Proceedings of CONF-SEML 2026 Symposium: Computational Analysis and Modeling in Complex Intelligent Systems
DOI: 10.54254/2755-2721/2026.GU33513

Music Genre Classification with Lightweight CNN

Yuxing Jiang

Shanghai Concord Bilingual School, Shanghai, China
202021060043@concordschool.com.cn

Abstract. The classification of music genres is a fundamental task in music information field. This project investigated how the characteristics of audio, the model architecture, and the selection of input segment length jointly affect the classification accuracy on the FMA small dataset. Lightweight self-designed convolution network CompactResNet trains on 3 second fragments by using these three major feature representations: Mel spectrogram, stft spectrogram, mfcc. Then it is compared with svm baselines with rbf kernels in those same features. This paper has done more research about the segments of length between 3 and 30: The CNN using MFE gets the best result, which is 90.5%; and the SVM with STFT gets 89.7%, so these two kinds of models like features differently: The shorter segment was better than the longer one: Accuracy decreased monotonically from 90.5% at three seconds and 53.8% at 30s, so data enrichment via segmentation is better than adding more time context.

Keywords: Music genre classification, convolutional neural network, mel spectrogram, support vector machine

1. Introduction

Music is a kind of art that organizes sounds in time, it has been developing constantly since ancient times till now and has become a significant part of human life [1]. The development of music has produced different genres which give people various types of music as well as emotional values. So far, there are about 10-20 large categories of music, and the number of small categories is endless [2]. The past few years have witnessed a tremendous growth in digital music platforms and thus people want an easier way to categorize commercial audio which resulted in the birth of machine learning based audio classification technology. Audio classification's process has 2 big steps: getting important features and then doing the classifying. Feature Extraction is when taking raw audio signals and turn that into some time-frequency representation of the sounds. Classification is used by ML and DL models to classify the audio on the basis of extracted features.

In its very beginning, with music genre classification most methods were based on hand crafted audio features in simple ML models. It's clear that the good side of using it. It can be trained very fast, and it has almost no parameters. But the quality of features greatly depends on it, so its generalization is rather poor [3]. Like if the data changes then also its performance is going to get effected [4]. Like classic support vectorchanges,e (svm) or k-nearest neighbor(knn), but they're just not as good as the deep neural networks. And it's even more difficult to put into practice the varied

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.GU33513

and complicated acoustic data on music platforms [5,6]. With the development of this field, the approach to solving the classification problem of music genres has gradually been adopted and refined, first converting the data into a spectrogram and then treating the audio problem as an image classification problem. In [7], the authors compared a self-built convolutional neural network (CNN) model, a classic CNN architecture, VGG16 with fully connected layers, and an XGBoost classifier in traditional machine learning to determine the architecture with the best accuracy. In [8], a parallel CNN ensemble architecture used three different features (Discrete Wavelet Transform, Mel-Frequency Cepstral Coefficients, Short-Time Fourier Transform), and three independent CNNs were constructed on top of them. Capuchin Search Algorithm (CapSA) was used to automatically optimize the hyperparameters of CNNs. Finally, the three models were fused, and softmax was used to complete the classification. In [9], the large ImageNet model ResNet, along with time series models such as Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM), were used to handle the spatial characteristics and temporal variations of music, respectively. Although these solutions are all aimed at improving accuracy, this paper hopes to explore reaching a balance between accuracy and model size to enable the use of self-designed small models on edge platforms.

Based on this idea, this paper has built a lightweight CNN, CompactResNet, to classify music from the Free Music Archive (FMA) dataset [10]. The model's architecture consists of stacked residual blocks with progressively expanding channels, along with SpecAugment data augmentation for regularization. This paper compares three feature extraction layers: Mel Spectrogram (MFE), STFT Spectrogram, and MFCC under the same model architecture and training process to evaluate their performance. To compare traditional machine learning with deep learning, this paper set a well-established Support Vector Machine as the baseline using the same features. Furthermore, this paper investigated the impact of segment length on classification performance using MFE features. In the end, the CompactResNet with MFE features achieved the best accuracy of 90.5% on 8 genres. The SVM baseline achieved 89.7%, 88.6%, and 79.3% on STFT, MFE, and MFCC, respectively, while the STFT-based model achieved 83.8% and the MFCC-based model achieved 82.4% on the same dataset. The 30-second full-track approach only achieved 53.8% due to insufficient training samples.

2. Method

2.1. Dataset description and preprocessing

This paper has used FMA dataset which is given by EPFL [10]. It has 4 different subsets ranging in size from small to full. In terms of economics, this paper is using the FMA Small subset with 8k tracks and almost 27 hours of audio split equally into 8 different genres (1k songs per genre) Each song is thirty seconds long at forty four thousand one hundred andths. FMA Small is comprised of 8 main categories of music: electronic, experimental, folk, hip-hop, instrumental, international, pop and rock.

Preprocessing was made up of 3. First, all the audios were resampled to 22,050Hz and divided by certain length. In different experiments, it ranged from 3s to 15s or even 30s as well, so this changes how many segments there are per audio clip as well as the overall sample size. For feature extraction, this paper compares 3 representations on the same STFT: n_fft=2048, hop_length=512. MFE is the 128-bin log-power mel spectrogram (log power spectrogram). The STFT magnitude spectrogram in log amplitude and 13 MFCCs were also included. Representation set to have a frame numer of certain segemnt lenfth, and this is kept same for each Cnn input dimesion. The obtained samples were categorized into training and testing sets (80/20), with even sampling across genres to preserve genre proportions.

DOI: 10.54254/2755-2721/2026.GU33513

Second, for CNNs, each segment was z-score–normalized after feature extraction. For SVMs, per-frequency statistics were computed and then fitted on the training set using StandardScaler. MFE and STFT were expressed in a logarithmic scale before normalization, while MFCCs were not separately log-scaled in the same way. Third, SpecAugment (random masking of the frequency and time axes) was applied to the training set to reduce overfitting.

2.2. Compactresnet

This paper uses the Residual Neural Network structure and also add in the Skip Connections which helps overcome Vanishing Gradients issue for Deep Learning Models so that the model was trained better and learnt some better more discriminative feature representation using less of a Model. As with audio classification problems, it will be a time-frequency representation. So this paper keeps the low level temporals as well as high level info, this improves the classification. Also has a certain degree of regularization through the form of the remainder. With combined shallow architectures and some simple data augmentations this paper does not want to overfit: so it's fine to use for training if you have little data.

Table 1. CompactResNet architecture (308K parameters)

Layer	Output Shape	Description
Input	(1, 128, 130)	Mel Spectrogram
Conv2d + BN + ReLU	(32, 128, 130)	Stem
MaxPool → ResBlock	(32, 64, 65)	Block 1
MaxPool → ResBlock	(64, 32, 32)	Block 2
MaxPool → ResBlock	(128, 16, 16)	Block 3
MaxPool → GAP → Dropout → Dense	(8,)	Classifier

And this paper doesn't use the bigger resnet which takes from imagenet But this paper used a self-made CompactResNet. The architecture is made up of a convolution, 3 residual blocks that have increasing channels (32 -> 64 -> 128), GAP and then regular dropout classifier giving us 8 class logit. Each residual block contains two 3×3 convolutions with batch normalization and a skip connection (identity shortcut or 1×1 convolution when channels change). The model took single-channel inputs as illustrated and processed the input in the model to generate eight logits, which were then passed through a softmax to obtain genre probabilities. As a baseline, SVM models with an RBF kernel (C = 10) are trained on a statistical feature vector for comparison. Furthermore, the effect of segment duration for CNN+MFE was studied by varying the window length while keeping the network and optimizer fixed. The architecture of the CompactResNet for mel spectrogram input is shown in Table 1, and the structure of one residual block in CompactResNet is shown in Table 2.

Table 2. Structure of one residual block in CompactResNet

Layer / Operation	Description
Input x	Feature map entering the block
Conv2d (3×3) + BN + ReLU	First convolution on the main path
Conv2d (3×3) + BN	Second convolution on the main path
Shortcut from x	Identity or 1×1 Conv + BN if channels change
Add	Element-wise sum of main path and shortcut

DOI: 10.54254/2755-2721/2026.GU33513

Table 2. (continued)

ReLU	Output activation: ReLU(F(x) + s(x))

2.3. Hyperparameter settings for experiments

Table 3 presents the hyperparameter settings employed in the CNN and SVM experiments.

Table 3. Hyperparameter settings for all experiments

Hyperparameter	Value
CNN — Optimizer	Adam (β_1=0.9, β_2=0.999)
CNN — Initial learning rate	1×10^{-3}
CNN — LR scheduler	Cosine Annealing (T_max=50)
CNN — Loss function	Cross-Entropy
CNN — Epochs	50
CNN — Batch size	64
CNN — Dropout rate	0.3
CNN — Data augmentation	SpecAugment (training set only)
CNN — Hardware	Apple MPS GPU
SVM — Kernel	RBF
SVM — Regularization C	10
SVM — Kernel coefficient γ	scale (auto)
SVM — Preprocessing	StandardScaler (fit on training set)
Shared — Train / test split	80% / 20% (stratified by genre)
Shared — Random seed	42
Shared — Evaluation metrics	Accuracy, macro F1, per-genre F1

For the CNN experiments, this project employed the Adam optimizer with an initial learning rate of $\eta_0 = 1\times10^{-3}$, and it used a cosine annealing schedule over 50 epochs:

$$\eta_t = \eta_{\min} + \tfrac{1}{2}(\eta_0 - \eta_{\min})\left(1 + \cos\left(\tfrac{t}{T_{\max}}\pi\right)\right), \quad T_{\max} = 50 \tag{1}$$

where t is the current epoch and $\eta_{min} = 0$.

The loss function was standard cross-entropy, which was applied to the softmax output over C = 8 genre classes:

$$L = -\tfrac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\, log\hat{p}_{i,c}, \quad \hat{p}_{i,c} = \frac{e^{z_{i,c}}}{\sum_{j=1}^{C} e^{z_{i,j}}} \tag{2}$$

where $z_{i,c}$ denotes the logit for sample i and class c, $y_{i,c} \in \{0,1\}$ is the one-hot ground truth, and $\hat{p}_{i,c}$ is the softmax probability.

DOI: 10.54254/2755-2721/2026.GU33513

To regularize the model, a dropout rate of p = 0.3 was implemented prior to the final dense layer. During training, each activation was independently set to zero with probability p, and the remaining activations were scaled by 1/(1−p). Additionally, SpecAugment, which involves random masking along both the frequency and time axes, was applied solely to the training data. All the CNN models were trained with an Apple MPS GPU, and the mini-batch size was 64.

For the SVM baselines, an RBF kernel with C = 10 and γ = scale was used:

$$K(\mathbf{x}_i, \mathbf{x}_j) = exp\left(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2\right), \quad C = 10,\ \gamma = scale \tag{3}$$

where γ = 1/(d · Var(X)) with d being the feature dimension. The input features on the training split are standardized by StandardScaler, and no data augmentation was applied.

All experiments adopted an 80/20 stratified train–test split (random seed 42) to ensure a fair comparison. The assessment is made by analyzing the overall accuracy and the macro-averaged F1 score:

$$\text{Macro-F1} = \frac{1}{C} \sum_{c=1}^{C} F_{1,c}, \quad F_{1,c} = \frac{2 P_c R_c}{P_c + R_c} \tag{4}$$

where P_c and R_c are the precision and recall of class c, as well as per-genre F1 scores and confusion matrices.

3. Results and discussion

3.1. Feature comparison with CNN (3-second segments)

Table 4 presents the classification accuracy of CompactResNet trained on three input features using 3-second segments.

Table 4. CNN accuracy across three input features (3 s, 8,000 samples)

Feature	Input Shape	Accuracy	Epoch Time
MFE	1×128×130	90.5%	~22 s
STFT	1×1025×130	83.8%	~85 s
MFCC	1×13×130	82.4%	~1 s

MFE achieved the highest accuracy at 90.5%, leading in seven of eight genres, with Electronic (F1 = 0.95) and Hip-Hop (0.93) performing best. STFT lagged at 83.8%, with Pop suffering the most (F1 = 0.69 vs. 0.82 for MFE). MFCC reached 82.4% but trained in only ~1 s per epoch.

With Mel scale this paper has 1025 linfreq compressed to 128 meaningfull percievable bands and also filtersout high freq noise but keep the genre associated structre. The CNN picks it up with some local convos. STFT is fully in frequency resoulution but it's input was 8x bigger, which caused overfitting on 308K params. This paper needs to compress(13,of 128 mfcc_coeff). The CNN does not have many learning capabilities here because of this.

DOI: 10.54254/2755-2721/2026.GU33513

3.2. Feature comparison with SVM (3-second segments)

The SVM result with the statistic features(mean, std, max, min) of every representations are shown in Table 5.

Table 5. SVM accuracy on the three feature set of 3s, 8000

Feature	Dim	Accuracy	Time
STFT	4,100	89.7%	50 s
MFE	512	88.6%	5 s
MFCC	52	79.3%	1 s

The STFT is 89.7% which beats MFE (88.6%), the inverse ranking from CNN STFT led for 5 out of the 8 genres, namely Experimental(0.91 vs 0.86)and Folk (0.91 vs 0.89) The worst performer is MFCC with 79.3%, and Pop was only F1=0.68:

Different from CNN,SVM works on 1D statistical vectors rather then on a 2d spectrograms. STFT's 4,100 dimensions kept all its frequencies, so it gave the RBF kernel more info to work with. On the other hand, Mel compressing MFE is helpful to CNN which removes high freqency statistical information but still can be used with kernels. This is the inverted relation that the optimal feature depends on model type.

3.3. CNN vs. SVM comparison

In Table 6, this paper compares the two models on every feature:

Table 6. CNN vs. Accuracy of SVM (3s segemnts)

Feature	CNN	SVM
MFE	90.5%	88.6%
STFT	83.8%	89.7%
MFCC	82.4%	79.3%

CNN had better performance than SVM on MFE(+1.9%);MFCC(+3.1%) ;STFT(+5.9%);SVM had better performance than CNN on STFT(+5.9%); No one model dominated them all.

MFE's spatial pattern was obtained by CNN,and SVM used the high dimensional statistical vector of stft with rbf kernel. That meant it is required to take into account which models and features went well together; CNN worked best with MFE, and SVM worked best with STFT.

3.4. Effect of segment length (CNN + MFE)

Table 7 shows the accuracy for different segments duration with all others training condition being fixed.

Table 7. The impact of segment length on the accuracy of CNN + MFE(800track)

Duration	Segments	Accuracy	Macro F1
3 s	8,000	90.5%	0.90
5 s	4,312	87.5%	0.88

DOI: 10.54254/2755-2721/2026.GU33513

Table 7. (continued)

6 s	3,513	85.5%	0.86
10 s	1,915	72.8%	0.73
15 s	1,116	62.1%	0.63
30 s	640	53.8%	0.52

From 90.5% (3s) it went down to 53.8% (30s). Pop was the most fragile one, F1: 0.86 -> 0.30 and Hip-Hop is the strongest one, F1: 0.94 -> 0.70. The 30s model is very much over-fitted: train 73.9%, test 53.8%.

The main thing was data volume; 3 s segment gave us 8k training sample out of 800 tracks, this is important for augmenting 308K parameter model. Longer segs reduced samples, so overfitting occurred. Between 3s & 5s(+)3.0%vs between 5sand +10%the result was negative, which indicated the diminishing returns, so it is known that by having between 3-5 second, enough time can be obtained to capture every genre-defining moment and still get a decent sample size of it. Global average polling also penalized long sequences for compression of hundreds of time steps into single things.

4. Conclusion

This project looks at how feature representation, models used, and segment length effects Music Genre Classification with the FMA Small Dataset. This paper is comparing the lightweightCompactResnet (308k) with a svm on 6 segments, lengths being from 3 – 30 also with three different kinds of features: mfe stfft and mfcc which equals a total amount of 11 combinations.

The best performer is the CNN with MFE feature, 90.5% acc. It used 3sec segments so mel scales may have worked especially well with convolutions. More interestingly, the SVM with STFT features got similar result which is 89.7%. In contrast to CNN that works well for structured and imagelike representation, SVM prefers a high dimension statistical representation.

Looking at just the pieces themselves there's definitely a decrease with the longer segments - it goes from 90.5% accuracy at 3sec down all the way to 53.8% at 30sec. In other words, more is better if it's training samples but less is better for long temporal context. In every experiment Pop was the one that proved to be the most difficult and Hip hop was the easiest.

There are some things worth noting as limitations: It was done with somewhat less than half as many tracks, 800. When scaled up to the full 8000-track dataset, it was at 81.3% after 100 epochs and therefore there is currently insufficient capacity in the model for larger sized datasets. And this paper used a single train – test split instead of doing k - fold cross validation which is going to hurt us.

In the future work, it is possible to increase Model capacities, add attention to it and try using other datasets like GTZAN, or even explore into multitask learning, such as genres but also instruments.

References

[1] Kamien, R. (2018). Music: An appreciation (13th ed.). McGraw-Hill Education.
[2] Brackett, D. (2016). Categorizing sound: Genre and twentieth-century popular music. Univ of California Press.
[3] Scaringella, N., Zoia, G., & Mlynek, D. (2006). Automatic genre classification of music content: A survey. IEEE Signal Processing Magazine, 23(2), 133–141.

DOI: 10.54254/2755-2721/2026.GU33513

[4] Sturm, B. L. (2013). The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use. arXiv: 1306.1461.
[5] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5), 293–302.
[6] Li, T., Ogihara, M., & Li, Q. (2003). A comparative study on content-based music genre classification. In Proc. 26th ACM SIGIR, 282–289.
[7] SuriyaPrakash, J., & Kiran, S. (2022). Obtain better accuracy using music genre classification systemon gtzan dataset. In 2022 IEEE North Karnataka subsection flagship international conference (NKCon) (pp. 1-5). IEEE.
[8] Zhang, Y., & Li, T. (2025). Music genre classification with parallel convolutional neural networks and capuchin search algorithm. Scientific reports, 15(1), 9580.
[9] Zhang, J. (2023). Music Genre Classification with ResNet and Bi-GRU Using Visual Spectrograms. arXiv: 2307.10773.
[10] Defferrard, M., Benzi, K., Vandergheynst, P., & Bresson, X. (2017). FMA: A Dataset for Music Analysis. arXiv: 1612.01840.