TRABAJO PRÁCTICO INTEGRADOR

IA4.4 Procesamiento de Imágenes

Tecnicatura Universitaria en Inteligencia Artificial

Trabajo realizado por:
Katia Otterstedt, Lautaro Florenza y Sebastián Palacio

Ejercicio 1: Ecualización Local de Histograma

Introducción

El objetivo del primer ejercicio fue revelar detalles ocultos presentes en distintas zonas de la imagen provista. Para ello, implementamos un proceso de ecualización local del histograma, con el fin de realzar el contraste de manera adaptativa en regiones específicas de la imagen, permitiendo identificar información que no se distingue mediante una ecualización global.

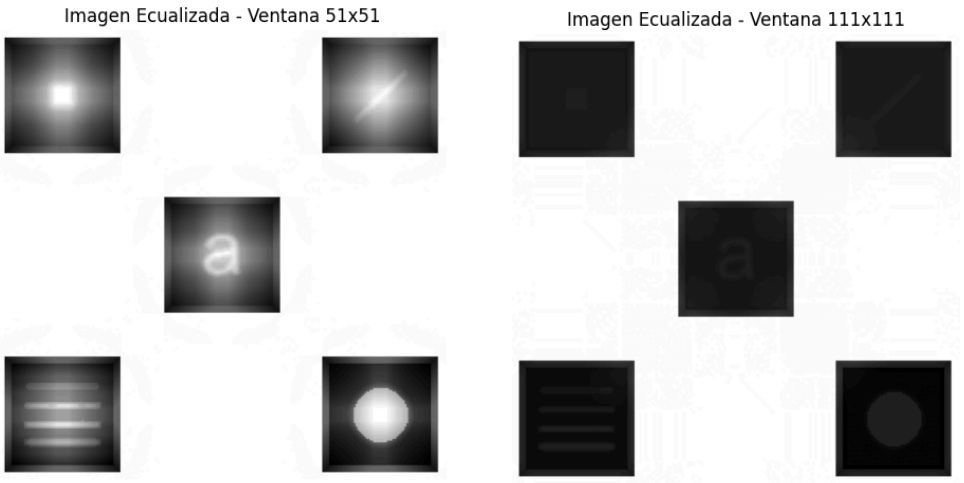
Desarrollo

El ejercicio 1 presenta una imagen con detalles ocultos que no se ven a simple vista. El problema plantea un análisis local de la imagen, a partir de la ecualización del histograma, para hacer visibles los detalles ocultos en diferentes zonas de la imagen.
Para ello, en primer lugar, se generó un suavizado de la imagen para disminuir el ruido. Posteriormente, se extendió la imagen original con un borde (o padding) para que la ventana pueda centrarse en los píxeles de los bordes; utilizando cv2.BORDER_REPLICATE, que replica el valor del píxel del borde. Esto es un paso muy importante ya que sin padding, al centrar la ventana en un píxel del borde, la ventana se saldría de la imagen.

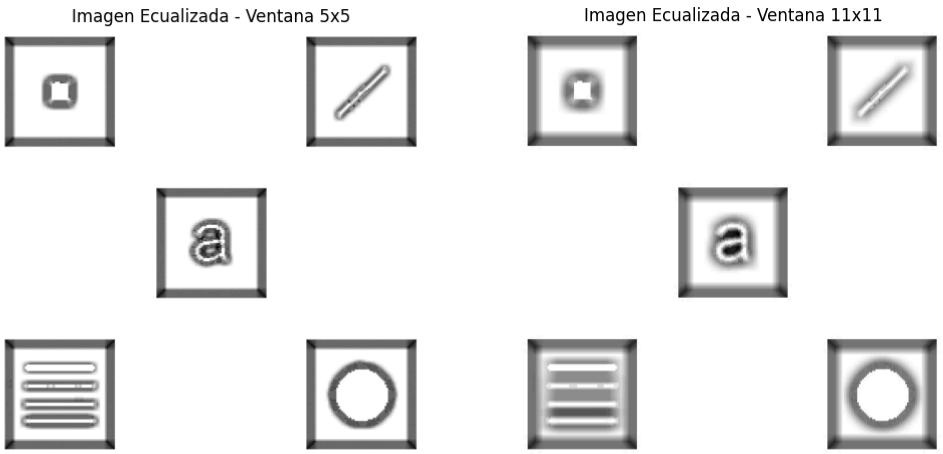
A su vez, se utilizaron kernels de distintos tamaños, tanto cuadradas como rectangulares, para poder analizar el efecto del tamaño de la ventana en la imagen ecualizada. En función de esto, se realiza el cálculo y normalización del histograma local.
Para facilitar la visualización de los resultados, se generó un diagrama en el cual se muestra por un lado, la imagen procesada, y por el otro, los histogramas, tanto de la imagen original como de la imagen ecualizada, para poder realizar una comparación entre ambas instancias.
Un punto clave de este ejercicio es el tamaño de la ventana, porque una ventana demasiado pequeña resulta muy sensible al ruido, mientras que una ventana muy grande, resulta en un histograma local muy similar al histograma global de la imagen, por lo que resulta en una ecualización similar a la global.
Para solventar este problema, iteramos el código a partir de distintos tamaños de ventana. Dado que el objetivo era identificar detalles escondidos en las diferentes zonas de la imagen, el probar distintos tipos de ventanas nos permitió identificar aquellos tamaños más apropiados para el lograrlo.

A partir de los análisis realizados, observamos que:

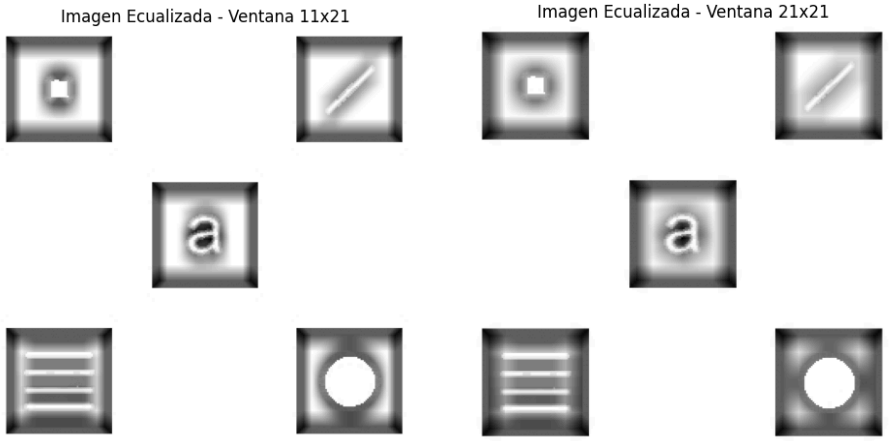
Ventanas grandes (por ejemplo, 51×51 y 111×111) generan una imagen con menor nivel de detalle local y una apariencia más homogénea, con un efecto similar al de una ecualización global, lo cual resulta desfavorable para poder visualizar detalles ocultos.



Ventanas pequeñas (como 5×5 y 11×11) producen un marcado contraste local. Esto permite identificar con claridad detalles ocultos y texturas finas, aunque incrementa la sensibilidad al ruido.



Ventanas intermedias (11×21 y 21×21) ofrecen un equilibrio adecuado entre nitidez y uniformidad. Estas dimensiones resultaron ser las más apropiadas para el propósito del ejercicio, ya que realzan detalles relevantes sin degradar la imagen con exceso de ruido.



Conclusión

Concluimos que la ecualización local del histograma es una técnica eficaz para revelar detalles ocultos en imágenes con contrastes no uniformes. El tamaño de la ventana constituye un parámetro crítico, debiendo seleccionarse en función de la escala de los detalles que se desea resaltar. En este ejercicio, las ventanas intermedias ofrecieron los mejores resultados, ya que permitieron realzar la información relevante sin introducir irregularidades visuales indeseadas.

La principal dificultad del ejercicio residió en la selección de la ventana óptima, ya que se requirió probar múltiples configuraciones para identificar aquella que maximizara la visualización de los detalles ocultos sin incrementar el ruido ni generar pérdida de uniformidad.

Ejercicio 2: Validación de formulario

Introducción

El propósito del segundo ejercicio fue desarrollar un procedimiento que permitiera validar de manera automática un formulario manuscrito a partir de su imagen digitalizada. El objetivo consistió en segmentar correctamente las celdas del formulario, identificar el contenido escrito en cada una de ellas y aplicar criterios de validación sobre los campos, verificando el cumplimiento de las consignas establecidas.

Desarrollo

El primer paso fue convertir la imagen en escala de grises a una imagen binaria mediante umbralización. Seleccionamos el valor de 150 después de analizar el histograma de las imágenes, determinando que este umbral separa efectivamente el contenido del fondo.
Para detectar las líneas que dividen las filas del formulario, utilizamos un enfoque basado en gradientes y proyecciones.
Sin embargo, notamos que las líneas tienen grosor, por lo que implementamos un filtrado de detecciones redundantes mediante análisis de distancia. Este proceso implementa elimina detecciones separadas por menos de 10 píxeles (las consideramos como parte de la misma línea). Aplicamos el mismo procedimiento en dirección horizontal para detectar las columnas.
Para evitar incluir píxeles de las líneas divisorias en las celdas, aplicamos márgenes, generando un conjunto de sub-imágenes correspondientes a cada celda del formulario, limpias de bordes.
Para contar caracteres y palabras, implementamos proyecciones binarias. Implementamos una operación morfológica de cierre en 1D para unir elementos cercanos.
Al estimar la cantidad de caracteres, nos encontramos con el problema de que el conteo directo de caracteres incluía los espacios entre palabras como separadores. Lo solucionamos ajustando la fórmula.
Para la validación de campos, generamos un diccionario con criterios definidos, verificando todas las restricciones simultáneamente:

Para las preguntas Sí/No, implementamos lógica de exclusividad mutua (XOR):
Inicialmente exploramos la clasificación automática mediante análisis de perfiles de proyección del encabezado, extrayendo características como:
- Valle central (detectar hueco en letra "A")
- Asimetría izquierda-derecha (letra "C" abierta)
- Densidad central (distinguir "B" llena vs "C" abierta)
Pero nos encontramos con el problema de que las letras en la tipografía utilizada tenían características geométricas muy similares, dificultando la discriminación automática con umbrales fijos.
Sin embargo, encontramos que las letras utilizadas en la tipografía del encabezado presentaban características geométricas muy similares, lo que dificultó su discriminación mediante umbrales fijos. Por esta razón, se adoptó un criterio más simple basado en la relación entre las densidades superior e inferior del perfil horizontal, permitiendo una clasificación aproximada entre formularios tipo A, B o C.

Finalmente, desarrollamos un procesamiento por lote que permite analizar automáticamente todos los archivos de formularios del directorio. Por cada uno, se genera un registro con los resultados de validación de cada campo y se guarda un archivo CSV con el resumen general. Además, se genera una imagen de salida (validacion_formularios.png) donde se visualiza el estado de cada formulario con recuadros verdes (“OK”) o rojos (“MAL”), indicando si cumple con todas las condiciones.

Conclusión

El sistema desarrollado cumple satisfactoriamente el objetivo de validar formularios manuscritos de manera automática, aplicando técnicas de procesamiento digital de imágenes, proyecciones binarias y reglas de validación definidas.

La estructura modular del código permitió dividir el proceso en etapas: segmentación, conteo de caracteres y palabras, validación de criterios y clasificación del tipo de formulario.

Las principales dificultades encontradas fueron la detección de bordes en formularios con bajo contraste y con la similitud geométrica de los encabezados al clasificar los tipos A, B y C.