Mel-Spectogram Music Classification Research

Marko Vukotić
markovukotic@uns.ac.rs
Faculty of Technical Sciences,
University of Novi Sad,
Trg Dositeja Obradovića 6, Novi Sad

Abstract —Background: This paper presents a study on automatic music-genre classification that converts MP3 audio into mel-spectrogram images and trains convolutional neural networks to predict genre labels. Using the Free Music Archive (FMA) as data source, where mel-spectrograms are generated. Mel-spectograms are then normalized and filtered where low-energy/empty segments exist. Custom CNN (~2M parameters), ResNet50V2 (~25M), and EfficientNetB0 (~5.5M) are evaluated under both from-scratch and transfer-learning regimes. Hardware limitations shaped choices for input resolution, batch size, augmentation, sampling, and class-weighting. Across extensive experiments it is observed consistently high top-k performance but poor per-class metrics on the full multi-class task. Restricting the task to well-populated genres or removing tiny classes substantially improves F1 and stability.This research also provides an inference pipeline for unseen MP3s and discusses practical mitigation strategies and directions for future work.

Keywords — mel-spectrogram; music genre classification; convolutional neural networks; transfer learning; class imbalance;

1. Introduction

Music is considered an inseparable part of our culture and tradition [1]. The rapid growth of online social networks and cloud technologies has driven an enormous increase in the demand for online data storage and sharing services. In this context, **Music Information Retrieval (MIR)** systems have become increasingly popular, finding applications in areas such as **music similarity and genre classification, emotion recognition, source separation, acoustic analysis, and automatic music transcription.** [2]. In this paper the problem of automatic music-genre classification is addressed by converting MP3 audio into mel-spectrogram images and training convolutional neural networks to predict genre labels. The goal is to leverage image-based deep models—both custom CNNs and off-the-shelf image architectures (ResNet50V2, EfficientNetB0)—to perform large-scale multi-class classification on the Free Music Archive (FMA) dataset. The work focuses on the practical pipeline from raw audio to image, model training under constrained hardware, and empirical evaluation across multiple modeling and data-preparation choices.

This problem is important because reliable automatic genre tagging supports music retrieval, recommendation, metadata enrichment, and musicological analysis at scale. Using mel-spectrograms exploits advances in computer vision (transfer learning, well-tuned CNN backbones, and image augmentation) while keeping the input representation compact and interpretable. If robust, such systems can help catalog large audio collections and power downstream music applications without manual labeling.

The task is difficult for several reasons. Genre labels are noisy and often ambiguous; many tracks cross styles or contain mixed instrumentation. The FMA corpus (while using top-genre tag) is also severely imbalanced—some genres have tens of thousands of tracks while others have only a few hundred—causing standard training to favor dominant classes. Finally, working at scale introduces engineering constraints: high-resolution spectrograms and large models produce out-of-memory (OOM) failures and unstable long runs on low and mid-range GPUs (GTX 1060, RTX 4070), forcing tradeoffs in input resolution and batch size.

In this paper a complete pipeline is presented (audio → mel-spectrogram → image preprocessing → model training → inference) and an empirical study of the design choices above. Custom CNNs are evaluated (~2M parameters), ResNet50V2 (~25M) and EfficientNetB0 (~5.5M) using resolutions of 224×224, 124×124, and 72×72 and report accuracy, macro/weighted F1, balanced accuracy, and top-k metrics. Key findings include consistently high top-3 accuracy (≈0.76–0.90) but poor macro F1 and balanced accuracy on the full multi-class task (macro F1 ≈0.03–0.05); restricting the task to well-populated genres or removing tiny classes substantially improves per-class performance. This research also documents practical engineering issues (OOM, TensorFlow instability) and provides an inference pipeline for unseen MP3s.

The remainder of this paper is organized as follows. Section 2 reviews related work in music genre classification using spectrograms and convolutional neural networks. Section 3 describes the dataset, preprocessing pipeline, and mel-spectrogram representation used in the experiments. Section 4 outlines the model architectures, training procedure, and implementation details. Section 5 presents the experimental setup, baseline models, and results. Section 6 discusses findings, limitations, and connections to prior research. Finally, Section 7 concludes the paper and suggests directions for future work.

2. Related Work

Mel-spectrograms and other time–frequency representations are standard inputs for audio classification because they capture temporal and spectral structure in a format amenable to convolutional models [3]. Prior music-genre work has applied hand-crafted features and shallow classifiers as well as deep convolutional networks trained on spectrograms; many studies report good performance on smaller or balanced benchmarks but dataset will not always be perfect and if it's imbalanced it would need much more experiments [4]. Recent work adapting image backbones to audio—via transfer learning with ResNet-style or EfficientNet backbones—has shown improved sample efficiency, but these approaches have not been systematically evaluated across multiple input resolutions and large-scale imbalance in the way it is done in this paper [5]. Several papers also explore imbalance mitigation (oversampling, undersampling, class weights, and focal losses), but results are often dataset-dependent and sensitive to augmentation and preprocessing choices [6].

Another interesting study is SpectNet [7] where a learnable spectrogram layer (instead

of fixed mel-spectrogram) is used. This introduced an *end-to-end audio classification* approach based on **learnable spectrogram features**. Unlike traditional mel-spectrograms with fixed filter banks, SpectNet integrates a *trainable gammatone filterbank layer* directly into a CNN architecture, allowing the network to adapt the center frequencies and bandwidths of filters during training. This enables data-driven optimization of the spectrogram representation itself, rather than relying on handcrafted front-end parameters.

Another comparative study by Yigang Meng [8] examined music genre classification using both *deep learning* and *traditional machine learning* techniques. The paper compared a custom Convolutional Neural Network (CNN), VGG16, and XGBoost across two feature types — Mel-spectrograms and Mel-Frequency Cepstral Coefficients (MFCCs) — to analyze their respective strengths. The study found that the XGBoost model trained on MFCCs achieved the highest overall accuracy, outperforming CNN-based architectures on the same dataset. Interestingly, the authors noted that shorter segmented audio windows (e.g., 3 seconds) improved CNN performance by providing finer-grained temporal information, demonstrating that preprocessing and feature design play a crucial role in determining model effectiveness.

This work highlights that while CNNs perform well on image-like spectrograms, classical algorithms such as XGBoost can still compete or even surpass them when applied to carefully engineered audio features.

This research complements this literature by (i) providing a reproducible pipeline for converting FMA MP3s to mel-spectrogram images at multiple resolutions, (ii) empirically comparing a custom CNN and two image backbones both from-scratch and with transfer learning, and (iii) evaluating sampling, class-removal, and class-weighting strategies under realistic GPU constraints.

3. Running Example / Preliminaries

This section describes the dataset, preprocessing pipeline, and representation used throughout the experiments. The FMA dataset is used as a running example to illustrate the conversion of raw MP3 tracks into mel-spectrogram images suitable for convolutional neural networks.

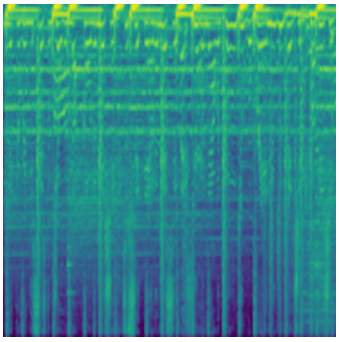
3.1 Dataset

The *Free Music Archive (FMA) Small* dataset contains 8 balanced genres, with 8000 tracks of 30 seconds each, sampled at 44.1 kHz [9]. This dataset was used for the first initial experiments with the model. The *Free Music Archive Large* dataset contains 15 unbalanced genres with over 100,000 MP3 files nearing almost 100GB in size. Each track is stored as an MP3 file with metadata including genre labels. Due to class imbalance and noisy samples, some classes were filtered or downsampled during experiments to stabilize training and avoid bias toward overrepresented genres.

3.2 Mel-Spectogram Representation

To convert each MP3 track into a visual representation suitable for convolutional neural networks, mel-spectrograms are computed with a *librosa* library. A mel-spectrogram is a time–frequency image where the **horizontal** axis represents **time**, the **vertical** axis represents **frequency** bands spaced according to the mel scale (which approximates human pitch perception), and

each **pixel's intensity (color intensity)** encodes the log-scaled energy (loudness/strength) at that frequency and time. Compared to the raw waveform or a standard linear spectrogram, the mel-scaled version compresses high frequencies and yields a more perceptually meaningful, compact representation. Each spectrogram is normalized and resized to fixed image dimensions (tested at 224 × 224, 124 × 124, and 72 × 72 pixels) before being fed into CNN-based classifiers.



Example of mel-spectogram representation

3.3 Audio to Mel-Spectrogram Conversion

Each track was divided into fixed-length segments (typically 3–10 seconds) and transformed into mel-spectrograms using the Librosa library. All audio signals are produced using the Short Time Fourier Transform (STFT) with Melfrequency rather than normal frequency [10]. The short-time Fourier transform (STFT) was computed with a window length of 2048 samples and a hop size of 512, producing 128 mel bands. The power spectrograms were converted to decibel (dB) scale and normalized to the [0, 1] range. The resulting images were resized to 224×224, 124×124, or 72×72 pixels depending on GPU memory constraints.

Mathematically, the mel-spectrogram is computed as:

$$S_{\text{mel}}(t,m) = 10\log_{10}\left(\sum_k M_{m,k}|X(t,k)|^2 + \epsilon\right)$$

where X(t,k) is the STFT of the audio frame and M(m,k) is the mel filterbank matrix.

3.4 Data Splits and Augmentation

Depending on the experiment, the dataset was divided into 80% training, 10% validation, and 10% test **or** 85% training and 15% validation. To improve generalization, data augmentation is applied, including random time shifts, dynamic range scaling, and small Gaussian noise in the log-power domain. Augmentation was applied on-the-fly using TensorFlow's `ImageDataGenerator`.

3.5 Models Overview

Five models were evaluated:

- **Custom CNN:** a lightweight convolutional network designed for grayscale/rgb mel-spectrograms, trained from scratch.
- **ResNet50V2 Transfer Learning:** pretrained on ImageNet, fine-tuned on spectrograms using transfer learning.
- **ResNet50V2 (Scratch):** same architecture trained without pretrained weights.
- **EfficientNetB0 (Scratch):** an efficient architecture without pretrained weights.
- **Custom Model With Residual Blocks:** a smaller residual-style CNN (~250K–300K parameters) used for sampling and class-reduction

experiments.

All models used categorical cross-entropy loss, Adam or AdamW optimizers, mixed precision, and early stopping with validation monitoring. Checkpoints saved the best model per run.

3.6 Evaluation Metrics

I evaluated models using macro-averaged F1 score, balanced accuracy, and top-3 accuracy. Macro-F1 was chosen to mitigate bias from class imbalance. Loss and learning-rate schedules (including ReduceLROnPlateau and cosine annealing) were monitored to ensure training stability.

4. Method / Algorithm / Design

4.1 Overview

I designed and implemented several convolutional models for automatic music-genre classification using mel-spectrogram images derived from the Free Music Archive (FMA) dataset.

The pipeline covers data loading, augmentation, model construction, training with callbacks and mixed precision, and evaluation using standard classification metrics.

4.2 Implementation Details

Run Directory.
Each run creates a unique timestamped folder (`runs/run_YYYYMMDD_HHMMSS`) to store model weights, plots, and metrics for versioning and reproducibility.

GPU Setup.
All experiments are GPU-accelerated (GTX 1060 and later RTX 4070) with CUDA, cuDNN, TensorRT, and TensorFlow (with GPU support). Before training, logs are filtered, sessions cleared, and mixed precision enabled.

Hyperparameters.
Example hyperparameters, that are adapted to the model that is experimented on.

```
TOP_K = 3
BATCH_SIZE = 100
INPUT_SHAPE = (124, 124)
EPOCHS = 100
REGULARIZER = L2(1e-4)
```

4.3 Data Loading and Augmentation

Images are loaded using Keras' `ImageDataGenerator.flow_from_directory`. Augmentation is implemented as a Keras layer sequence. Some of the augmentations applied to images are: random zoom, random contrast, random brightness and gaussian noise.

4.4 Models Architectures

4.4.1 My Custom CNN

This is a **custom** model created from scratch using tensorflow containing ~ 2.5 million parameters.
This sequential model is created with:

- augmentation,
- multiple Conv2D blocks (various filter sizes: 84,144,160,172,194,264...),
- BatchNormalization, Dropout, MaxPooling,
- GlobalAveragePooling2D,
- Dense(128), Dense(224), Dropout(0.5),

- final Dense(num_classes, activation='softmax', dtype='float32') and many Convs use L2 regularizer.

4.4.2 Resnet (Transfer Learning)

ResNet stands for **Residual Neural Network**, introduced by *He et al.* in 2015 in the paper *"Deep Residual Learning for Image Recognition"*[10].

Key idea:
As neural networks get deeper, they can suffer from *vanishing gradients* and *degradation* — meaning accuracy gets worse even though more layers are added.
ResNet solved this by introducing **residual (skip) connections**, where the input to a layer is added directly to its output:

$$y=F(x)+x$$

This lets the network learn only the *residual* mapping $F(x)=y-x$, making optimization easier and allowing **very deep networks** (e.g., 50, 101, 152 layers) to train effectively.

I implemented a transfer learning approach using the **ResNet50** (~ 25 million parameters) architecture pretrained on **ImageNet**. The model was adapted to classify mel-spectrogram images of music genres while leveraging pretrained visual features from natural images.

Model design:
The ResNet50 base (without top layers) was imported with pretrained ImageNet weights and **frozen** (trainable = False) during initial training to retain generic low-level features. Batch normalization layers were kept non-trainable to prevent distribution drift during transfer learning.

A custom classification head was appended:

- Global Average Pooling
- Dense(256, ReLU, L2 regularization)
- Dropout(0.4)
- Dense(num_classes, Softmax)

Fine-tuning stage:
After the initial frozen-phase training converged, the top convolutional blocks of ResNet50 were selectively **unfrozen** to enable deeper adaptation to spectrogram data. Specifically, higher-level feature extractors were made trainable while keeping early convolutional and batch normalization layers frozen to preserve general low-level representations.

4.4.3 Resnet (Scratch)

To evaluate the baseline performance of ResNet50V2 (~ 25 million) without pretrained weights, the architecture is **entirely trained from scratch** on the mel-spectrogram dataset. The model was initialized with random weights and compiled with the same optimizer, learning rate, and regularization settings used in transfer learning experiments, ensuring fair comparison.

4.4.4 Efficientnet (Scratch)

EfficientNet is a **family** of convolutional neural networks (CNNs) introduced by Mingxing Tan & Quoc Le at Google in 2019 in the paper: *"EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks."* [11].

Earlier CNNs (like ResNet, VGG, Inception) were made larger by **increasing depth, width, or input resolution**, but usually in a **manual, unbalanced** way — which made them inefficient.

EfficientNet introduced a *systematic* way to scale a model **uniformly** using a simple **compound coefficient (ϕ)** that balances:

- **Depth** (number of layers)
- **Width** (number of channels per layer)
- **Resolution** (input image size)

The idea is to find a *base network* (EfficientNet-B0), then scale it up efficiently:

- **EfficientNet-B0 → B1 → B2 → ... → B7**
- Each step increases all three dimensions in a balanced way.

EfficientNet models use:

- **MBConv blocks** (Mobile Inverted Bottleneck Convolutions) from *MobileNetV2*,
- **Squeeze-and-Excitation (SE)** attention layers,
- **Swish activation** instead of ReLU (smoother gradients).

This combination gives better accuracy with fewer parameters and FLOPs (computations).

EfficientNetB0 was trained under the same protocol as ResNet50V2 but with its architecture initialized **without pretrained weights** to assess baseline generalization capacity. EfficientNetB0, with roughly **5.5 million parameters**, is significantly smaller and more computationally efficient, allowing the use of **larger batch sizes (up to 200)** and higher image resolution (128 - 224 px).

4.4.5 Small Model With Residual Blocks

To further investigate the effect of class imbalance and model capacity on genre classification, a **custom convolutional neural network with residual blocks** was designed. This lightweight model (~250 k parameters) was used to test both **data sampling** strategies (make an equal playing field for all classes) and **reduced-class experiments** (remove classes that have too few samples). The aim was to determine whether poor generalization in large-scale runs stemmed primarily from architectural limitations or from data imbalance.

Sampling Model

An **external sampling script** was developed to create balanced training sets prior to model training. The script uniformly sampled images from each genre and produced multiple balanced subsets, enabling controlled comparisons between class distributions. A small test split was also included to maintain evaluation consistency. The sampling-based model contained approximately **200 k parameters**, optimized with the same augmentation and mixed-precision setup as in previous experiments.

Three balanced sampling configurations were tested:

- **All classes (15 total):** 200 samples per class were drawn to equalize representation.
- **Classes with 1000+ samples (11 total):** 1000 samples per class were used, excluding underrepresented genres.
- **Classes with 4000+ samples (8 total):** 4000 samples per class were retained to emphasize dominant categories.

This procedure revealed that balancing even a reduced subset of genres significantly stabilized training and improved macro F1 scores compared to the full imbalanced dataset.

Four-Class Residual Model

To further isolate the imbalance effect, the same **residual CNN network** is trained on only **four most populated genres**.

Training ran for **200 epochs** with **early stopping**, using input images of **224×224** pixels and a **batch size of 8**. Despite its simplicity, this model achieved stable convergence and clear class separation, supporting the hypothesis that **dataset imbalance—rather than model architecture—was the main limiting factor** in full-scale experiments.

5. Experiments

This section describes the dataset, implementation details, baselines, evaluation metrics, and results obtained during training and evaluation of multiple models for music genre classification from mel-spectrogram images.

5.1 Datasets and Implementation Details

Dataset.
The experiments were conducted on the **FMA Large dataset**, which consists of ~ 100k tracks (30 s each) across 15 unbalanced genres. Each track was converted to a **mel-spectrogram** using *Librosa* (n_mels=128, hop_length=512) and then split into 10 non-overlapping segments, producing approximately ~ **100,000-175,000 224×224 grayscale or rgb images** (depending on script config).
Additional versions were resized to 124×124 and 72×72 in model scripts for computational efficiency testing.
The dataset exhibited **heavy class imbalance**, with certain genres (e.g., *Rock*, *Experimental*) having over 10× more samples than others.

Preprocessing.
Each mel-spectrogram was normalized to [0, 1] and stored in per-genre folders for easy loading with TensorFlow's ImageDataGenerator. Augmentation was performed on-the-fly with:

- **Random Zoom (±10%)** – to simulate variations in scale and distance.
- **Random Contrast (±20%)** – to introduce variability in lighting conditions.
- **Random Brightness (±20%)** – to account for brightness differences across samples.

Additionally, **horizontal flipping** and **small rotations (±5%)** were considered but later removed, as they were not expected to provide meaningful variation for mel-spectrogram data.

Spectogram Parameters.
Colormaps: 'magma' (purple–orange) and 'green', reflecting the RGB image channels.

Segment Duration: Tracks were split into *n*-second segments, producing variable numbers of images per track.

Amplitude Thresholding: Very quiet segments were discarded to remove uninformative samples.

Empty Segments: Segments with padding (from tracks not divisible by *n*) were removed.

Mel Bands: Number of frequency bins per spectrogram ranged from 64 to 256.

FFT Size: Window length for Fourier transform ranged from 1,024 to 4,096 samples.

Hop Length: Step between successive FFT windows ranged from 4 to 8 samples.

Software Environment.
Experiments were conducted using **Python 3.8.20+**, with dependencies specified in requirements.txt. The main deep learning

framework was **TensorFlow 2.19.0** with GPU acceleration (tensorflow[and-cuda]).

5.2 Baselines and Metrics

Baselines
Several models were evaluated to serve as baselines for music genre classification:

1. **ResNet50V2 (from scratch & transfer learning)** – Large model (~25M parameters), trained with reduced batch size and input resolution due to GPU memory constraints.

2. **EfficientNetB0 (from scratch)** – Moderate-sized model (~5.5M parameters) allowing higher batch size and input resolution, more GPU-friendly.

Sampling & Class Reduction Experiments

Sampling Model: Balanced samples per class using an external script to reduce class imbalance. These experiments used a small custom model (~200k–250k parameters) with residual blocks. This resulted in:

- **All-Classes Training**: Limited samples per class to **200**.

- **Classes with 1000+/4000+ samples**: Selective sampling for larger classes.

- **Four-Class Training**: Focus on the four most populated classes to evaluate the effect of extreme imbalance without sampling.

Custom CNN Models

- **Sequential Model**: ~ 2 million parameter model that used all classes on custom *Sequential* tensorflow setup

Resnet Transfer Learning Model

- ~ 25 million parameter pre-trained Resnet model on ImageNet and fine-tuned

Metrics

Evaluation was performed using multiple metrics to capture overall and per-class performance:

- **Top-K Accuracy** (Top-3)

- **Macro F1 Score** – treats all classes equally

- **Weighted F1 Score** – accounts for class imbalance

- **Balanced Accuracy** – average of recall per class

- **Train/Validation Loss and Accuracy** – monitored during training

- **Per-class Precision, Recall, F1, Support** – detailed confusion analysis

During training also was monitored:

Runtime – wall-clock training time per epoch

Memory usage – GPU memory consumption for large models (nvidia-smi)

Observations from Baselines

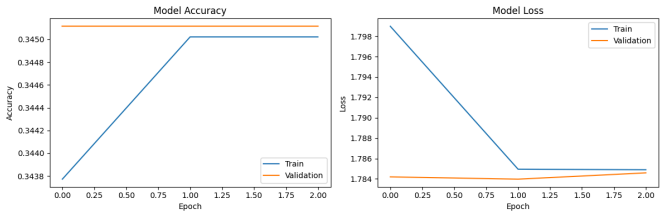
- Models trained on all classes struggled with dataset imbalance; smaller classes were often ignored.

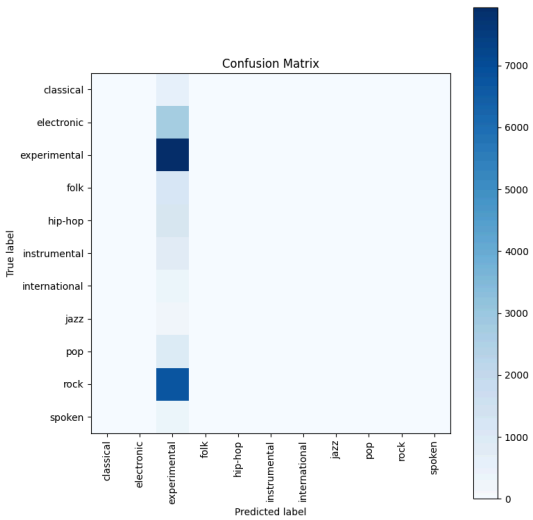
- Sampling and class reduction improved macro F1 and balanced accuracy.
- ResNet50V2 was significantly slower (~2 hr/epoch on RTX 4070), whereas EfficientNetB0 handled higher batch sizes and input sizes with less training time (~9 min/epoch).
- Top-3 accuracy remained relatively high for all models, indicating the model could capture at least some correct classes despite imbalance.

5.3 Results

5.3.1 Baseline Models

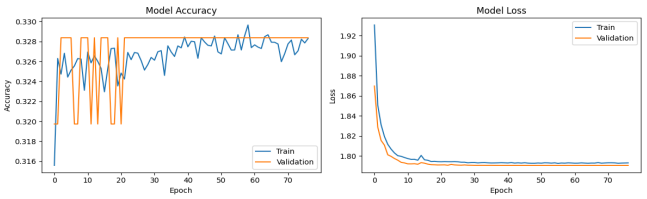
Resnet Results

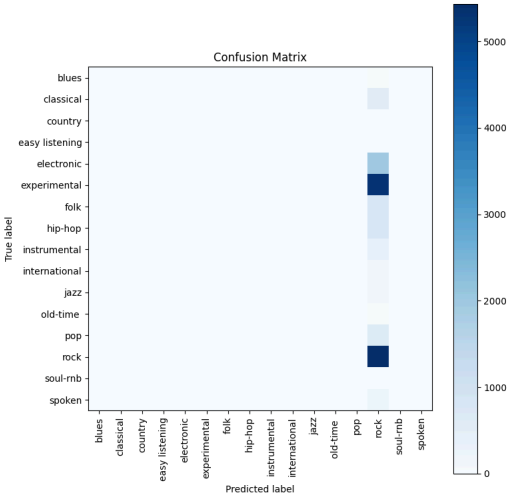




Confusion matrix clearly showing imbalance problem of the dataset. Run shown ran ~6 hours with small batch size [4] and smaller image size 124x124. One epoch took ~ 2 hrs.

Efficientnet Results





Again, same issue. The model is guessing one class with a lot more samples than others and successfully blindly guessing over 30%. This training ran for ~ 11hrs and it's much more forgiving on the GPU than Resnet. Higher batch size [200] and image size [224x224] were used.

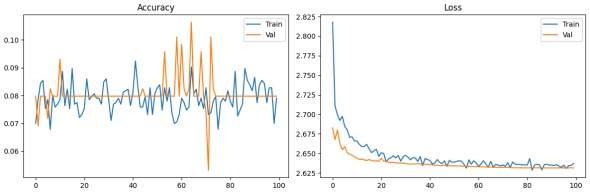
```
=== Metrics Summary ===
Macro F1: 0.03090028204894914
Weighted F1: 0.16235159067342483
Balanced Accuracy: 0.328
Top-3 Accuracy: 0.768914672467972
```

5.3.2 Sampling and Class Reduction Models

All of these experiments use a custom made neural network with residual blocks (~200k parameters).

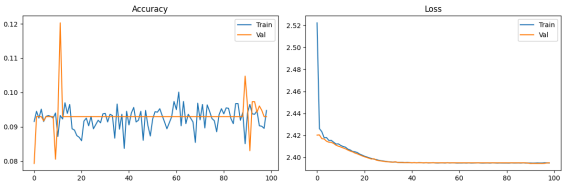
All Classes Training [15] with 200 Samples

200 samples were taken from all classes.



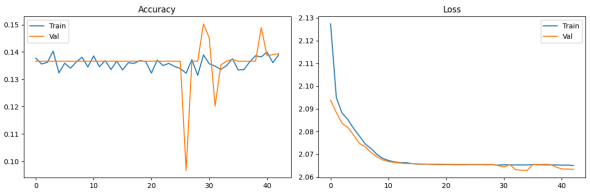
Class Training [11] with 1000+ Samples

1000 samples were taken from four classes that have more than a thousand images.



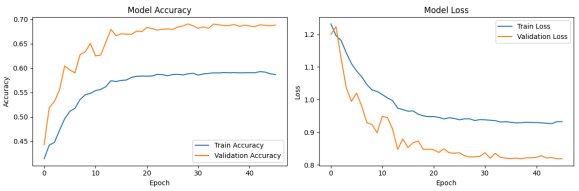
Class Training [8] with 4000+ Samples

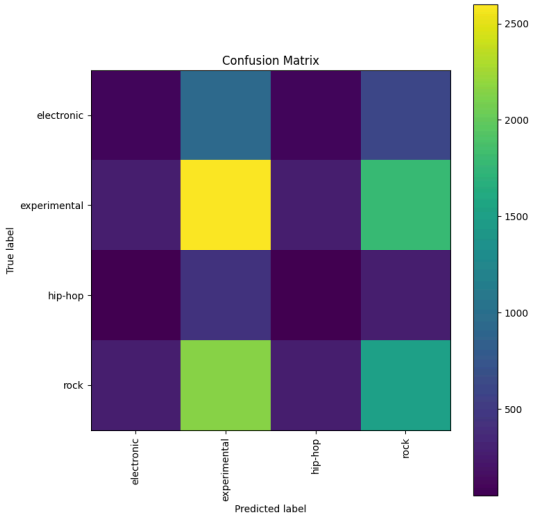
4000 samples were taken from eight classes that have more than four-thousand images.



Four Class Training

Here **four** most populated classes are used to showcase if the dataset imbalance is truly the problem. The model ran on 200 epochs with early stopping with an image size of 224x224 and 8 batch size. This ~ 250k parameter model showed this results:

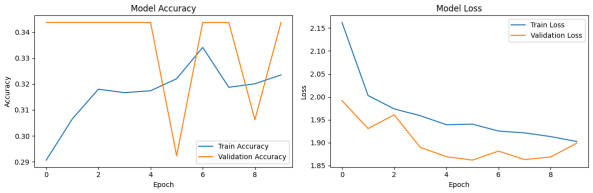


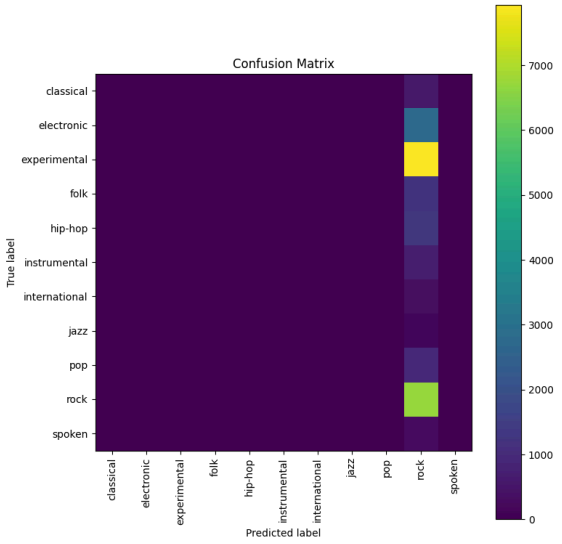


When small classes were removed, the experiment had a lot of improvements, the model had better metrics and also didn't guess only one class but tried with all of them.

```
=== Metrics Summary ===
Top-3 Accuracy (sklearn): 0.9015946107273812
Macro F1: 0.2456439095456165
Weighted F1: 0.3440269725234586
Balanced Accuracy: 0.25209415697447407

=== Train Metrics ===
Train Accuracy: 0.5868
Train Loss: 0.9322

=== Validation Metrics ===
Validation Accuracy: 0.6883
Validation Loss: 0.8188

Per-class P / R / F1 / Support:
electronic: 0.13988919667590027
0.057223796033994336 0.08122235625251306 1765
experimental: 0.42425726412014364
0.5252627324171383 0.46938775510204084 4948
hip-hop: 0.07913669064748201
0.06832298136645963 0.07333333333333333 805
rock: 0.3597036328871893 0.3575671180803041
0.3586321934945788 4209
```

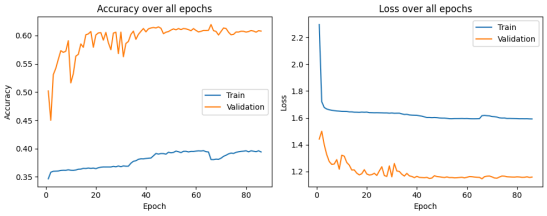
5.3.3 Custom CNN Model

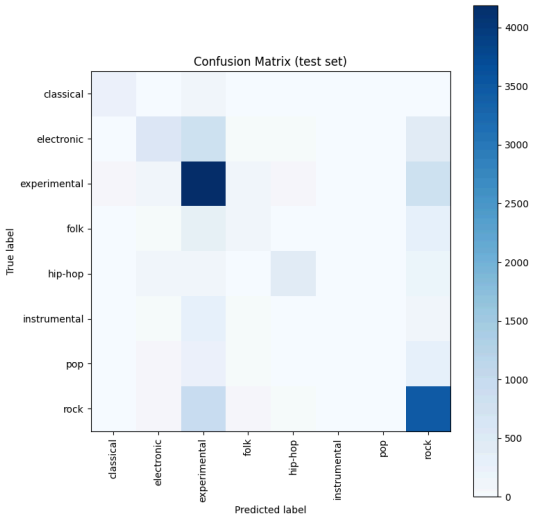




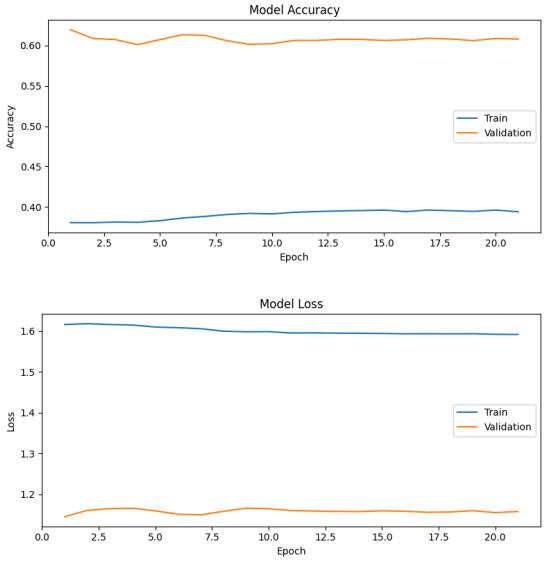
A ~2 million parameter model using custom architecture. Depending on the runs accuracy was between 30% and 50%. **Same issue**, model was trying to guess only one class and had some success with it.

5.3.4 Resnet Transfer Learning Model





And fine-tune metrics:



Transfer learning showed really good results having really good training and validation results and even better results after fine-tuning.

6. Discussion and (Additional) Related Work

The experiments demonstrate several key insights about mel-spectrogram-based music genre classification:

1. **Impact of Class Imbalance** – Models trained on the full FMA dataset struggled to generalize across underrepresented classes. Most architectures (Custom CNN, ResNet50V2, EfficientNetB0) tended to predict dominant genres like Rock and Experimental, which explains low macro F1 and balanced accuracy despite relatively high top-3 scores.

2. **Effectiveness of Sampling and Class Reduction** – Balanced sampling and restricting experiments to well-populated classes significantly improved per-class metrics. This suggests that dataset composition, rather than architecture alone, is the primary bottleneck in multi-class

performance.

3. **Architecture Observations** – Large models like ResNet50V2 provided strong feature extraction but were limited by GPU memory and runtime constraints, leading to smaller batch sizes and lower input resolutions. EfficientNetB0, being more GPU-efficient, allowed higher batch sizes and resolution, resulting in faster training while achieving comparable top-3 accuracy.

4. **Limitations and Failure Modes** –
    - Small classes were frequently ignored.
    - GPU memory constraints and OOM issues limited image resolution and batch size, affecting convergence.
    - Transfer learning helped but did not fully resolve dataset imbalance.
    - Temporal information beyond short spectrogram segments was not captured, which may limit capturing genre-specific patterns across longer tracks.

5. **Comparison to Prior Work** – Previous studies on small or balanced datasets report higher per-class F1, but these results do not generalize to large, highly imbalanced collections like FMA. This study highlights the importance of explicit sampling, class removal, and careful preprocessing when working with large, real-world music datasets.

7. Conclusion

Conclusion.
This research presented a full workflow for transforming audio tracks into mel-spectrogram images and applying convolutional neural networks for music genre classification. Through experiments with both custom CNNs and large pretrained architectures such as ResNet50V2 and EfficientNetB0, the study revealed that dataset imbalance has a far greater impact on performance than architecture complexity. Even though top-3 accuracy remained relatively high, low macro-F1 and balanced-accuracy scores confirmed that minority genres were consistently under-represented in predictions. The results emphasize that proper sampling, augmentation, and class selection are critical when dealing with highly imbalanced music datasets. How good the dataset is, that's how the results are going to be.

REFERENCES

[1] A. Kumar, A. Rajpal and D. Rathore, “Genre classification using feature extraction and deep learning techniques,” in 2018 10th Int. Conf. on Knowledge and Systems Engineering (KSE)

[2] Texture Feature and Mel-Spectrogram Analysis for Music Sound Classification by M. E. ElAlami , S. M. K. Tobar , S. M. Khater, Eman. A. Esmaeil, Computer Science Department-Faculty of Specific Education, Mansoura University, Mansoura, Egypt

[3] Deep Learning for Speech Emotion Recognition: A CNN Approach Utilizing Mel Spectrograms by Niketa Penumajji Department of Computer Science, Kansas State University, Manhattan, Kansas

[4] A Review of Machine Learning Techniques in Imbalanced Data and Future Trends Elaheh Jafarigola, Theodore B. Trafalisa , Neshat Mohammadia; School of Industrial and Systems Engineering University of Oklahoma, 202 W. Boyd St., Room 124, Norman, Oklahoma 73019, USA

[5] Implementing transfer learning for sound event classification using the realised audio database by I. Mohino-Herranz, J. García-Gómez, S. Alonso-Diaz, J.G. Gallegos, F.J. Perez-Sanz, M. Aguilar-Ortega, R. Gil-Pita

[6] A systematic study of the class imbalance problem in convolutional neural networks by Mateusz Buda, Atsuto Maki Maciej A. Mazurowski; Department of Radiology, Duke University School of Medicine, Durham, NC, USA, Royal Institute of Technology (KTH), Stockholm, Sweden and Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA

[7] M. Ansari and T. Hasan, "SpectNet: A Learnable Spectrogram Frontend for Audio Classification," Proc. Interspeech, 2020, pp. 1972–1976.

[8] Y. Meng, "Music Genre Classification: A Comparative Analysis of CNN and XGBoost Approaches with Mel-Frequency Cepstral Coefficients and Mel Spectrograms," Department of Statistics, University of California, Davis, 2023.

[9] FMA: A DATASET FOR MUSIC ANALYSIS by Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, Xavier Bresson, LTS2, EPFL, Switzerland & SCSE, NTU, Singapore

[10] DIFFERENTIABLE ADAPTIVE SHORT-TIME FOURIER TRANSFORM WITH RESPECT TO THE WINDOW LENGTH ACCEPTED | 2023 by Maxime Leiber, Yosra Marnissi , Axel Barrau , Mohammed El Badaoui | INRIA, DI/ENS, PSL Research University and Safran Tech, Digital Sciences & Technologies Univ Lyon, UJM-St-Etienne, LASPI

[11] Deep Residual Learning for Image Recognition by Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun | Microsoft Research

[12] EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks by Mingxing Tan & Quoc V. Le | Google