Clasificación de patologías en la retina a partir de imágenes OCT con Deep Learning

Isabella Bermón Rojas
Departamento de Ingeniería Electrónica
Universidad de Antioquia
Medellín, Colombia
isabella.bermonr@udea.edu.co

***Abstract*—The application of machine learning techniques to OCT aims to improve the accuracy and speed of image classification, providing valuable assistance to ophthalmologists in the diagnosis and monitoring of these diseases. This project aims to train different types of CNNs and vision transformer architectures to find the best possible solution to classify diseases into OCT scans, as well as to study the effects of preprocessing on this type of medical images.**

Keywords: CNNs, vision transformers, machine learning, image processing, medical imaging.

I. Dataset

El dataset utilizado es el **Labeled Optical Coherence Tomography (OCT)** disponible en Kaggle. Este conjunto de datos contiene imágenes de OCT que han sido etiquetadas según las 4 categorías de enfermedades o estados de salud ocular. El conjunto de datos está compuesto por imágenes de OCT en formato PNG, cada una con etiquetas correspondientes a las cuatro categorías mencionadas.

El dataset consta de 109,309 imágenes de OCT, ocupando alrededor de 6.70 GB en disco. El dataset se encuentra desbalanceado entre las cuatro clases, como se muestra en la siguiente figura.

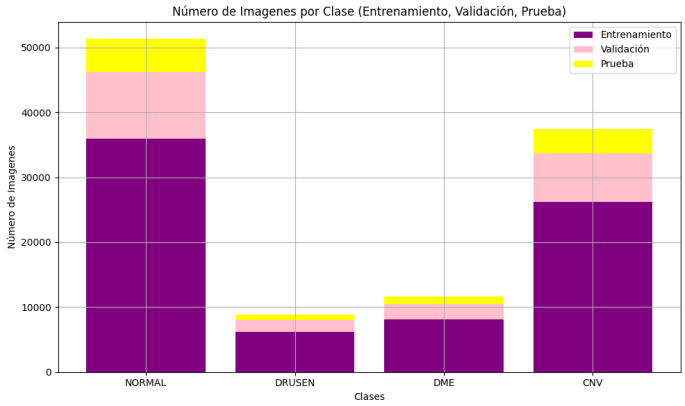

Fig. 1. Representación de base de datos dividida por patologías y su respectiva división en subconjuntos.

II. Estructura de los notebooks

01 - exploración de datos: Como indica el nombre del notebook su código se utiliza para analizar los datos existentes de la base de datos. Primero se deben cargar de Kaggle y cargar al entorno de Colab. Después se importan librerías y se cargan las funciones para graficar y cargar imagenes aleatorias. En la última sección de exploración se hace una gráfica de los datos por cada clase para subconjunto, adicionalmente se visualizan ejemplos de cada clase y las variaciones que hay en la base de datos (tamaño y ángulo en las imágenes).

02 - preprocesado: Este notebook repite los primeros pasos del notebook anteriores, descarga de datosy carga al entorno de Colab. Después se define una función para aplicar 4 filtros a la imagen (sobel, canny, gausiano y ecualizador). El objetivo de esta exploración de filtros es encontrar una representación de los datos que permita resaltar las diferencias entre cada clase, es decir, las características de cada enfermedad. Con base en los resultados obtenidos de los filtros, se encontró que el mejor contraste entre las clases se encuentra con el ecualizador, ya que resalta mucho mejor el patrón de las clases DME y CNV.

03 - arquitecturas de linea de base: En este notebook se repiten los primeros pasos de los dos notebooks anteriores, descarga de datos y carga al entorno de Colab. Se definen las funciones para cargar los datos, pre-procesar las imágenes por medio de transformaciones en tamaño y normalización, y finalmente se definen las funciones para evaluar los modelos finales, visualizar las predicciones y generar los reportes de las métricas obtenidas. Después se realiza el entrenamiento de 3 arquitecturas (Resnet,Swin transformer y ConvNext) y se cargan los pesos para realizar la evaluación de los modelos, calculando sus métricas y graficando su matriz de confusión.

04 - arquitecturas con datos filtrados: Se repiten los primeros pasos de los dos notebooks anteriores, descarga de datos y carga al entorno de Colab. También se repite el entrenamiento y prueba de los modelos pero con un cambio, ya que el entrenamiento de tantos datos toma mucho tiempo, esta vez se carga el 50% de la base de datos. Se entrenan dos

tipos de modelos, con los datos originales pero el cambio es que son con el 50% de la base de datos original y los datos filtrados con esta misma base de datos reducida.

05 - resultados: Este notebook repite los primeros pasos de los dos notebooks anteriores, descarga de datos y carga al entorno de Colab, además carga todos los modelos entrenados en un computador aparte de Colab debido al límite de créditos que permite con uso de GPU (los entrenados con toda la base de datos y el 50%), evalúa sus métricas y grafica sus matrices de confusión

III. Solución propuesta

En este proyecto se entrenaron 3 arquitecturas, 2 redes neuronales convolucionales y 1 transformador de visión, para comparar el rendimiento de diferentes arquitecturas de *deep learning* para clasificar imágenes OCT en cuatro categorías: NORMAL, CNV (neovascularización coroidea), DME (edema macular diabético) y DRUSEN: NORMAL, CNV (neovascularización coroidea), DME (edema macular diabético) y DRUSEN. El objetivo era seleccionar el mejor modelo para predecir el estado de enfermedad (una de las cuatro clases mencionadas) de la retina a partir de imágenes de OCT.

A continuación se muestra un ejemplo de imágenes aleatorias de cada clases y cada subconjunto. Se puede evidenciar que hay en algunos casos una rotación en el ángulo de las imágenes y el tamaño varía en muchos casos. Por esto se consideró hacer un preprocesamiento para rectificar el ángulo con procesamiento de imágenes pero al verificar en la base de datos se encontraron suficientes imágenes con estos casos por lo cual se decidió no hacer un preprocesamiento extra ya que hay suficientes ejemplos de estos casos y no va a sesgar el modelo en su entrenamiento.

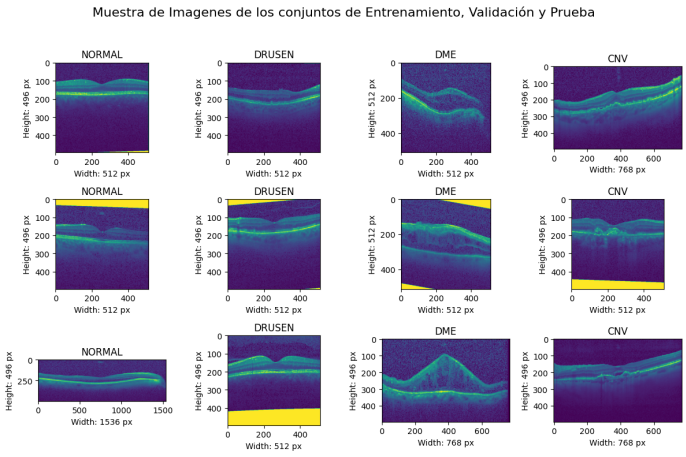

Fig. 2. Ejemplos de la base de datos.

A. Preprocesado

Los filtros de procesamiento de imágenes (Canny, Sobel y ecualización del histograma) tienen diferentes propósitos, y su utilidad depende del contexto y del tipo de modelo que se utiliza. La ecualización del histograma en una imagen mejora el contraste de la imagen distribuyendo mejor los niveles de intensidad de píxeles. Esto lo hace útil para mejorar la visibilidad de detalles ocultos en imágenes con poca iluminación o bajo contraste, que es el caso de esta base de datos. A continuación se muestra el efecto de cada uno de los filtros en las diferentes clases del problema.

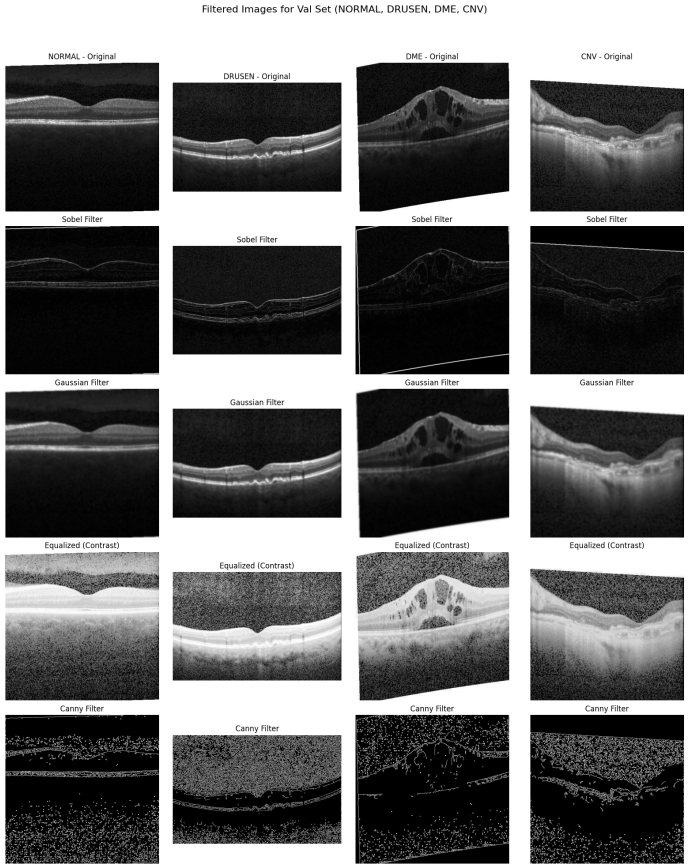

Fig. 3. Ejemplos de la base de datos filtrada.

Aparte del filtro implementado en la base de datos, se realizaron unas modificaciones a las imágenes. Los transforms en el código cumplen varias funciones fundamentales en el preprocesamiento y la preparación de los datos para entrenar modelos. Estas transformaciones se definen utilizando torchvision.transforms y su trabajo es cambiar el tamaño de las imágenes a un formato fijo (224x224) para la entrada de las tres arquitecturas. Adicionalmente, escalan los valores de los píxeles a un rango con media 0 y desviación estándar 1, utilizando valores promedio de imágenes de ImageNet ya que se utilizan modelos preentrenados en el desarrollo del proyecto.

B. Arquitectura

Se desarrolló un código para entrenar 3 tipos de arquitecturas para la clasificación de imágenes, utilizando varias arquitecturas preentrenadas y ajustándolas para un conjunto específico de clases. A continuación, se describen las arquitecturas utilizadas, las características del entrenamiento y los parámetros modificados durante el proceso de entrenamiento:

1. Arquitecturas utilizadas

El código permite entrenar tres tipos diferentes de modelos preentrenados mediante el reemplazo de la capa final (fully connected) por una nueva capa que tiene 4 unidades.

- **ResNet-50:** Se utiliza el modelo ResNet-50 preentrenado.
- **ConvNeXt:** Este modelo es una variante de los transformadores convolucionales, concretamente el modelo "ConvNeXt Tiny".
- **Swin Transformer:** Utiliza el modelo Swin Transformer preentrenado.

Modelo	Parámetros	Velocidad	Hardware Necesario
ResNet	Alto	Moderada	General
ConvNeXt	Bajo	Moderada	GPU preferible
Swin Transformer	Variable	Baja a moderada	GPU/TPU avanzada

TABLE I
COMPARACIÓN GENERAL DE LAS 3 ARQUITECTURAS

En imperativo recalcar el uso de la GPU para el entrenamiento de estos datos ya que es una gran cantidad y en las pruebas realizadas se encontró que para el entrenamiento de una sola epoca de la arquitectura ResNet en Colab se podía demorar más de 6 horas.

2. Características del Entrenamiento

El modelo se entrena utilizando las siguientes características clave:

- **Cargado de Datos:** Se entrenan los modelos base con toda la base de datos y el resto de modelos con sus variaciones en la base de datos reducida al 50%.
- **Pérdida (Loss function):** La función de pérdida utilizada es la `CrossEntropyLoss`, comúnmente utilizada para problemas de clasificación multiclase.
- **Epochs:** El modelo se entrena por un número definido de épocas, con un valor predeterminado de 4 para los modelos base y 10 para la base de datos reducida(`num_epochs`).

3. Parámetros modificados al entrenar

Durante el proceso de entrenamiento, se modifican varios parámetros clave:

- **Optimizador:** El optimizador utilizado en este caso es el `SGD (Stochastic Gradient Descent)` con un `learning rate` de 0.001 y un `momentum` de 0.9. Aunque también se hicieron pruebas con `Adam`, que es otro optimizador popular.
- **Tasa de aprendizaje (Learning rate):** En el optimizador `SGD`, el `learning rate` se establece inicialmente en 0.001. Este parámetro se ajusta durante el entrenamiento utilizando el scheduler `CosineAnnealingLR`.
- **Scheduler:** Se utiliza un `CosineAnnealingLR` para ajustar la tasa de aprendizaje durante el entrenamiento, lo que permite una disminución gradual y cíclica de la tasa de aprendizaje, ayudando al modelo a converger más rápidamente hacia una solución óptima. También se tiene la opción de usar un `StepLR`, que disminuye la tasa de aprendizaje en intervalos regulares pero el mejor desempeño se obtuvo con `CosineAnnealingLR`.

C. Métricas de desempeño

Al finalizar el entrenamiento, el modelo se evalúa utilizando el conjunto de prueba para obtener su desempeño en datos no vistos durante el entrenamiento. Se utilizaron las siguientes métricas en base a las métricas comúnmente reportadas en la literatura:

- **Exactitud (Balanced Accuracy):** Proporción de predicciones correctas sobre el total de predicciones.
- **Precisión (Precision):** Medida de la exactitud de las predicciones positivas.
- **Recall (Sensibilidad):** Capacidad del modelo para identificar correctamente las instancias positivas.
- **F1-Score:** Media armónica entre precisión y recall, útil para balancear ambas métricas.
- **Matriz de Confusión:** Para visualizar el desempeño del modelo en cada clase específica.

Para evaluar el impacto en el negocio y en la práctica clínica se consideran las siguiente métricas que se obtienen del proceso de evaluación de los modelos:

- **Tiempo de Diagnóstico:** El tiempo requerido para obtener un diagnóstico realizado por el modelo.
- **Costo de Implementación:** Requerimientos de hardware.

Se resalta el uso de utilizar **Balanced Accuracy** dentro de las métricas de evaluación ya que el desbalance entre las clases es siginificativo. Por ejemplo, la clase Normal tiene más de 50,000 imágenes mientras que la clase DRUSEN tiene menos de 10,000. Esto no solo implica un posible

sesgo al momento de entrenar sino al momento de evaluar el modelo debido a que el **Accuracy** clásico puede ofrecer resultados engañosos al ser influenciado por la predominancia de la clase mayoritaria. Por esta razón, **Balanced Accuracy** resulta más adecuada, ya que calcula la media de las tasas de acierto por clase, proporcionando una evaluación más equitativa del desempeño del modelo en todas las clases, independientemente de su proporción en el conjunto de datos.

IV. Resultados

La siguiente Tabla II, al final del informe, resume las métricas clave obtenidas para las tres arquitecturas (**ResNet-50**, **ConvNeXt**, y **Swin Transformer**) en tres experimentos diferentes para una de las BD (base de datos).

El mejor modelo identificado en los experimentos es el **Swin Transformer** con la base de datos original. En comparación con las demás arquitecturas (ResNet-50 y ConvNeXt) y experimentos (reducción de datos y ecualización), el Swin Transformer sobresale al obtener las métricas más altas.

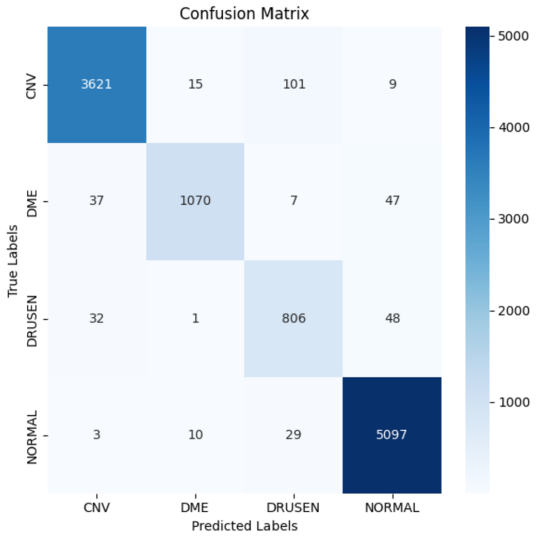

Fig. 4. Matriz de confusión de Swin Transformer.

Además, los resultados de la matriz de confusión refuerzan este desempeño,evidenciando que el desbalance de las clases no afectó significativamente el desempeño del modelo y que obtuvo:

1) Altas tasas de predicción correcta para todas las clases (CNV, DME, DRUSEN y NORMAL).
2) Errores mínimos entre clases (valores fuera de la diagonal principal son pequeños).

Aunque ResNet-50 tiene un desempeño cercano al Swin Transformer en la base de datos original (94.32% vs. 94.72% en Balanced Accuracy), el Swin Transformer lo supera en todas las métricas principales. Por otro lado, ConvNext presenta un desempeño notablemente inferior. Adicionalmente se

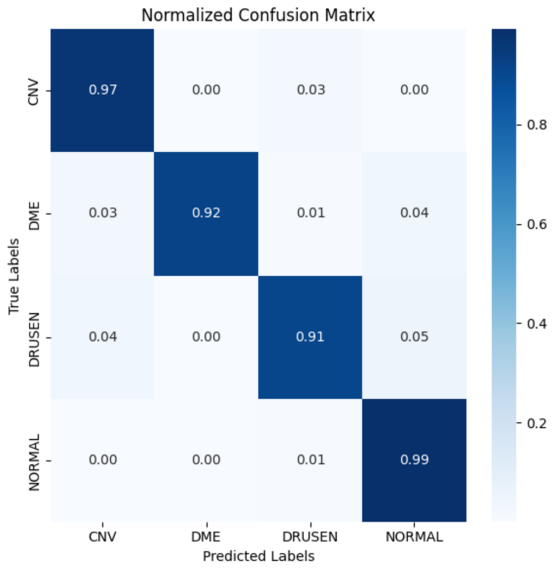

Fig. 5. Matriz de confusión de Swin Transformer.

adjunta una muestra aleatoria del conjunto de prueba al que aleatoriamente se le extrajeron 10 ejemplos con sus respectivas predicciones.

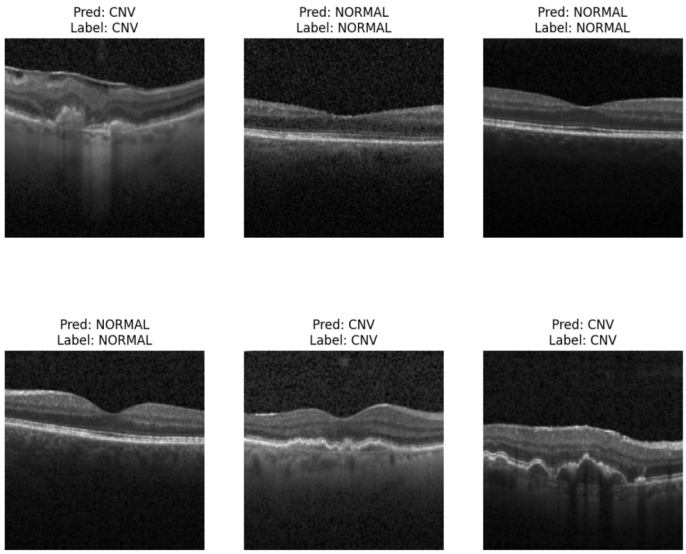

Fig. 6. Predicciones de Swin Transformer.

V. Discusión

Al observar las métricas en los experimentos con datos reducidos o ecualizados, las métricas tienden a disminuir. Esto sugiere que el modelo aprovecha mejor la riqueza de los datos originales. La arquitectura transformer, Swin Transformer, demuestra ser altamente efectiva con los datos originales, posiblemente debido a su capacidad para capturar relaciones globales y locales en imágenes.

Arquitectura	Experimento	Balanced Accuracy	Precision	Recall	F1-Score
ResNet-50	BD original	94.32	96.74	96.69	96.70
	BD 50%	94.1	96.76	96.77	96.76
	BD 50% + ecualizado	92.75	96.31	96.35	96.31
ConvNeXt	BD original	85.52	92.98	93.16	92.87
	BD 50%	89.21	93.26	93.1	93.16
	BD 50% + ecualizado	77.49	88.86	88.42	87.89
Swin Transformer	BD original	**94.72**	**96.96**	**96.9**	**96**.91
	BD 50%	93.72	96.5	96.5	96.5
	BD 50% + ecualizado	92.71	96.11	96.13	96.11

TABLE II
COMPARACIÓN DE MÉTRICAS PARA LAS ARQUITECTURAS.

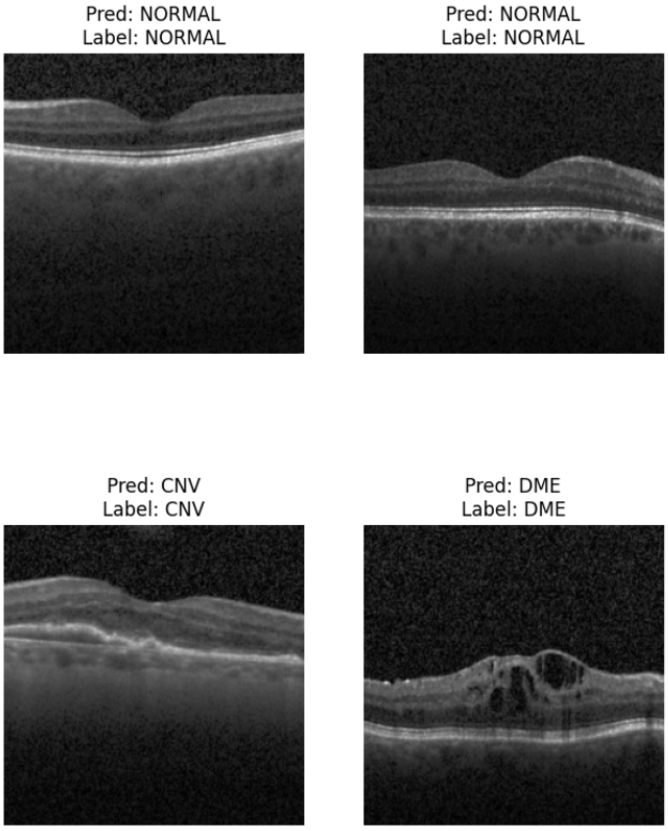

Fig. 7. Predicciones de Swin Transformer.

REFERENCES

[1] Elsharif, A. A. E. F., Abu-Naser, S. S. (2022). Retina diseases diagnosis using deep learning.

[2] (2022). Classification of Retinal OCT Images Using Deep Learning. doi: 10.1109/icci54379.2022.9740985

[3] De Fauw, J., Ledsam, J. R., Romera-Paredes, B., Nikolov, S., Tomasev, N., Blackwell, S., ... Ronneberger, O. (2018). Clinically applicable deep learning for diagnosis and referral in retinal disease. Nature medicine, 24(9), 1342-1350.