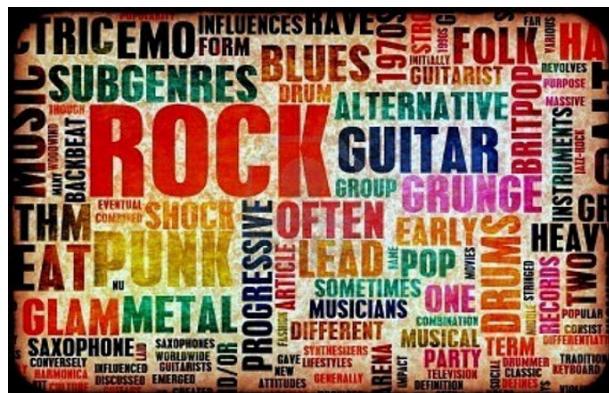


UNIVERSITAT AUTÒNOMA DE BARCELONA
Escola d'Enginyeria

Music Genre Classification



Xarxes Neuronals i Aprendentatge Profund

Carles Galletto Carner (1639390) carles.galletto03@gmail.com
Almoujtaba Kharrat Motabite (1639431) moujtabakh123@gmail.com
Alejandro Jaime Cabrera (1636745) alejandrojosejaime1@gmail.com
Joel Tapia Salvador (1638962) Joel.TapiaS@uab.cat

1. Introducción

La clasificación de géneros musicales a partir de audio es un desafío importante dentro del ámbito del aprendizaje profundo. Con el crecimiento constante del contenido musical disponible en internet, disponer de herramientas automáticas que permitan organizar y categorizar canciones de forma precisa se ha vuelto cada vez más relevante. Esta tarea resulta especialmente compleja debido a que una misma pista puede pertenecer a múltiples géneros simultáneamente, y a que dichos géneros suelen estar organizados en estructuras jerárquicas, desde categorías generales hasta subgéneros muy específicos.

El presente proyecto se centra en el desarrollo de un sistema capaz de predecir los géneros musicales de una pista a partir de su señal de audio en crudo. Para ello, se ha utilizado el dataset Free Music Archive (FMA), que cuenta con más de 100.000 pistas musicales etiquetadas con un total de 164 géneros, distribuidos en distintos niveles jerárquicos. Cada archivo de audio tiene una duración fija de 29 segundos y un sample rate de 16kHz, lo que genera señales de alta dimensión que requieren un procesamiento adecuado.

El trabajo parte de la forma de onda original (waveform) a su transformación en espectrogramas de Mel, una representación frecuentemente utilizada en tareas de clasificación de audio, ya que permite visualizar cómo varía la energía de distintas frecuencias a lo largo del tiempo.

A partir de esta representación, se han entrenado y evaluado diferentes modelos de aprendizaje profundo. En primer lugar, se ha implementado una red recurrente GRU, orientada a capturar las dependencias temporales de la señal. Posteriormente, se han probado arquitecturas convolucionales personalizadas y modelos preentrenados como ResNet18. Finalmente, se ha diseñado una arquitectura Multihead que permite realizar predicciones diferenciadas para cada nivel jerárquico del género musical. Todos los modelos han sido entrenados utilizando la función de pérdida BCEWithLogitsLoss, adaptada a problemas multilabel, y se ha incorporado el parámetro pos_weight para compensar el desbalanceo entre clases.

Todo el código de este proyecto se puede encontrar en este <https://github.com/XarNeu-EngDades/project24-25-01>

1.1. Objetivos

El objetivo principal de este proyecto es desarrollar un sistema de clasificación automática de géneros musicales a partir de señales de audio utilizando técnicas de aprendizaje profundo. Para lograr este objetivo general, se han definido los siguientes objetivos específicos:

- Explorar representaciones del audio adecuadas para tareas de clasificación, concretamente la forma de onda (waveform) y el espectrograma de Mel.
- Preprocesar y estructurar el dataset FMA, adaptando los datos a las necesidades del modelo y respetando las características multilabel y jerárquicas del problema.
- Implementar diferentes arquitecturas de red neuronal profunda, incluyendo redes recurrentes (GRU), convolucionales (CNN) y modelos preentrenados como ResNet18.
- Diseñar una arquitectura Multihead que permita realizar una clasificación jerárquica, asignando una salida por cada nivel de la estructura de géneros.
- Evaluar y comparar el rendimiento de los modelos utilizando métricas adecuadas para clasificación multilabel, loss y precisión.
- Ajustar hiperparámetros clave como pos_weight, número de capas, dimensiones ocultas y técnicas de regularización para optimizar el comportamiento del sistema.
- Analizar los errores de clasificación, especialmente en géneros amplios o ambiguos, e interpretar el comportamiento del modelo ante etiquetas difíciles.

2. Metodología

Para abordar la clasificación de géneros musicales a partir de audio crudo, se ha seguido una metodología estructurada basada en aprendizaje profundo. A continuación, se describen las principales fases del desarrollo:

2.1. Estructura y Preparación de los datos

El *Free Music Archive* (FMA) es un conjunto de datos de dominio público compuesto por 106 574 pistas musicales, cada una con una duración aproximada de 30 segundos (recortada a 29 segundos para que todas sean de igual longitud para los batches), muestreadas a 16 kHz en formato mono.

Para este proyecto se utilizaron:

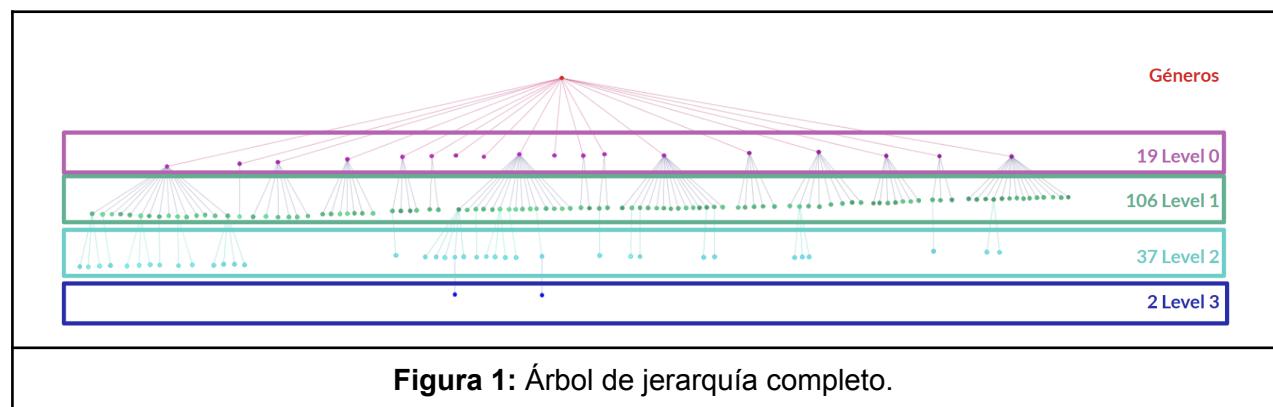
fma_large: Contiene el audio en formato .MP3.

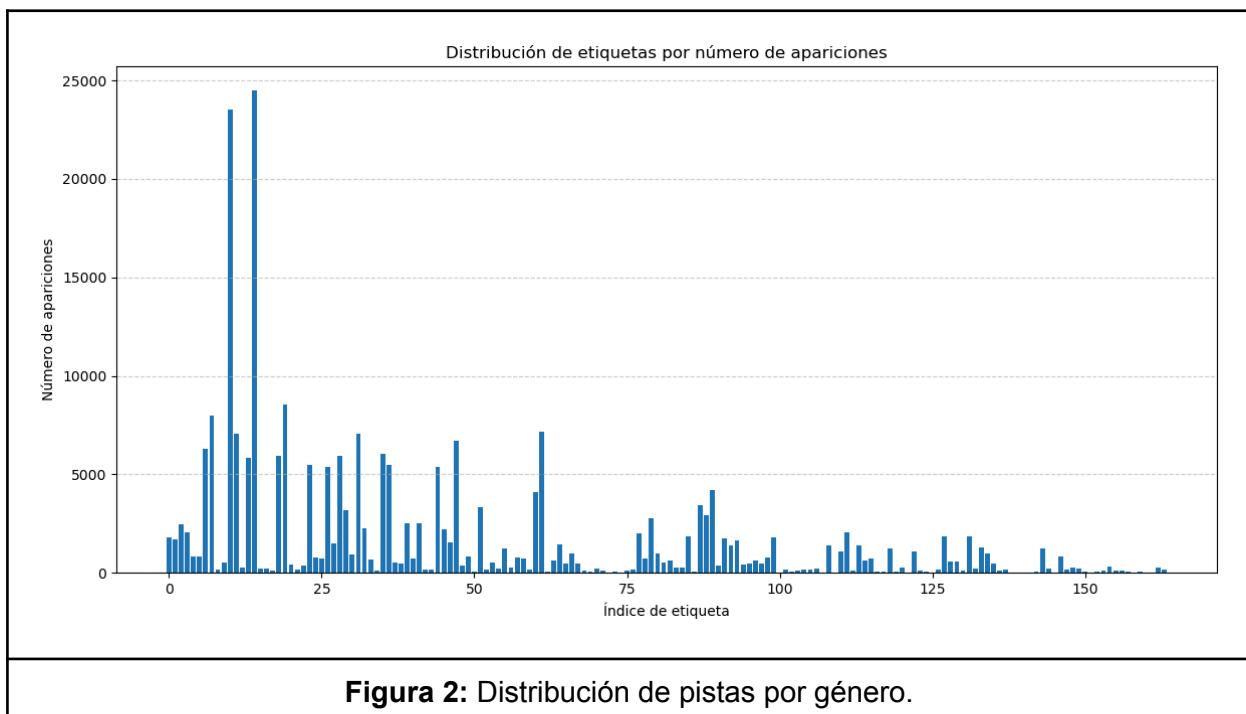
fma_metadata: Proporciona las etiquetas de género, organizadas jerárquicamente.

El conjunto incluye **164 géneros musicales** distribuidos en 4 niveles jerárquicos (0 a 3), que forman una estructura tipo árbol. Por ejemplo:

```
Level 0: Rock
└── Level 1: Loud-Rock
    └── Level 2: Noise-Rock
        └── Level 3: Sludge
```

Esta jerarquía no es estrictamente balanceada, hay ramas con más profundidad que otras, y muchos géneros hoja tienen muy pocas muestras.





Proceso de depuración

Antes de iniciar el preprocesamiento, se realizó una auditoría de integridad basada en checksums SHA-1 para detectar archivos corruptos o incompletos. Además, se filtraron:

- Archivos mal formados de duración muy inferior a 29 s.
- Tracks corruptos que no se podían cargar correctamente.

El conjunto final quedó reducido a **102 221 pistas válidas**.

Transformación a espectrogramas de Mel

Transformamos los waveforms crudos en espectrogramas de Mel, una representación frecuencial que permite capturar mejor las características acústicas relevantes. Este proceso incluye el cálculo del espectrograma con **n_fft = 1024, hop_length = 1024, 128 bandas de Mel y conversión a escala logarítmica en decibelios**.

Waveforms crudos: amplitud en el dominio del tiempo.

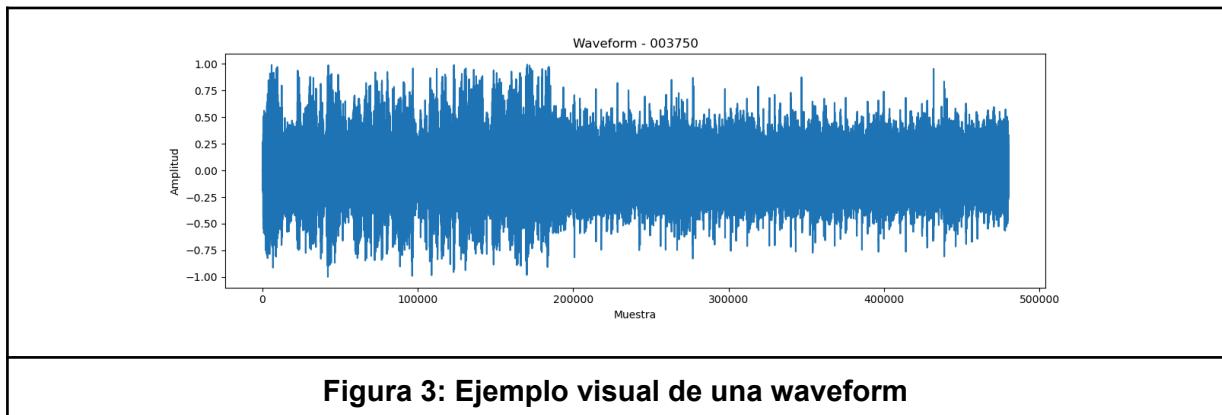


Figura 3: Ejemplo visual de una waveform

Espectrogramas de Mel: representación tiempo-frecuencia logarítmica utilizada como entrada al modelo.

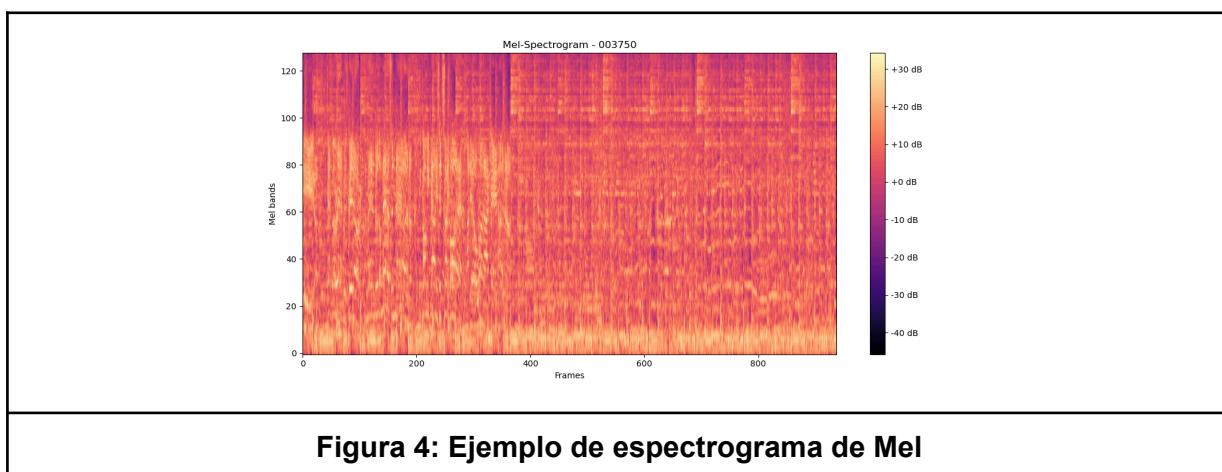


Figura 4: Ejemplo de espectrograma de Mel

2.2. División de los datos

Para garantizar la validez estadística de los resultados en un problema multilabel jerárquico, se utilizó la estrategia Multilabel Stratified Shuffle Split, implementada en `data_splitting.py` → `levels_split()`, que permite preservar la distribución de etiquetas en los distintos niveles de la jerarquía.

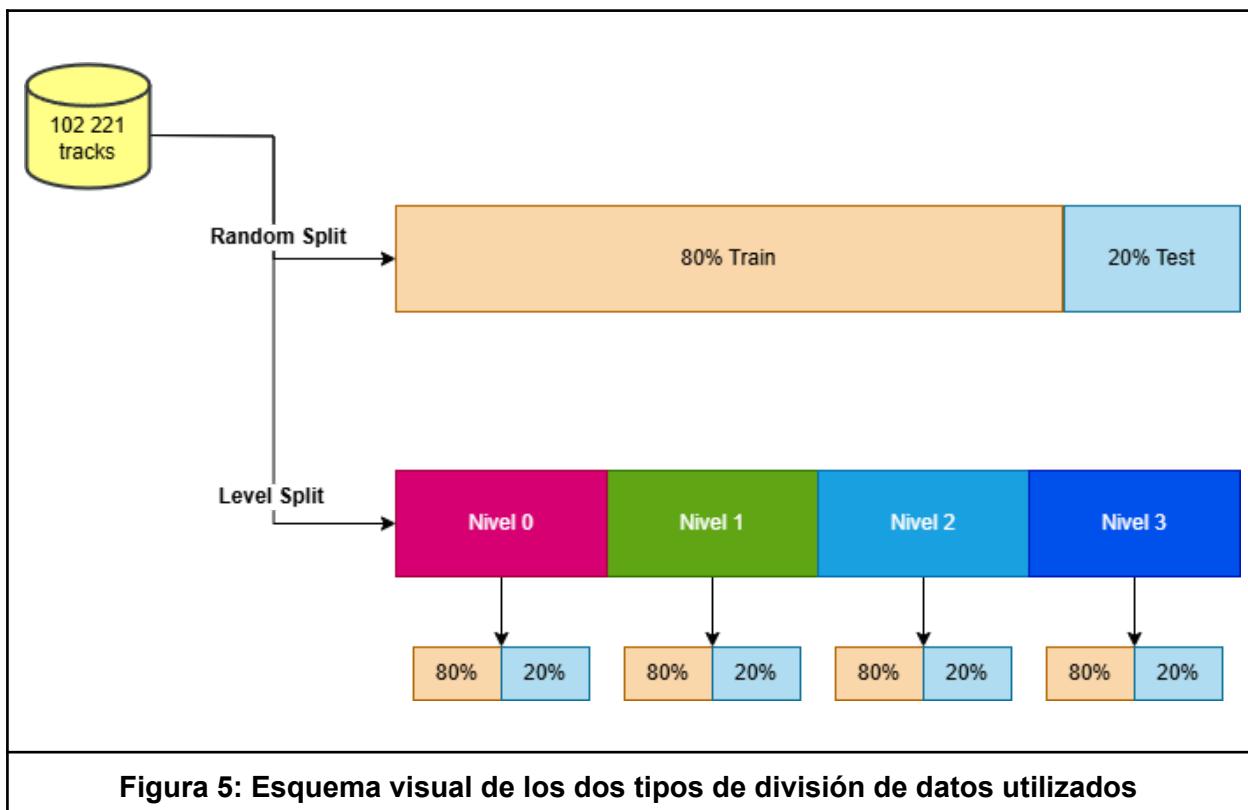
Durante fases preliminares del proyecto o en configuraciones sin separación explícita por niveles, se recurrió al método estándar `random_split`, aunque este no garantiza un equilibrio entre clases ni entre niveles jerárquicos.

Ratio: 80 % entrenamiento / 20 % validación.

Estratificación jerárquica: La rutina replica la distribución de etiquetas en los cuatro niveles de la taxonomía (género ↔ sub-género) preservando la co-ocurrencia entre niveles.

Aleatoriedad controlada: `random_state = 0` para reproducibilidad plena.

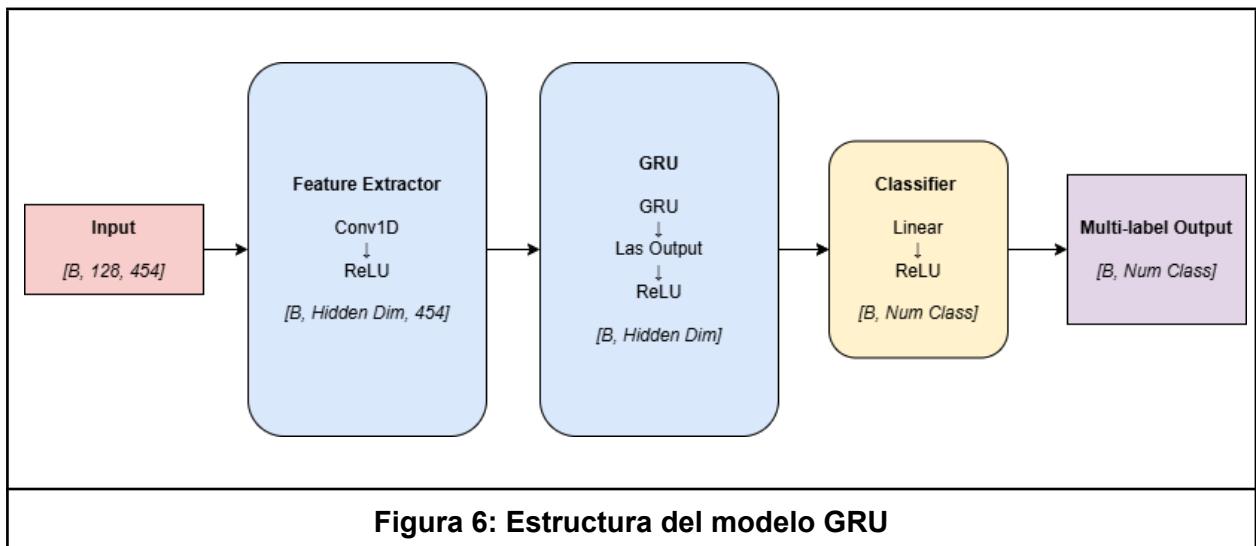
Exportación: Cada split generado se guarda como un volcado de memoria que contiene los paths de las pistas y sus correspondientes géneros, lo que permite reutilizar directamente los datos sin necesidad de recalcular la partición.



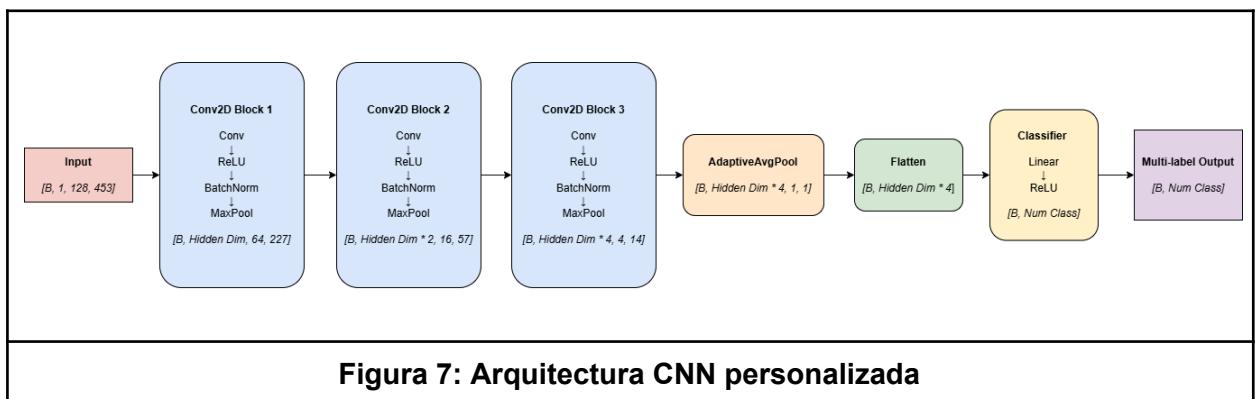
2.3. Diseño e implementación de modelos

Se han desarrollado diferentes modelos de aprendizaje profundo para abordar la tarea de clasificación multilabel. A continuación se detallan las principales arquitecturas implementadas:

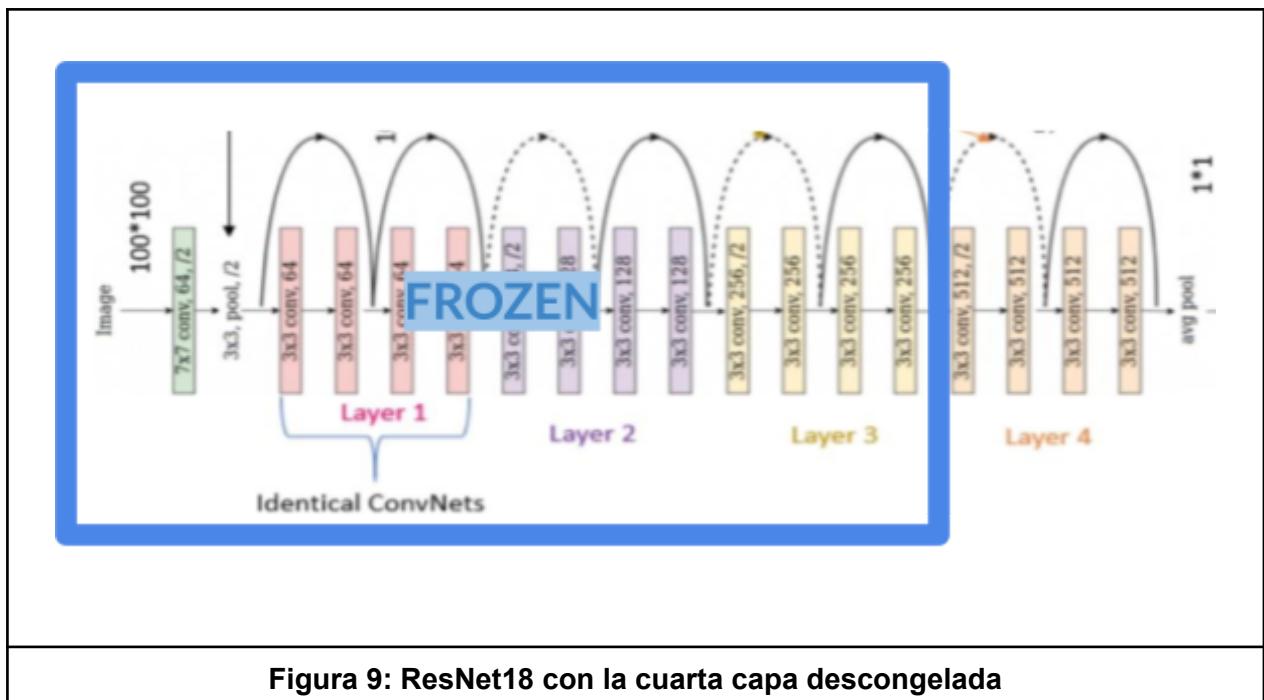
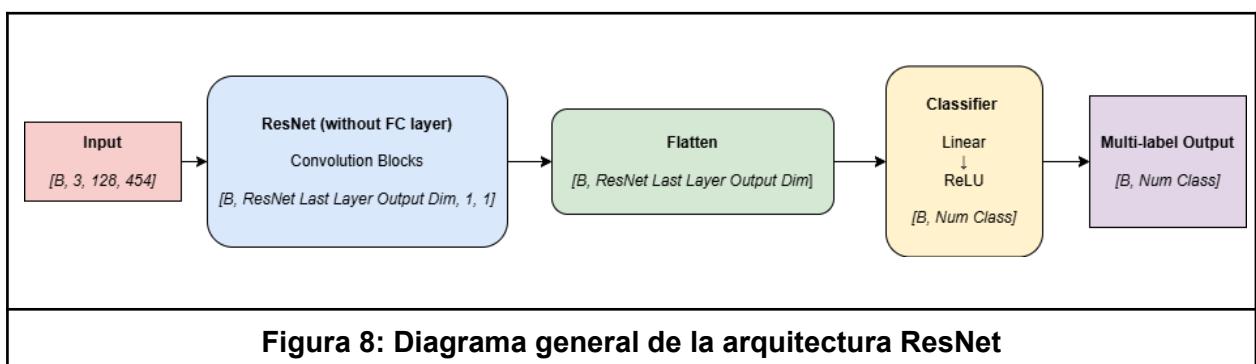
- **GRU (Gated Recurrent Unit):** Se diseñaron dos variantes de red recurrente GRU para procesar las características extraídas del audio en el espectrograma por cada unidad de tiempo. Ambas recibían como entrada una secuencia temporal de tamaño fijo. Se probaron dos configuraciones para la dimensión oculta: 256 y 512. La salida final de la GRU se conectaba a un clasificador simple basado en capas lineales y función de activación ReLU. Esta variante se evaluó utilizando el conjunto completo de géneros (todos los niveles de la jerarquía fusionados).



- **CNN personalizada:** Se desarrolló una red convolucional adaptada específicamente al formato del espectrograma de Mel. Esta red incluía varias capas convolucionales con normalización por batches y activación ReLU, seguidas de capas lineales para la predicción multilabel. Se probó, como la anterior, utilizando el conjunto completo de géneros a la vez.



- **ResNet (18, 50 y 101):** Se exploraron tres versiones de la arquitectura ResNet preentrenada sobre ImageNet: ResNet18, ResNet50 y ResNet101. En todos los casos, se reemplaza la capa final por un clasificador *multi label* propio. Primero se congelaron las capas convolucionales y posteriormente se aplicó *fine tuning* sobre ResNet18 para permitir el ajuste completo de los pesos. La ResNet18 se probó tanto con el conjunto completo de géneros como exclusivamente con los géneros de nivel 0 (top-level), para analizar el rendimiento en tareas más generales, mientras que las demás ResNet sobre el conjunto completo de géneros.



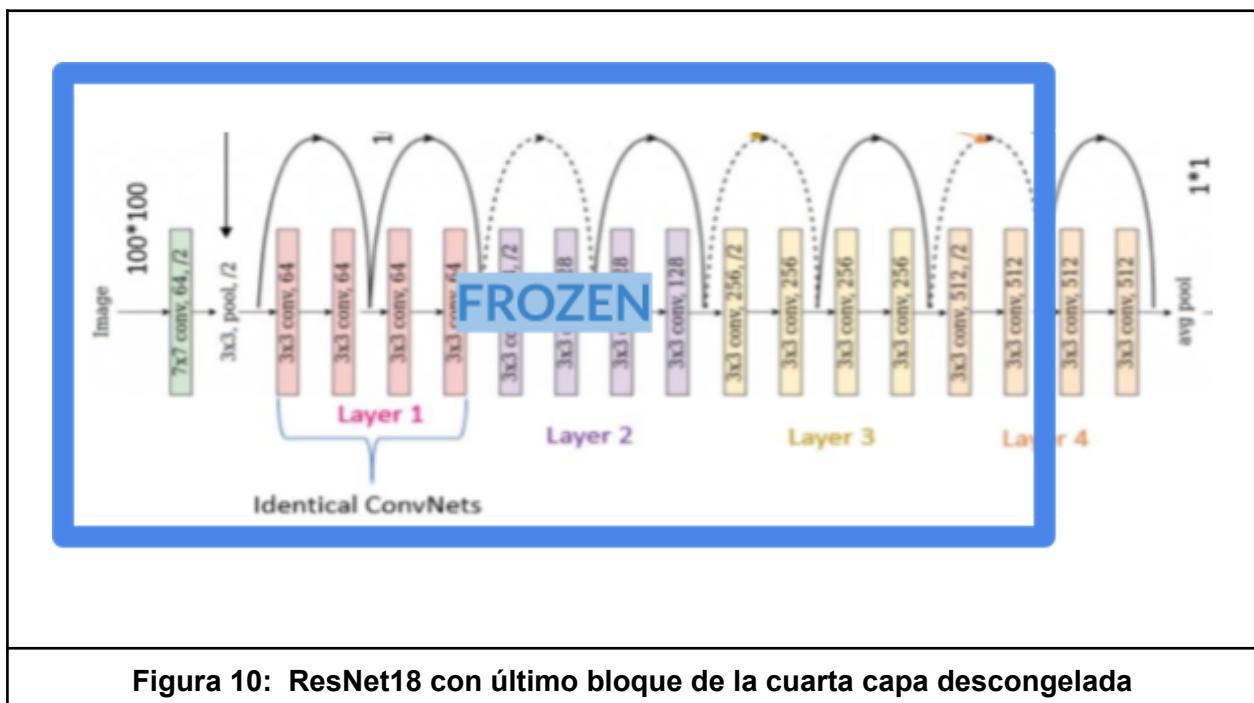


Figura 10: ResNet18 con último bloque de la cuarta capa descongelada

- **Arquitectura MultiHead:** Finalmente, se implementó una arquitectura Multihead, donde se añadieron múltiples cabezas de salida al modelo base (ResNet18). Cada cabeza estaba destinada a predecir los géneros correspondientes a un nivel específico de la jerarquía.

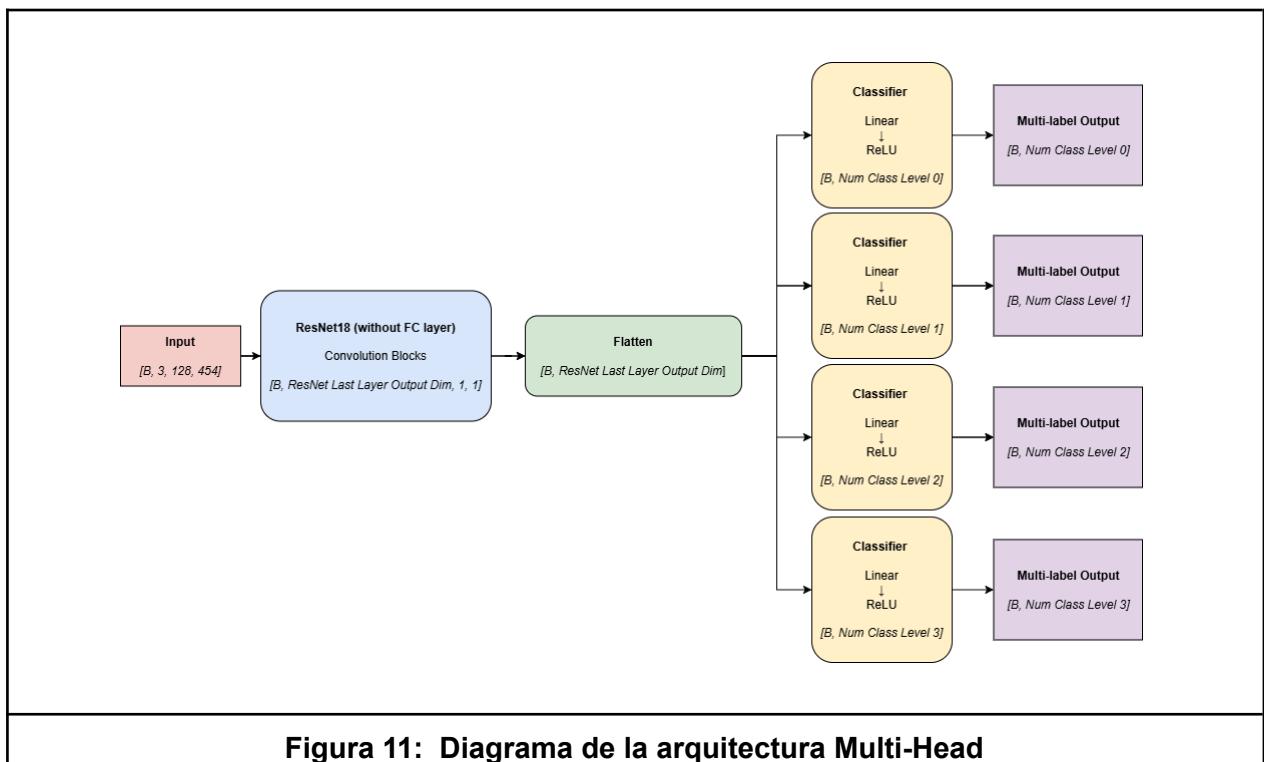


Figura 11: Diagrama de la arquitectura Multi-Head

2.4. Función de pérdida y desbalanceo

Dado que se trata de una tarea de clasificación *multi label* con fuerte desbalance de clases, se utilizó la función BCEWithLogitsLoss junto con el parámetro pos_weight, que permite ponderar la loss según la frecuencia de cada clase. Inicialmente, se probó con el método logaritmo inverso para dar más peso a las clases poco frecuentes, aunque después se probaron varios diferentes para ver cómo afectaban y si la suposición inicial fue correcta.

2.5. Métricas de evaluación

Para evaluar los resultados y modelos se han utilizado tre métricas:

Métrica	Dónde se Calcula	Propósito
BCEWithLogits Loss	Train y Validation (cada epoch)	Señal de convergencia y overfitting.
Accuracy	Train y Validation (cada epoch)	Medida de aciertos globales.
Matrices de confusión	Train y Validation (cada epoch)	1) Global 2×2 (TP, FP, FN, TN) para comprobar balance general de errores. 2) $L \times L$ por cabeza para identificar qué géneros se confunden entre sí.

2.6. Entrenamiento y optimización

Los modelos fueron entrenados durante 100 épocas utilizando el optimizador Adam. Se utilizó como Learning Rate Scheduler “OneCycleLR” que comienza con la learning rate en un punto medio bajo ($1/25$ la learning rate máxima $\rightarrow 1.2 \times 10^{-4}$) y lentamente va aumentando la learning rate hasta que al cabo del 30% de los steps alcanza la learning rate máxima (3×10^{-3}) y lentamente, siguiendo la forma de una función coseno alcanza la learning rate final ($1/\text{la learning rate inicial} \rightarrow$).

Hemos inicializado los pesos utilizando Xavier Uniform. Y los entrenamientos han sido de 100 épocas (excepto que fuise claro que la red no estaba mejorando o estaba haciendo mucho overfitting que entonces se paraba) con 12 workers, con preload de 2 batches, y utilizando el batch size máximo que permitía la memoria de la GPU dando a considerar los pesos del modelo a entrenar. Para entrenar utilizamos un dropout de 50% para el modelo GRU y FullCNN y un dropout del 70% para todos los modelos con ResNet. Sin utilizar clip del gradiente.

3. Diseño experimental

3.1 Descripción del conjunto de datos

La descripción completa del dataset FMA, incluyendo su jerarquía de géneros y proceso de depuración, se encuentra en el apartado **2.1**. Véanse las **Figuras 1, 2 y 3** para la jerarquía de géneros y la distribución de pistas por nivel.

3.2 Pre-procesado y transformaciones

El preprocessamiento del audio (normalización, generación de Mel-spectrogramas y almacenamiento en caché) se detalla en **2.1**.

3.3 Partición de los datos

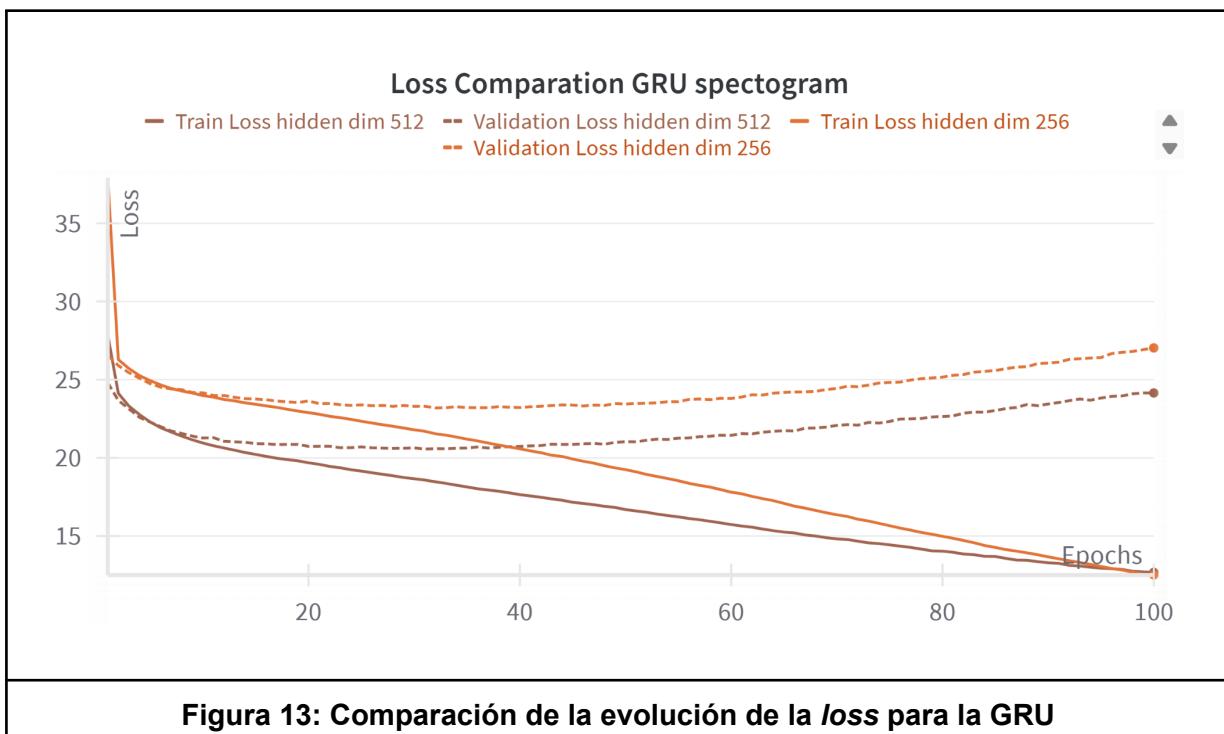
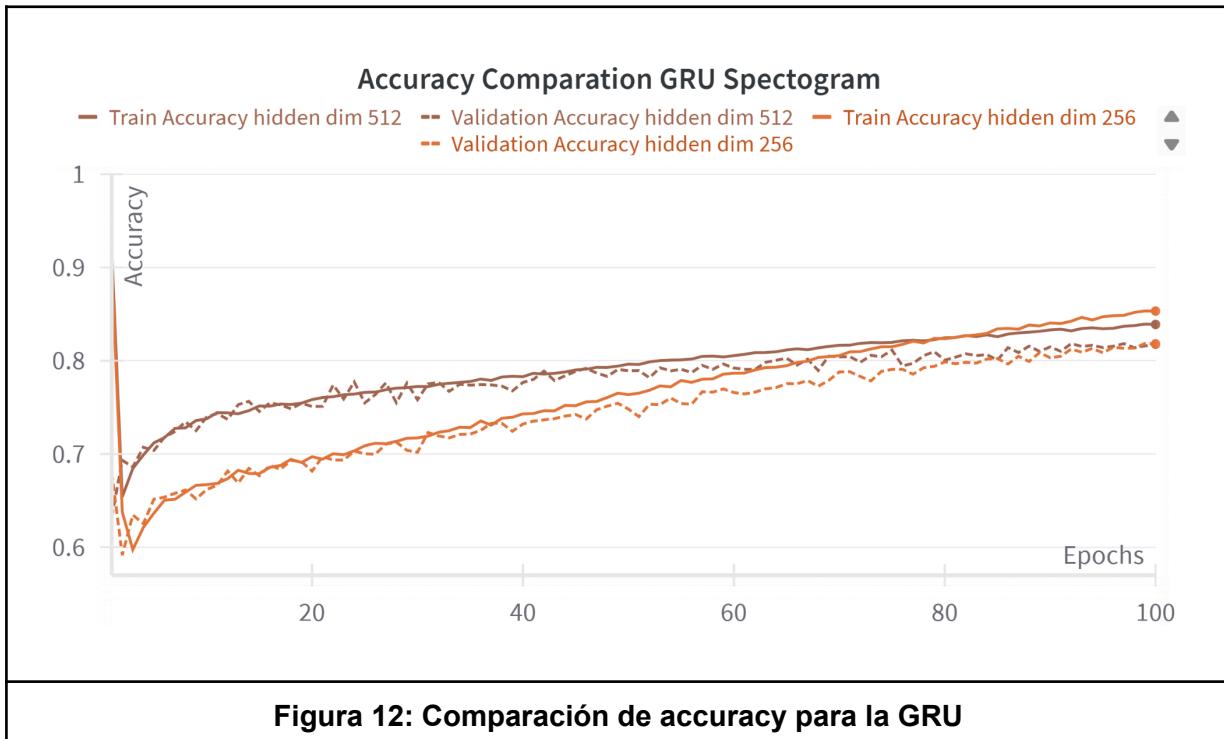
En el Experimento 1, 2, 3 y 4 la partición de datos es sobre todos los géneros a la vez sin importar el nivel, de forma aleatoria. En el Experimento 5 se aplica el Stratified Shuffle y se utilizan solo los géneros del nivel 0. En el Experimento 6 se aplica otra vez el Stratified Shuffle y se separan los géneros por niveles.

3.4 Experimentos

Nº	Arquitectura	Objetivo
E1	GRU + Classifier	Probar un modelo RN.
E2	Custom CNN 2-D + Classifier	Probar modelos CNN que comparar con E1.
E3	Frozen ResNet18/50/101 + Classifier	Probar rendimiento con CNN ya entrenadas congeladas.
E4	Fine Tuning ResNet 18 + Classifier	Aplicar Fine Tuning para comparar con E2.
E5	Fine Tuning ResNet 18 + Classifier: Only Top Level	Evaluar si limitarse a los géneros de nivel 0 simplifica el problema y mejora la precisión.
E6	ResNet18 Multi-Head	Comparar los resultados con clasificadores dedicados por nivel en vez de todos los géneros simultáneamente, y el efecto del balanceo de los pesos de los géneros.

4. Resultados

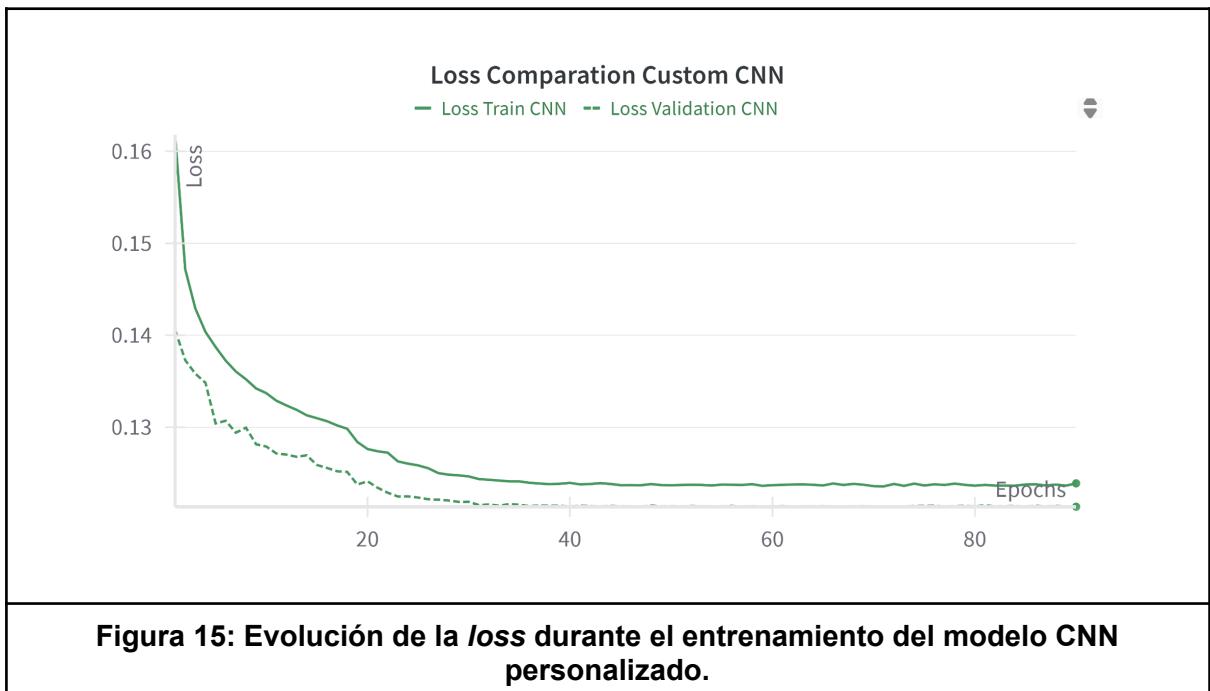
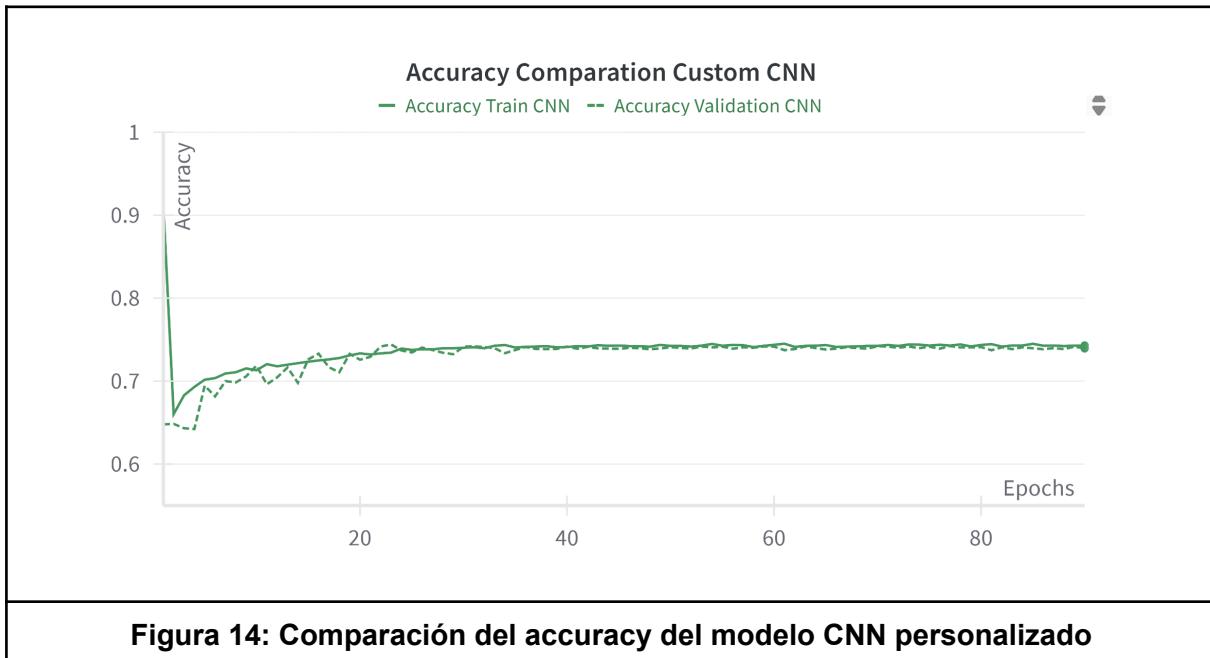
4.1. Experimento 1



Como se puede ver en la figura 13, a partir de aproximadamente la época 10 ambos modelos de GRU sufren un gran overfitting aumentando la loss de validación. Aun así, en la figura 12 de la accuracy este overfitting no se empieza a reflejar hasta la época 40.

En la figura 12 se puede ver que ambos modelos no han convergido al cabo de 100 épocas y terminan sobre una accuracy del 80%. Inicialmente, el modelo con hidden size 512 tiene mejores resultados, al final del entrenamiento se puede ver al de 256 sobrepasarlo aunque la loss de validación del de 256 no mejora como la de entrenamiento, esto puede dar a deducir que está mejorando final del de 256 se debe a un overfitting muy grande y que el de 512 generaliza mejor.

4.2. Experimento 2



En este modelo de CNN personalizada podemos observar que el modelo converge sin overfitting con una accuracy aproximada del 74%.

4.3 Experimento 3

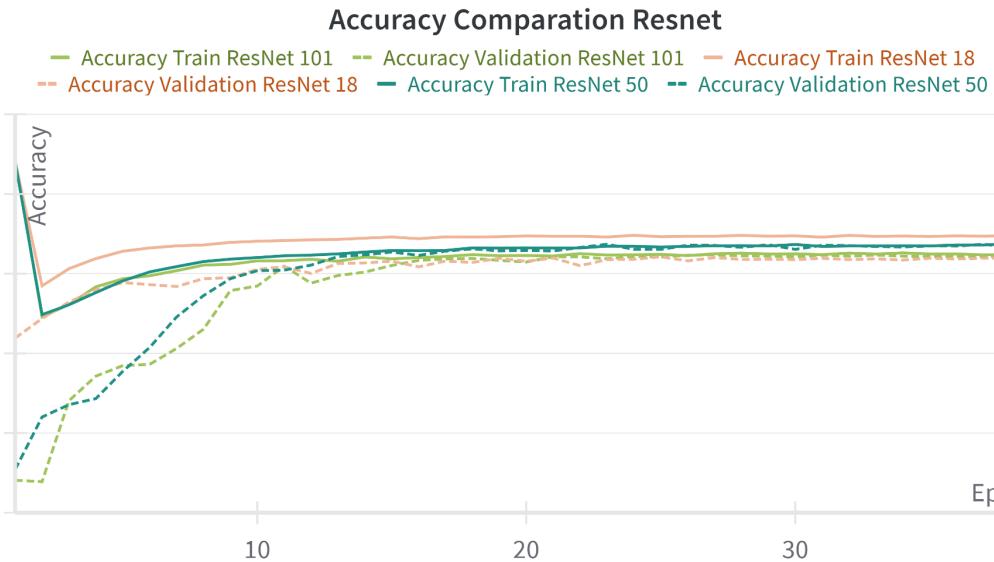


Figura 16: Comparación del accuracy para las arquitecturas ResNet18, ResNet50 y ResNet101.

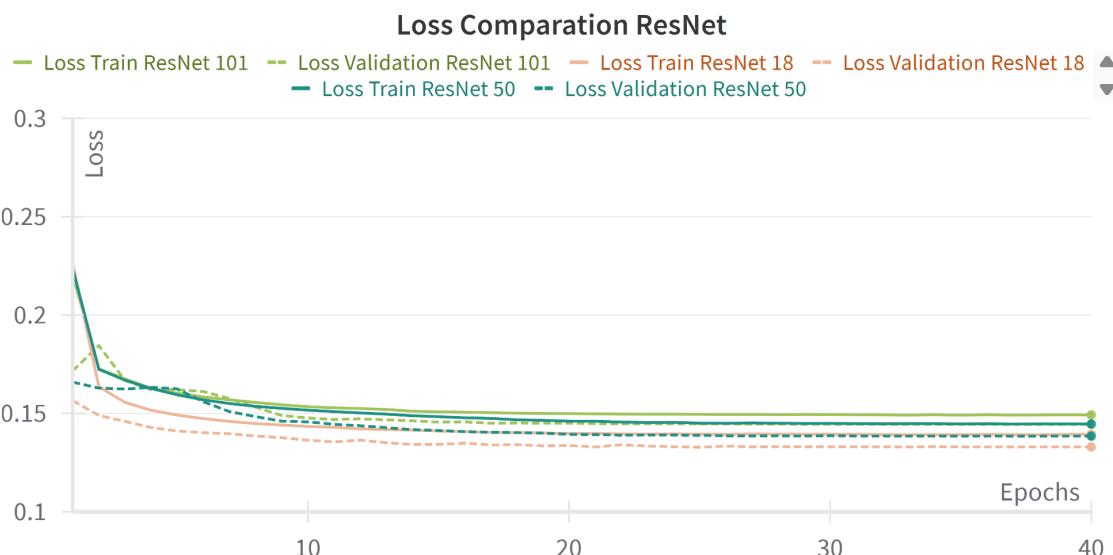


Figura 17: Comparación de la loss durante el entrenamiento de las arquitecturas ResNet18, ResNet50 y ResNet101.

Podemos observar en la figura 17 que las tres versiones, ResNet18, ResNet50, ResNet101 convergen en su entrenamiento sin mucho overfitting.

Tanto en la figura 16 y 17 se puede observar que la ResNet18 obtiene mejores resultados que las ResNet50 que obtiene mejores resultados que la ResNet101, observando que no porque el modelo sea más grande, dará mejores resultados. Por eso los siguientes experimentos que se plantearon para los modelos de ResNet todos se hacen con la ResNet18.

Se obtiene una accuracy del 60% aproximadamente, menor que la de la CNN personalizada, pero aquí solo se está entrenando el classifier, por lo tanto, se espera que al hacer fine running mejoren los resultados.

4.4. Experimento 4

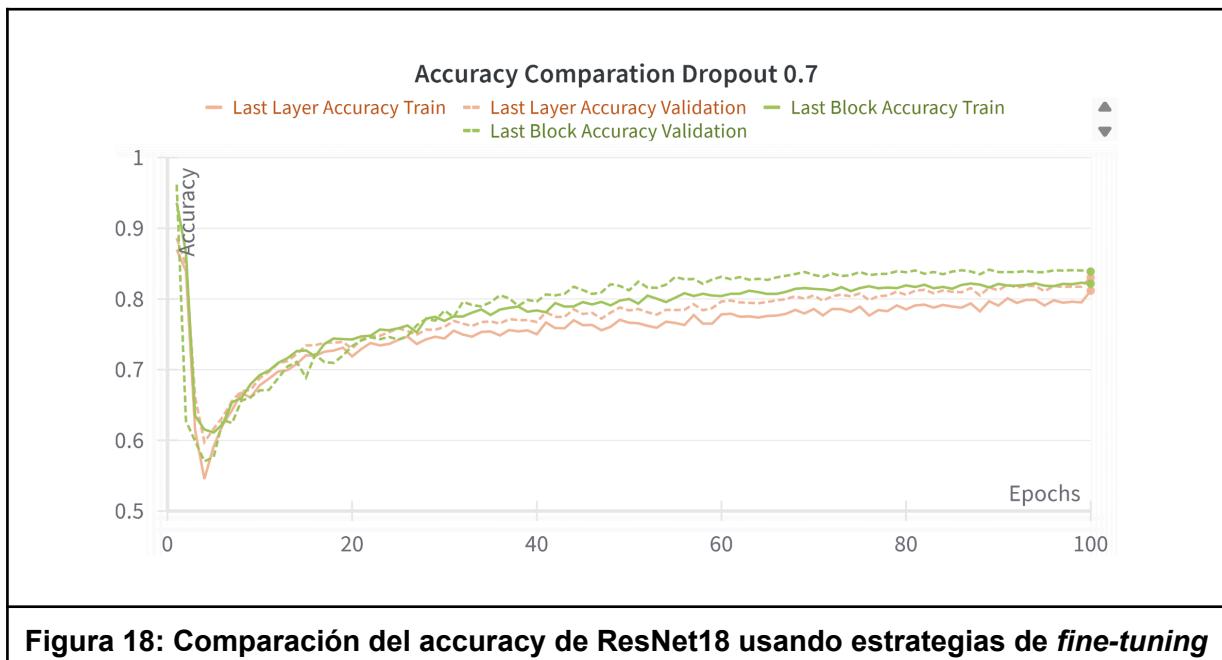


Figura 18: Comparación del accuracy de ResNet18 usando estrategias de *fine-tuning*

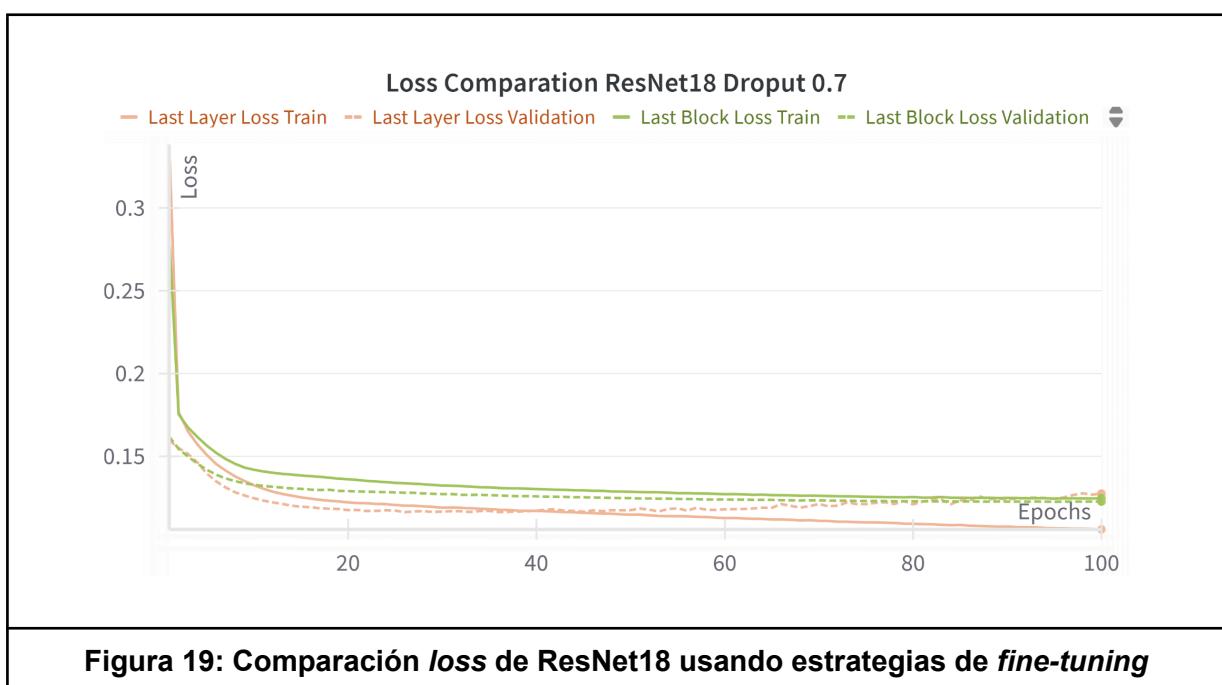
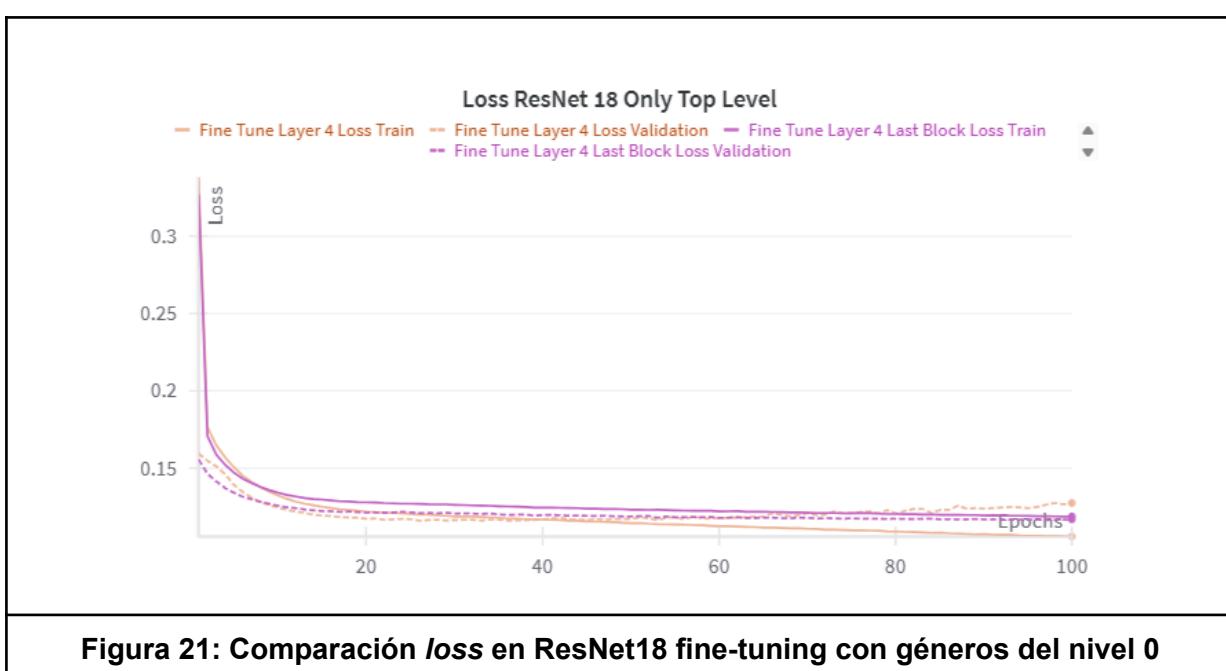
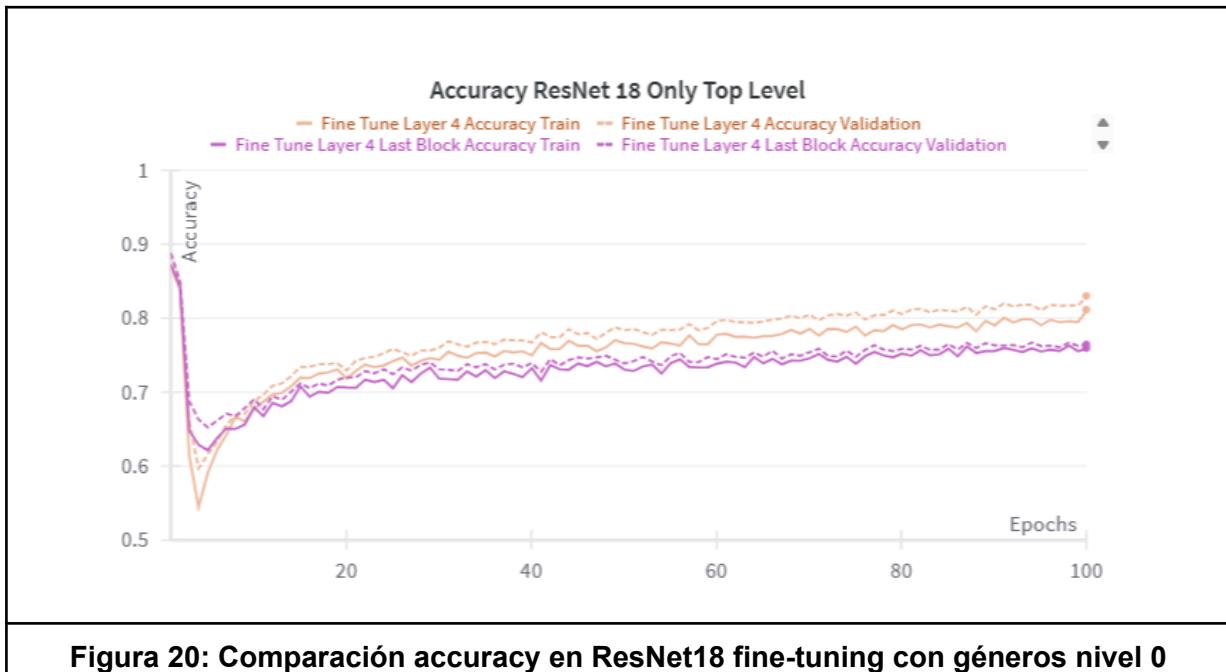


Figura 19: Comparación loss de ResNet18 usando estrategias de *fine-tuning*

En la figura 19 se puede observar que ambas versiones convergen, pero la versión que descongela la layer 4 entera en comparación con la versión que descongela únicamente el último bloque de esta capa tiene un overfitting significativo.

En la figura 18 podemos ver qué a causa de este overfitting la versión donde solo se descongela el último bloque de la última capa tiene una accuracy ligeramente mejor con un 81%. Estos resultados son mejores también que la accuracy del modelo CNN personalizado, así que para los próximos experimentos utilizaremos esta versión.

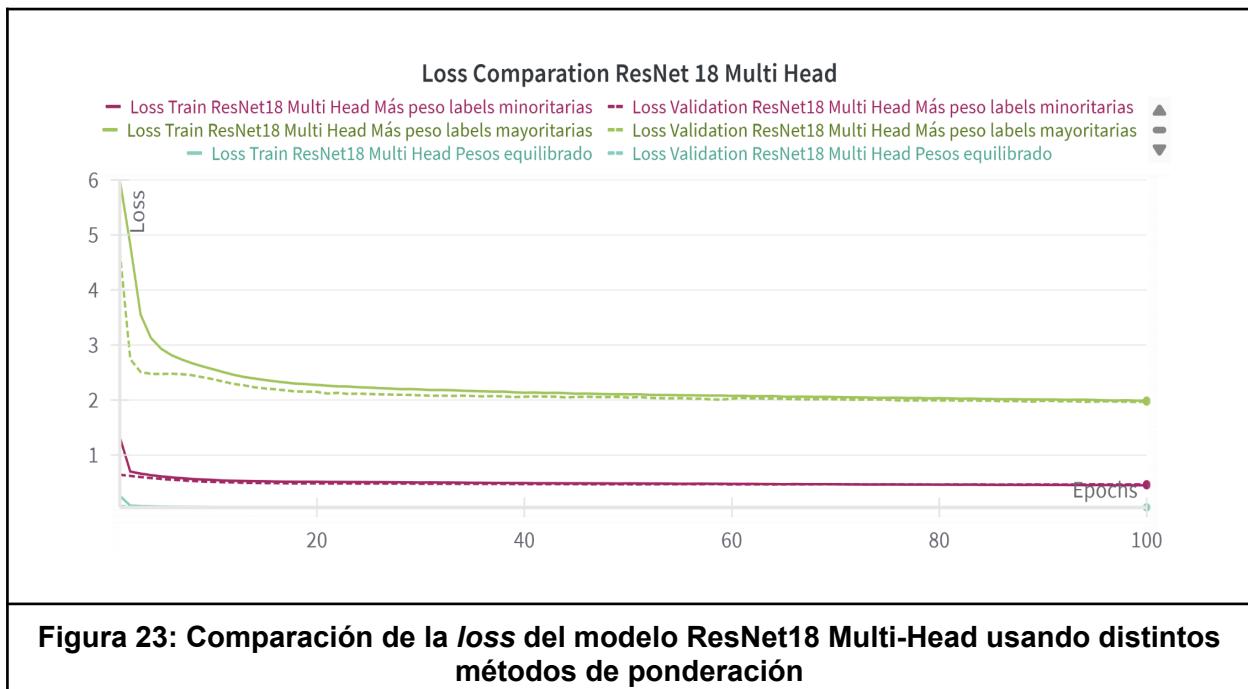
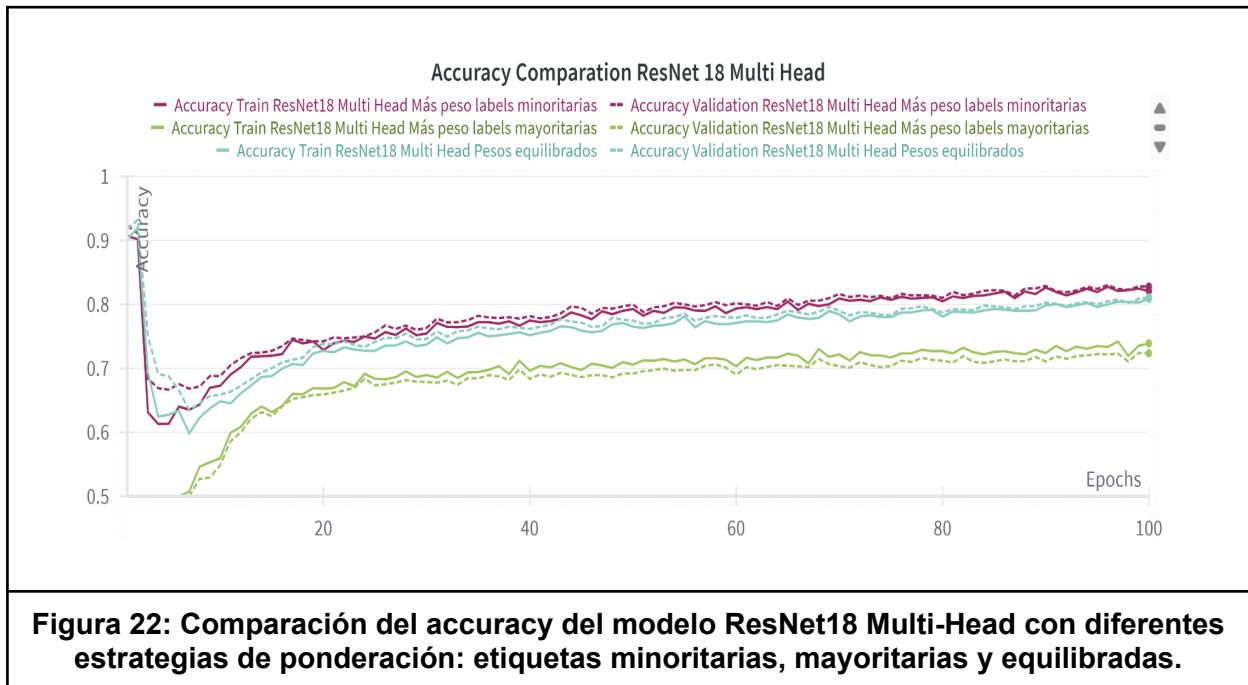
4.5. Experimento 5



Hasta ahora hemos probado clasificando todos los géneros a la vez, por lo tanto, hemos decidido probar clasificar solo los top levels con las versiones de fine tuning para ver cómo cambian los resultados.

Podemos observar en la figura 21 que pasa algo similar que con todos los géneros, la versión de descongelar la layer 4 entera tiene un overfitting significativo. Y, como se ve en la figura 20, el modelo entrenado solo el último bloque de la layer 4 tiene una accuracy del 76%.

4.6. Experimento 6

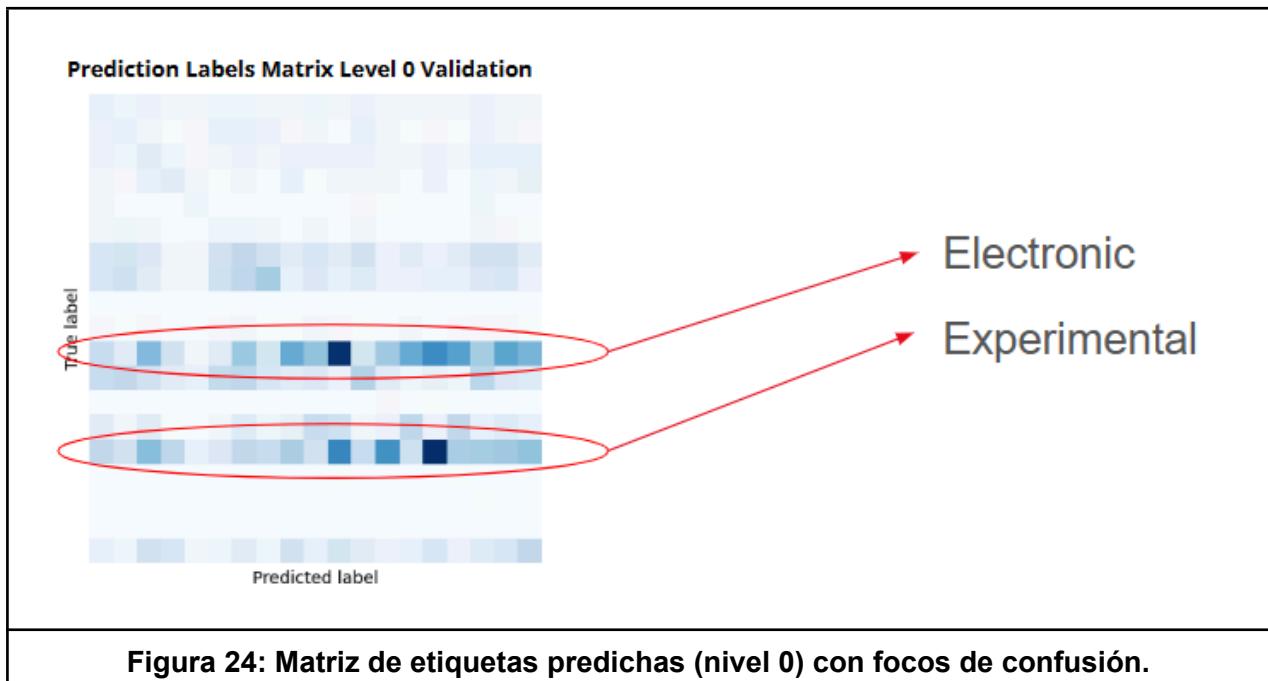


A continuación, con la versión de la ResNet18 haciendo fine tuning sobre el último bloque de la layer 4 intentamos predecir los géneros con 4 heads diferentes, una por cada nivel. Y compraremos el efecto al modificar el pos_weight que recibe la loss para cada clase.

El modelo converge y podemos observar que al darle más peso a las clases mayoritarias el modelo tienen peores resultados, al darle el mismo peso a todas las clases los resultados mejorar, pero sigue siendo mejor dándole más peso a las clases minoritarias como se había supuesto.

5. Discusión

Dados estos resultados nos gustaría saber por qué, que hace con las clases, como las clasifica porque se equivoca. Para eso miramos las matrices de confusión de las clases.



Como podemos observar en la figura 24 (en este caso el nivel 0, pero ocurre en cada nivel) hay clases que la red intenta clasificar como casi todas las demás clases. Si nos fijamos en los espectrogramas de la figura 25 podemos ver que esto se debe a que estos géneros son muy variados entre sí, hasta cierto punto que hay tracks que se parecen más a las tracks de otros géneros que a las otras del mismo género.

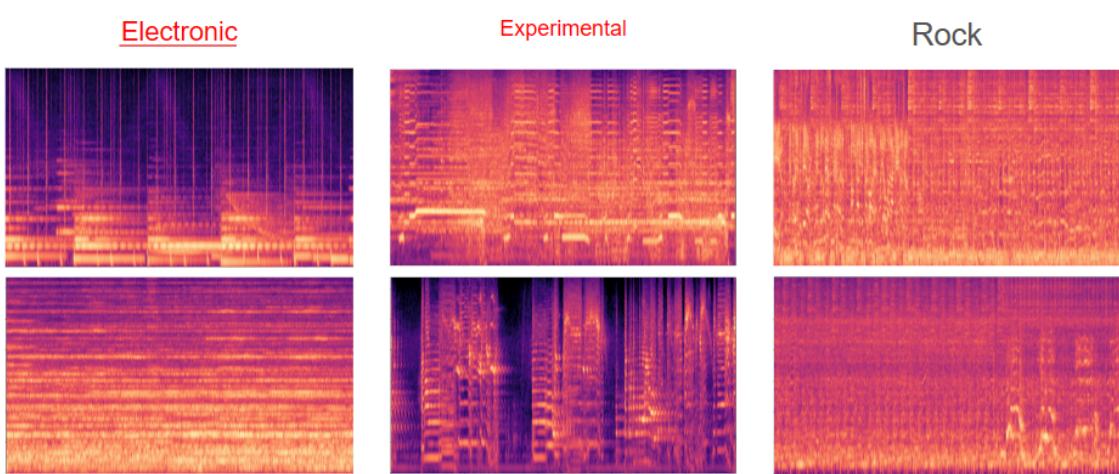


Figura 25: Espectrogramas representativos de tres géneros.

Si, en cambio, nos fijamos en la matriz de confusión general del modelo, podemos observar que hay muchos valores negativos (recordemos moda de 3 géneros por tracks y hay 164 géneros) esto hace que el modelo clasifique casi todo como negativo y se arriesgue poco a clasificar una clase como positivo (recordemos es multilabel, por lo tanto, cada clase tiene su probabilidad), esto acaba con que el modelo tiene una cantidad muy alta de falsos negativos.

Confusion Matrix Level 0 Validation

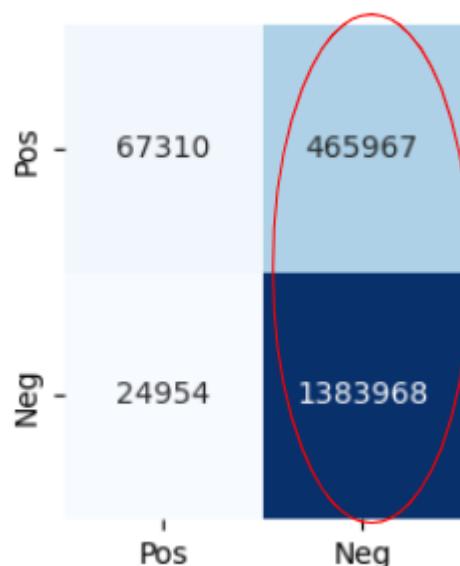


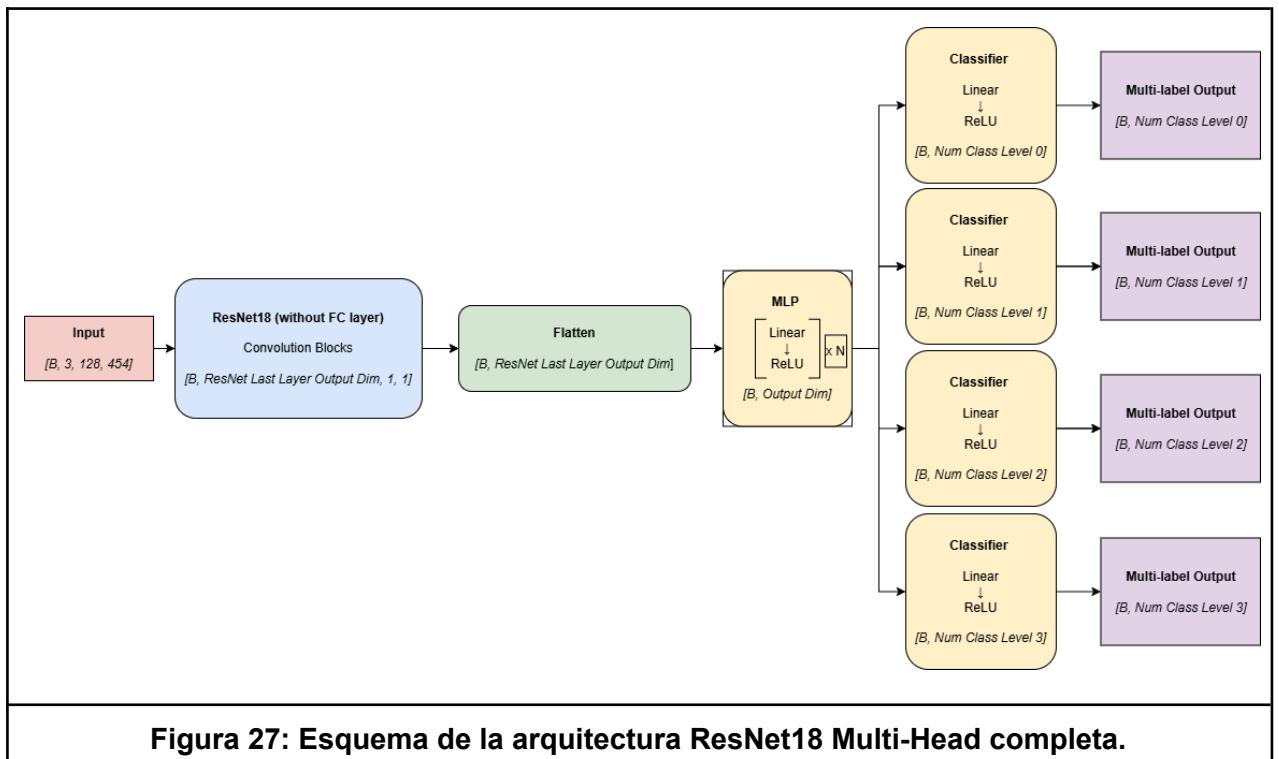
Figura 26: Matriz de confusión global 2×2 para el nivel 0.

Para mitigar estos problemas teorizamos que implementar una loss que tenga en cuenta la estructura del árbol de géneros y que penalice de forma mayor cuando clasifica un género más distanciado en el árbol (nodos hermanos < nodos primos < nodos primos segundos < ...) a diferencia del actual que solo penaliza por correcto e incorrecto.

Y también que la nueva loss penalice muchísimo más por obtener un falso negativo que un falso positivo, esperando que así el modelo deje de clasificar casi todo como negativo y deje de tener tantos falsos negativos.

También sugerimos nuevos modelos que probar que no nos ha dado tiempo a hacer, pero podrían ser interesantes:

- **Fine Tuning ResNet multihead con capa MLP común:** en el modelo multihead cada head entrena su clasificador y todas influyen en el fine tuning de los embeddings, con este modelo esperamos que el MLP común aprenda y procese información común de los embeddings para facilitar el trabajo de los clasificadores.



- **GRU con multihead:** En este trabajo al fin del todo la GRU no ha recibido mucha atención, pero, similar a como se ha hecho con las CNN se podría añadir múltiples cabezas (una por cada nivel de géneros) para ver cómo afecta a los resultados.

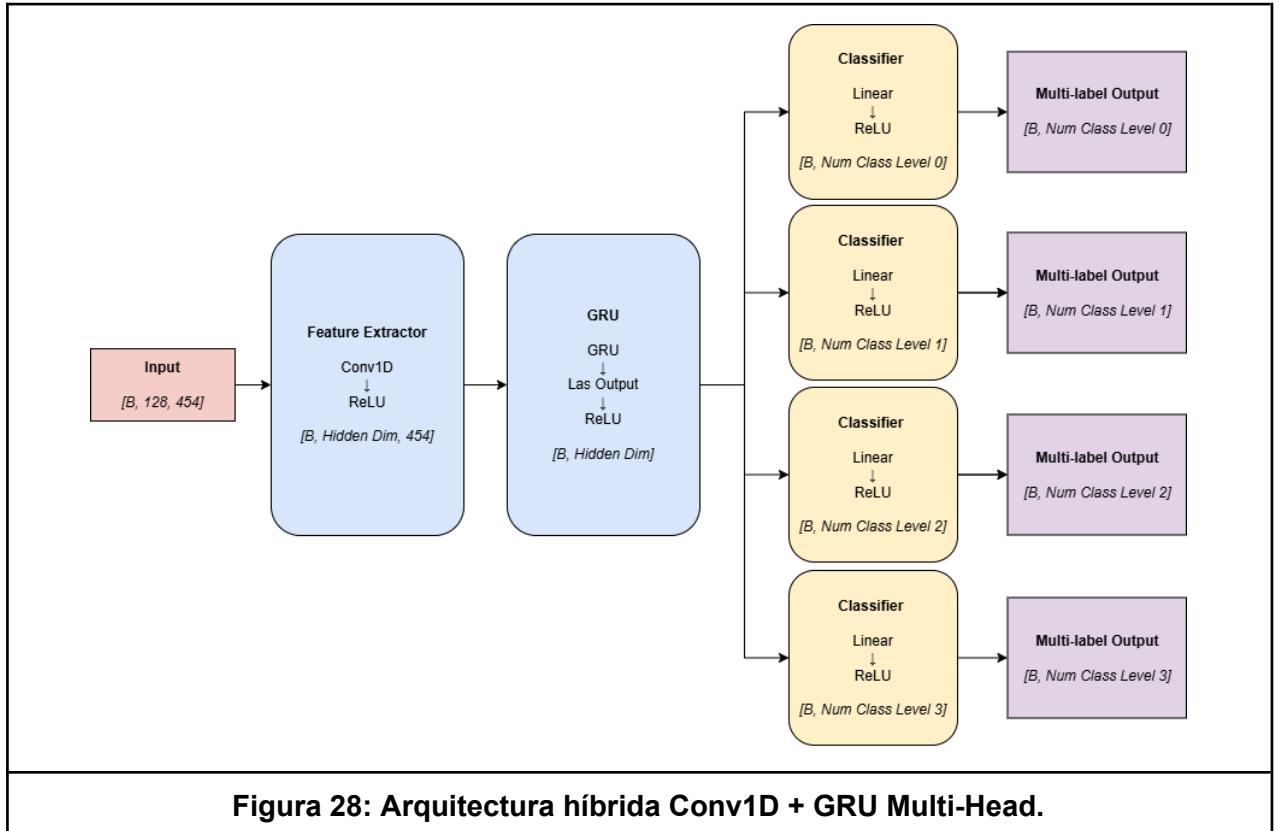


Figura 28: Arquitectura híbrida Conv1D + GRU Multi-Head.

- **CNN con multihead secuenciales:** Teorizamos que este diseño podría aprender de forma innata la jerarquía de géneros de forma mejor y tener en cuenta la decisión de géneros del nivel anterior para la siguiente (ya que sabemos que una canción no tiene un género de nivel 1 si no tiene el género padre de nivel 0 de este).

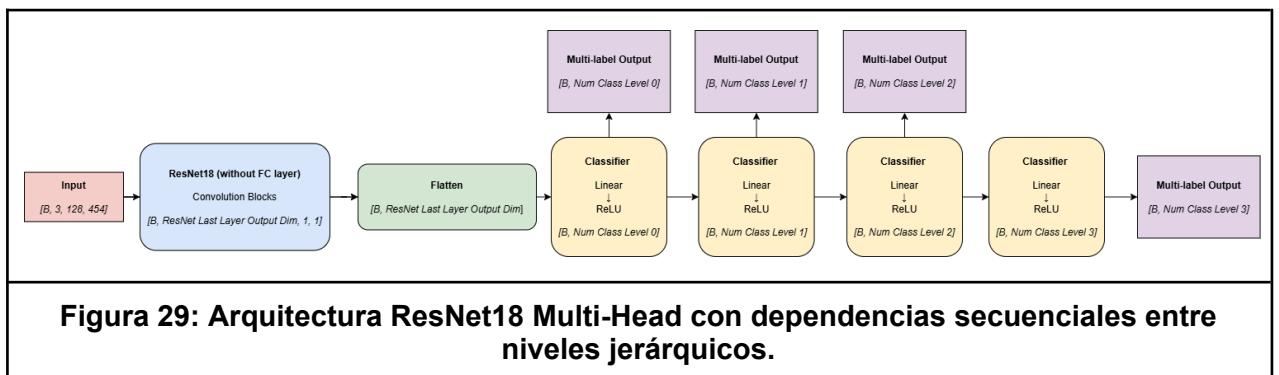


Figura 29: Arquitectura ResNet18 Multi-Head con dependencias secuenciales entre niveles jerárquicos.

- **Attention:** Ya que se ha probado una versión recurrente con GRU porque no probar la versión secuencial similar a este, sería mucho más complicado pero a lo mejor da mejores resultados.

6. Conclusiones

Este trabajo se ha centrado en diseñar un sistema capaz de clasificar géneros musicales a partir del audio en crudo, explorando diferentes representaciones y arquitecturas de redes neuronales profundas. A lo largo del proyecto se evaluaron diversas estrategias y modelos, enfrentándose al reto de un problema multilabel con estructura jerárquica.

Entre las principales conclusiones, destacan las siguientes:

- Los Mel-Spectrogramas han demostrado ser una representación muy útil para extraer información relevante del audio en tareas de clasificación.
- Las arquitecturas preentrenadas como ResNet han superado con claridad a los modelos entrenados desde cero.
- El fine tuning progresivo (descongelando bloques superiores de la red) ha sido clave para mejorar el rendimiento.
- La arquitectura Multihead ha sido especialmente efectiva para abordar la jerarquía de géneros, permitiendo una predicción más estructurada y precisa.

A pesar de los buenos resultados, todavía hay margen de mejora. Como propuestas de mejora o líneas futuras, se podrían considerar:

- Implementar y probar las nuevas arquitecturas discutidas
- Implementar la nueva función de loss discutida que tenga en cuenta la jerarquía de géneros y penalice falsos negativos.
- Más dedicación a las arquitecturas GRU.
- Buscar hiperparámetros con mejores resultados.
- Aplicar K-Fold para medidas más reales de validación.