

Tarea 2: Aplicación de filtros sobre imágenes y Chatbot

Edgar André Araya Vargas
Ingeniería en Computación
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
andrearaya1234@estudiantec.cr

Jose Daniel Rojas Calderon
Ingeniería en Computación
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
josedanielrojas5@estudiantec.cr

Abstract—Este trabajo se va a realizar el preprocesamiento de imágenes para clasificadores basados en redes neuronales convolucionales (CNN) y la creación de un chatbot utilizando procesamiento de lenguaje natural (NLP).

I. INTRODUCCIÓN

En esta tarea abordamos dos áreas fundamentales de la inteligencia artificial: el preprocesamiento de imágenes para clasificadores basados en redes neuronales convolucionales (CNN) y la creación de un chatbot utilizando procesamiento de lenguaje natural (NLP). El punto de unión de esta tarea es el hecho de que ambos enfoques están diseñados para mejorar la capacidad de los sistemas de tomar decisiones precisas basadas en datos que están "ruidosos" o son demasiado complejos, así como lo son las imágenes médicas y texto en lenguaje natural.

Para nuestra primera área de preprocesamiento de imágenes buscamos estudiar y aplicar una serie de filtros de imagen sobre un conjunto de datos de imágenes médicas relacionado con Covid-19, con el propósito de mejorar el rendimiento de clasificadores basados en CNN [1]. Los filtros aplicados incluyen técnicas comunes como el filtro de Blur, Sobel, Canny, y el Bilateral Filter, que permiten suavizar la imagen, reducir el ruido y preservar los bordes importantes para la detección de características [2].

Mientras que en nuestra segunda área parte del componente de procesamiento de lenguaje natural (NLP), se quiere desarrollar y estudiar un chatbot que utiliza embeddings de texto para seleccionar respuestas predefinidas basadas en la similitud con preguntas de entrada. Los embeddings de texto, que transforman palabras en vectores en un espacio semántico, son una técnica popular para medir la similitud entre frases, y se utilizan ampliamente en aplicaciones de NLP como chatbots, asistentes virtuales y sistemas de búsqueda.

El propósito de este proyecto es demostrar y al mismo tiempo enseñar cómo el preprocesamiento con filtros de imagen puede mejorar la clasificación en sistemas de visión por computadora y cómo el uso de embeddings en NLP puede facilitar la creación de un chatbot funcional. Ambos enfoques reflejan la importancia del preprocesamiento de datos en tareas de clasificación y respuesta automatizada.

II. REVISIÓN DE LITERATURA

A. Filtros de Imagen

1) **Blur Filter:** El filtro de desenfoque, o Blur Filter, se utiliza comúnmente para suavizar una imagen. Este proceso ayuda a reducir el ruido y a eliminar detalles innecesarios que podrían interferir con el aprendizaje del modelo. Al aplicar un desenfoque, se logra una representación más uniforme de la imagen, lo que puede ser beneficioso para evitar el sobreajuste durante el entrenamiento del clasificador. Este filtro es particularmente útil en escenarios donde las imágenes contienen variaciones de iluminación o ruido aleatorio que pueden afectar negativamente la precisión del modelo [3].

2) **Sobel Edge Detection:** La detección de bordes es un paso crítico en el análisis de imágenes, y el filtro Sobel es uno de los métodos más utilizados para este propósito. Este filtro calcula la derivada espacial de la imagen, resaltando las áreas donde hay cambios significativos en la intensidad del color. La aplicación del filtro Sobel permite identificar contornos y estructuras dentro de la imagen, lo que facilita la extracción de características importantes que las CNN pueden utilizar para clasificar imágenes. La capacidad del filtro Sobel para resaltar bordes hace que sea una herramienta valiosa en tareas donde la forma y los contornos son relevantes [4].

3) **Canny Edge Detection:** El método Canny es considerado uno de los algoritmos más efectivos para la detección de bordes en imágenes. A diferencia del filtro Sobel, el algoritmo Canny utiliza un enfoque multi-etapa que incluye suavizado, detección de bordes y supresión no máxima. Este proceso resulta en bordes más definidos y menos susceptibles al ruido. La precisión y robustez del método Canny lo convierten en una opción preferida para aplicaciones donde se requiere una detección precisa de bordes, lo cual es fundamental para mejorar el rendimiento de los clasificadores basados en CNN [5].

4) **Bilateral Filtering:** El filtrado bilateral es un método avanzado que combina las propiedades del filtrado espacial y la preservación de bordes. Este filtro suaviza la imagen al mismo tiempo que mantiene los bordes nítidos, lo cual es crucial para evitar la pérdida de información importante durante el preprocesamiento. El filtrado bilateral es especialmente útil cuando se trabaja con imágenes ruidosas, ya que

permite reducir el ruido sin comprometer la calidad visual de las características relevantes. Su capacidad para equilibrar la reducción del ruido y la preservación de detalles hace que sea un componente valioso en la preparación de datos para modelos de aprendizaje profundo [6].

B. Embeddings en NLP

III. METODOLOGÍA

A. Dataset de Imágenes

Para este estudio, utilizamos el Dataset Covid-19 Image Dataset disponible en Kaggle [1]. Este conjunto de datos es particularmente relevante para nuestro trabajo, ya que contiene imágenes médicas relacionadas con el COVID-19, lo que nos permite explorar la aplicación de filtros en un contexto real y de alta importancia.

El proceso de selección de imágenes fue el siguiente:

- 1) Descargamos el conjunto de datos completo de Kaggle.
- 2) Del set de entrenamiento, seleccionamos tres imágenes representativas, una por cada clase disponible en el dataset. Este proceso se realizó mediante el siguiente código:

```
categories = ['Covid', 'Normal', 'Viral_
Pneumonia']
images = {}
for category in categories:
    image_path = os.path.join(dataset_path,
category, os.listdir(os.path.join(
dataset_path, category))[0])
    images[category] = cv2.imread(image_path)
```

- 3) Debido a la naturaleza de las imágenes, las convertimos a un único canal (escala de grises) para mejorar los resultados. Esto se logró con la siguiente función:

```
def preprocess_image(image):
    return cv2.cvtColor(image, cv2.
COLOR_BGR2GRAY)

preprocessed_images = {cat: preprocess_image
(img) for cat, img in images.items() }
```

- 4) Estas imágenes fueron elegidas para representar la variabilidad presente en el conjunto de datos y para demostrar la efectividad de los filtros en diferentes tipos de imágenes médicas.

La diversidad en la selección de imágenes nos permite evaluar cómo los diferentes filtros afectan a distintos tipos de características visuales presentes en las imágenes médicas relacionadas con COVID-19.

B. Filtros de Imagen

Para el procesamiento de las imágenes seleccionadas, implementamos cuatro filtros diferentes utilizando la biblioteca OpenCV. Los filtros aplicados fueron:

- 1) **Filtro de Desenfoque (Blur Filter):** Utilizamos la función `cv2.blur()` para aplicar un desenfoque [7]. El parámetro clave aquí es el *tamaño del kernel*. Se variaron los tamaños del kernel (5,5) y (15,15) para

observar cómo afectaba la reducción de ruido y la preservación de detalles críticos en las imágenes.

- 2) **Detección de Bordes Sobel:** Implementamos el filtro Sobel usando `cv2.Sobel()` [8]. Los parámetros importantes incluyen el *orden de la derivada* y el *tamaño del kernel*. Se probaron tamaños de kernel en 'x' y 'y' desde 5 hasta 21 para optimizar la definición de los bordes.
- 3) **Detección de Bordes Canny:** Aplicamos el algoritmo Canny con `cv2.Canny()`. Este método proporciona resultados más precisos y es menos sensible al ruido en comparación con Sobel [9]. Se experimentó con *valores de umbral* y se utilizó `cv2.GaussianBlur()` para suavizar la imagen con un kernel de (5, 5) antes de la detección de bordes. Con Canny se ajustaron umbrales como (50, 150), (75, 175) y (150, 250) para obtener diferentes niveles de detalle.
- 4) **Filtro Bilateral:** Como filtro adicional de nuestra elección, implementamos el filtro bilateral usando `cv2.bilateralFilter()` [7]. Este filtro es interesante porque reduce el ruido mientras preserva los bordes, crucial en imágenes médicas. Se probaron parámetros como el *diámetro del filtro* (5 y 9), *sigma-Color* (25 y 100), y *sigmaSpace* (25, 75 y 100) con el fin de mantener la precisión en los detalles. A diferencia de los otros filtros, se utilizaron imágenes originales con 3 canales en lugar de las preprocesadas que fueron convertidas a un único canal, esto debido a la naturaleza del bilateral filtering que trabaja con color.

Para cada imagen seleccionada, aplicamos estos cuatro filtros y comparamos los resultados. El proceso de comparación incluyó:

- Análisis visual de las diferencias entre la imagen original y las imágenes filtradas.
- Evaluación de la efectividad de cada filtro en términos de reducción de ruido y mejora de características relevantes.
- Consideración de cómo cada filtro podría impactar en el rendimiento de un clasificador CNN al procesar estas imágenes.

C. Implementación en Jupyter Notebook

Para la implementación práctica de nuestro análisis, optamos por utilizar un *Jupyter Notebook* con Python y OpenCV. Esta elección nos proporcionó un entorno integrado ideal para nuestro estudio, permitiéndonos combinar código, visualizaciones y documentación en un solo documento interactivo. El uso de Jupyter Notebook facilitó la organización de nuestro trabajo, permitiendo una iteración rápida en la aplicación de filtros y el análisis inmediato de los resultados. Además, esta aproximación mejoró la reproducibilidad de nuestro estudio, ya que el notebook completo puede ser fácilmente compartido y ejecutado por otros investigadores.

Esta metodología nos proporciona una base sólida para evaluar el impacto de diferentes técnicas de preprocesamiento de imágenes en el contexto de la clasificación de imágenes médicas relacionadas con COVID-19, lo cual es crucial para

mejorar la precisión y robustez de los sistemas de diagnóstico asistidos por computadora.

D. Chatbot con Embeddings

IV. RESULTADOS

A. Resultados de los Filtros de Imagen

En esta sección presentaremos los resultados de los distintos filtrados de imágenes y exploraremos los puntos mencionados en la metodología. Además, se incluyen las imágenes originales a continuación para facilitar una comparación directa con los resultados obtenidos tras aplicar los filtros.



Fig. 1. Imagen Original

1) **Blur Filter:** Utilizando el filtro Blur (ver Figura 2) no se nota un gran cambio a diferencia de las imágenes originales, únicamente elimina ruido para notar más claramente las zonas afectadas.

El filtro de desenfoque puede reducir el ruido en las imágenes, ayudando al clasificador CNN a centrarse en características generales. Sin embargo, un desenfoque excesivo podría eliminar detalles importantes para diferenciar condiciones pulmonares, reduciendo la sensibilidad en la detección de patrones sutiles asociados con COVID-19 o neumonía viral. Encontrar el grado óptimo de desenfoque es esencial para equilibrar la reducción de ruido y la preservación de información diagnóstica.

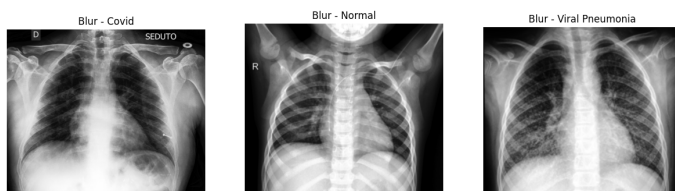


Fig. 2. Imagen Blur

2) **Sobel Edge Detection:** Con el filtro Sobel (ver Figura 3) podemos notar a diferencia del original que se resaltan los bordes pero aún mantiene ciertos gradientes de intensidad. Se denota como se enfatizan los límites pulmonares y los bordes con las costillas, de gran utilidad para la diferenciación entre COVID y neumonía. Al utilizar el filtro Sobel notamos cómo la presencia de COVID en el cuerpo y pulmones del paciente disminuye exponencialmente la enfatización de los bordes de las costillas y pulmones, mientras que el uso del filtro en pacientes con neumonía nos muestra más detalladamente estos bordes en comparación con un paciente sano, además

de la presencia de denotación de bordes o "ruido" dentro de los pulmones; así mostrando la inflamación causada por la neumonía más claramente [10].

En el caso del filtro Sobel, la diferenciación de bordes y gradientes mejora la capacidad del clasificador CNN para detectar cambios en los pulmones. Al mantener la información de intensidad, este filtro ayuda a identificar patrones de opacidad y consolidación asociados con COVID-19 y neumonía viral, permitiendo que la CNN reconozca mejor las diferencias en la textura pulmonar y los bordes anatómicos.



Fig. 3. Imagen Sobel

3) **Canny Edge Detection:** Los resultados del filtro Canny (ver Figura 4) daban imágenes ruidosas y mal definidas. Al utilizar el filtro gaussiano notamos una gran mejora en la extracción de bordes, mostrando contornos de pulmones y costillas. Se pierde parte de la información de la intensidad original, pero a cambio de eso obtenemos una alta diferenciación entre COVID, normalidad y neumonía. En un paciente sano vemos únicamente el contorno de sus costillas. En el paciente con COVID podemos notar cómo parece delimitarse la opacidad en vidrio esmerilado, definido como áreas más claras y difusas en la imagen indicando una afectación leve a moderada en el tejido pulmonar [11]. En el caso de la neumonía se resaltan los alvéolos inflamados aún más claramente que con el filtro Sobel [10].

Canny genera una representación binaria de los bordes, simplificando las imágenes. Esto podría facilitar al clasificador CNN detectar cambios estructurales en los pulmones, pero la pérdida de información de intensidad puede dificultar la diferenciación de patologías específicas. Usar Canny como un canal adicional junto con la imagen original permitiría que la CNN aprenda tanto características estructurales como detalles de intensidad.

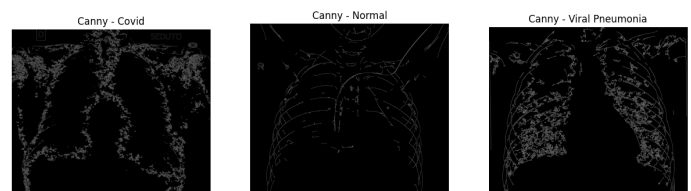


Fig. 4. Imagen Canny

4) **Bilateral Filtering:** Utilizando el Bilateral Filtering (ver Figura 6) podemos notar una exponencial mejora en la calidad de las imágenes. En general se resalta más claramente el sistema óseo y los pulmones con más claridad. Se disipa ligeramente el ruido y mejora la calidad de las imágenes. Este sería

un buen filtro para iniciar a tratar las imágenes. Este filtrado parece de extrema importancia para el preprocesamiento de imágenes. Nuevamente notamos los síntomas del COVID y neumonía claramente en los pulmones, más claro que en las originales y el paciente sano se ve más definido.

El filtro bilateral puede equilibrar el procesamiento de imágenes médicas en un clasificador CNN. Al reducir el ruido y preservar bordes, mantiene detalles críticos de estructuras anatómicas y anomalías patológicas. Esto podría mejorar la detección de sutilezas en la textura pulmonar, sin sacrificar la definición de los límites anatómicos, lo que potencialmente incrementaría la precisión en la clasificación de condiciones clínicamente significativas.



Fig. 5. Imagen Bilateral

B. Resultados del Chatbot

C. Resultados del Chatbot

Para el chatbot se selecciono el contexto de un hospital, para la realizacion de preguntas, ademas se selecciono el idioma ingles. El chatbot fue diseñado para brindar información a pacientes y visitantes de un hospital.

Carga de datos:

- Modelo de lenguaje: Se carga el modelo de lenguaje de spaCy, que proporciona una representación vectorial de palabras y frases en inglés.
- Respuestas: Se cargan las respuestas predefinidas desde un archivo JSON.

Creación de embeddings:

- Vectorización: Cada respuesta se convierte en un vector numérico (embedding) utilizando el modelo de spaCy. Estos vectores capturan la semántica de cada respuesta.
- Matriz de embeddings: Los embeddings de todas las respuestas se almacenan en una matriz NumPy.

Preprocesamiento de texto:

- Limpieza: La función preprocess realiza un preprocesamiento básico del texto, convirtiendo las palabras a su forma lema, pasando todo a minúsculas y eliminando las stop words (palabras comunes que aportan poco significado).

Cálculo de similitud:

- Vector de pregunta: La pregunta del usuario también se convierte en un vector utilizando el modelo de spaCy.
- Similitud coseno: Se calcula la similitud coseno entre el vector de la pregunta y cada uno de los vectores de las respuestas. Esta métrica mide qué tan similares son dos vectores en un espacio vectorial.

Selección de la respuesta más similar:

- Ordenamiento: Se ordenan las respuestas según su similitud con la pregunta.
- Selección: Se selecciona la respuesta con la mayor similitud.

V. ANÁLISIS DE LOS RESULTADOS DEL CHATBOT

A partir de la muestra de interacciones proporcionada, se puede realizar un análisis preliminar del desempeño del chatbot.

VI. RESPUESTA DE INTERACCION CHATBOT

```
Question: What are the visiting hours?
Selected response: Patients can have two visitors at a time during visiting hours.
-----
Question: What are the visiting rules?
Selected response: Family members can call the patient's room directly for updates.
-----
Question: How can I get a medical assistance?
Selected response: For emergency medical assistance, dial 911 immediately.
-----
Question: Is the emergency room open 24 hours?
Selected response: The emergency room is open 24 hours a day.
-----
Question: Information about Lab results
Selected response: Physical therapy sessions can be scheduled by referral from a doctor.
-----
Question: lab results
Selected response: Lab test results are usually available within 24 to 48 hours.
-----
Question: Physical therapy sessions?
Selected response: Family members can call the patient's room directly for updates.
-----
Question: Is there free WiFi?
Selected response: The hospital offers free WiFi for patients and visitors.
-----
Question: Is there a hospital cafeteria?
Selected response: Family members can call the patient's room directly for updates.
-----
```

Fig. 6. Respuestas de chatbot

Aqui podemos observar como en la mayoría de los casos las respuestas fueron congruentes con lo que se le pregunto, sin embargo en otros casos no fue asi. Se podria pensar que el uso de preguntas diferentes o la formulacion de la respuesta podria mejorarse para encontrar una similitud mas acertada.

VII. CONCLUSIONES

Las respuestas del chatbot fueron acertadas en la mayoría de los casos, sin embargo hay algunos que aunque no se salio del contexto, no fue la respuesta esperada.

Observaciones: Falta de especificidad: Las respuestas son a menudo genéricas y no abordan las preguntas de manera precisa. Repetición de respuestas: Se observan respuestas idénticas para preguntas similares, lo que sugiere una falta de flexibilidad en el modelo. Incomprensión de la intención del usuario: En algunos casos, el chatbot parece no captar el sentido de la pregunta. Posible sesgo en el entrenamiento: El modelo podría estar sesgado hacia ciertas preguntas o temas.

REFERENCES

- [1] <https://www.kaggle.com/datasets/pranavraikokte/covid19-image-dataset>
- [2] <https://www.freedium.cfd/https://sumitrsharma-ai.medium.com/image-filtering-in-computer-vision-ec60ec8a3e1>
- [3] https://docs.opencv.org/4.x/d4/d13/tutorial_py_filtering.html
- [4] <https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm>

- [5] https://www.cs.buap.mx/~daniel.valdes/docs/Deteccion_de_bordes_mediante_el_algoritmo_de_Canny.pdf
- [6] <https://www.cse.iitd.ac.in/~pkalra/col783-2017/bilateral-filtering.pdf>
- [7] https://docs.opencv.org/4.x/d4/d13/tutorial_py_filtering.html
- [8] https://docs.opencv.org/4.x/d5/d0f/tutorial_py_gradients.html
- [9] https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html
- [10] <https://www.mayoclinic.org/es/diseases-conditions/pneumonia/multimedia/chest-x-ray-showing-pneumonia/img-20005827>
- [11] <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7685043/#:~:text=Los%20hallazgos%20radiol%C3%B3gicos%20ms%20frecuentes,predominio%20en%20los%20campos%20inferiores.>

Criterios	Puntuación máxima	Puntuación obtenida
Descarga y mostrar imágenes del dataset original	10	
Aplicar y analizar filtro Blur	15	
Aplicar y analizar filtro Sobel Edge Detection	15	
Aplicar y analizar filtro Canny Edge Detection	15	
Aplicar y analizar filtro seleccionado	15	
Estructura y claridad del informe	30	
Rúbrica Tarea NLP		
Criterios	Puntuación máxima	Puntuación obtenida
Definición de propósito de chatbot y respuestas	20	
Convertir las respuestas en embeddings y almacenarlo en un dataset de conocimiento	15	
Convertir la pregunta del usuario en embeddings	15	
Comparar los embeddings con similitud de coseno y mostrar la respuesta más acorde.	50	