Segmentación de Capas de Grafeno con U-Net y EfficientNet-B5: Un Enfoque Robusto para Datos Limitados

Camilo Andres Cuellar Benito
Facultad de Ingeniería
Universidad de Antioquia
Medellin, Antioquia
camilo.cuellar@udea.edu.co

Leonardo Jose Amaris Dominguez
Facultad de Ingeniería
Universidad de Antioquia
Medellin, Antioquia
leonardo.amaris@udea.edu.co

***Resumen*—Graphene is a two-dimensional material whose properties are strongly dependent on the number of layers in each sample. Distinguishing between monolayer, few-layer, and thick graphene is critical for various applications, although conventional characterization methods such as Raman spectroscopy or atomic force microscopy present practical limitations in cost and scalability.**

In this work, we developed a semantic segmentation model based on convolutional neural networks to automatically identify graphene flakes in optical microscopy images. The proposed pipeline includes dataset transformation, preprocessing adapted to class imbalance, and the training of a U-Net with a pretrained encoder.

To mitigate the effects of limited data, a composite loss function was employed, combining weighted Cross-Entropy, Dice Loss, and Focal Loss. Results show promising performance, with a mean IoU close to 0.70 and accuracy above 80 % for the bulk and few-layer classes, confirming the feasibility of the approach and highlighting its potential to be extended to other two-dimensional materials.

***Index Terms*—graphene, deep learning, convolutional neural networks, optical microscopy, class imbalance, few-shot learning**

I. Resumen

El grafeno es un material bidimensional cuyas propiedades dependen directamente del número de capas presentes en cada muestra. La distinción entre monocapa, pocas capas y grafeno grueso resulta esencial para múltiples aplicaciones, aunque los métodos convencionales de caracterización, como espectroscopía Raman o microscopía de fuerza atómica, presentan limitaciones prácticas de costo y escalabilidad.

En este trabajo se desarrolló un modelo de segmentación semántica basado en redes neuronales convolucionales para identificar automáticamente escamas de grafeno en imágenes de microscopía óptica. El pipeline propuesto incluye la transformación del dataset, un preprocesamiento adaptado al desbalance de clases y el entrenamiento de una U-Net con encoder preentrenado.

Para mitigar los efectos de la escasez de datos, se incorporaron funciones de pérdida combinadas (Cross-Entropy ponderada, Dice Loss y Focal Loss). Los resultados muestran un desempeño notable, con una mIoU cercana a 0.70 y una precisión superior al 80 % en las clases bulk y few-layer, lo que confirma la viabilidad del enfoque y abre la posibilidad de extender esta metodología a otros materiales bidimensionales.

II. Introducción

El grafeno es un material bidimensional cuyas propiedades dependen directamente del número de capas presentes en la muestra. La identificación entre monocapa, pocas capas y grafeno grueso resulta crítica para aplicaciones en electrónica, energía y nanomateriales. Las técnicas tradicionales de caracterización, como la espectroscopía Raman o la microscopía de fuerza atómica (AFM), ofrecen precisión pero presentan limitaciones en costo y escalabilidad. En contraste, la microscopía óptica constituye una herramienta accesible, aunque dependiente de la pericia humana.

En este contexto, la integración de modelos de segmentación basados en deep learning emerge como una solución innovadora para la identificación automática de capas de grafeno en imágenes ópticas. El presente trabajo se orienta al entrenamiento y validación de modelos de segmentación semántica adaptados a escenarios de pocos datos, con el objetivo de alcanzar una precisión superior al 80 % en métricas globales de desempeño.

II-A. Objetivos

Como objetivo general tenemos el desarrollar y evaluar un modelo de segmentación automática de grafeno en imágenes de microscopía óptica, capaz de distinguir entre grafeno bulk, few-layer y background.

Para esto tendremos que seguir unos objetivos específicos como lo son:

- Optimizar el pipeline de preprocesamiento descrito en anteriores versiones del proyecto, buscando que proporcionen mejores resultados en el proceso de entrenamiento.
- Evaluar y comparar modelos de transfer learning y self-supervised learning adaptados al dominio de microscopía óptica, integrando métodos de calibración para cuantificar el impacto en la confiabilidad de sus predicciones.
- Desarrollar un pipeline reproducible con reportes de métricas avanzadas.

- Documentar y publicar los resultados para su posible reutilización en otros problemas de clasificación de materiales.

III. Marco Teórico

El presente marco teórico aborda la fundamentación necesaria para el desarrollo del modelo propuesto; se exploran conceptos clave relacionados con el material grafeno, las técnicas de microscopía y las distintas arquitecturas de Redes Neuronales Convolucionales (CNN) para la segmentación de imágenes.

III-A. Grafeno: Propiedades y Clasificación por Capas

El grafeno es un material bidimensional que ha capturado una atención significativa a nivel mundial desde su descubrimiento en 2004 [4]. Se define como una nanohoja bidimensional compuesta por átomos de carbono con hibridación sp2. Su interés principal radica en sus propiedades inusuales y aplicaciones prometedoras en campos como la electrónica, la espintrónica, la medicina, el almacenamiento de energía y el procesamiento de luz [1].

La caracterización del grafeno a menudo implica distinguir el número de capas que lo componen, ya que sus propiedades excepcionales disminuyen a medida que aumenta el número de capas [1]. El grafeno se puede categorizar según el número de capas (L) apiladas de la siguiente manera:

- **Grafeno de una sola capa (SLG):** Una única capa atómica. Posee propiedades mecánicas, eléctricas y térmicas excepcionales [1].
- **Grafeno de pocas capas (FLG):** Generalmente se refiere a menos de 5 capas [1].
- **Grafeno de múltiples capas (MLG):** Usualmente se define como menos de 10 capas. Un espesor de más de 10 capas es el punto donde el grafeno comienza a aproximarse al límite tridimensional y se comporta como grafito a granel [1].

La exfoliación mecánica del grafito, uno de los métodos más eficientes para obtener muestras de grafeno de alta calidad, a menudo produce una mezcla de grafeno de una sola capa, de doble capa y de múltiples capas [4].

Distinguir estas capas es crucial, ya que las propiedades del material son altamente dependientes de su espesor [1]. En las imágenes de microscopía óptica, el número de capas modifica ligeramente la oscuridad [4]. Para la identificación con Deep Learning, las capas se categorizan comúnmente como *mono* (1 capa), *few* (2–10 capas) y *thick* (10–40 capas) [3].

III-B. Microscopía Óptica para la Caracterización de Grafeno

La microscopía óptica (MO) utiliza luz visible y un sistema de lentes para generar imágenes magnificadas de muestras pequeñas. Se emplea frecuentemente para estudiar morfologías a bajas magnificaciones y es una herramienta importante para estimar el número de capas en las escamas de grafeno [1].

A diferencia de otras técnicas de caracterización:

- **Microscopía Electrónica de Barrida (SEM):** Proporciona imágenes escaneando las superficies de las muestras con un haz de electrones enfocado, que interactúan con los átomos para producir señales con información sobre la morfología de la superficie [1].
- **Microscopía de Fuerza Atómica (AFM)** y **Espectroscopia Raman:** Son métodos que también permiten la visualización y confirmación del número de capas de grafeno. Sin embargo, la AFM y la espectroscopia Raman son procesos técnicamente más elaborados [4].

La MO permite un reconocimiento simple de las láminas de grafeno y puede visualizar el óxido de grafeno en un área mucho mayor (>200 nanohojas por imagen) de lo que es posible con AFM o SEM [1]. Esto la convierte en un método de bajo costo y eficiente para determinar el número de capas. Sin embargo, la identificación precisa de grafeno de una sola capa (SLG) bajo un microscopio óptico requiere un entrenamiento continuo y un esfuerzo humano significativo.

Aquí es donde la introducción de la segmentación de imágenes basada en aprendizaje profundo tiene un gran potencial para la determinación automática de capas de grafeno, superando las limitaciones de la inspección manual y los métodos más complejos como la espectroscopia Raman y la microscopía de fuerza atómica [4].

III-C. Arquitecturas CNN Relevantes para la Segmentación de Imágenes

El desarrollo de las CNN ha llevado a una diversidad de arquitecturas especializadas en segmentación de imágenes. Algunas de las más relevantes incluyen:

- ***U-Net*:** presentada por Ronneberger, es una red en forma de Ü"de extremo a extremo. Esta red puede aprender el patrón de manera eficiente y funcionar bien con un número reducido de imágenes anotadas.
- ***U-Net++*:** basada en la arquitectura U-Net, Zhou propusieron *U-Net++* para la segmentación de imágenes médicas. Se caracteriza por sus conexiones de salto (skip connections) densas y rediseñadas, donde cada bloque entre el codificador y el decodificador aprende características semánticas de todos los bloques convolucionales anteriores, permitiendo a la red extraer diferentes niveles de información semántica.
- ***DeepLab*:** Chen presentaron *DeepLab*, una arquitectura que utiliza convoluciones atrous para ampliar el campo receptivo sin aumentar los parámetros ni reducir la resolución espacial. Esto le permite capturar información contextual a distintas escalas, mejorando la segmentación de bordes y detalles finos.
- ***Pyramid Scene Parsing Network (PSPNet)*:** Zhao propusieron *PSPNet* (*Pyramid Scene Parsing Network*), que incorpora un módulo de *Pyramid Pooling* para integrar características globales y locales. Gracias a ello, logra una segmentación más coherente en escenas complejas y con objetos de diferentes tamaños.

Durante el entrenamiento de estas arquitecturas, las imágenes se normalizan mediante la técnica de *Min-Max scaling*,

ajustando los valores de intensidad al rango [0, 1] para mejorar la estabilidad y convergencia del modelo. Además, se emplea el procesamiento en *mini-batches*, dividiendo el conjunto de datos en pequeños lotes para optimizar el uso de memoria y mantener la estabilidad del gradiente durante la optimización.

III-D. ***Optimización Bayesiana para Ajuste de Hiperparámetros***

La optimización bayesiana constituye un enfoque probabilístico para la búsqueda eficiente de hiperparámetros en modelos de aprendizaje profundo. Su objetivo es minimizar una función objetivo desconocida y costosa de evaluar (por ejemplo, la pérdida de validación) mediante la construcción iterativa de un modelo sustituto que aproxima su comportamiento. A diferencia de métodos como la búsqueda aleatoria o *grid search*, que exploran el espacio de forma exhaustiva o no informada, la optimización bayesiana equilibra explícitamente la exploración y explotación, lo que la hace particularmente adecuada en escenarios con recursos computacionales limitados [10].

Dado un espacio de hiperparámetros $\mathcal{X}$ y una función objetivo $f : \mathcal{X} \mapsto \mathbb{R}$, se modela la distribución posterior sobre f a partir de observaciones previas . A partir de este modelo, se define una función de adquisición, cuya maximización determina los puntos donde evaluar el modelo real. Este proceso iterativo permite concentrar el cómputo en regiones prometedoras, acelerando la convergencia hacia configuraciones óptimas.

Tree-structured Parzen Estimator (TPE): El método TPE es un enfoque de optimización bayesiana que modela directamente la densidad posterior condicionada a valores de la función objetivo [11]. En lugar de asumir un modelo conjunto sobre variables y resultados, el algoritmo separa los valores observados en dos subconjuntos: configuraciones de alto desempeño y de bajo desempeño, definidos mediante un umbral cuantílico. Mediante estimadores de densidad kernel, TPE construye modelos $l(x)$ y $g(x)$ para las regiones buenas y malas del espacio de hiperparámetros, respectivamente. La selección de nuevos puntos se realiza maximizando la razón $l(x)/g(x)$, lo que favorece regiones donde es más probable encontrar mejoras, manteniendo una exploración adaptativa y eficiente del espacio.

Median Pruner: Para reducir el costo computacional asociado a múltiples ejecuciones de entrenamiento, se implementó un mecanismo de *pruning* basado en el algoritmo *Median Pruner* [12]. Este método interrumpe tempranamente configuraciones cuyo desempeño intermedio es estadísticamente inferior a la mediana de ensayos previos en el mismo punto de evaluación. Si el valor actual de la métrica se encuentra por debajo de dicho umbral, el entrenamiento se detiene, permitiendo reasignar recursos a configuraciones más prometedoras. Este enfoque resulta especialmente valioso cuando el entrenamiento por configuración es prolongado y los recursos de cómputo son limitados.

IV. METODOLOGÍA

IV-A. ***Dataset***

El conjunto de datos se obtuvo desde la plataforma Roboflow bajo el formato COCO. Para facilitar el entrenamiento y la evaluación, se transformó a una estructura explícita de *imágenes–máscaras*, donde cada píxel codifica la clase correspondiente (0 = *background*, 1 = *bulk*, 2 = *few-layer*).

Durante la inspección inicial se observó que la clase monolayer no contenía ejemplos válidos, por lo que fue reasignada a la categoría background. El dataset resultante presenta un marcado desbalance de clases (84 % background, 16 % bulk y 0.3 % *few-layer*), aspecto que condicionó la selección de funciones de pérdida y estrategias de evaluación.

IV-B. ***Selección de Arquitecturas***

Se entrenaron y compararon cuatro arquitecturas de segmentación semántica: **U-Net**, **U-Net++**, **DeepLabV3+** y **PSPNet**, todas implementadas mediante la librería segmentation.models.pytorch (SMP).

Cada modelo fue probado con dos tipos de *encoders* preentrenados en ImageNet: **ResNet34** y **EfficientNet-B4**. El primero se empleó como baseline de referencia por su estabilidad y bajo costo computacional, mientras que el segundo se utilizó como alternativa más profunda y eficiente, capaz de capturar representaciones más complejas.

El objetivo de esta comparación fue evaluar el impacto de la arquitectura y del encoder en el desempeño global de la segmentación, manteniendo constantes el resto de los hiperparámetros y la configuración del entrenamiento.

Durante el entrenamiento de estas arquitecturas, las imágenes se normalizan mediante la técnica de *Min-Max scaling*, ajustando los valores de intensidad al rango [0, 1] para mejorar la estabilidad y convergencia del modelo. Además, se emplea el procesamiento en *mini-batches*, dividiendo el conjunto de datos en pequeños lotes para optimizar el uso de memoria y mantener la estabilidad del gradiente durante la optimización.

IV-C. ***Funciones de Pérdida***

IV-C0a. ***Cross-Entropy ponderada****:* La *Cross-Entropy* mide la divergencia entre la distribución de probabilidad predicha y la verdadera. En su versión ponderada, se asigna a cada clase c un peso w_c inversamente proporcional a su frecuencia:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} w_c\, y_{i,c}\, \log(p_{i,c})$$

donde:

- N es el número total de píxeles,
- C es el número de clases,
- $y_{i,c}$ es la etiqueta real (1 si el píxel i pertenece a la clase c, 0 en caso contrario),
- $p_{i,c}$ es la probabilidad predicha para la clase c en el píxel i.

IV-C0b. ***Dice Loss****:* La *Dice Loss* deriva del coeficiente de Dice, el cual mide el solapamiento entre la máscara predicha y la real. Su forma diferenciable para entrenamiento se define como:

$$\mathcal{L}_{Dice} = 1 - \frac{2\sum_{i=1}^{N} p_i y_i + \epsilon}{\sum_{i=1}^{N} p_i + \sum_{i=1}^{N} y_i + \epsilon}$$

donde ϵ es un término de estabilidad numérica (generalmente en el rango de 10^{-8} a 10^{-6}) que evita divisiones por cero. Este término favorece el aprendizaje de regiones pequeñas o desbalanceadas, al optimizar directamente la métrica de solapamiento espacial.

IV-C0c. ***Focal Loss****:* La *Focal Loss* enfatiza ejemplos difíciles, reduciendo la influencia de los píxeles correctamente clasificados con alta confianza. Se expresa como:

$$\mathcal{L}_{Focal} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}(1 - p_{i,c})^{\gamma}\, y_{i,c}\, \log(p_{i,c})$$

donde $\gamma > 0$ controla el grado de enfoque en ejemplos difíciles (usualmente $\gamma = 2$). Cuando $\gamma = 0$, la función se reduce a la Cross-Entropy tradicional.

IV-C0d. ***Combinación total de pérdidas****:* Los pesos relativos de cada componente se ajustaron empíricamente tras pruebas iniciales. La pérdida total se define como:

$$\mathcal{L}_{total} = \lambda_{CE} \cdot \mathcal{L}_{CE} + \lambda_{Dice} \cdot \mathcal{L}_{Dice} + \lambda_{Focal} \cdot \mathcal{L}_{Focal}$$

donde:

- $\mathcal{L}_{CE}$, $\mathcal{L}_{Dice}$ y $\mathcal{L}_{Focal}$ representan las pérdidas individuales de *Cross-Entropy ponderada*, *Dice Loss* y *Focal Loss*, respectivamente;
- λ_{CE}, λ_{Dice} y λ_{Focal} son sus pesos de contribución en la combinación total.

En este trabajo, los valores se fijaron en $(\lambda_{CE}, \lambda_{Dice}, \lambda_{Focal}) = (0{,}25, 0{,}5, 0{,}25)$ tras una validación inicial.

IV-D. Entrenamiento

El protocolo de entrenamiento se implementó en `PyTorch` empleando el optimizador `AdamW` con una tasa de aprendizaje inicial de 1×10^{-4} y un *weight decay* moderado para controlar la magnitud de los pesos. Se utilizó entrenamiento de precisión mixta (*mixed precision training*) para aprovechar al máximo la GPU y reducir el consumo de memoria. El tamaño de lote se fijó en 4 debido a las restricciones de memoria de la GPU. El aprendizaje se reguló mediante *early stopping* basado en la métrica *mean Intersection over Union (mIoU)* del conjunto de validación, con el objetivo de prevenir sobreajuste. Los pesos del modelo se guardaron automáticamente cuando se alcanzó el mejor valor de mIoU. Cada ejecución se repitió con diferentes semillas aleatorias para estimar la variabilidad experimental.

Durante el entrenamiento se registraron las siguientes métricas: *IoU* por clase, *mIoU*, *Dice*, *Precision*, *Recall* y *F1-score*. Dado que se trata de un problema multiclase, todas las métricas se calcularon bajo el esquema de **macropromedio**, promediando de forma equitativa el desempeño entre clases independientemente de su frecuencia. Este enfoque permite evaluar con mayor sensibilidad el rendimiento en clases minoritarias, a diferencia del micropromedio, que pondera las clases según su tamaño y tiende a favorecer el *background* en datasets desbalanceados.

IV-D0a. ***Curvas de aprendizaje****:* Se generaron las curvas de entrenamiento y validación para las métricas de *loss* y *mIoU* en función de las épocas, con el fin de diagnosticar fenómenos de sobreajuste o subentrenamiento. Estas gráficas permitieron observar la estabilidad de la convergencia y la generalización del modelo.

IV-D0b. ***Curvas de precisión-recall****:* Adicionalmente, se calcularon las *Precision–Recall Curves* por clase y su área bajo la curva (AUC–PR) para evaluar la relación entre sensibilidad y precisión en distintos umbrales de decisión. Estas curvas son especialmente útiles en escenarios desbalanceados, ya que revelan la capacidad del modelo para mantener una alta precisión sin sacrificar el recall en clases poco representadas.

IV-D0c. ***Curva de calibración****:* Como complemento, se generó una *curva de calibración* que compara la confianza media de las predicciones con la frecuencia real de aciertos. Una calibración adecuada indica que las probabilidades de salida del modelo son confiables como estimadores de incertidumbre, lo cual es deseable en aplicaciones científicas o de análisis automatizado. En conjunto, este protocolo de entrenamiento permitió comparar cuantitativamente la evolución de las distintas arquitecturas (U-Net, U-Net++, DeepLabV3+ y PSPNet) bajo las mismas condiciones de aprendizaje y evaluación.".

IV-E. Evaluación de Desempeño

El desempeño de los modelos de segmentación se evaluó mediante un conjunto robusto de métricas cuantitativas y análisis gráficos, priorizando el rendimiento en clases minoritarias.

Métricas Cuantitativas y Estrategia de Promedio: Las métricas primarias utilizadas incluyen: el **mIoU** (Mean Intersection over Union) global, **IoU por clase**, **Dice Coeficient**, **Precision**, **Recall** y **F1-score**. Debido al desbalance inherente en el dataset (materiales 2D versus *background*), todas las métricas de resumen se reportaron utilizando el **Macropromedio** (*Macro-Averaging*). Esta estrategia asegura que el rendimiento en las clases de grafeno minoritarias sea ponderado de manera equitativa.

Análisis Gráfico y Confiabilidad: Los resultados se complementaron con el siguiente análisis:

- **Curvas de Aprendizaje:** Se monitorearon las curvas de *loss*, *mIoU* y *recall* (*Macro-Averaging*) en los conjuntos de entrenamiento y validación para diagnosticar la convergencia y el sobreajuste.
- **Matrices de Confusión:** Se generaron matrices normalizadas por fila para detallar el rendimiento píxel a píxel entre las clases (True Positive Rate).
- **Curva Precision vs. Recall (Curva PR):** Se calculó el promedio **MACRO** de esta curva para visualizar el *trade-*

off entre la capacidad de detección (*Recall*) y la exactitud (*Precision*), en particular para las clases desbalanceadas.

- **Curva de Calibración de Modelos (*Reliability Diagram*):** Este gráfico se utilizó para evaluar la **confiabilidad** de las probabilidades de salida del modelo. Esto valida la efectividad del *Temperature Scaling* implementado en la función de pérdida para corregir la sobreconfianza.

Finalmente, se realizó una inspección visual cualitativa de las segmentaciones sobre el conjunto de validación, enfocándose en la coherencia espacial y la precisión de los contornos del material.

IV-F. Optimización de Hiperparámetros

Con el fin de mejorar el desempeño del modelo y maximizar la eficiencia computacional, se llevó a cabo un proceso sistemático de optimización de hiperparámetros. Inicialmente, se definió un espacio de búsqueda amplio compuesto por nueve hiperparámetros relevantes, abarcando funciones de activación, tasas de aprendizaje, parámetros de regularización, configuraciones de *dropout* y componentes estructurales del modelo.

La primera fase consistió en una búsqueda exploratoria empleando *Tree-structured Parzen Estimator (TPE)* como estrategia bayesiana de muestreo, combinada con el mecanismo *Median Pruner* para detener tempranamente configuraciones con desempeño deficiente. Este procedimiento permitió evaluar un conjunto significativo de configuraciones manteniendo el costo computacional dentro de límites razonables.

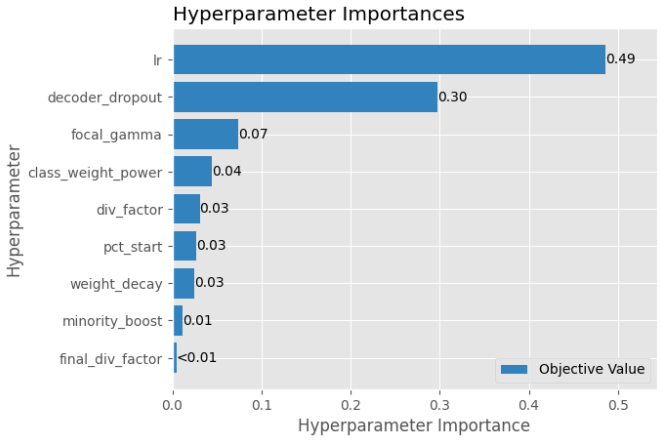

Figura 1. Importancia relativa de los hiperparámetros evaluados mediante Optuna.

Al concluir esta etapa, se generó una gráfica de importancia de hiperparámetros (Figura 1), la cual reveló la contribución relativa de cada parámetro al desempeño del modelo. A partir de este análisis, se seleccionaron los tres hiperparámetros más influyentes, mientras que los restantes se fijaron con valores constantes derivados de las mejores configuraciones previas. Esta estrategia redujo el espacio de búsqueda de nueve a tres dimensiones, refinando el proceso de optimización y concentrando la exploración en las variables de mayor impacto.

Este enfoque permitió no solo mejorar la eficiencia del entrenamiento, sino también obtener un conjunto de hiperparámetros más estable y consistente con las limitaciones computacionales del entorno experimental.

V. Resultados y Análisis

Los resultados obtenidos en la evaluación comparativa de las distintas arquitecturas (U-Net, U-Net++, DeepLabV3+, PSPNet) y encoders (ResNet34, EfficientNet-B4 y EfficientNet-B5) evidencian un desempeño notable y variado en la tarea de segmentación de grafeno. A continuación, se presentan las Curvas de Aprendizaje (Loss, mIoU, Recall) para analizar la estabilidad de la convergencia, la Curva Precision-Recall, y la Curva de Calibración para validar la confiabilidad de las probabilidades de cada modelo y la Matriz de Confusión para detallar el rendimiento por clase.

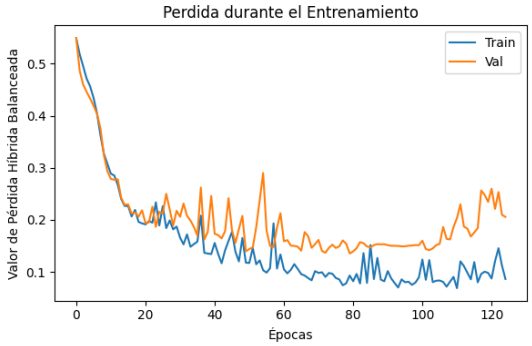

Figura 2. Curva de Aprendizaje: Pérdida (Loss Total) frente a Época para el Modelo U-Net con Encoder Efficientnet-B5.

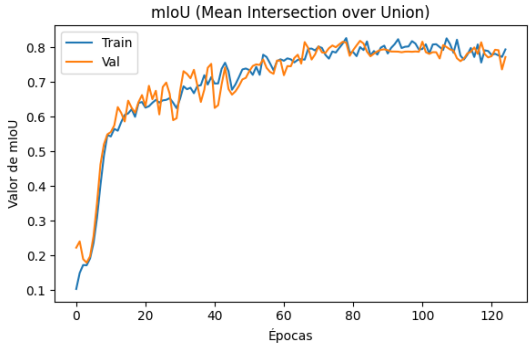

Figura 3. Curva de Aprendizaje: mIoU (Macropromedio) frente a Época para el Modelo U-Net con Encoder Efficientnet-B5.

V-A. Configuración de Entrenamiento

Todos los modelos se entrenaron bajo un conjunto uniforme de hiperparámetros para garantizar la comparabilidad de los resultados. El entrenamiento se ejecutó durante 120 épocas (`EPOCHS = 120`) para asegurar la convergencia completa, dado el uso de una función de pérdida híbrida. La tasa de aprendizaje inicial (LR) fue de $6e{-}4$, utilizando el optimizador Adam con un programador de tasa de aprendizaje por meseta (*plateau scheduler*).

Cuadro I
COMPARACIÓN CUANTITATIVA DE DESEMPEÑO DE MODELOS DE SEGMENTACIÓN DE GRAFENO EN EL CONJUNTO DE VALIDACIÓN

Arquitectura	Encoder	mIoU (Macro)	Recall (Macro)	Recall (*few-layer*)	Recall (*bulk*)	Pérdida (Val.)
U-Net	ResNet34	0.747	**0.94**	**91.80 %**	87.10 %	0.185
U-Net	EfficientNet-B4	0.755	0.88	76.43 %	92.33 %	0.199
U-Net	**EfficientNet-B5**	**0.819**	**0.94**	90.58 %	92.99 %	0.157
U-Net++	ResNet34	0.758	0.92	63.71 %	**94.34 %**	0.185
U-Net++	EfficientNet-B4	0.758	0.93	89.45 %	92.49 %	**0.146**
DeepLabV3+	ResNet34	0.769	0.91	63.71 %	**94.34 %**	0.208
DeepLabV3+	EfficientNet-B4	0.765	0.84	64.13 %	91.81 %	0.215
PSPNet	ResNet34	0.778	0.89	67.64 %	92.43 %	0.249
PSPNet	EfficientNet-B4	0.727	0.88	83.21 %	90.26 %	0.250

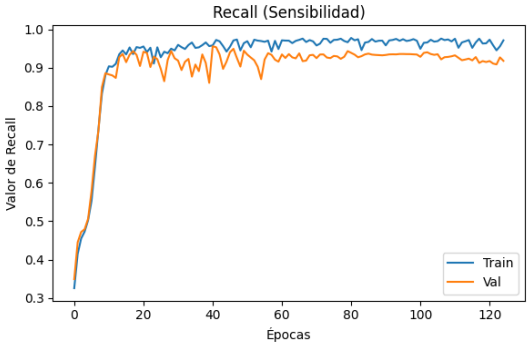

Figura 4. Curva de Aprendizaje: Recall (Macropromedio) frente a Época para el Modelo U-Net con Encoder Efficientnet-B5.

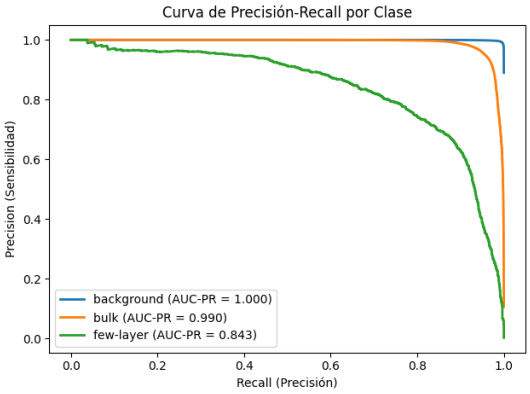

Figura 5. Curva Precision-Recall (PR) por Clase para el Modelo U-Net con Encoder Efficientnet-B5.

V-B. Análisis de las Curvas de Convergencia

Las Figuras 2, 3 y 4 muestran las curvas de aprendizaje para el modelo seleccionado, *U-Net / EfficientNet-B5*, considerado el mejor desempeño general dentro de la evaluación comparativa.

V-B1. ***Estabilidad y Pérdida:*** La curva de pérdida (*Loss*) en la Figura 2 exhibe una rápida disminución inicial seguida de una convergencia estable. El modelo *U-Net / EfficientNet-B5* alcanzó una de las pérdidas de validación más bajas del conjunto comparado (0.157) entre las variantes de U-Net (Tabla I), lo que refleja una alta confianza en las predicciones

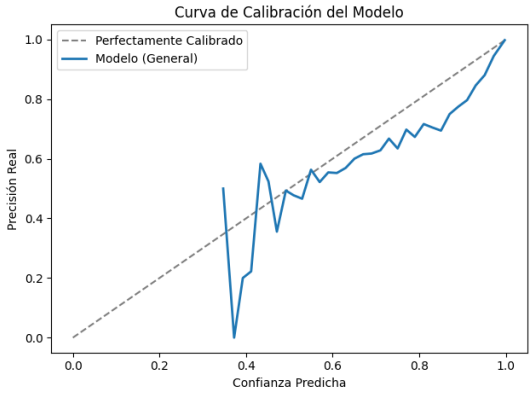

Figura 6. Curva de Calibración (*Reliability Diagram*) para el Modelo U-Net con Encoder Efficientnet-B5.

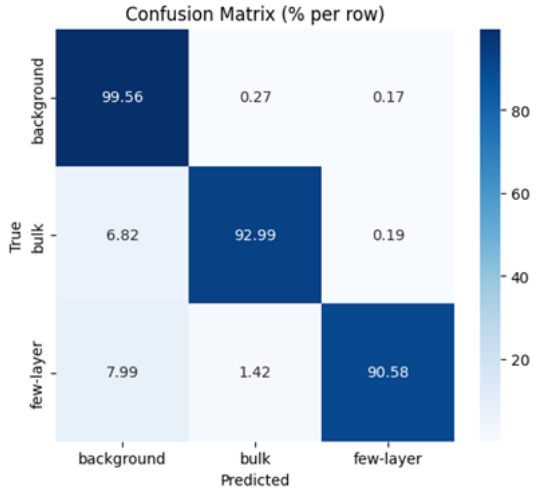

Figura 7. Matriz de Confusión Normalizada para U-Net con Encoder Efficientnet-B5 (ejemplo de desempeño por clase).

y una adecuada generalización.

V-B2. Generalización de mIoU y Recall: La Figura 3 demuestra que el *mIoU* de validación sigue de cerca al de entrenamiento, evidenciando una adecuada generalización y ausencia de sobreajuste. En términos de sensibilidad, el modelo mantiene un Recall (Macro) de 0.94, lo que confirma su capacidad de detección equilibrada en todas las clases, especialmente en la clase crítica *few-layer*.

V-C. Resultados tras la optimización de hiperparámetros

Las figuras 8, 9 y 10 presentan la evolución del *loss*, *mIoU* y *recall* tras la optimización bayesiana de hiperparámetros mediante Optuna [12]. A diferencia del entrenamiento inicial, el modelo logró una convergencia más estable y progresiva, evitando oscilaciones marcadas y episodios de estancamiento prematuro.

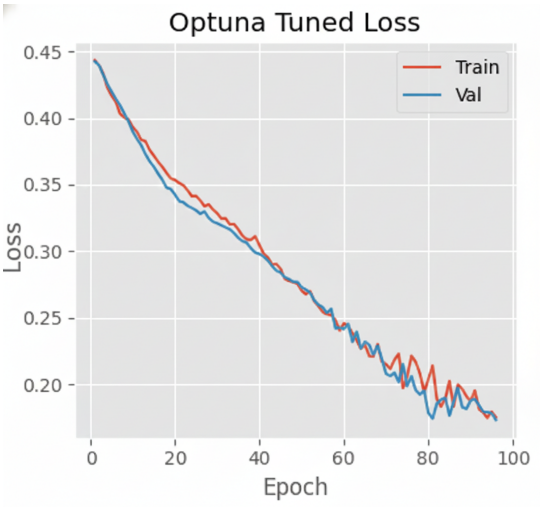

Figura 8. Curva de Loss tras la optimización bayesiana.

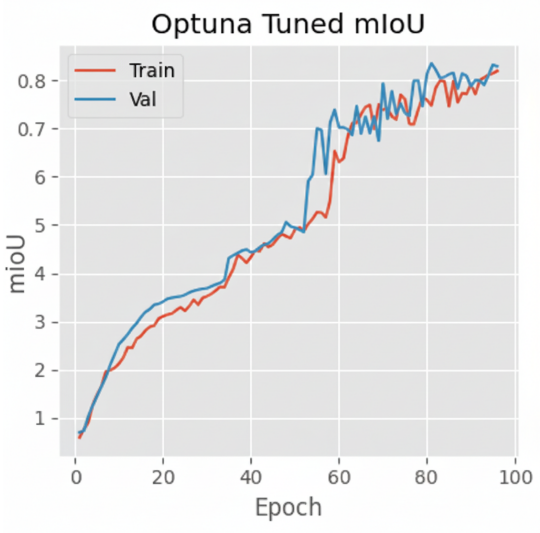

Figura 9. Curva de mIoU tras la optimización bayesiana.

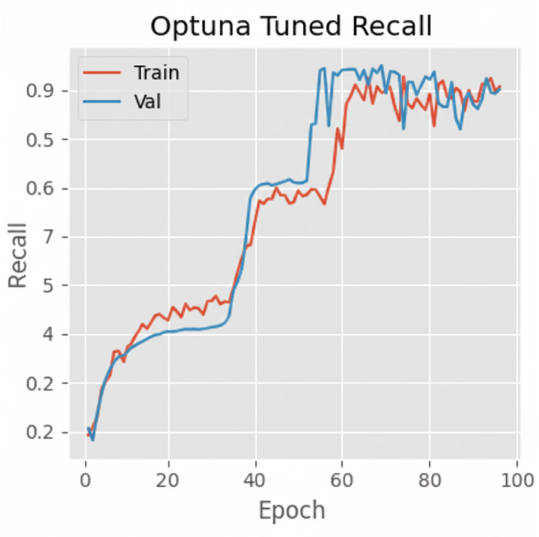

Figura 10. Curva de Recall tras la optimización bayesiana.

Se observa una disminución continua del *loss* a lo largo de las 100 épocas, sin indicios significativos de sobreajuste. Adicionalmente, el *mIoU* alcanzó valores superiores a 0.80 con una menor brecha entre entrenamiento y validación, lo que evidencia una generalización más sólida y una dinámica de aprendizaje menos volátil respecto a la fase previa.

El *recall* mostró un incremento sostenido y se mantuvo estable por encima de 0.90 tras la convergencia, demostrando una mayor capacidad del modelo para recuperar correctamente las regiones segmentadas, incluso bajo un escenario de fuerte desbalance de clases.

En conjunto, estos resultados demuestran que la optimización permitió alcanzar un entrenamiento más eficiente y controlado, obteniendo curvas más suaves, un mejor rendimiento global y mayor robustez frente al ruido del conjunto de validación en la tarea de segmentación de grafeno.

V-D. Desempeño Cuantitativo por Arquitectura

La Tabla I resume los resultados finales, destacando el equilibrio entre la Precisión Global (mIoU) y la Detección de la Clase Crítica (*few-layer*).

- **Líderes en Precisión (mIoU):** El modelo *U-Net / EfficientNet-B5* obtuvo el mayor mIoU (0.819), superando a las variantes con ResNet34 y EfficientNet-B4, mientras mantenía una pérdida de validación reducida.
- **Equilibrio en Detección:** A diferencia del modelo *U-Net / ResNet34*, que priorizó la clase *few-layer* a costa de la precisión general, la versión con EfficientNet-B5 logró un balance superior con Recall (*few-layer*) = 90.58 % y Recall (*bulk*) = 92.99 %.

V-E. Selección del Modelo Óptimo: U-Net/EfficientNet-B5

Se selecciona la arquitectura *U-Net / EfficientNet-B5* como el modelo óptimo por su mayor precisión global y estabilidad de entrenamiento.

Este modelo alcanzó un mIoU de 0.819 y un Recall (Macro) de 0.94, con una pérdida de validación mínima (0.157). Además, mantuvo un excelente desempeño en las clases críticas, con Recall (*few-layer*) = 90.58 % y Recall (*bulk*) = 92.99 %, demostrando una detección robusta y equilibrada.

La combinación del decodificador clásico de U-Net y el potente *encoder* EfficientNet-B5 permitió capturar información contextual multiescala con alta eficiencia, optimizando la sensibilidad sin sacrificar la precisión espacial.

V-F. *Análisis de Calibración y Precision-Recall*

Las Figuras 5 (Curva Precision-Recall) y 6 (Curva de Calibración) validan el comportamiento probabilístico del modelo seleccionado. La curva PR refleja un excelente equilibrio entre precisión y sensibilidad, mientras que la curva de calibración evidencia una ligera **sobreconfianza** en el rango medio de probabilidad, un patrón común en arquitecturas de segmentación profunda. Este comportamiento sugiere la conveniencia de aplicar *Temperature Scaling* como calibración posterior.

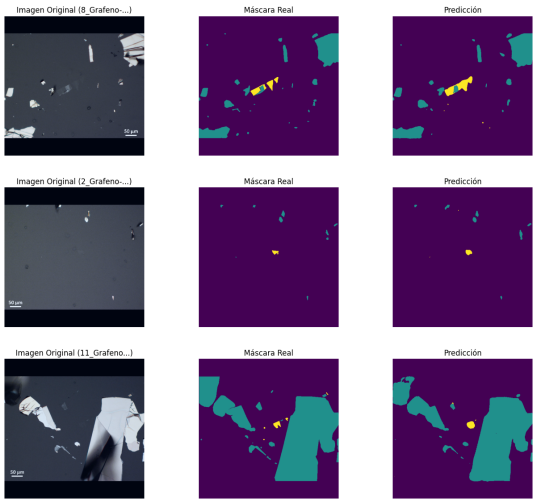

Figura 11. Ejemplo de predicción: imagen original, máscara predicha y superposición (*overlay*).

La comparación visual (Figura 11) confirma que el modelo identifica con alta precisión las regiones *bulk* y mantiene una detección sólida de las zonas *few-layer*, aunque estas últimas pueden ser ocasionalmente subestimadas debido a su bajo contraste.

V-G. *Análisis Estadístico del Entrenamiento*

Para cuantificar la estabilidad del entrenamiento del modelo *U-Net/EfficientNet-B5*, se calcularon las medias y desviaciones estándar de las métricas principales (Pérdida, mIoU y Recall) a lo largo de las épocas de entrenamiento y validación. Los resultados se presentan en la Tabla II.

Los valores presentados en la Tabla II reflejan la fase de convergencia estable del modelo. Es importante destacar que, para este análisis, los estadísticos (media y desviación estándar) fueron calculados a partir de la época 20, ya que el comportamiento del modelo en las épocas iniciales no fue relevante para evaluar la estabilidad. La similitud en las medias de mIoU y Recall entre ambos conjuntos evidencia una adecuada generalización, mientras que la baja desviación estándar y la ligera diferencia en la pérdida indican una regularización efectiva sin sobreajuste.

Cuadro II
ESTADÍSTICOS DESCRIPTIVOS DEL ENTRENAMIENTO Y VALIDACIÓN DEL MODELO U-NET/EFFICIENTNET-B5.

Métrica	Conjunto	Media	Desv. Estándar
Loss	Entrenamiento	0.1179	0.0390
	Validación	0.1780	0.0352
mIoU	Entrenamiento	0.7559	0.0544
	Validación	0.7519	0.0546
Recall	Entrenamiento	0.9635	0.0117
	Validación	0.9249	0.0164

El código del proyecto y los datos de entrenamiento se encuentran disponibles en nuestro repositorio de GitHub: https://github.com/leonaidasup/PI1_unet-graphene-segmentation.git

VI. CONCLUSIONES

El análisis comparativo confirmó la superioridad del aprendizaje profundo para la segmentación de escamas de grafeno en microscopía óptica. La arquitectura *U-Net / EfficientNet-B5* demostró el mejor desempeño general, con la mayor precisión global y una detección equilibrada entre clases.

VI-A. *Validación de la Estrategia de Pérdida*

La combinación de pérdidas ponderadas (Cross-Entropy, Dice y Focal) resultó efectiva para mitigar el desbalance de clases, alcanzando un Recall (Macro) superior a 0.94. Esto confirma la efectividad del enfoque para priorizar la detección sin comprometer la estabilidad del entrenamiento.

VI-B. *El Trade-off entre Precisión Global y Detección Crítica*

El estudio evidenció un compromiso entre la precisión global y la sensibilidad hacia la clase minoritaria:

- Los modelos con mIoU elevado (por ejemplo, PSPNet/ResNet34) sacrificaron la detección de *few-layer*.
- El modelo *U-Net / EfficientNet-B5* logró el mejor equilibrio, manteniendo alta precisión global y una sensibilidad competitiva en las clases críticas.

VI-C. *Selección Final del Modelo Óptimo*

Se concluye que la arquitectura *U-Net / EfficientNet-B5* es la opción más robusta y equilibrada para la tarea de segmentación de grafeno, con el mIoU más alto (0.819), Recall (Macro) = 0.94 y una pérdida de validación mínima (0.157).

VI-D. *Limitaciones y Trabajo Futuro*

Pese a su alto rendimiento, la calibración de probabilidad muestra un leve sesgo de sobreconfianza. Por ello, los futuros trabajos deberían:

1. Implementar *Temperature Scaling* para mejorar la calibración.

2. Aumentar la diversidad del conjunto de datos para mejorar la generalización.
3. Explorar *fine-tuning* del encoder EfficientNet-B5 en imágenes de microscopía para refinar la extracción de características específicas del grafeno.

REFERENCIAS

[1] V. Kumar, A. Kumar, D.-J. Lee, y S.-S. Park, "Estimation of Number of Graphene Layers Using Different Methods: A Focused Review," *Materials*, vol. 14, núm. 16, p. 4590, 2021.

[2] X. Zhao et al., "A review of convolutional neural networks in computer vision," *Artif. Intell. Rev.*, vol. 57, núm. 99, 2024.

[3] S. Masubuchi et al., "Deep-learning-based image segmentation integrated with optical microscopy for automatically searching for two-dimensional materials," *npj 2D Materials and Applications*, vol. 4, núm. 3, 2020.

[4] B. Yang, M. Wu, y W. Teizer, "Modified UNet++ with attention gate for graphene identification by optical microscopy," *Carbon*, vol. 195, pp. 246–252, 2022.

[5] G. Jiang, "Few-shot Learning using Data Augmentation: A Literature Review," *Dean&Francis*, 2024.

[6] O. Ronneberger, P. Fischer, y T. Brox, "U-net, Convolutional Networks for Biomedical Image Segmentation," *arXiv:1505.04597*, May, 2015.

[7] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, y J. Liang, "Unet++: A Nested U-Net Architecture for Medical Image Segmentation," *arXiv:1807.10165*, Jul., 2018.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, y A. L. Yuille, "DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, núm. 4, pp. 834–848, 2018.

[9] K. He, G. Gkioxari, P. Dollár, y R. Girshick, "Mask R-CNN," en *Proc. IEEE Int. Conf. Computer Vision*, 2017, pp. 2961–2969.

[10] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, y N. De Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE*, vol. 104, núm. 1, pp. 148–175, 2016.

[11] J. Bergstra, R. Bardenet, Y. Bengio, y B. Kégl, "Algorithms for hyper-parameter optimization," en *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[12] T. Akiba, S. Sano, T. Yanase, T. Ohta, y M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," en *Proc. 25th ACM SIGKDD Intl. Conf. Knowledge Discovery & Data Mining (KDD)*, 2019, pp. 2623–2631.