UNIVERSIDAD TORCUATO DI TELLA
TD VI: Inteligencia Artificial

TP 3: Clasificación de audio

Integrantes:
Isabel Núñez
Camilo Suárez
Valentina Vitetta

10 de Noviembre del 2024

Contents

1 Introducción

El aprendizaje profundo ha revolucionado el campo de la inteligencia artificial, especialmente en áreas complejas como el procesamiento de audio, donde la clasificación de sonidos y música plantea desafíos únicos.

Este informe presenta un análisis detallado del entrenamiento y optimización de una red neuronal diseñada para clasificar géneros musicales a partir de fragmentos de audio. Para ello, se utilizó la base de datos GTZAN, la cual contiene mil canciones clasificadas en diez géneros distintos como rock, jazz, hip hop, entre otros.

El objetivo de este trabajo es examinar el impacto de diversos hiperparámetros en el rendimiento del modelo de clasificación de audio, evaluando cómo configuraciones como la estructura de la red, funciones de activación, optimizadores, técnicas de regularización y representaciones de entrada (espectrogramas y waveforms) influyen en la capacidad de la red para identificar correctamente los géneros musicales.

A través de este enfoque, se pretende no solo construir un modelo eficaz, sino también comprender las decisiones de diseño que afectan la precisión y robustez del clasificador.

2 Configuración

Para monitorear y registrar el entrenamiento del modelo, se utilizó la plataforma Weights & Biases. Cada experimento fue almacenado con un nombre diciente que permitiera identificar la configuración de sus hiperparámetros.

Para entrenar y evaluar el modelo, se dividió el conjunto de datos en entrenamiento (80% de los datos), validación (10%) y test (10%). Además, para que la separación sea reproducible y para que los experimentos sean comparables, se utilizó una semilla aleatoria fija.

Durante el entrenamiento, se registraron las métricas de loss en train y validación en Weights & Biases luego de cada epoch. Esto permitió observar la evolución de la loss para contemplar si el modelo estaba mejorando o si se estaba produciendo overfitting.

En todos los experimentos de este trabajo, implementamos early stopping con un máximo de 20 epochs para optimizar el uso del tiempo de cómputo, evitando entrenamientos innecesarios. Sin embargo, en los experimentos de la sección de Entrenamiento, decidimos omitir esta restricción con el fin de analizar el impacto de modificar la cantidad máxima de epochs durante el proceso de entrenamiento.

3 Arquitectura

El objetivo de esta sección es explorar diferentes configuraciones arquitectónicas de la red neuronal, variando la cantidad de capas densas, nodos y capas ocultas. Para ello, llevamos a cabo experimentos que buscan identificar cómo estas variaciones afectan el rendimiento del modelo, destacando el mejor y peor desempeño obtenido.

La estrategia utilizada consistió en ajustar la cantidad de capas densas y el número de neuronas en cada capa oculta. Se probaron nueve configuraciones distintas con entre tres y cinco capas densas, variando el tamaño de las capas intermedias para analizar cómo la profundidad influían en la capacidad del modelo.

Los nombres de los experimentos, como **4fcLayers-1024-512-512**, indican el número de capas densas (4 en este caso) y la cantidad de neuronas en las capas ocultas (1024, 512 y 512, respectivamente).

En general, encontramos que las redes con cuatro capas densas tuvieron los mejores resultados. En particular, los experimentos **4fcLayers-1024-512-512** y **4fcLayers-1024-1024-512** obtuvieron una accuracy en validación de 0.3. Entre estas dos configuraciones, la primera mostró una loss más baja en validación, indicando una mejor capacidad de generalización.

Por otro lado, el experimento con peor desempeño fue **3fcLayers-2048-1024**, alcanzando una accuracy de 0.21. Esto puede deberse al posible sobreajuste del modelo al conjunto de entrenamiento, dado el elevado número de parámetros en relación con la profundidad de la red.

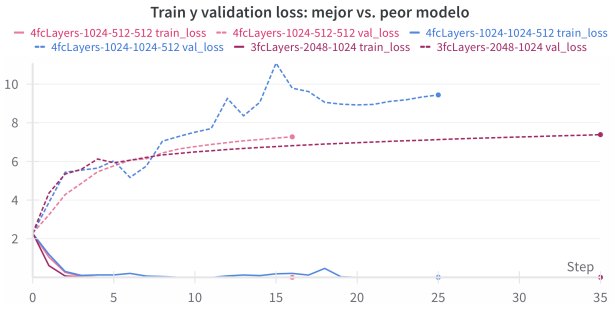

Figure 1: Arquitectura - Train y validation loss

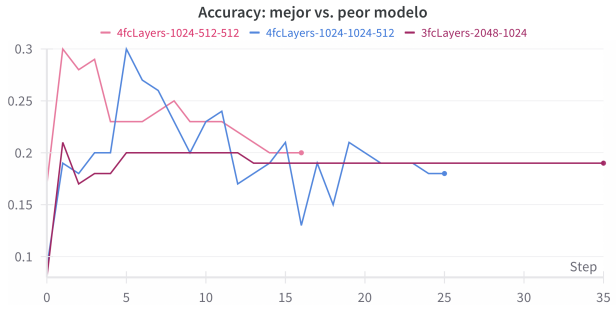

Figure 2: Arquitectura - Accuracy

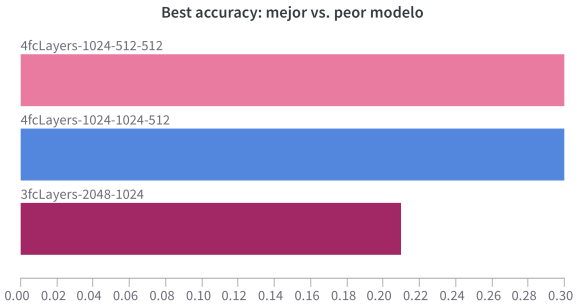

Figure 3: Arquitectura - Best Accuracy

4 Arquitectura CNN

Con el fin de mejorar el rendimiento del clasificador, extendimos el análisis de la arquitectura utilizando capas convolucionales. En esta sección, comparamos el desempeño de una red neuronal convolucional (CNN) con la arquitectura densa evaluada previamente.

Los catorce experimentos realizados incorporaron entre una y tres capas convolucionales seguidas por capas fully connected. Utilizamos espectrogramas como entrada del modelo, lo cual nos permitió capturar algunas características espaciales relevantes. En cada caso, las capas convolucionales utilizaron un tamaño de kernel de 3, stride de 1, padding de 1, funciones de activación LeakyReLU, y max pooling para la reducción de dimensión. De esta manera, nos centramos en la variación del número de capas, la cantidad de filtros en cada capa convolucional y la cantidad de neuronas en las capas densas.

Los nombres utilizados para los experimentos, como **3cnn-32-64-128–2fc-128**, indican el número de capas convolucionales y el tamaño de los filtros en cada una (32, 64, y 128 en este caso), seguido por la cantidad y tamaño (128 neuronas) de las capas completamente conectadas.

El mejor resultado se obtuvo con el experimento **3cnn-32-64-128–2fc-128**, que alcanzó una accuracy en validación de 0.5. Por otra parte, las configuraciones **1cnn-32–3fc-256-128** y **1cnn-32–2fc-128** alcanzaron los peores resultados, con una accuracy de 0.39.

Estos resultados parecen mostrar que una cantidad menor de capas convolucionales parece ser insuficiente para capturar la complejidad del dataset, provocando underfitting. En cambio, los experimentos con tres capas convolucionales mostraron una mejor capacidad del modelo, alcanzando un equilibrio entre profundidad y capacidad de generalización.

En comparación con las redes densas del ejercicio anterior, las capas convolucionales lograron mejores resultados en validación, lo que sugiere que las convoluciones permiten capturar mejor las características espaciales de los espectrogramas de audio.

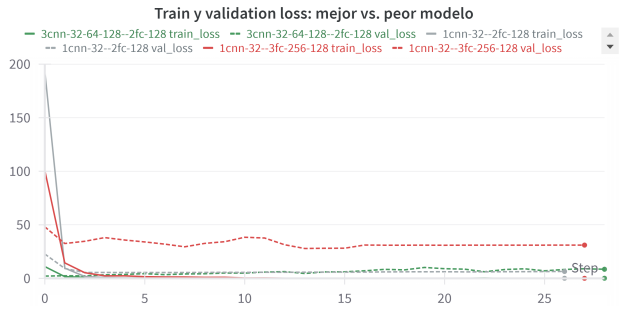

Figure 4: CNN - Train y validation loss

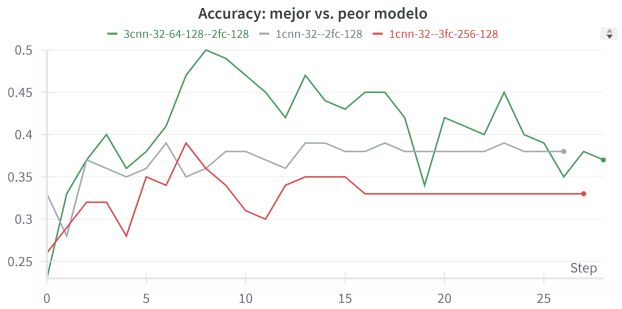

Figure 5: CNN - Accuracy

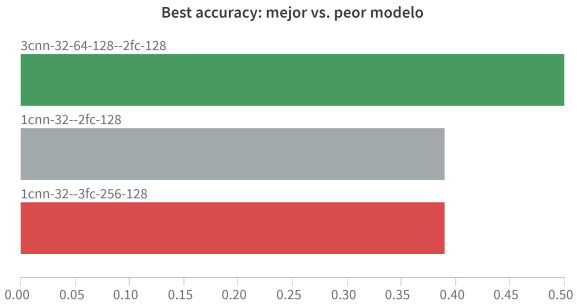

Figure 6: CNN - Best Accuracy

5 Funciones de Activación

Para explorar una mejor configuración de la red neuronal desarrollada, decidimos utilizar uno de los mejores modelos experimentados en el inciso anterior, específicamente el modelo **3cnn-32-64-128–5fc-256-256-256-128**, para experimentar con distintas combinaciones de funciones de activación. Optamos por este modelo y no el mejor absoluto, ya que, al tener más capas, nos permitió probar una mayor variedad de combinaciones.

Se evaluaron, a través de 8 experimentos, diversas funciones de activación como Shifted ReLU, ELU, Leaky ReLU, MaxOut y Swish, junto con sus diferentes parámetros (por ejemplo, alpha, shift, etc.), así como combinaciones entre ellas. Al comparar los resultados de estos experimentos, identificamos las funciones de activación que mejor se ajustaban a la tarea de clasificación de

géneros musicales.

Los mejores resultados se obtuvieron con los modelos **3cnn5fc-ELU- Shifted ReLU0.3-intercalado** y **3cnn5fc-ELU-1alpha,** alcanzando una accuracy en validación de 0.54. En el primer caso, **3cnn5fc-ELU-ShiftedReLU0.3-intercalado**, se utilizó una configuración en la que se alternaban capa a capa las funciones ELU y Shifted ReLU, comenzando por ELU y aplicando un shift de 0.3 en la función Shifted ReLU. En el segundo caso, **3cnn5fc-ELU-1alpha**, se utilizó ELU en todas las capas.

En contraste, el experimento con peor desempeño fue **3cnn-LeakyReLU-5fc-Maxout**, que alcanzó una precisión en validación de solo 0.43. En este experimento, se utilizó Leaky ReLU en las capas convolucionales y MaxOut en las capas fully connected.

Como podemos observar, la elección de las funciones de activación tiene un impacto significativo en el desempeño de la red. La combinación de funciones como ELU y Shifted ReLU parece ajustarse mejor a esta tarea, posiblemente debido a su capacidad para capturar patrones complejos en los datos. Dado que MaxOut selecciona el valor máximo entre varios subconjuntos de neuronas, creemos que, al trabajar con información de alta variabilidad, como los audios musicales, puede haber resultado demasiado "agresivo" en la selección de activaciones, perdiendo detalles sutiles que funciones como ELU o ReLU lograron capturar mejor.

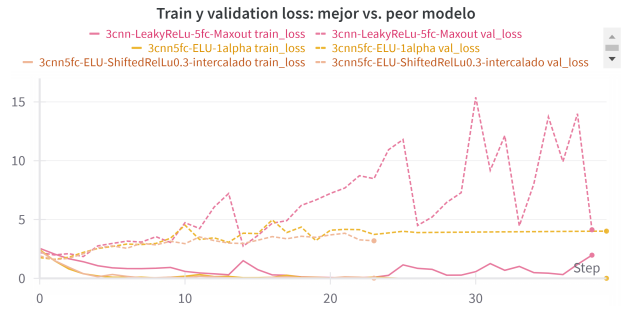

Figure 7: Función de Activación - Train y validation loss

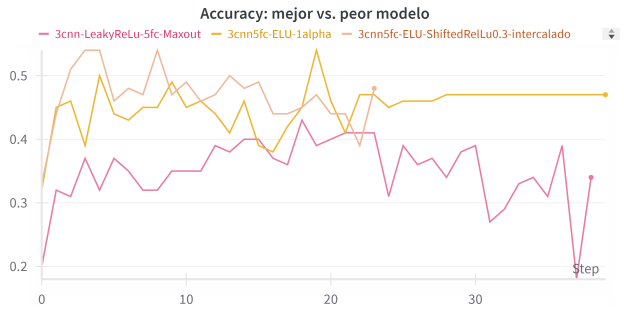

Figure 8: Función de Activación - Accuracy

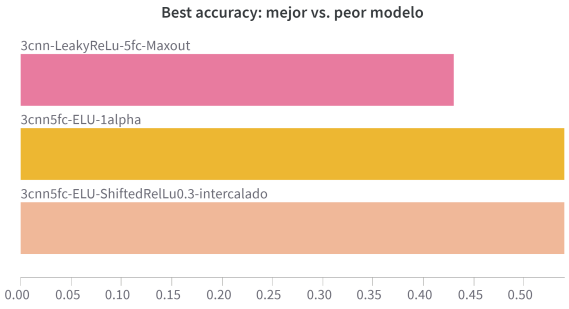

Figure 9: Función de Activación - Best Accuracy

6 Optimizadores

En esta sección evaluamos distintos optimizadores (Adagrad, Adam, RMSprop) junto con dos schedulers (StepLR y ExponentialLR). Para todos los experimentos utilizamos uno de los mejores modelos obtenidos en el ejercicio de convoluciones, el cual cuenta con tres capas convolucionales con 32, 64 y 128 filtros respectivamente, seguidas por cinco capas fully connected con 256, 256, 256 y 128 neuronas.

Como estrategia, probamos distintas combinaciones de cada optimizador con ambos tipos de scheduler, utilizando distintos valores para el parámetro gamma (0.1, 0.05 y 0.01 en StepLR y 0.99, 0.95 y 0.9 en ExponentialLR) y un step_size de 5 en StepLR.

El experimento cuyo optimizador fue RMSprop con el scheduler ExponentialLR con un parámetro gamma de 0.9 mostró el mejor rendimiento, logrando una accuracy en validación de 0.54. En cambio, el peor resultado se obtuvo con el experimento de Adagrad y StepLR con un gamma de 0.01, que alcanzó una accuracy de 0.37.

En promedio, los experimentos con RMSprop alcanzaron una accuracy de validación de 0.49, mientras que Adam obtuvo un promedio de 0.455 y Adagrad de 0.4267. Estos resultados sugieren que el optimizador RMSprop tiene una levemente mejor capacidad de predicción de géneros musicales que los optimizadores Adam y Adagrad.

Por otro lado, las configuraciones con ExponentialLR lograron una accuracy de 0.4811, mientras que las de StepLR obtuvieron 0.4333 en promedio. Esto subraya la importancia de elegir un scheduler que permita una adaptación más gradual, a comparación de uno con ajustes un poco más bruscos en momentos específicos, como es el caso de StepLR.

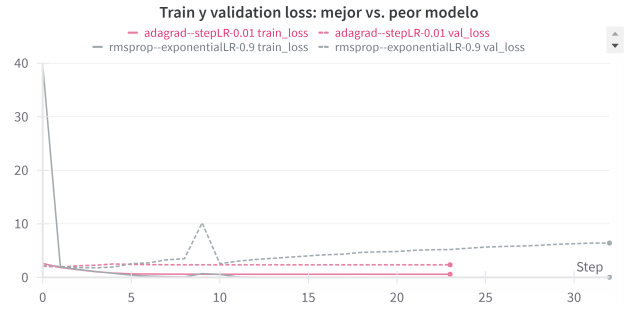

Figure 10: Optimizadores - Train y validation loss

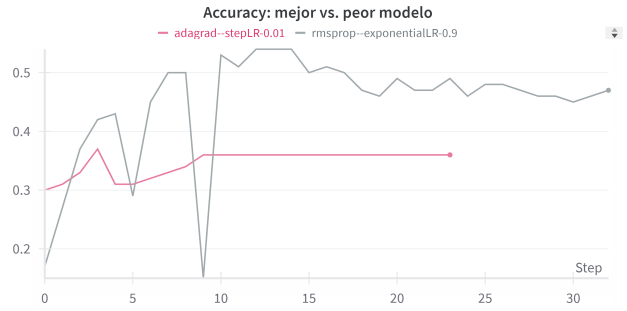

Figure 11: Optimizadores - Accuracy

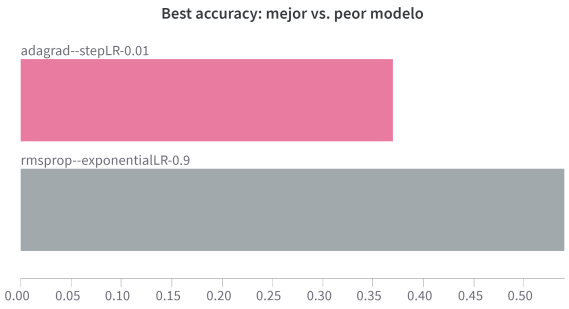

Figure 12: Optimizadores - Best Accuracy

7 Entrenamiento

En esta sección, evaluamos el impacto de los hiperparámetros batch size y número de epochs en la capacidad del modelo. Para ello, utilizamos uno de los modelos que mostró mejor desempeño en el análisis previo de funciones de activación, con una arquitectura que combina capas convolucionales y fully connected, y emplea ELU y Shifted ReLU en las capas de activación.

Como estrategia, probamos combinaciones de batch sizes de 16, 32, 64 y 128, y número de epochs de 5, 20 y 35. Esto nos permitió observar cómo el tamaño del batch procesado en cada paso de entrenamiento y la duración del entrenamiento afectaban la precisión del modelo a la hora de predecir géneros musicales.

Entre las doce combinaciones probadas, encontramos que los mejores resultados se alcanzaron con un batch size de 128 y con un número de epochs de 20 y 35, obteniendo una accuracy de 0.53 en ambos casos. Estos resultados sugieren que un tamaño de batch relativamente grande y un número de epochs suficientemente alto pueden mejorar la estabilidad y capacidad del modelo.

En cambio, el peor rendimiento fue el experimento con batch size de 128 y 5 epochs, alcanzando una accuracy de 0.48. Esto sugiere que un número insuficiente de epochs impide que el modelo converja adecuadamente, lo que causa underfitting.

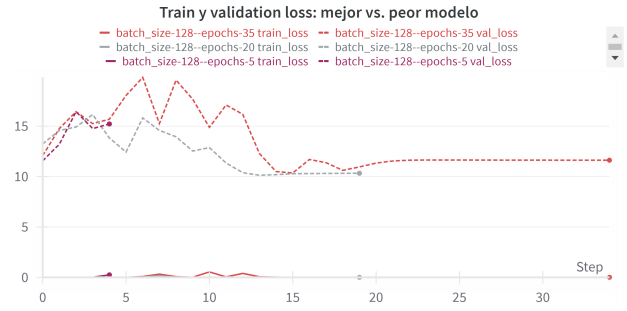

Figure 13: Entrenamiento - Train y validation loss

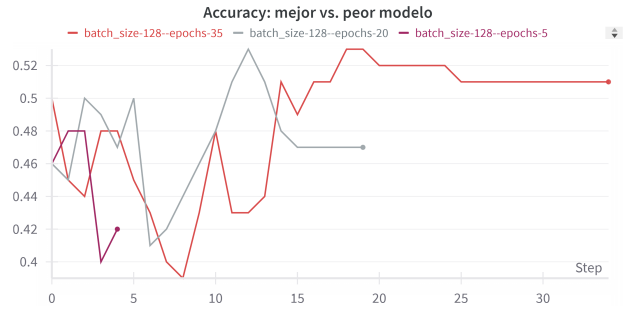

Figure 14: Entrenamiento - Accuracy

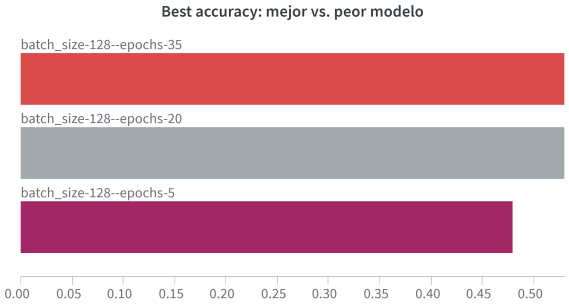

Figure 15: Entrenamiento - Best Accuracy

Al observar las accuracies promedio para los distintos batch sizes, nos sorprendieron los resultados obtenidos, ya que esperábamos que un menor batch size llevaría a un mayor overfitting. Consideramos que un batch size pequeño introduce más ruido en el cálculo del gradiente, lo cual permite aprender patrones más específicos, pero también puede hacer que el modelo se ajuste más a los datos de entrenamiento. Sin embargo, si bien el batch size de 128 tuvo un rendimiento ligeramente superior, los valores de accuracy promedio fueron bastante similares entre diferentes batch sizes.

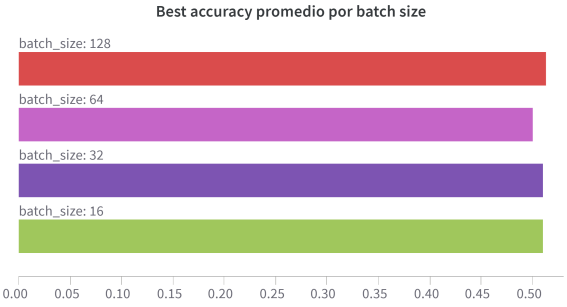

Figure 16: Entrenamiento - Best accuracy promedio

8 Regularización

Con el objetivo de evitar el sobreajuste y mejorar la generalización de los modelos, decidimos experimentar con distintos métodos de regularización.

8.1 L2

Implementamos un bucle para evaluar distintos valores del parámetro de regularización **L2** (weight decay), que ayuda a reducir el sobreajuste penalizando los pesos grandes en la red. En este proceso, probamos una lista de valores de **weight decay** (1e-5, 1e-4, 5e-4, 1e-3, 5e-3, 1e-2 y 0.1) y registramos los resultados de cada uno. Concluimos que el valor de **weight decay** que mejor equilibró la precisión de entrenamiento y validación, fue 1e-5, con el cual alcanzamos una precisión en validación de 0.56. En contraste, el valor de 0.1 resultó ser el menos efectivo, obteniendo una precisión en validación de solo 0.41.

Observamos además que, al incrementar el valor del parámetro de **L2**, la precisión del modelo disminuyó, indicando que valores altos de regularización pueden llevar a una pérdida de capacidad predictiva en este caso.

8.2 Time Shifting - Data Augmentation

Implementamos time shifting en los audios de entrada para mejorar la generalización del modelo, buscando que no dependa estrictamente de la alineación temporal exacta de las características del audio. Este método introduce una variación en la posición temporal de los eventos sonoros sin alterar la secuencia general de notas o ritmos. Para ello, probamos diferentes valores de desplazamiento, aplicando un rango de $\pm 15\%$ en un experimento y $\pm 20\%$ en otro. Esto permite desplazar el audio hacia adelante o atrás de manera aleatoria, manteniendo el contenido esencial del audio pero variando su posición temporal. En el experimento con un desplazamiento de $\pm 15\%$, obtuvimos una accuracy de 0.50 en validación, mientras que en el de $\pm 20\%$ la accuracy fue de 0.45.

8.3 Dropout

Experimentamos con dropout para mejorar la generalización del modelo y prevenir el sobreajuste. En este caso, aplicamos diferentes valores de dropout tanto en las capas convolucionales como en las capas fully connected. Probamos tres configuraciones distintas:

- Dropout del 10% en las capas convolucionales y un 20% en las capas FC.
- Dropout del 30% en las capas convolucionales y 50% en las FC.
- Dropout del 20% en las capas convolucionales y un 40% en las FC.

El mejor rendimiento se obtuvo en **dropout0.1cnn-0.2fc**, alcanzando una accuracy de 0.56 en validación. Por otro lado, el peor resultado fue en **dropout0.3cnn-0.5fc** con una accuracy en validación de 0.54

Los experimentos de regularización que mejor rendimiento obtuvieron fueron el de L2 con un parámetro de 1e-5, y el de dropout con una configuración de 10% en las capas convolucionales y 20% en las fully connected, ambos alcanzando una accuracy de validación de 0.56. Por otro lado, el experimento menos efectivo fue el de L2 con un parámetro de 0.1, que obtuvo una accuracy de validación de 0.41.

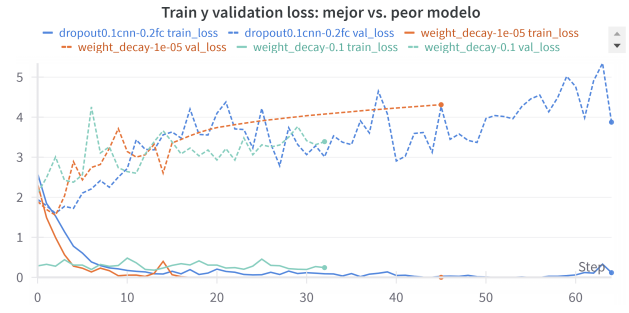

Figure 17: Regularización - Train y validation loss

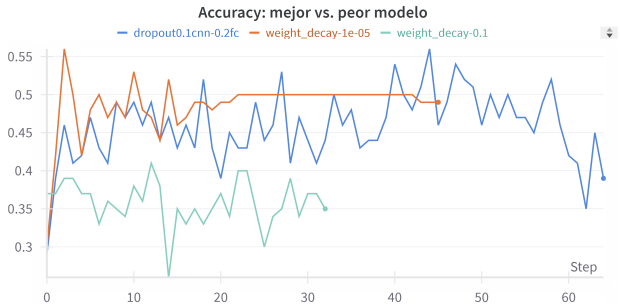

Figure 18: Regularización - Accuracy

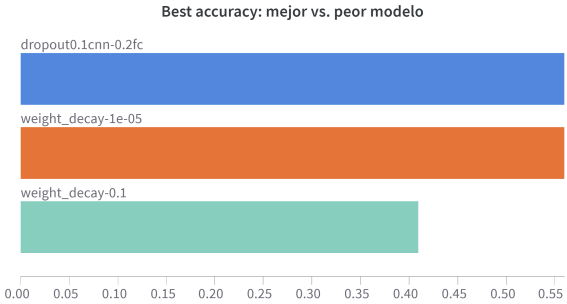

Figure 19: Regularización - Best Accuracy

9 Input

En esta sección, evaluamos los resultados de utilizar espectrogramas contra utilizar waveforms. Para ello, utilizamos uno de los mejores modelos obtenidos en los ejercicios anteriores, cuyo nombre es **dropout0.1cnn-0.2fc**. Dicho modelo consiste en tres capas convolucionales seguidas de cinco capas densas, utilizando dropout y funciones de activación ELU y ShiftedReLu. En el caso de las waveforms, utilizamos convoluciones y max pooling 1D en lugar de 2D, ajustando las dimensiones en donde fue necesario.

La comparación entre el uso de espectrogramas y waveforms como input para el modelo mostró una diferencia significativa en la capacidad de clasificación. Utilizando espectrogramas, el modelo alcanzó una accuracy de 0.56, mientras que con waveforms obtuvo 0.29.

Estos resultados sugieren que los espectrogramas presentan mejor las características relevantes

de los audios, facilitando la identificación de patrones útiles para la clasificación. Además, las dimensiones temporal y frecuencial de los espectrogramas permiten capturar de forma explícita las variaciones de frecuencia que son clave en la identificación de géneros musicales.

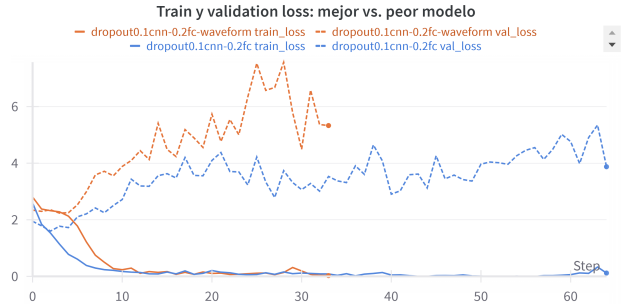

Figure 20: Input - Train y validation loss

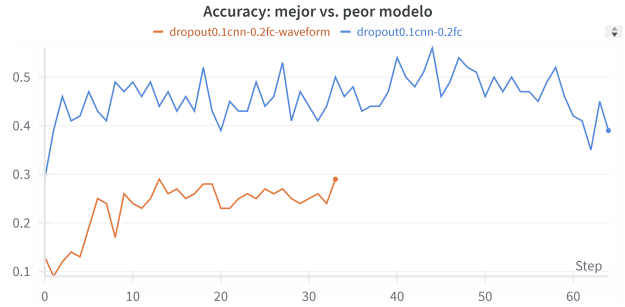

Figure 21: Input - Accuracy

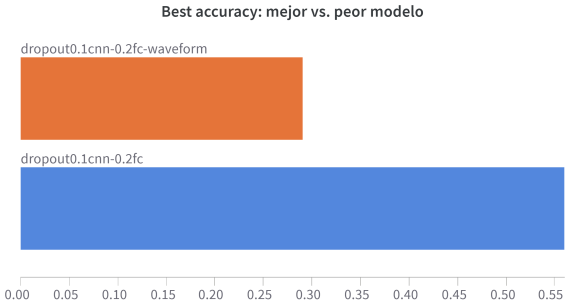

Figure 22: Input - Best Accuracy

10 Evaluación final

Para la evaluación final, se probaron los dos mejores modelos obtenidos en los experimentos anteriores: **weight_decay-1e-05** y **dropout0.1cnn-0.2fc**, ambos con un rendimiento de 0.56 de accuracy en el conjunto de validación. Al evaluar estos modelos en el conjunto de test, el modelo **weight_decay-1e-05** alcanzó una precisión de 0.51, mientras que el modelo **dropout0.1cnn-0.2fc** obtuvo una precisión de 0.52.

11 Conclusión

En este trabajo, exploramos diversas configuraciones de redes neuronales aplicadas a la clasificación de géneros musicales a partir de fragmentos de audio. A través de distintos enfoques arquitectónicos, regularizadores y técnicas de optimización, logramos identificar los mejores modelos y parámetros que optimizan el rendimiento. Los modelos con weight decay con un valor de 0.00001 y el de dropout de 0.1 en las capas convolucionales y 0.2 en las capas fully connected fueron los que alcanzaron el mejor desempeño en validación, con una precisión de 0.56.

Sin embargo, al evaluar estos modelos en el conjunto de test, se observó una ligera disminución en la precisión, lo que podría sugerir que algunos ajustes aún podrían mejorarse para lograr una mayor generalización. Los resultados también destacaron la importancia de las representaciones de entrada, como los espectrogramas, y de técnicas como el dropout y la regularización L2 para evitar el sobreajuste y mejorar la capacidad de generalización.

Este proyecto nos permitió profundizar en el uso de redes neuronales, comprendiendo mejor los desafíos y decisiones clave que influyen en el rendimiento del modelo. A futuro, la experimentación con más técnicas de regularización o diferentes arquitecturas podría seguir mejorando los resultados obtenidos.