Music Genre Classification

Abdul Wahid Rukua
University of Souhampton
36628611
awr1u24@soton.ac.uk

1. Data Preprocessing

The dataset used in this experiment is the precomputed spectrogram images and original audio from the GTZAN dataset. In accordance with the coursework guidelines, the spectrogram images were resized to 180×180 pixels, with no additional preprocessing applied. Regarding the original audio, it was found that there is one corrupted audio, and the number of datasets drops to 999, just as the spectrogram images. As for the preprocessing step, the audio was divided into 10 segments and normalised to 660,000 to avoid any mismatch due to the different lengths of the waveforms. Subsequently, the audio features, such as MFCC, spectral contrast, spectral centroid, and chroma, are extracted. The feature extraction process was carried out using the following configuration: n_ftt = 1024, hop_length = 4084, and centroid = False were used for all audio extraction processes. This preprocessing pipeline was inspired by the implementation on GitHub and Kaggle. Both of these datasets were divided into 70% for training, 20% for testing, and 10% for testing.

2. Setup Overview and Architecture

In this experiment, hyperparameter tuning was performed manually and guided by commonly used values in deep learning literature. The primary focus, however, was on examining the influence of different learning rates rather than conducting exhaustive hyperparameter optimisation, as emphasised by [2]. As for the learning rate values, 0.01, 0.001, 0.0005, and 0.0001 were evaluated with the combination of two widely used optimisers: Adam and RMSprop. These configurations were explored using a trial-and-error approach to observe their effects on training behaviour and model performance. The following section presents the architectural details of the six models evaluated in this study.

2.1. Fully Connected Layer (Net1)

A multilayer perceptron with a flatten input layer, two hidden layers (512 neuron + ReLU), and an output layer (10 neuron). This larger number of hidden units was chosen to enable the model to capture more features, as stated by [2]

2.2. CNN (Net2)

The following model architecture was designed according to coursework guidelines, with parameter choices informed by [1]. The use of a 3×3 kernel and a small initial number of filters was adopted based on their demonstrated effectiveness.

Layer	Output Shape	Kernel / Stride / Pool
Input	(3, 180, 180)	-
Conv2d + ReLU	(32, 180, 180)	3×3 / 1
Conv2d + ReLU	(32, 180, 180)	3×3 / 1
MaxPool2d	(32, 90, 90)	2×2 / 2
Conv2d + ReLU	(64, 90, 90)	3×3 / 1
Conv2d + ReLU	(128, 90, 90)	3×3 / 1
MaxPool2d	(128, 45, 45)	2×2 / 2
Flatten	(259200)	-
Linear + ReLU	(256)	-
Linear	(10)	-

Table 1. Architecture details of Net2 model.

2.3. CNN with Batch Normalisation (Net3)

An extension of the base CNN architecture in (2.2), with batch normalisation layers added after each convolutional layer.

2.4. RMSprop Optimizer Variant (Net4)

The architectures described in (2.3) is trained using the RMSProp optimizer.

2.5. RNNs (Net5)

A two-layer stacked LSTM architecture with 128 hidden units was employed in a many-to-one setting. Observations from the training of previous models indicated severe overfitting, which motivated the application of a dropout rate of 0.3 to improve generalisation. Furthermore, fluctuations in validation performance were used as a signal to trigger

early stopping during training. This is also supported by [4], which explains when the model's training should be terminated. In this model, `patience=7` was applied after experimenting with `patience=4,5,6`.

2.6. Data Augmentation Via GANs (Net6)

The architecture described in Section 2.5 is trained for audio synthesis using a Conditional GAN (CGAN). CGAN are employed to model the conditional audio distribution across different musical genres. To improve computational efficiency, the training dataset consists of temporally segmented audio clips. The generator consists of a 10-dimensional genre label embedding, which is concatenated with a 100-dimensional noise vector to form a 110-dimensional input. This input is passed through three fully connected layers with 512, 1024, and 2048 units, each activated by ReLU functions. The final output layer comprises 66,000 units, representing the synthesised audio waveform. The discriminator takes the real or generated audio data, concatenated with the same label embedding, with a total input size of 66,010 dimensions. This input is passed through two leaky ReLU-activated layers (1024 and 512 units), followed by a final classification layer. No hyperparameter tuning was conducted in this training. Furthermore, the training procedure fixed the number of epochs at 100, without dynamic balancing between the generator and discriminator. The generated audio segments underwent the same preprocessing pipeline as the real audio before being concatenated.

2.7. Metrics

The test set was evaluated using four standard metrics: accuracy (overall correct predictions), precision (correct positives among predicted positives), recall (correctly identified actual positives), and F1-score (harmonic mean of precision and recall).

3. Result

3.1. Training Behavior and Convergence

Initial experiments were conducted on four network configurations (Net1–Net4), each evaluated at 50 and 100 training epochs. Models were trained using RMSprop and Adam optimisers with a learning rate of 0.0001. While almost all models achieved over 99% training accuracy, validation accuracy remained lower, approximately 40% to 50%. These findings suggest that these models experienced an overfitting. Adam-trained models have shown more stable performance than RMSprop-trained models. For instance, Net3 at 50 epochs with Adam showed the most consistent performance, maintaining a low and stable training loss (approximately 0.005–0.01). In contrast, Net4 experienced an extreme loss and accuracy spike at several moments. Subsequent experiments on Net5 and Net6 also revealed similar tendencies; however, both networks achieved significant improvements. Specifically, Net5 with Adam achieved a validation accuracy of 83.7% at epoch 43, before early stopping was triggered due to no improvement over seven epochs.

3.2. Performance evaluation

As shown in Table 2, models trained using the Adam optimiser consistently outperformed those using RMSprop. For instance, in configurations such as Net5 and Net6, which achieved balanced and high scores across all metrics (precision, recall, and F1-score all at 85%). These results indicate strong discriminatory power and generalisation capability. In contrast, RMSprop-trained models demonstrated inferior performance, with F1 scores peaking at 82% (Net5) but dropping as low as 35% in suboptimal configurations (Net1), reflecting reduced stability and convergence.

4. Discussion

4.1. Perfomance

Based on the experiment above there are several key concerns that can simplified as follows:

1. **Inductive bias affects model performance**

   The performance of Net 1 compared to Nets 2, 3, and 4 in similar tasks indicates that CNNs are more robust for handling image datasets. As discussed by [3], this advantage stems from the absence of suitable inductive biases in fully connected layers. While CNNs are designed to capture local spatial patterns and maintain some translation equivariance, fully connected architectures treat input features independently and lack such priors. Similarly, Long Short-Term Memory (LSTM) networks, used as the backbone for Nets 5 and 6, outperform earlier models in this experiment due to their inductive bias towards sequential data structures. Since the dataset used is audio, models that rely solely on Mel spectrogram representations overlook important temporal dynamics.

2. **Optimization and learning rate affect performance**

   During learning rate tuning, it was observed that RMSprop is highly sensitive to changes in the learning rate. This was evident from the initial experiments, where using a learning rate of 0.001 resulted in highly variable performance across different models. In contrast, Adam demonstrated greater robustness, maintaining relatively consistent results under the same conditions. However, in certain cases, RMSprop proved to be more efficient, as seen in the per-

Model	Epochs	Optimizer	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Optimizer: Adam						
Net 1	50	Adam	55	68	55	56
Net 1	100	Adam	51	61	51	51
Net 2	50	Adam	69	71	69	69
Net 2	100	Adam	65	67	65	65
Net 3	50	Adam	69	70	69	69
Net 3	100	Adam	71	74	71	71
Net 5	43	Adam	85	86	85	85
Net 6	51	Adam	80	80	80	80
Optimizer: RMSprop						
Net 1	50	RMSprop	39	52	39	35
Net 1	100	RMSprop	60	67	60	60
Net 2	50	RMSprop	61	63	61	61
Net 2	100	RMSprop	59	63	59	60
Net 4	50	RMSprop	57	66	57	58
Net 4	100	RMSprop	71	68	68	68
Net 5	49	RMSprop	83	85	83	83
Net 6	77	RMSprop	76	76	76	75

Table 2. Performance metrics of six model architectures evaluated at different epoch settings, grouped by optimizer.

formance of Net 6, which achieved convergence within only 55 epochs.

3. **Longer training does not guarantee improved generalisation**

   The performance trends observed in Net 2 and Net 3 at both 50 and 100 epochs suggest that simply extending training time does not mitigate overfitting. For instance, in Net 1, the validation accuracy peaked around epoch 40 and declined thereafter. This highlights how extended training might degrade generalisation. These findings strengthen the idea of early stopping, as it prevents continuing training beyond the point of optimal validation performance.

4. **Impact of Distribution Mismatch in Data Augmentation**

   Since the core objective of GANs is to model the underlying data distribution, directly augmenting raw audio using a CGAN led to suboptimal results in this experiment. The performance of Net 6, trained with CGAN-augmented audio, dropped to 80% compared to 85% achieved by Net 5 without such augmentation. This suggests that the CGAN-generated audio added noise rather than informative variation to the training data. The likely cause is the inability of CGANs with fully connected layers to effectively capture the sequential and phase-sensitive nature of audio signals, which in turn degrades the quality of extracted features. However, in further observation outside the main experimental setup, applying CGAN-based augmentation directly to extracted features (e.g., MFCCs, spectral contrast, chroma) yielded an improved accuracy of 92%, indicating that feature-domain augmentation may better preserve the statistical properties relevant to learning.

References

[1] Wafaa Shihab Ahmed et al. The impact of filter size and number of filters on classification accuracy in cnn. In *2020 International conference on computer science and software engineering (CSASE)*, pages 88–93. IEEE, 2020.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[4] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 2002.