Coursework 1 – Music Genre Classification

Aris Karalis - 36622966

Abstract

This report explores deep learning approaches for Music Genre Classification using the GTZAN dataset, evaluating models based on Mel spectrogram images and MFCC sequences. Convolutional Neural Networks (CNNs) processing spectrograms demonstrated superior performance compared to a Fully Connected Network and LSTM models using MFCCs. The best results were achieved by a CNN variant incorporating Batch Normalisation and RMSprop. While LSTM models exhibited early overfitting, augmentation with GAN-generated MFCCs unexpectedly decreased performance, likely because of the challenges of training generative models on limited datasets and computational power limiting the quality of synthetic features. This work highlights the effectiveness of well-tuned CNNs for this task on Mel spectrograms and the difficulties of data augmentation via GANs on small datasets.

1. Introduction

Music Genre Classification (MGC) is a core task in Music Information Retrieval (MIR) for applications such as music recommendation and library organisation [1]. This report details the implementation and comparison of deep learning models for MGC using the GTZAN [2, 3] dataset. Additionally, the development of a Conditional Generative Adversarial Network based on genre is explored for the generation of music samples.

Six architectures were investigated:

1. Fully Connected Neural Network (FCNN)
2. Convolutional Neural Network (CNN)
3. CNN with Batch Normalisation (CNN+BN)
4. CNN+BN with RMSprop optimiser
5. Long Short-Term Memory Network (LSTM)
6. LSTM with C-GAN Generated Samples

All models focused on classification accuracy.

2. Methodology

2.1. Data Preparation & Features

The GTZAN dataset includes 1000 30-second music files and Mel spectrogram images with 100 songs for 10 different genres. The audio files have been used for all 6 architectures and were split stratified (70% training, 20% validation and 10% testing datasets). Then different preprocessing steps were implemented for different architectures.

2.1.1. Models 1 - 4

The first four models utilised the same preprocessed dataset. Converting and resizing of the pre-existing coloured images leads to loss of information, that is why new greyscale Mel spectrograms were generated in 180×180 pixel size with the Librosa library [4]. Then data augmentation was applied on the images, more specifically, random crop where random parts of the. Data augmentation (random crop, affine, erase) was applied to the training set. Finally, the data were normalised.

2.1.2. Models 5 & 6

For the LSTM-based architectures of models 5 and 6 the raw audio waveforms were utilised. The .wav files were segmented into 3-second samples. A sliding window method was implemented with 50% overlap to ensure comprehensive feature capture across time.

The input features chosen for the LSTM models were sequences of Mel-Frequency Cepstral Coefficients (MFCCs), which are suitable for LSTMs as they have proved that are efficient at representing audio timbre [5]. Utilising the Librosa library, 13 MFCCs were extracted independently from each 3-second audio segment. These features were then transposed to align with the conventional LSTM input shape of `[time_steps, features]`. Following extraction, the features were structured into train(70%), validation(20%) and test(10%) sets corresponding to the original data split. Finally, the features were standardised before being fed to the model for training, validation and testing.

2.2. Model Architectures & Training

All six model architectures were implemented with PyTorch [6] and trained on an M3 Pro chip with MPS acceleration. These models were categorised based on their feature input. Models 1 through 4 utilised the preprocessed Mel spectrogram images, while models 5 and 6 were trained and evaluated on MFCC features extracted from raw audio. The metrics of accuracy, cross entropy loss, micro-F1 and macro-F1 were calculated for all models to ensure comparable assessment.

2.3. Models 1-4: Mel Spectrogram-based Classifiers

These models were designed to classify music genre directly from 180×180 greyscale Mel spectrogram images. All models used the same training function with Cross-Entropy Loss and a learning rate of 1×10^{-5}. Furthermore, these models were trained with 50 and 100 epochs each.

Model 1: Fully-Connected Network (FCNN). The first architecture was a standard fully connected neural network. It consisted of an input layer flattening the $180 \times 180 = 32400$ pixels, followed by two hidden layers with 512 and 256 neurons respectively, both including ReLU activation functions. The network concluded with a 10-neuron output layer corresponding to the genre classes. Training employed the Adam optimiser [7] because it has shown great performance.

Model 2: Convolutional Neural Network (CNN). A basic CNN was implemented as the second model. It features four convolutional layers with increasing output channels (32, 64, 128, 256), using 3×3 kernels and `padding=1`. ReLU activation follows each convolutional layer and max pooling layers (2×2) were applied after the second and fourth convolutional layer to reduce output dimensions. The output from the convolutional blocks is then flattened and passed through a 256-neuron fully connected layer with ReLU activation before the final 10-class output layer. The Adam optimiser used for training.

Model 3: CNN with Batch Normalisation (CNN + BN). Building upon the CNN from model 2, model 3 integrates Batch Normalisation (BN) layers. BN was placed after each convolutional layer and hidden fully connected layer but before each ReLU activation function. This aimed to improve training stability and accelerate convergence by normalising layer inputs. The Adam optimiser was used for training in this model too.

Model 4: CNN with BN and RMSprop. Model 4 has identical architecture with model 3 with only distinction the optimiser. Model 4 was trained using the RMSprop optimiser instead of Adam, this way we can compare the differences in performance on this specific network architectures.

2.3.1. Models 5 & 6: MFCC-based Classifiers

Models 5 and 6 processes sequential MFCC features from the audio data as described in Subsection 2.1.2. The input of these models consisted of sequences of 13 MFCC features per time step, derived from 3-second audio segments.

Model 5: Long Short-Term Memory Network (LSTM). The fifth model employed an LSTM network with two bidirectional LSTM (BiLSTM) layers, each configured with a hidden size of 128 units. Dropout ($rate = 0.3$) was applied between the LSTM layers. The concatenated output from the last time step of the bidirectional final LSTM layer was passed through a dropout layer of the same size and then to a fully-connected layer for the final 10-class classification. The input dimension to the LSTM was 13, matching the number of MFCC features, and the sequence length was determined by the number of time speps in a 3-second segment. Training used Cross-Entropy Loss and the Adam optimiser with a learning rate of 1×10^{-3} and a batch size of 32. A maximum training duration of 50 epochs was set, but early stopping with a patience of 10 epochs based on validation loss was implemented to automatically select the best performing model state and prevent overfitting.

Model 6: LSTM with cGAN Augmentation. Model 6 focused on the impact of data augmentation generated by a Conditional Generative Adversarial Network (cGAN) on classification performance. First, a cGAN consisted of a Generator and a Discriminator was trained to generate MFCC sequences conditioned on genre labels. The Generator took a noise vector and genre embedding to produce synthetic sequences while the Discriminator learned to distinguish these from real MFCCs. This GAN was trained adversarially using Binary Cross-Entropy Loss and the Adam optimiser with $LR = 2 \times 10^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.999$ for 50 epochs. Then MFCC sequences were generated corresponding to 100 3-second audio segments for each of the 10 genres with the trained cGAN to form an augmented dataset. Model 6, an identical LSTM to the one of model 5 but was trained on both the training set with the extracted MFCC features from the GTZAN dataset and the augmented dataset. The hyperparameters used to train model 6 were also identical to model 5 to isolate the effect of the data augmentation.

3. Results

Classification performance was evaluated on the test set. Model 6 (GAN) was evaluated qualitatively based on generated audio samples.

The key performance metrics for all implemented models and training configurations are summarised in Table 1 and an example of a generated MFCC is shown in Figure 1.

Table 1. Performance Summary

Model	Epochs	Tr Acc(%)	Val Acc(%)	Te Acc(%)	μ-F1	m-F1
Net1	50	86.84	55.28	51.49	51	50
Net1	100	89.41	57.29	52.49	52	52
Net2	50	89.56	63.32	61.39	61	60
Net2	100	91.99	66.33	63.37	63	61
Net3	50	92.70	73.37	66.34	66	65
Net3	100	95.42	**74.88**	69.31	69	69
Net4	50	92.56	66.83	64.36	64	61
Net4	100	**94.43**	71.35	**74.26**	**74**	**72**
Net5	17	92.45	63.11	67.60	68	67
Net6	19	92.72	66.17	62.85	66	66

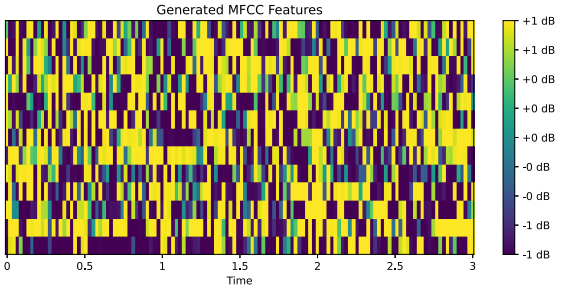

Figure 1. cGAN Generated MFCC

4. Discussion

Evaluation of the six implemented models reveals key insights into effective strategies for music genre classification on the GTZAN dataset, as summarised by the metrics in Table 1. An initial observation is the significant performance advantage of models utilising convolutional layers for processing spectrograms images compared to the basic Fully Connected Network (Net 1). Net 1 variants achieved test accuracies around 50% with not a large difference between 50 to 100 epochs runs, demonstrating the limitations of no spatial information through flattening. In contrast, CNNs (Net 2 3 and 4) consistently yielded higher test accuracies, ranging from 63% to 74%, highlighting the CNNs ability to effectively capture spatial information and patterns within the Mel spectrogram representations.

Within the CNN models, different refinements yielded better results. The addition of Batch Normalisation (BN) in Net 3 improved the model's accuracies over all metrics by 3-10% and especially Test accuracy increased by 10% at 50 epochs and 9% at 100 epochs, showing the importance of BN. Then comparing Net 3 which has Adam

optimiser and Net 4 which has RMSprop optimser we can see that in net 4 test accuracy, μ-F1 and m-F1 improved by 6%, 5 and 3 respectively but also the training accuracy decreased by a small margin of 1% meaning that the model generalises slightly better and does not learn unnecessary features from the training data. The Net4-100 variant achieved the highest test accuracy (74.26%), μ-F1 (74), and m-F1 (72) among all models. Training with 100 epochs showed better results for all models from 1 to 4, indicating that the models benefited from extended training within this setup.

The MFCC-based BiLSTM approach (Net 5) achieved a test accuracy of 67.6%, placing it above the model 1 FCNN and basic model 2 CNN, but below the enhanced CNNs of models 3 and 4. A notable characteristic of both Net 5 and 6 was the early termination of training by the early stopping mechanism which was triggered at epochs 17 for Net 5 and epochs 19, well before the maximum defined epochs. This suggests that, while training accuracy continued to rise the validation loss stopped improving and likely began to increase rapidly, showing a significant degree of overfitting to the training data.

Model 6, which trained the LSTM on original and GAN-augmented MFCC data, performed relatively worse than the basic Net5 BiLSTM, achieving 62% test accuracy compared to the slightly higher Net5's accuracy. The that Net 6 trained slightly longer (2 more epochs) before early stopping might suggest that the GAN-augmented training set offered a marginal reduction in overfitting compared to the original data, but not enough to significantly improve overall generalisation. The generated MFCC features as visually illustrated in Figure 1, when compared to original samples appear less structured and potentially fail to capture the critical features necessary for accurate genre discrimination. This limitation in the quality of generated features is most likely dues to the simple architecture of the cGAN and the limited size of the GTZAN dataset as well as the limited compute power available for this experiment, as a longer training of more than 200 epochs would probably offer better results. As a result, the synthetic samples likely acted more as noise than beneficial augmentation, limiting the classifier's performance.

5. Conclusion

In summary, this study evaluated six deep learning models for music genre classification on GTZAN. CNNs on Mel spectrograms significantly outperformed the FCNN. The Net 4 trained with 100 epochs CNN, utilising Batch Normalisation and RMSprop, achieved the highest test accuracy (74.26%). Both LSTM variants (Net5 and Net6) overfit severely, stopping training early. Notably, GAN-augmented training (Net6) worsened LSTM performance compared to Net5 (62.85% vs 67.60%), indicating the GAN failed to generate beneficial features, likely due to the dataset's small size limiting generation quality. Thus, well-tuned CNNs proved most effective for this task on GTZAN.

References

[1] Pingping Wu, Weijie Gao, Yitao Chen, Fangfang Xu, Yanzhe Ji, Juan Tu, and Han Lin. An improved ViT model for music genre classification based on mel spectrogram. *PLOS One*, 20(3):e0319027, March 2025. 1

[2] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, July 2002. 1

[3] GTZAN Dataset - Music Genre Classification. 1

[4] Brian McFee, Colin Raffel, Dawen Liang, Daniel Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and Music Signal Analysis in Python. pages 18–24, Austin, Texas, 2015. 1

[5] Yinhui Yi, Xiaohui Zhu, Yong Yue, and Wei Wang. Music Genre Classification with LSTM based on Time and Frequency Domain Features. In *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*, pages 678–682, April 2021. 1

[6] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. October 2017. 1

[7] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. arXiv:1412.6980 [cs]. 1