Similitud de Jaccard en el caso promedio.

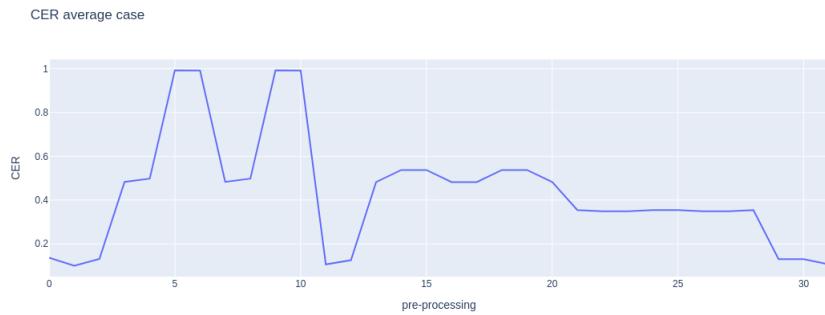


Figure 7: Gráfico de Character Error Rate (CER) en el caso promedio.

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.

4.5.5 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), con el objetivo de corregir errores del OCR, mejorando la precisión del texto extraído. Completar fragmentos de información faltantes, basándose en el contexto del documento y optimizar la calidad del texto final, asegurando que sea más comprensible y estructurado.

Para esta tarea, empleamos Gemma2, un modelo LLM de OpenAI, que nos permitió enriquecer el texto extraído y mejorar su coherencia.

Existen métodos tradicionales para corregir errores de OCR, como el uso de diccionarios de palabras, modelos de corrección ortográfica o técnicas basadas en reglas, presentan varias limitaciones cuando se aplican a documentos históricos. En estos casos, el texto puede contener palabras antiguas o poco comunes que no siempre están registradas en los diccionarios modernos, lo que dificulta su corrección automática.

Además, el OCR no solo puede cometer errores en el reconocimiento de caracteres, sino también en la estructura del texto, separando incorrectamente palabras o fusionando caracteres de manera incorrecta, lo que un corrector ortográfico convencional no siempre puede detectar. Por otro lado, los enfoques basados en reglas requieren ajustes manuales para adaptarse a cada documento, lo que los hace poco escalables cuando se trabaja con grandes volúmenes de datos.

Para evaluar la efectividad, 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

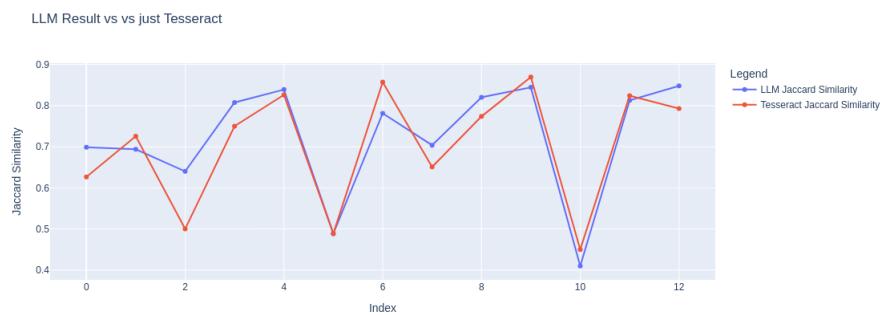


Figure 8: LLM vs. solo Tesseract utilizando similitud de Jaccard.

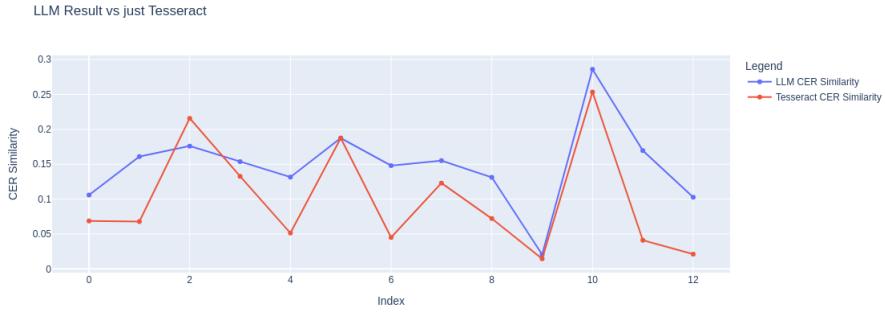


Figure 9: LLM vs. solo Tesseract utilizando CER.

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.

4.5.6 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 (Contrastive Language–Image Pre-training), un modelo diseñado para conectar representaciones visuales y textuales dentro de un mismo espacio de embeddings. CLIP fue entrenado con un gran conjunto de datos de pares imagen-texto.

A diferencia de otros enfoques que requieren una clasificación explícita o un entrenamiento supervisado extenso, CLIP opera bajo un método de aprendizaje contrastivo. Esto significa que el modelo aprende a colocar cerca entre sí los embeddings de imágenes y sus descripciones correspondientes, mientras aleja las imágenes de descripciones que no les pertenecen. Esta característica es clave para nuestro proyecto, ya que nos permite realizar asociaciones precisas incluso en casos donde los textos y las imágenes no siguen un patrón fijo o previamente etiquetado.

El modelo se compone de dos redes separadas: un encoder de imágenes, basado en redes neuronales convolucionales, y un encoder de texto, basado en transformers. Ambos generan representaciones vectoriales que se ubican en un espacio común, permitiendo medir su similitud mediante la métrica de coseno de similitud. Una puntuación alta en esta métrica indica una fuerte relación entre una imagen y un texto determinado, lo que facilita la tarea de identificación y emparejamiento.

La decisión de utilizar CLIP se basó en varias razones. En primer lugar, su capacidad de generalización sin entrenamiento adicional (Zero-Shot Classification) nos permitió aplicarlo directamente a nuestro problema sin requerir una fase de ajuste con datos específicos.

Se consideraron otras alternativas para esta tarea, como MKL-VisITA, que emplea Multi-Kernel Learning y funciona bien en bases de datos grandes. Sin embargo, estos modelos suelen requerir entrenamiento adicional y no ofrecen la misma capacidad de alineación inmediata entre imágenes y texto. CLIP, en cambio, nos brindó una solución lista para usar con una eficiencia computacional mucho mayor.

En nuestro caso, CLIP resultó ser la mejor opción porque permitió establecer una conexión precisa entre los textos y las imágenes de la colección sin necesidad de entrenamientos costosos. Esto redujo significativamente el tiempo de desarrollo y garantizó una asociación efectiva de los elementos en el proceso de digitalización de documentos.

4.6 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 *imágenes* 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.

4.7 Conclusiones

El proyecto de digitalización de la Colección Factivia de Emilio Roig de Leuchsenring logró desarrollar un sistema automatizado basado en técnicas avanzadas de *Deep Learning* para la detección, extracción y asociación de textos e imágenes en documentos históricos. Entre los principales logros se destacan:

1. **Detección Precisa de Elementos:** Mediante YOLOv11 OBB, se alcanzó una detección robusta de textos, imágenes, *captions* y escritos a mano, incluso en documentos con rotaciones, fondos degradados y deterioro. La precisión mejoró significativamente tras ajustes de hiperparámetros y aumento de datos etiquetados.
2. **Optimización del Preprocesamiento:** Técnicas como escala de grises y CLAHE demostraron ser críticas para mejorar el contraste y reducir el ruido, facilitando la segmentación de elementos.

3. **Extracción Efectiva de Texto:** La combinación de Tesseract OCR con inversión de colores y postprocesamiento mediante LLM (Gemma2) permitió corregir errores y completar información faltante, mejorando la calidad del texto digitalizado.
4. **Asociación Imagen-Texto:** CLIP demostró ser una herramienta eficaz para vincular imágenes con sus descripciones textuales, aprovechando su capacidad de aprendizaje contrastivo sin necesidad de entrenamiento adicional.

El mejor modelo entrenado fue el que utilizó preprocesamiento de las fotografías, lo que subraya la importancia de esta etapa en el pipeline de procesamiento. Sin embargo, se identificaron desafíos persistentes:

- La variabilidad en la calidad de los documentos afectó la consistencia del OCR, especialmente en textos desvanecidos o con fondos complejos.
- La clase *caption* presentó menor rendimiento debido a su diversidad estructural y escasez de datos etiquetados.
- La dependencia de modelos preentrenados limitó la adaptación a características únicas de documentos históricos.

A pesar de la complejidad de la solución, nos sentimos satisfechos con los resultados en la práctica. Para ilustrar su efectividad proponemos visitar un ejemplo de despliegue de esta aplicación mediante el framework Streamlit, accesible en la siguiente dirección:

<https://ml-facticia.streamlit.app/>

Este ejemplo permite explorar las capacidades del modelo en un entorno interactivo, brindando una demostración clara de su utilidad en tareas reales y que no requiere de dispositivos con muchas prestaciones para correr, y puede ser fácilmente integrado a un software especializado en el futuro.

4.8 Propuestas a Seguir

...

- Incrementar la diversidad de ejemplos para *captions* y textos manuscritos, enfocándose en casos críticos (rotaciones extremas, fondos irregulares).
- Incorporar técnicas de *data augmentation* sintética (p. ej., simulación de deterioro) para mejorar la generalización del modelo.
- Regularizar el *dataset* con un mayor equilibrio de clases para evitar sesgos en el modelo y alcanzar mejores resultados.

Mejora del Pipeline de OCR

- Evaluar motores OCR alternativos (Kraken, EasyOCR) para documentos históricos, combinándolos con modelos de lenguaje especializados en español antiguo.
- Implementar un módulo de corrección contextual basado en LLMs, entrenado con corpus históricos para abordar términos arcaicos o poco comunes.
- Probar técnicas avanzadas de preprocesamiento, como las descritas en Tolstoy AI - Transcripción de Periódicos, como alternativa para las imágenes con mejor calidad.

Optimización del Modelo de Detección

- Experimentar con arquitecturas como Mask R-CNN para segmentación precisa de elementos superpuestos o deformados.
- Experimentar con modelos recientes y como Segment Anything (SAM), y evaluar su desempeño en nuestro problema.
- Integrar *attention mechanisms* en YOLO para priorizar regiones críticas (p. ej., firmas, fechas).
- Probar versiones más robustas de YOLO, como YOLOv11 OBB M, que requieren mayor capacidad de cómputo pero podrían ofrecer mejoras significativas en la precisión.

Refinamiento de la Asociación Imagen-Texto

- Fine-tuning de CLIP con pares imagen-texto de la colección para adaptarlo a terminología y contextos específicos.
- Explorar modelos multimodales como MKL-VisITA para capturar relaciones semánticas complejas en documentos estructurados.

Despliegue en Plataforma Accesible

- Desarrollar una interfaz web que permita a historiadores y archivistas visualizar, buscar y corregir resultados de forma interactiva.
- Integrar herramientas de retroalimentación para recopilar correcciones humanas y reentrenar modelos de manera iterativa.

Escalabilidad y Transferibilidad

- Documentar un protocolo estandarizado para aplicar el sistema a otras colecciones históricas con mínima adaptación.
- Publicar *benchmarks* y conjuntos de datos anotados para fomentar la colaboración abierta en la comunidad académica.

4.9 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.
- [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/>.