

Redes Neuronales para el Reconocimiento de Voz a Partir de Espectrogramas

1rd Luis Benavides Villegas

Escuela de Ingeniería en Computación

Instituto Tecnológico de Costa Rica

Cartago, Costa Rica

lubenavides@estudiantec.cr

2st Juan Jiménez Valverde

Escuela de Ingeniería en Computación

Instituto Tecnológico de Costa Rica

Cartago, Costa Rica

juand0908@estudiantec.cr

3nd Alex Naranjo Masis

Escuela de Ingeniería en Computación

Instituto Tecnológico de Costa Rica

Cartago, Costa Rica

alnaranjo@estudiantec.cr

Abstract—Automatic sound recognition has gained increasing relevance in applications such as robotics, hearing assistance systems, and environmental monitoring. This work presents a convolutional neural network (CNN) approach for multi-class classification of short audio clips through their log-Mel spectrogram representations. Two architectures were implemented: a modified version of the classical LeNet-5 and a customized adaptation of ResNet-18, both trained on the ESC-50 dataset, which contains 2000 labeled recordings across 50 sound classes. Additionally, the SpecAugment technique was applied to increase the diversity of the training set through random masking in the time and frequency domains. All models were trained and evaluated under multiple hyperparameter configurations, with metrics logged using the Weights & Biases platform. The obtained results demonstrate the effectiveness of CNNs in recognizing spectral-temporal audio patterns, showing that ResNet-18 achieves higher generalization and training stability.

Index Terms—Audio Classification, Spectrogram, Convolutional Neural Networks, SpecAugment, ESC-50, Deep Learning.

I. INTRODUCCIÓN

El reconocimiento automático de sonidos constituye un área fundamental dentro del campo de la inteligencia artificial y el aprendizaje profundo. Este problema se relaciona con la capacidad de los sistemas computacionales para identificar eventos acústicos en señales de audio, permitiendo aplicaciones en detección de ambientes, interfaces hombre-máquina, vigilancia acústica y sistemas inteligentes de asistencia. Tradicionalmente, las técnicas de procesamiento de audio se han basado en descriptores manuales como MFCCs o energía de banda, los cuales, aunque efectivos en escenarios controlados, presentan limitaciones para capturar patrones temporales y frecuenciales complejos en entornos ruidosos o no estacionarios.

Diversos autores han demostrado que las redes neuronales convolucionales (CNN) pueden superar estas limitaciones al operar directamente sobre representaciones espectrales del sonido, extrayendo jerarquías de características relevantes sin intervención manual. LeCun et al. [1] introdujeron la arquitectura LeNet-5, que sentó las bases para el aprendizaje jerárquico mediante convoluciones y *pooling*. Posteriormente, He et al. [2] propusieron las redes residuales (ResNet), que solventan el problema del *vanishing gradient* en redes profundas mediante conexiones de identidad, permitiendo entrenar modelos más estables y precisos. En el ámbito específico del

audio, Piczak [3] presentó el dataset ESC-50, ampliamente utilizado para la clasificación de sonidos ambientales, mientras que Park et al. [4] desarrollaron *SpecAugment*, una técnica de aumento de datos en el dominio espectral que mejora la capacidad de generalización de los modelos al enmascarar regiones de tiempo y frecuencia de manera aleatoria.

Basándose en estos avances, el presente trabajo propone un sistema de clasificación de sonidos ambientales a partir de sus espectrogramas log-Mel utilizando dos arquitecturas contrastantes: una versión adaptada de LeNet-5, optimizada para entradas RGB de 224×224 píxeles, y una implementación personalizada de ResNet-18 con regularización y *weight decay*. Se emplea además *SpecAugment* como técnica de *data augmentation*, evaluando su impacto en la convergencia y desempeño de ambos modelos.

La principal contribución de este estudio radica en la comparación controlada de ambas arquitecturas bajo condiciones reproducibles, cuantificando el efecto de la aumentación espectral y los hiperparámetros sobre métricas clave como *accuracy*, *precision*, *recall*, *F1-score* y *AUC*. Los resultados permiten analizar la relación entre profundidad de la red, capacidad de generalización y robustez frente al sobreajuste.

Finalmente, el documento se estructura de la siguiente manera: la Sección II describe la metodología experimental y el flujo general del proyecto; la Sección III detalla las características del dataset ESC-50 y el proceso de generación de espectrogramas; las Secciones IV y V presentan las arquitecturas LeNet-5 y ResNet-18 respectivamente; la Sección VI aborda los entrenamientos y la configuración de hiperparámetros; la Sección VII analiza los resultados comparativos de ambos modelos; y la Sección VIII expone las conclusiones generales del estudio.

II. METODOLOGÍA

La metodología seguida en este trabajo se estructuró en cuatro etapas principales: adquisición y preparación de datos, generación de espectrogramas, diseño e implementación de modelos convolucionales, y evaluación experimental mediante métricas cuantitativas y seguimiento en línea de los entrenamientos.

A. Adquisición y preparación de datos

El conjunto de datos empleado fue *ESC-50*, recopilado por Piczak [3], el cual contiene 2000 grabaciones de audio en formato .wav, cada una de cinco segundos de duración y categorizada en una de cincuenta clases balanceadas. Este dataset fue descargado desde su repositorio público y utilizado como base para el entrenamiento y validación de los modelos.

Cada archivo de audio se convirtió en una representación visual mediante la obtención de su espectrograma log-Mel, una transformación que proyecta la energía de la señal sobre el dominio tiempo-frecuencia utilizando una escala perceptualmente alineada con el oído humano. Para reducir el costo computacional, las señales fueron remuestreadas a 22 050 Hz y transformadas mediante la *Short-Time Fourier Transform* (STFT) con una ventana de 2048 muestras y salto de 512. Los espectrogramas resultantes se escalaron al rango [0,255] y se mapearon al espacio RGB con el esquema de color *magma*, produciendo imágenes de 224×224 píxeles listas para ser utilizadas por modelos de visión.

Posteriormente, se aplicó la técnica *SpecAugment* [4] para incrementar la diversidad del conjunto de entrenamiento. Esta técnica introduce enmascaramientos aleatorios en los ejes de tiempo y frecuencia, forzando al modelo a aprender representaciones más invariantes y robustas a la occlusión de informaciónpectral. Se generaron tres versiones aumentadas por cada muestra original, manteniendo los conjuntos de validación y prueba sin alteraciones para asegurar evaluaciones justas.

Finalmente, los datos fueron divididos de manera estratificada siguiendo una proporción de 60% para entrenamiento, 20% para validación y 20% para prueba. Esta división garantizó la representación uniforme de todas las clases en cada subconjunto y evitó solapamientos entre ellos.

B. Diseño e implementación de los modelos

Siguiendo las especificaciones del proyecto, se desarrollaron dos modelos desde cero utilizando la biblioteca PyTorch, limitando el uso de abstracciones de alto nivel para preservar el control sobre la arquitectura y la reproducibilidad de los experimentos.

El **Modelo A** se basó en la arquitectura clásica *LeNet-5* [1], adaptada para la clasificación de espectrogramas a color. Se introdujeron modificaciones específicas en el número de canales, normalización por lotes (*Batch Normalization*) y regularización mediante *Dropout*, con el fin de mejorar la estabilidad y capacidad de generalización. Por su parte, el **Modelo B** correspondió a una adaptación de *ResNet-18* [2], una arquitectura moderna con conexiones residuales que permiten un flujo más eficiente del gradiente en redes profundas. Ambas redes fueron entrenadas utilizando la función de pérdida *CrossEntropyLoss* y el optimizador *AdamW*, incorporando un esquema de reducción de tasa de aprendizaje (*StepLR*) y detención temprana (*early stopping*) para prevenir el sobreajuste.

C. Entrenamiento y registro de resultados

Cada modelo fue entrenado tanto con el conjunto original como con la versión aumentada, realizando múltiples ejecuciones con distintas combinaciones de hiperparámetros. Se exploraron variaciones en el tamaño de lote, tasa de aprendizaje, *weight decay* y probabilidad de *dropout*, manteniendo una semilla global para asegurar la reproducibilidad.

Durante el proceso, las métricas de pérdida, precisión, *F1-score*, *recall* y *AUC* fueron registradas automáticamente mediante la plataforma *Weights & Biases*, que permitió visualizar la evolución de los entrenamientos y detectar comportamientos de sobreajuste o subentrenamiento en tiempo real. Al finalizar, se seleccionaron los mejores modelos para cada configuración y se evaluaron sobre el conjunto de prueba, generando sus respectivas matrices de confusión y reportes comparativos.

Esta metodología permitió desarrollar un flujo experimental completo, desde la obtención de los datos hasta la comparación objetiva de resultados, garantizando trazabilidad, reproducibilidad y validez científica en el análisis final.

III. DATASET: ESC-50

El conjunto de datos utilizado en este proyecto es *ESC-50*, un *dataset* público ampliamente empleado para la clasificación de sonidos ambientales. Fue recopilado y curado por Karol J. Piczak [3], y contiene un total de 2000 grabaciones de audio en formato .wav, cada una con una duración de cinco segundos y frecuencia de muestreo de 44.1 kHz en un solo canal (*mono*).

Cada muestra de audio se encuentra etiquetada con una de las 50 clases de sonido distribuidas equitativamente, abarcando cinco categorías principales: sonidos de animales, sonidos naturales y ambientales, sonidos humanos no verbales, sonidos domésticos y sonidos de exterior o maquinaria.

Además, incluye un archivo de metadatos que, para cada registro, incluye campos como *filename*, *fold* y *category*, que facilitan la organización y división de los datos en subconjuntos de entrenamiento, validación y prueba.

A. Generación de Espectrogramas

Para permitir el uso de arquitecturas convolucionales de visión, cada archivo de audio fue transformado en una imagen mediante la extracción de su *espectrograma log-Mel*. Este procedimiento convierte la energía de la señal de audio en una representación bidimensional de tiempo y frecuencia, donde el eje horizontal corresponde a los instantes temporales y el eje vertical a las bandas de frecuencia Mel, una escala perceptualmente más cercana a la audición humana. Para lograr esto, se siguieron los siguientes pasos:

- 1) Cada audio se re-muestra a 22 050 Hz para reducir el volumen de datos sin pérdida perceptible de información. Según el teorema de Nyquist, esta frecuencia permite representar señales de hasta 11 kHz, suficiente para el rango auditivo útil en sonidos ambientales.
- 2) Se calcula el espectrograma Mel aplicando la Transformada Rápida de Fourier de Ventana Corta (STFT), que analiza la energía en pequeños intervalos de tiempo. Se usan los parámetros $n_{fft} = 2048$, $hop_length = 512$

- y $n_{mels} = 128$ para equilibrar la resolución temporal y frecuencial, generando una matriz que resume la distribución energética de la señal.
- 3) La energía se transforma al dominio logarítmico (decibelios), resaltando diferencias perceptibles y atenuando picos de alta intensidad.
 - 4) Se normalizan los valores al rango [0, 255], se invierte el eje vertical para ubicar las frecuencias graves abajo y se aplica el mapa de color magma para obtener imágenes RGB compatibles con modelos de visión.
 - 5) Las imágenes se redimensionan a 224×224 píxeles mediante interpolación bicúbica. Aunque esta reducción puede suavizar detalles finos, conserva la estructura global de energía y patrones acústicos, manteniendo la información relevante para la clasificación.

Este proceso genera representaciones visuales compactas y normalizadas que capturan la estructura espectro-temporal del sonido, listas para su uso en modelos convolucionales.

B. Data Augmentation de Espectrogramas

Con el objetivo de incrementar la diversidad del conjunto de entrenamiento y mejorar la capacidad de generalización de los modelos, se implementó la técnica *SpecAugment* [4] aplicada sobre los espectrogramas generados a partir del conjunto ESC-50. Esta técnica introduce perturbaciones sintéticas en el dominio espectral mediante el enmascaramiento aleatorio de regiones en los ejes de tiempo y frecuencia, preservando la coherencia global de la estructura espectro-temporal del sonido.

Para cada espectrograma original, se generaron tres versiones aumentadas independientes, aplicando dos máscaras de frecuencia y dos máscaras temporales, con posiciones y tamaños seleccionados aleatoriamente. Estas modificaciones eliminan de forma controlada fragmentos de información, lo que obliga al modelo a aprender representaciones más invariantes ante oclusiones o variaciones en la señal original.

Adicionalmente, se realizaron pruebas con la variante que incluye *time warping*, como se describe en la literatura original de *SpecAugment*. Sin embargo, bajo las mismas condiciones de entrenamiento y evaluación, los modelos presentaron un desempeño inferior respecto a la versión que aplica únicamente enmascaramientos de tiempo y frecuencia. Por este motivo, el conjunto final de entrenamiento se construyó exclusivamente con la versión basada en máscaras.

La Fig. 1 ilustra ejemplos de las técnicas aplicadas en el mismo espectograma.

C. División Estratificada y Normalización

El dataset ESC-50 ya se encuentra normalizado y libre de valores atípicos, pues todas las grabaciones fueron preprocesadas por su autor original; cada clip tiene igual duración, rango de amplitud coherente y calidad homogénea. Por tanto, no fue necesario realizar normalizaciones adicionales sobre las señales de audio ni sobre las imágenes generadas.

La división de los datos se realizó de manera estratificada por los folds originales del dataset, manteniendo la distribución

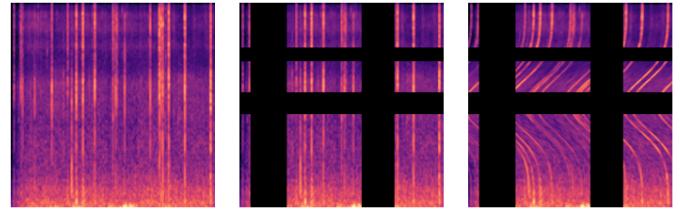


Fig. 1. Comparativa visual de técnicas de aumento: (izquierda) espectrograma original, (centro) versión aumentada con máscaras de tiempo y frecuencia, (derecha) versión con *time warping* adicional.

uniforme de clases en cada subconjunto. Se emplearon los folds 1, 2 y 3 para entrenamiento, el fold 4 para validación, y el fold 5 para prueba. Esto asegura que no exista solapamiento entre los datos y que la proporción de clases se mantenga constante.

Cabe destacar que la técnica de *data augmentation* se aplicó únicamente al conjunto de entrenamiento, dejando intactos los conjuntos de validación y prueba. Esto evita introducir sesgos artificiales en las evaluaciones y garantiza una comparación justa entre los modelos.

Finalmente, durante la carga de datos para el entrenamiento, las imágenes fueron convertidas a tensores y normalizadas canal a canal mediante el promedio y desviación estándar ($\mu = 0.5$, $\sigma = 0.5$), centrando sus valores en el rango $[-1, 1]$. Esta normalización facilita la convergencia del entrenamiento y mantiene la coherencia con las prácticas estándar de inicialización en redes convolucionales.

La Tabla I resume la distribución de clases y tamaños de los subconjuntos antes y después de la augmentación:

Tabla I
DISTRIBUCIÓN DE CLASES Y MUESTRAS EN LOS DISTINTOS SUBCONJUNTOS DEL DATASET ESC-50.

Conjunto	Total de muestras	Muestras por clase
Dataset completo	2000	40
Training set (Original)	1200	24
Training set (Aumentado)	3600	72
Validation set	400	8
Testing set	400	8

Esta división garantiza un balance perfecto de clases en todos los subconjuntos, condición esencial para evaluar correctamente la capacidad de generalización de los modelos convolucionales propuestos.

IV. MODELO A: LENET-5

El **Modelo A** se basa en la arquitectura clásica *LeNet-5*, propuesta por LeCun et al. [1], adaptada para la clasificación de espectrogramas log-Mel de tamaño 224×224 y tres canales RGB. Dicha red fue originalmente concebida para la clasificación de dígitos manuscritos del dataset MNIST, cuyas imágenes son en escala de grises de 32×32 píxeles. Por tanto, se realizaron diversas modificaciones estructurales y funcionales con el fin de adecuar su profundidad, número de parámetros y estabilidad de entrenamiento a las características del conjunto ESC-50.

La Tabla II resume las diferencias principales entre la arquitectura original y la versión implementada en este trabajo.

Tabla II
COMPARACIÓN RESUMIDA ENTRE LA ARQUITECTURA ORIGINAL DE LE NET-5 Y LA VERSIÓN MODIFICADA.

Capa	LeNet-5 original	Versión modificada
Entrada	$32 \times 32 \times 1$	$224 \times 224 \times 3$
Conv1	Conv2d(1, 6, 5, 1, 0)	Conv2d(3, 6, 5, 1, 2)
BatchNorm1	–	BatchNorm2d(6)
AvgPool1	AvgPool2d(2, 2)	AvgPool2d(2, 2)
Conv2	Conv2d(6, 16, 5, 1, 0)	Conv2d(6, 16, 5, 1, 2)
BatchNorm2	–	BatchNorm2d(16)
AvgPool2	AvgPool2d(2, 2)	AvgPool2d(2, 2)
Conv3	Conv2d(16, 120, 5, 1, 0)	Conv2d(16, 120, 5, 1, 2)
BatchNorm3	–	BatchNorm2d(120)
AvgPool3	–	AvgPool2d(2, 2)
FC1	Linear(120, 84)	Linear(<i>flatten</i> , 120)
Dropout	–	Dropout(0.5)
FC2	–	Linear(120, 84)
FC3	Linear(84, 10)	Linear(84, 50)
Inicialización	Aleatoria simple	Xavier uniform
Optimizador	SGD	AdamW

Las modificaciones estructurales del modelo fueron definidas de manera iterativa tras un proceso de experimentación, en el que se probaron distintas configuraciones de capas, funciones de activación y esquemas de normalización. Cada ajuste se evaluó comparando el rendimiento en validación y las curvas de convergencia registradas en *Weights&Biases*, hasta obtener la arquitectura final que mostró el mejor equilibrio entre capacidad representacional y estabilidad de entrenamiento. Durante este proceso se procuró mantener la estructura fundamental del *LeNet-5* original, conservando su estilo clásico basado en convoluciones secuenciales y *average pooling*, pero introduciendo ajustes mínimos que permitieran adaptarlo de forma efectiva a las características del conjunto ESC-50.

Cada modificación se introdujo con un propósito específico:

- El cambio a **tres canales de entrada** permite procesar spectrogramas RGB generados mediante el mapa de color.
- El uso de **padding** en todas las convoluciones mantiene la resolución espacial de las características (224×224) durante las primeras capas, evitando reducciones prematuros de información en bordes y preservando patrones de energía de alta frecuencia.
- Se añadió una **tercera capa de pooling** tras la última convolución, necesaria para reducir la dimensionalidad resultante del incremento de tamaño de entrada (de 32×32 a 224×224), permitiendo un vector aplanado manejable y evitando un número excesivo de parámetros en las capas densas.
- Las capas de **Batch Normalization** fueron integradas después de cada convolución para estabilizar la distribución de activaciones y acelerar la convergencia, mejorando la robustez frente a valores extremos y la sensibilidad a la inicialización.
- Se añadió una **capa fully connected intermedia** ($120 \rightarrow 84 \rightarrow 50$) que incrementa la capacidad representativa del modelo ante el aumento del número de clases (50 en lugar de 10).

cional del modelo ante el aumento del número de clases (50 en lugar de 10).

- El **Dropout** se aplica tras la primera capa densa, contribuye a la regularización del modelo mitigando el sobreajuste derivado del gran número de parámetros en la etapa de clasificación.
- La inicialización **Xavier uniform** mejora la propagación de gradientes en redes con activaciones tanh, garantizando una varianza estable entre capas.
- El optimizador **AdamW** se empleó en lugar de SGD clásico debido a su manejo explícito del término de regularización *weight decay*. Este penaliza pesos grandes y favorece una mejor generalización al evitar que las capas densas dominen el aprendizaje.

Todas las capas convolucionales y densas utilizan la función de activación *tanh*, elegida por su estabilidad en datos normalizados en el rango $[-1, 1]$ y su naturaleza centrada que facilita la propagación del gradiente. La salida final del modelo no aplica una *softmax* explícita, ya que la función de pérdida *CrossEntropyLoss* combina internamente *LogSoftmax* y *NLLLoss*, permitiendo una interpretación probabilística directa de las clases sin modificar la arquitectura.

V. MODELO B: RESNET-18

El **Modelo B** corresponde a una implementación personalizada de la arquitectura *ResNet-18* propuesta por He et al. [2], diseñada para resolver el problema del *vanishing gradient* en redes profundas mediante conexiones residuales o *skip connections*. En este trabajo, la red se adaptó para la clasificación de los spectrogramas log-Mel del conjunto ESC-50, manteniendo la estructura fundamental de la ResNet original pero con ajustes específicos en regularización, inicialización y número de clases.

A diferencia de *LeNet-5*, que presenta una topología fija, *ResNet-18* se basa en un patrón modular compuesto por bloques residuales. Cada bloque implementa un camino directo que permite que el gradiente fluya sin restricciones entre capas, mejorando la estabilidad del entrenamiento y permitiendo redes más profundas sin pérdida de información. La arquitectura se describe en la Tabla III a continuación:

Tabla III
RESUMEN ESTRUCTURAL DE LA IMPLEMENTACIÓN DE RESNET-18
ADAPTADA AL DATASET ESC-50.

Etapa	Configuración
Entrada	$224 \times 224 \times 3$ (spectrogramas RGB)
Conv inicial	Conv2d(3, 64, kernel_size=7, stride=2, padding=3) + BatchNorm2d + ReLU
MaxPooling	MaxPool2d(kernel_size=3, stride=2, padding=1)
Bloque residual 1	2x BasicBlock(64, stride=1)
Bloque residual 2	2x BasicBlock(128, stride=2)
Bloque residual 3	2x BasicBlock(256, stride=2)
Bloque residual 4	2x BasicBlock(512, stride=2)
Promediado global	AdaptiveAvgPool2d(output_size=1)
Capa final	Dropout + Linear(512, 50)
Activaciones	ReLU en todas las capas convolucionales
Inicialización	Kaiming Normal (<i>fan_out</i> , ReLU)
Optimizador	AdamW con <i>weight decay</i> y <i>scheduler</i> (StepLR)

Al igual que para el modelo anterior, tras varias iteraciones con distintas configuraciones, se llegó a esta arquitectura en la que cada componente cumple un rol específico dentro de la red:

- La convolución inicial de 7×7 con reduce tempranamente la resolución espacial, adaptando las entradas a un tamaño manejable para los bloques residuales posteriores.
- Los **bloques residuales básicos** (BasicBlock) están compuestos por dos convoluciones 3×3 seguidas de normalización por lotes y activación ReLU. Cada bloque incorpora una conexión de identidad que suma la entrada original a la salida del bloque, permitiendo que la red aprenda una función residual. Cuando el tamaño espacial o el número de canales cambia (por ejemplo, al aplicar `stride=2`), se utiliza un camino alternativo de proyección (*downsample*) mediante una convolución 1×1 y normalización por lotes para igualar dimensiones antes de la suma. Esta estructura preserva la información de bajo nivel, mejora la propagación del gradiente y permite entrenar redes más profundas sin degradación del rendimiento.
- El **promediado adaptativo global** reemplaza capas densas intermedias, reduciendo el riesgo de sobreajuste y permitiendo procesar entradas de distinto tamaño sin redimensionamientos manuales.
- El **Dropout** se aplica antes de la capa totalmente conectada final, aportando regularización adicional.
- La inicialización **Kaiming Normal** se emplea por su idoneidad con activaciones ReLU, asegurando una propagación estable del gradiente incluso en redes profundas.

El entrenamiento de este modelo incorpora además un **ajuste dinámico de la tasa de aprendizaje** mediante un StepLR, que reduce el *Learning Rate* a la mitad cada 5 épocas. Esta estrategia permitió una convergencia más controlada en etapas avanzadas del entrenamiento, evitando oscilaciones y mejorando la estabilidad de validación.

La función de pérdida utilizada es también es *CrossEntropyLoss*, y el optimizador **AdamW** se mantuvo como en el Modelo A, aprovechando su capacidad para aplicar regularización *weight decay* para mejorar la generalización del modelo.

VI. ENTRENAMIENTOS DE LOS MODELOS

Ambos modelos fueron entrenados bajo un esquema reproducible y controlado, con registro automático de métricas mediante la plataforma Weights&Biases (wandb), lo que permitió visualizar en tiempo real la evolución de la pérdida, la precisión y las demás métricas asociadas a cada experimento.

Para cada modelo se realizaron cinco entrenamientos independientes sobre ambas versiones del conjunto de datos (con y sin *SpecAugment*), aplicando en total las cinco combinaciones de hiperparámetros mostradas en la Tabla IV. Cada ejecución se inicializó con la misma semilla aleatoria global para garantizar la reproducibilidad completa de los resultados y reducir la variabilidad debida al muestreo o a la inicialización de pesos.

Se utilizaron las mismas combinaciones para las cuatro configuraciones con el fin de mantener la comparabilidad directa entre arquitecturas y analizar de forma consistente

Tabla IV
COMBINACIONES DE HIPERPARÁMETROS UTILIZADAS DURANTE LOS ENTRENAMIENTOS.

#	Batch	Epochs	LR (α)	Weight Decay	Dropout	Patience
1	32	80	0.00005	0.00001	0.30	15
2	64	100	0.00010	0.00005	0.25	15
3	64	120	0.000050	0.00010	0.30	15
4	64	120	0.000020	0.000020	0.35	15
5	32	60	0.000075	0.00005	0.30	15

el efecto de los hiperparámetros sobre la convergencia y la generalización.

Cada configuración fue diseñada para evaluar el efecto de distintos hiperparámetros como el tamaño del lote, la tasa de aprendizaje, el *Dropout* y el *Weight Decay* sobre la estabilidad del entrenamiento. Durante cada ciclo de entrenamiento, se aplicó un esquema de **early stopping** basado en la pérdida de validación, deteniendo el proceso si no se observaban mejoras después de 15 épocas consecutivas. Esta estrategia evitó el sobreajuste, acortó los tiempos de entrenamiento y favoreció la convergencia hacia soluciones más estables y generalizables.

Las métricas de desempeño como *loss*, *accuracy*, *precision*, *recall*, *F1-score* y *AUC* se calcularon en cada época tanto para entrenamiento como para validación, y fueron registradas automáticamente en wandb para su análisis comparativo posterior. De esta forma, se documentó la evolución de la red a lo largo de los experimentos, facilitando la selección de las configuraciones más eficientes para cada arquitectura.

Finalmente, una vez completado el entrenamiento, se realizó una evaluación final sobre el conjunto de prueba para cada modelo. En esta etapa se calcularon nuevamente todas las métricas de rendimiento y se generó la matriz de confusión correspondiente, lo que permitió cuantificar la capacidad de generalización real de cada arquitectura en datos no vistos.

VII. RESULTADOS Y COMPARACIÓN DE LOS MODELOS

Esta sección presenta los resultados obtenidos a partir de los entrenamientos descritos previamente, analizando el desempeño de cada arquitectura bajo las distintas configuraciones de hiperparámetros y conjuntos de datos.

A. Resultados del Modelo A: LeNet-5

Como se observa en la Tabla V, el desempeño global del modelo LeNet-5 fue bastante malo en comparación con arquitecturas más profundas, mostrando *F1-scores* entre 21 % y 26 % en las distintas configuraciones.

El rendimiento más alto se obtuvo con la configuración 5, alcanzando un *F1-score* de 25.91 % sin *SpecAugment* y uno de 24.74 % con *SpecAugment*.

Aunque la versión aumentada del mismo experimento presentó un *F1-score* inferior, tiene un *loss* (2.92 contra 3.08) y un *AUC* mejor (84.47 contra 80.46), pero la diferencia entre ambas versiones es mínima, lo que sugiere que la técnica de aumento no generó mejoras significativas en esta arquitectura.

En general, los resultados indican que LeNet-5 presenta una capacidad de generalización muy limitada para la tarea de clasificación de audio del conjunto ESC-50. El rango de



Fig. 2. Evolución de la pérdida de entrenamiento y validación para la configuración 5 de LeNet-5 con el Dataset No Aumentado.

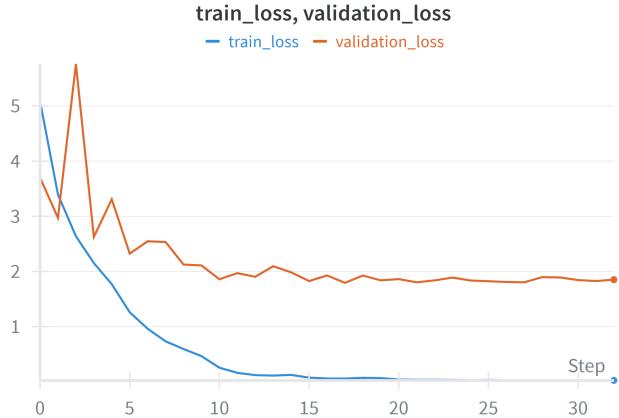


Fig. 3. Evolución de la pérdida de entrenamiento y validación para la configuración 3 de ResNet-18 con el Dataset Aumentado.

Tabla V

RESUMEN DE MÉTRICAS FINALES EN EL CONJUNTO DE PRUEBA PARA LAS CONFIGURACIONES DE *LeNet-5* CON Y SIN *SpecAugment*.

Configuración	F1-score (%)	AUC (%)	Loss
1_Augmented_LeNet5_test	24.22	82.51	2.98
2_Augmented_LeNet5_test	24.71	83.02	2.96
3_Augmented_LeNet5_test	25.29	85.63	2.95
4_Augmented_LeNet5_test	21.81	85.25	2.89
5_Augmented_LeNet5_test	24.74	84.47	2.92
1_NoAugmented_LeNet5_test	25.00	80.17	3.09
2_NoAugmented_LeNet5_test	24.78	79.59	3.09
3_NoAugmented_LeNet5_test	23.17	85.40	2.91
4_NoAugmented_LeNet5_test	23.78	82.26	3.00
5_NoAugmented_LeNet5_test	25.91	80.46	3.08

F1-scores obtenidos refleja que, a pesar de alcanzar valores de *AUC* relativamente altos (superiores al 80 %), el modelo no logra traducir esa separabilidad en predicciones precisas, evidenciando una alta tasa de falsos positivos y confusión entre clases.

El mejor desempeño se observó en la **configuración 5 sin SpecAugment**, que alcanzó el **mayor F1-score (25.91 %)** y una pérdida moderada (3.08).

Esta configuración combina una tasa de aprendizaje baja (7.5×10^{-5}) con un *Dropout* moderado (0.3) y un *Weight Decay* reducido (5×10^{-5}), lo que favoreció una convergencia más estable sin sobrepenalizar los pesos ni suprimir la capacidad del modelo para ajustar los datos. En comparación, las configuraciones con tasas de aprendizaje más altas (como la 3 y 4) mostraron pérdidas menores pero peor desempeño en *F1-score*, lo que sugiere un aprendizaje menos equilibrado entre clases.

En la Figura 2 se aprecia que la pérdida de validación se estabiliza rápidamente y luego presenta oscilaciones, lo que indica un **sobreajuste significativo**. Durante el entrenamiento, la pérdida de entrenamiento disminuyó de forma continua, mientras que la de validación se mantuvo alrededor de un valor constante después de la época 7, evidenciando que el modelo alcanzó su límite de aprendizaje. Esto es coherente con

Tabla VI

RESUMEN DE MÉTRICAS FINALES EN EL CONJUNTO DE PRUEBA PARA LAS CONFIGURACIONES DE *ResNet-18* CON Y SIN *SpecAugment*.

Configuración	F1-score (%)	AUC (%)	Loss
1_Augmented_ResNet18_test	46.21	93.35	3.07
2_Augmented_ResNet18_test	47.53	93.13	2.91
3_Augmented_ResNet18_test	59.59	95.97	2.79
4_Augmented_ResNet18_test	52.63	94.20	3.47
5_Augmented_ResNet18_test	51.22	93.87	3.21
1_NoAugmented_ResNet18_test	44.66	91.77	2.39
2_NoAugmented_ResNet18_test	43.97	92.67	2.34
3_NoAugmented_ResNet18_test	52.17	95.35	2.02
4_NoAugmented_ResNet18_test	51.05	93.83	2.30
5_NoAugmented_ResNet18_test	46.04	92.46	2.37

la arquitectura de *LeNet-5*, cuya profundidad y capacidad de representación son insuficientes para la complejidad espectral y temporal de los audios de 50 clases.

Finalmente, al comparar los resultados con y sin *SpecAugment*, no se observa una mejora clara en las métricas. Aunque el aumento de datos redujo ligeramente la pérdida y mejoró el *AUC*, no logró incrementar el *F1-score*, por lo que puede concluirse que el modelo no aprovechó adecuadamente la variabilidad introducida por la técnica de aumento. **En conclusión, la configuración 5 sin aumento de datos se considera la mejor versión de *LeNet-5***, al ofrecer el mejor equilibrio entre pérdida, *AUC* y capacidad de clasificación general, dentro de las limitaciones inherentes de la arquitectura.

B. Resultados del Modelo B: ResNet-18

Como se observa en la Tabla VI, el modelo *ResNet-18* alcanzó un rendimiento significativamente superior al de *LeNet-5*, con *F1-scores* en el rango de 43 % a 60 %.

La mejor configuración correspondió a la **configuración 3 con SpecAugment**, que logró un **F1-score de 59.59 %**, un **AUC de 95.97 %** y una pérdida de 2.79. Estos resultados superan por un amplio margen a las demás variantes, evidenciando una convergencia más eficiente y una capacidad

de representación más profunda gracias a las conexiones residuales características de la arquitectura.

Las configuraciones sin aumento de datos mostraron un rendimiento ligeramente inferior, destacando la configuración 3 con **52.17 % de F1-score** y un *loss* de 2.02 como la mejor dentro de ese grupo. La diferencia de más de 7 puntos porcentuales en *F1-score* respecto a su versión aumentada indica que la técnica *SpecAugment* mejoró la robustez del modelo frente a la variabilidad espectral, contribuyendo a una mejor generalización.

La Figura 4 evidencia un comportamiento de entrenamiento estable, con una rápida reducción de la pérdida durante las primeras épocas y una posterior estabilización. Se observa cierta inestabilidad inicial en la pérdida de validación, propia de los primeros pasos de ajuste del optimizador en redes profundas, pero luego ambas curvas mantienen una tendencia descendente hasta estabilizarse alrededor de los valores finales.

La brecha entre ambas curvas se mantiene moderada, lo que sugiere un **sobreajuste más limitado** y una adecuada capacidad de generalización. Este patrón concuerda con las métricas finales del modelo ($F1 = 59.59\%$, $AUC = 95.97\%$), confirmando que la configuración 3 con *SpecAugment* alcanzó una convergencia adecuada y un equilibrio sólido entre aprendizaje y regularización.

La **configuración 3** combina una tasa de aprendizaje moderada (5×10^{-4}) que se redujo progresivamente mediante el *scheduler*, junto con un *Weight Decay* de 1×10^{-4} , lo que permitió una exploración más amplia del espacio de pesos sin provocar inestabilidades numéricas ni oscilaciones abruptas en la pérdida. En comparación, las configuraciones 4 y 5, que emplearon tasas de aprendizaje más bajas (0.0002 y 0.000075, respectivamente) y mayor regularización, mostraron incrementos en la pérdida y reducciones en el *F1-score*. Esto sugiere que un aprendizaje demasiado conservador limitó la capacidad del modelo para escapar de mínimos locales, impidiendo alcanzar un ajuste óptimo de los pesos.

ResNet-18 demostró una mejora sustancial en todas las métricas respecto a *LeNet-5*. La capacidad de las conexiones residuales para preservar gradientes durante el entrenamiento facilitó la optimización de una red más profunda sin degradación de rendimiento. Además, el uso de *SpecAugment* resultó beneficioso en este modelo, incrementando el *F1-score* y el *AUC* sin introducir un aumento significativo en la pérdida.

En conclusión, la **configuración 3 con datos aumentados** se considera la mejor versión de *ResNet-18*, al ofrecer el mejor equilibrio entre precisión, estabilidad y capacidad de generalización, alcanzando casi el doble de rendimiento que *LeNet-5* en términos de *F1-score* (59.6 % frente a 25.9 %).

C. Mejor Modelo

A partir de los resultados obtenidos, el modelo que demostró el mejor desempeño global fue la **ResNet-18 con datos aumentados y los hiperparámetros de la configuración 3**, y sus resultados representan una mejora sustancial frente al mejor modelo *LeNet-5*.

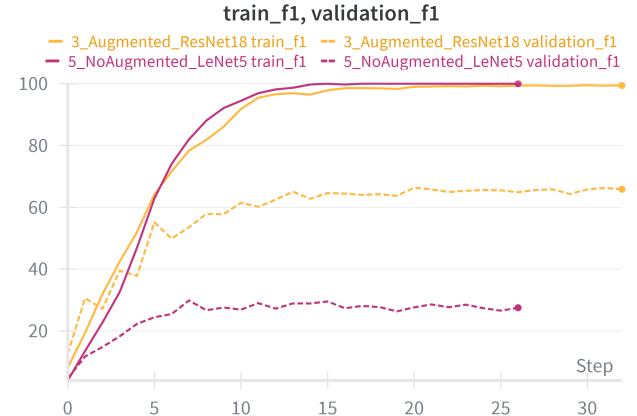


Fig. 4. Evolución del *f1-score* de entrenamiento y validación para la mejor configuración de cada modelo.

La diferencia entre el *AUC* y el *F1-score* refleja distintos aspectos del comportamiento del modelo. El *AUC* mide la capacidad global del clasificador para separar correctamente las clases, considerando todos los posibles umbrales de decisión. Por lo tanto, un valor alto de *AUC* (por encima del 95 %) indica que el modelo asigna, en general, probabilidades más altas a las clases verdaderas, aun cuando sus predicciones finales (top-1) puedan ser erróneas. En contraste, el ***F1-score*** se centra exclusivamente en las decisiones finales (las predicciones de mayor probabilidad), por lo que penaliza severamente los errores en las clases más difíciles o desequilibradas. Esto explica que un modelo pueda alcanzar un *AUC* elevado, demostrando una buena separación global en el espacio de características, pero un *F1-score* más bajo si las probabilidades correctas no son consistentemente las más altas en el top-1. En este sentido, la *ResNet-18* logró una buena calibración probabilística, pero aún presenta espacio de mejora en la discriminación final entre clases acústicamente similares.

La superioridad de *ResNet-18* puede atribuirse a tres factores principales:

- **Mayor profundidad y capacidad representacional:** permite capturar patrones espectrales y temporales complejos en los spectrogramas, mejorando la discriminación entre clases acústicamente similares.
- **Conexiones residuales:** facilitan el flujo del gradiente entre capas, previniendo la degradación del rendimiento y permitiendo un entrenamiento más estable en redes profundas.
- **Uso de *SpecAugment*:** introduce variaciones temporales y frecuenciales que simulan condiciones acústicas reales, fortaleciendo la robustez y la capacidad de generalización del modelo.

En conclusión, la **ResNet-18 aumentada (configuración 3)** se considera el **mejor modelo global**, logrando el equilibrio más favorable entre capacidad de generalización, estabilidad de entrenamiento y rendimiento en métricas de clasificación. Este modelo demostró no solo una separación efectiva en el espacio

de características (alto *AUC*), sino también una mejora tangible en las predicciones efectivas (*F1-score*) frente a *LeNet-5*, confirmando la ventaja de las arquitecturas profundas con regularización adecuada para tareas de clasificación acústica multiclas.

VIII. CONCLUSIONES

El presente trabajo implementó un sistema de reconocimiento de sonidos ambientales basado en redes neuronales convolucionales, utilizando el conjunto de datos ESC-50 y representaciones log-Mel de los audios como entrada. A partir del desarrollo de dos arquitecturas, una versión adaptada de LeNet-5 y una ResNet-18 personalizada, se demostró la viabilidad del aprendizaje profundo para la clasificación multiclas de señales acústicas.

Los resultados experimentales evidencian una diferencia sustancial de desempeño entre ambas arquitecturas. El modelo LeNet-5, pese a su sencillez estructural y baja demanda computacional, mostró limitaciones considerables para capturar la complejidad espectro-temporal del conjunto ESC-50, alcanzando un *F1-score* máximo de 25.91 % y un *AUC* promedio de 80.46 %. Esto refleja que, si bien el modelo logra cierta separabilidad global en el espacio de características, no consigue traducirla en decisiones de clasificación precisas. Además, la técnica de *SpecAugment* no produjo mejoras significativas en esta arquitectura, lo que sugiere que la capacidad de LeNet-5 resulta insuficiente para aprovechar la variabilidad introducida por la aumentación de datos.

Por otro lado, la arquitectura ResNet-18 demostró un rendimiento notablemente superior en todas las métricas evaluadas. Su mejor configuración, correspondiente al experimento 3 con datos aumentados, alcanzó un *F1-score* de 59.59 %, un *AUC* de 95.97 % y una pérdida final de 2.79. Este comportamiento confirma que las conexiones residuales y la mayor profundidad de la red permiten capturar patrones de frecuencia y tiempo más complejos, mejorando la discriminación entre clases acústicamente similares. Asimismo, la integración de *SpecAugment* favoreció la robustez y la generalización del modelo, reduciendo la brecha entre las curvas de entrenamiento y validación.

Comparando ambos modelos, se concluye que ResNet-18 superó a LeNet-5 por un margen aproximado del 130 % en *F1-score*, demostrando una capacidad de generalización significativamente mayor. Este resultado pone de manifiesto la importancia de las arquitecturas profundas con mecanismos de regularización adecuados en tareas de clasificación de audio, especialmente cuando se trabaja con representaciones espectrales de alta dimensionalidad.

En términos metodológicos, el uso de *Weights & Biases* permitió un monitoreo exhaustivo de los procesos de entrenamiento y validación, facilitando la identificación de sobreajuste y la selección objetiva de hiperparámetros. Además, la combinación de control de aleatoriedad, estratificación y registro sistemático de resultados garantizó la reproducibilidad y solidez experimental del estudio.

Finalmente, se concluye que la combinación de redes profundas como ResNet-18 con técnicas de aumentación espectral como *SpecAugment* representa una estrategia eficaz para el reconocimiento de sonidos ambientales. Como trabajo futuro, se propone explorar arquitecturas más eficientes, incorporar mecanismos de atención temporal, y ampliar el conjunto de datos con grabaciones propias que reflejen condiciones acústicas del entorno local, con el objetivo de incrementar la precisión y robustez del sistema en escenarios reales.

REFERENCES

- [1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv*, arXiv:1512.0385, 2015. [Online]. Available: <https://arxiv.org/abs/1512.0385>
- [3] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd ACM International Conference on Multimedia (MM '15)*, Brisbane, Australia: ACM, Oct. 2015, pp. 1015–1018. [Online]. Available: <https://dl.acm.org/doi/10.1145/2733373.2806390>
- [4] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," *arXiv*, arXiv:1904.08779, 2019. [Online]. Available: <https://arxiv.org/abs/1904.08779>