Comparing the Performance of Traditional Deep Learning Models on the GTZAN Dataset

Tianhao Liu
School of Computer Science
University College Dublin
tianhao.liu@ucdconnect.ie

Abstract

This paper delves into the realm of music genre classification, leveraging the renowned GTZAN dataset as our testing ground. Two distinct methodologies have been explored: training models from scratch and fine-tuning pre-trained models. Through rigorous experimentation, we compare the performance of Multilayer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) with LSTM and GRU units under the former approach. Meanwhile, the latter approach involves evaluating the efficacy of fine-tuning pre-trained models such as VGG19, ResNeXt, and SqueezeNet. The findings of this study will shed light on the most effective strategies for classifying music genres, offering valuable insights for future research and practical applications in the domain of audio processing.

1 Introduction

The GTZAN dataset has long served as a pivotal benchmark for evaluating music genre classification methods. In this study, we aim to discern the most effective approaches for classifying music genres using deep learning models on the GTZAN dataset. Our investigation revolves around two primary methodologies: training models from scratch and fine-tuning pre-trained models. In the former approach, we compare the performance of Multilayer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) equipped with Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) units. Conversely, the latter approach entails evaluating the efficacy of fine-tuning pre-trained models, including VGG19, ResNeXt, and SqueezeNet. Subsequent sections will delve into the related work, experimental setup, model architecture, and results, elucidating the key findings of our investigation.

2 Related Work

In the realm of music genre classification, a diverse array of traditional deep learning models has been extensively investigated. Convolutional neural networks (CNNs), originally designed for image classification, have demonstrated notable success in audio processing as well (Hershey et al. [1]). Alongside CNNs, various iterations of recurrent neural networks (RNNs) have exhibited robust performance in audio classification tasks. For example, Banuroopa et al. [2] achieved an outstanding accuracy of 98.8% using an LSTM model for classifying the UrbanSound8K dataset. Similarly, GRU (Gated Recurrent Unit) networks have shown significant efficacy in audio recognition tasks, outperforming LSTMs across all network depths in a large vocabulary continuous speech recognition task, as reported by Shubham Khandelwal et al. [3].

COMP47650 - Deep Learning Final Project Report, Dublin, Ireland.

Moreover, fine-tuning pre-trained models is a prevalent practice in music genre classification, in addition to training models from scratch. In genre classification tasks, a common approach involves using the spectrogram of the audio as input to fine-tune pre-trained models.

3 Experimental Setup

3.1 Dataset

The GTZAN dataset serves as a cornerstone benchmark for music genre classification, comprising 1000 audio tracks, each lasting 30 seconds and sampled at 22050Hz. This comprehensive dataset encapsulates 10 diverse music genres, with each genre meticulously represented by 100 tracks in .wav format. Encompassing a rich array of musical styles, the genres featured include blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock.

3.2 Baseline

A Multilayer Perceptron (MLP) is served as the baseline model for comparison. This MLP consists of three fully-connected hidden layers with ReLU activation functions. Each hidden layer has a decreasing number of units, starting with 512, followed by 256, and finally 64 units. To prevent overfitting, dropout with a 30% probability is applied after each hidden layer. The final output layer has 10 units, aligning with the 10 music genres present in the dataset.

3.3 Train from Scratch

3.3.1 Data pre-process

The dataset is perfectly balanced, with each genre containing an equal number of tracks, so no additional data balancing is required. Then the audio pre-processing is mainly about extracting Mel-frequency cepstral coefficients (MFCCs) from the audio tracks. The audio is segmented into 10 frames, each frame lasting 3 seconds, and 13 MFCCs are extracted from each frame. These MFCCs are then serves as the input features for the deep learning models.

3.3.2 Model Architecture

- **Convolutional Neural Network (CNN)**: This CNN model consists of three convolutional layers: conv1 with 32 output channels, conv2 with 64 output channels, and conv3 with 120 output channels. Each convolutional layer is followed by batch normalization and ReLU activation. The max pooling layers reduce the spatial dimensions of the feature maps by a factor of 2. The fully connected layer fc1 has 64 neurons, and the final output layer fc2 produces predictions for a specified number of classes (10). A dropout rate of 30% is applied to fc1 to prevent overfitting. Overall, the model architecture contains pproximately 328,840 trainable parameters.
- **Long Short-Term Memory (LSTM)**: This LSTM model is composed of an LSTM layer followed by two fully connected layers. The LSTM layer takes input sequences and processes them through two LSTM layers with a hidden size of 64. The output of the LSTM layer is passed through a fully connected layer (fc1) with 64 neurons. A ReLU activation function is applied after the first fully connected layer, followed by a dropout layer to prevent overfitting. Finally, the output is passed through the second fully connected layer (fc2), which produces the final predictions for a specified number of classes using softmax activation.
- **Gated Recurrent Unit (GRU)**: This GRU model is similar to the LSTM model, but with GRU layers instead of LSTM layers.

3.3.3 Train

1. **Split the dataset**: The dataset is split into training and test sets with a ratio of 70:30, where the training set is used to train the model, while the test set is used to evaluate the model's performance.

2. **Batch Loading**: The splitted dataset is loaded in batches to train the model by using the PyTorch Dataset and DataLoader, which allows for efficient training and memory management.
3. **Train the model**: The model is trained using the Adam optimizer with a learning rate of specified in the 'hparms.yaml'. I adopted the cross-entropy loss function as the loss function and adam as the optimizer.
4. **Logging**: In each epoch, the training and validation loss are logged to monitor the training process.
5. **Benchmark**: The model with the best validation loss will be saved as the benchmark model.
6. **Early Stopping**: To prevent overfitting, early stopping is applied to the training process. The training process is stopped when the validation loss does not decrease for a specified number of epochs. The patient - the number of epochs with no improvement after which training will be stopped - is also configured in the 'hparms.yaml'.

3.4 Fine-tuning Pre-trained Models

The GTZAN dataset also provides the spectrogram of the audio tracks, which can be used as the input to fine-tune the pre-trained models. The spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. Therefore, using the spectrogram as the input we can transform the audio classification task into an image classification task.

3.4.1 Data pre-process

The image data is loaded using OpenCV, then converted to RGB format and resized to specified dimensions. The dimension depends on the pre-trained model used, for example, VGG19 requires the input image to be 224×224. Finally, the image data is moved to the GPU for training.

3.4.2 Model Architecture

- **VGG19**: The first pre-trained model is VGG19, which is a convolutional neural network model designed by Simonyan et al.[4], which has been pre-trained on the ImageNet dataset. The model consists of 19 layers, including 16 convolutional layers and 3 fully connected layers as classifier. To fine-tune the model on the GTZAN dataset, the average pool layer is deleted and the fully connected layers are replaced by some new fully connected layers with 10 output units. The convolutional layers are frozen, and only the fully connected layers are trained on the GTZAN dataset. The new classifier architecture is shown below.

ConvNet Configuration						
A	A-LRN	B	C	D	E	
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers	
input (224 × 224 RGB image)						
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	
maxpool						
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	
maxpool						
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**	Frozen
maxpool						
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**	
maxpool						
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**	
maxpool -> identity						Modify & Train
FC-4096 -> FC-4096						
FC-4096 -> FC-512						
FC-1000 -> FC-256						
soft-max -> FC-10 + softmax						

Figure 1: Modified VGG19 Model Architecture

- **ResNeXt**: ResNext is a deeplearning model designed by Orhan, A. Emin [5], which has been pre-trained on the ImageNet dataset. To fine-tune the model on the GTZAN dataset, the last layer (fc) is replaced by a new fully connected layer with 10 output units. The layers before the last layer are frozen, and only the fully connected layer is trained on the GTZAN dataset.
- **SqueezeNet**: SqueezeNet is a deeplearning model designed by Iandola, Forrest N. [6], which has been pre-trained on the ImageNet dataset. To fine-tune the model on the GTZAN dataset, the last layer (classifier) is replaced by a new fully connected layer with 10 output units. The layers before the last layer are frozen, and only the fully connected layer is trained on the GTZAN dataset.

3.4.3 Train

The training process is similar to the training from scratch process, but the input data is the spectrogram of the audio tracks.

3.5 Evaluation

The performance of the models is evaluated using the test set with evaluation metrics include precision, recall, and F1 score.

4 Results

4.1 Baseline: MLP

- **Loss**: The curve is quite smooth, which indicates that the model is learning well. Since I applied early stopping, the training process stopped after around 160 epochs, which avoids overfitting. The loss curve is quite smooth, which indicates that the model is the learning rate is appropriate.

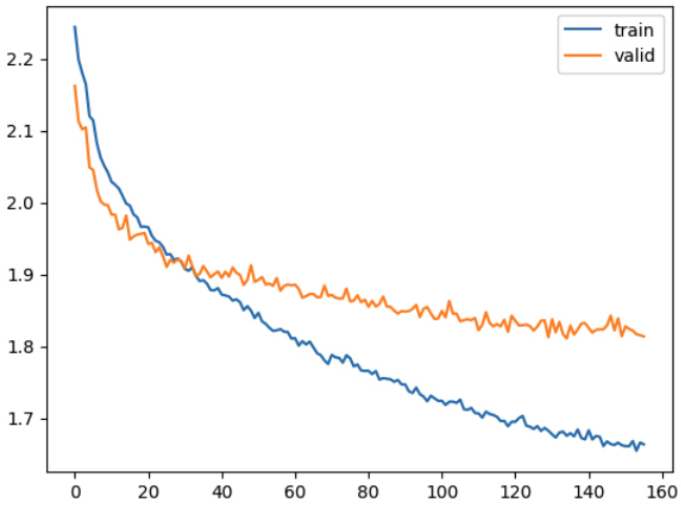

Figure 2: Loss of MLP Model

- **Evaluation**: Here we can see that the MLP model achieved an average accuracy of 65% on the test set. The F1 score of the model is 0.64, which is not very good.

4.2 Train from Scratch

4.2.1 CNN

- **Loss**: From the loss curve, we can see that CNN model takes shorter time to converge than the MLP model. Since I applied early stopping, the training process stopped after around 35

Table 1: MLP Classification Report

	Precision	Recall	F1-Score	Support
0	0.87	0.77	0.82	319
1	0.80	0.81	0.81	308
2	0.53	0.51	0.52	289
3	0.63	0.73	0.68	299
4	0.64	0.50	0.56	328
5	0.79	0.81	0.80	283
6	0.44	0.56	0.49	282
7	0.59	0.51	0.55	283
8	0.59	0.53	0.56	304
9	0.61	0.70	0.65	301
Macro Avg	0.65	0.64	0.64	2996
Weighted Avg	0.65	0.64	0.64	2996

epochs, which avoids overfitting. The loss curve is quite smooth, which indicates that the model is the learning rate is appropriate.

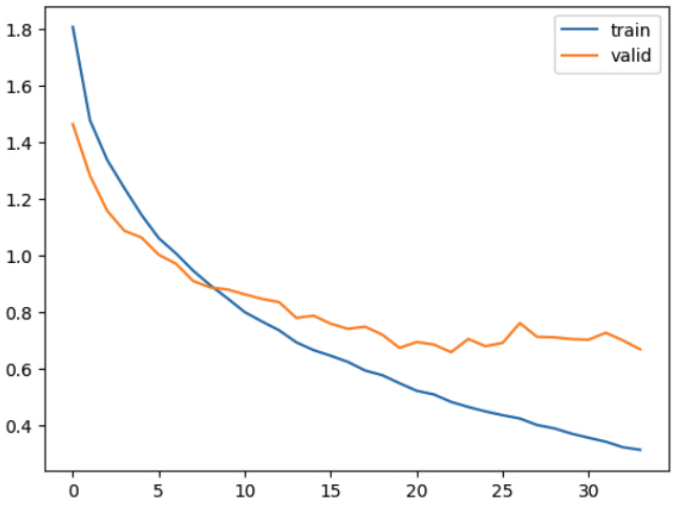

Figure 3: Loss of CNN Model

- **Evaluation**: The CNN model achieved an average accuracy of 79% on the test set, which is better than the MLP model. The F1 score of the model is 0.76, which is also better than the MLP model.

4.2.2 LSTM

- **Loss**: The loss curve of the LSTM model is not as smooth as the CNN model, the time it takes to converge is also longer. The model stopped training after around 300 epochs. However, the loss curve for training set and validation set are quite close, which indicates that the model is not overfitting.
- **Evaluation**: The LSTM model achieved an average accuracy of 63% on the test set, which is worse than the baseline model MLP. The F1 score of the model is 0.62, which is also worse than the baseline. From my perspective, the reason for the poor performance is due to the implementation goal of the LSTM is not suitable for this kind of task. LSTM is more suitable for tasks that require long-term memory, such as speech recognition. However, for this dataset, the short-term memory is more important, since the whole audio tracks are only 30 seconds long and has been splited into 10 segments.

Table 2: CNN Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.81	0.79	0.80	310
1	0.83	0.89	0.86	316
2	0.77	0.63	0.69	314
3	0.83	0.71	0.76	283
4	0.87	0.58	0.70	301
5	0.94	0.91	0.93	280
6	0.49	0.77	0.60	320
7	0.88	0.67	0.76	288
8	0.64	0.69	0.66	295
9	0.82	0.94	0.88	289
Macro Avg	0.79	0.76	0.76	2996
Weighted Avg	0.78	0.76	0.76	2996

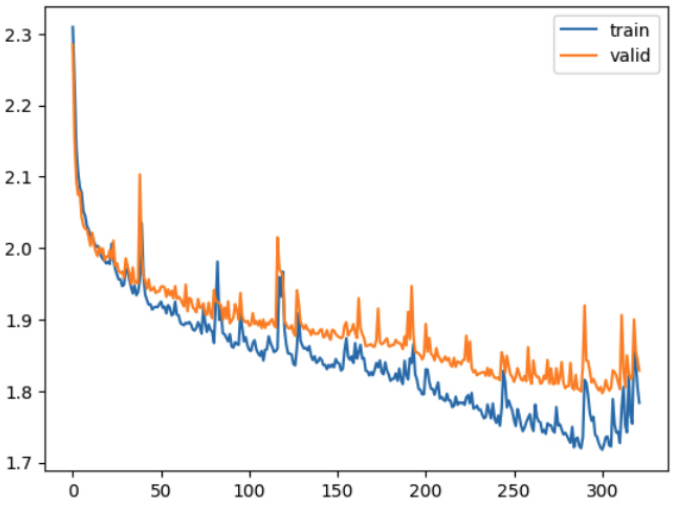

Figure 4: Loss of LSTM Model

4.2.3 GRU

- **Loss**: From the loss curve, we can see that the GRU model takes shorter time to converge than the LSTM model, but longer than the CNN model. Similar to the LSTM model, the loss curve for training set and validation set are quite close, which indicates that the model is not overfitting.
- **Evaluation**: In essence, the goal of the GRU model is similar to the LSTM model, which is to capture the long-term dependency in the audio tracks. Therefore, the performance of the GRU model is also worse than the CNN model. However, the GRU model is better than the LSTM model, which is consistent with the previous research [3].

4.3 Fine-tuning Pre-trained Models

4.3.1 VGG19

- **Loss**: The loss curve of the VGG19 model not as smooth as the models trained from scratch, this is because the pre-trained model has been trained on the ImageNet dataset, which is quite different from the GTZAN dataset. The loss curve for training set and validation set are getting far away from each other from the 15^{th} epoch, which indicates that the model is tend to overfit.

Table 3: LSTM Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.81	0.79	0.80	297
1	0.71	0.86	0.78	293
2	0.57	0.64	0.60	303
3	0.61	0.43	0.50	315
4	0.56	0.71	0.63	304
5	0.89	0.91	0.90	300
6	0.42	0.37	0.39	297
7	0.58	0.55	0.57	291
8	0.50	0.39	0.44	317
9	0.62	0.67	0.65	279
Macro Avg	0.63	0.63	0.62	2996
Weighted Avg	0.62	0.63	0.62	2996

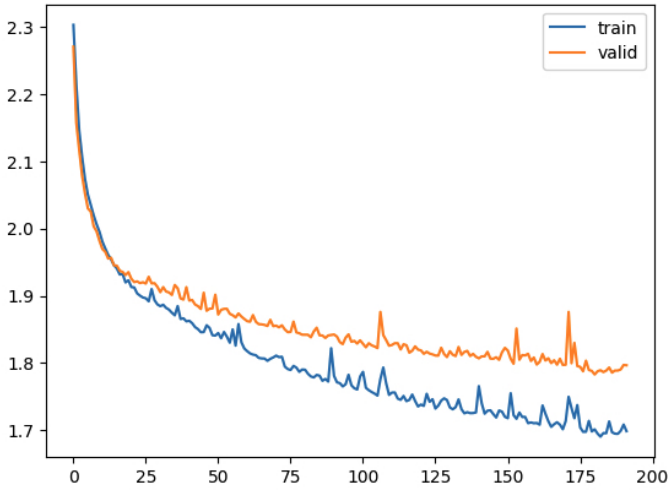

Figure 5: Loss of GRU Model

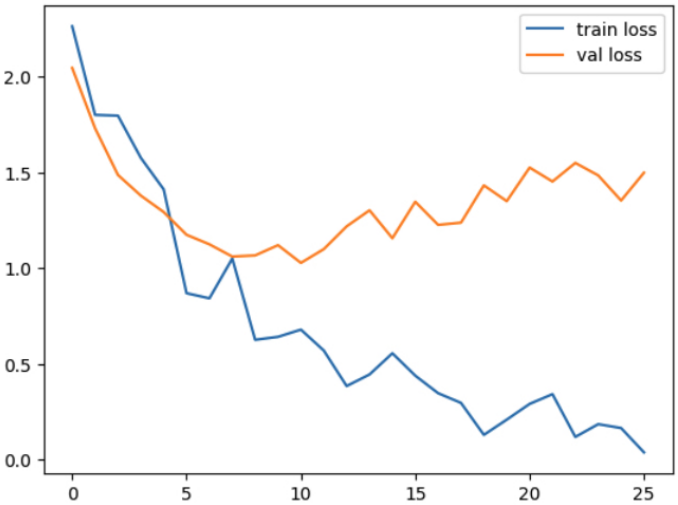

Figure 6: Loss of VGG19 Model

Table 4: GRU Classification Report

	Precision	Recall	F1-Score	Support
0	0.84	0.81	0.82	306
1	0.71	0.86	0.78	283
2	0.56	0.59	0.57	287
3	0.66	0.57	0.61	302
4	0.67	0.66	0.66	314
5	0.86	0.92	0.89	307
6	0.49	0.38	0.43	293
7	0.61	0.73	0.67	284
8	0.46	0.44	0.45	306
9	0.72	0.69	0.70	314
Macro Avg	0.66	0.67	0.66	2996
Weighted Avg	0.66	0.67	0.66	2996

- **Evaluation**: The VGG19 model achieved an average accuracy of 73% on the test set, which is better than the baseline, but worse than the CNN model. The F1 score of the model is 0.72, which is also better than the LSTM and GRU models.

Table 5: VGG Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.63	0.57	0.60	21
1	0.83	0.83	0.83	12
2	0.59	0.54	0.57	24
3	0.55	0.55	0.55	22
4	0.80	0.80	0.80	15
5	0.77	0.89	0.83	27
6	1.00	0.83	0.91	18
7	0.43	0.84	0.57	19
8	0.68	0.68	0.68	22
9	1.00	0.25	0.40	20
Macro Avg	0.73	0.68	0.67	200
Weighted Avg	0.72	0.67	0.66	200

4.3.2 ResNeXt

- **Loss**: The loss curve of the ResNeXt model is not as smooth as the models trained from scratch, this is because the pre-trained model has been trained on the ImageNet dataset, which is quite different from the GTZAN dataset. The loss curve for training set and validation set are getting far away from each other from the 10^{th} epoch, which indicates that the model is tend to overfit.

- **Evaluation**: The ResNeXt model achieved an average accuracy of 61% on the test set, which is lower than the baseline MLP model. Additionally, the F1 score of the model is 0.60, also lower than the baseline. In my view, the discrepancy in performance can be attributed to the significant dissimilarity between the pre-training dataset, ImageNet, and the target dataset, GTZAN. ImageNet is a diverse image dataset, enabling the model to learn general features applicable to various images. In contrast, the GTZAN dataset focuses on music genre classification, requiring the model to discern subtle differences in spectrogram images. This disparity in dataset characteristics may hinder the model's ability to generalize effectively to the GTZAN dataset, resulting in the observed performance decrement.

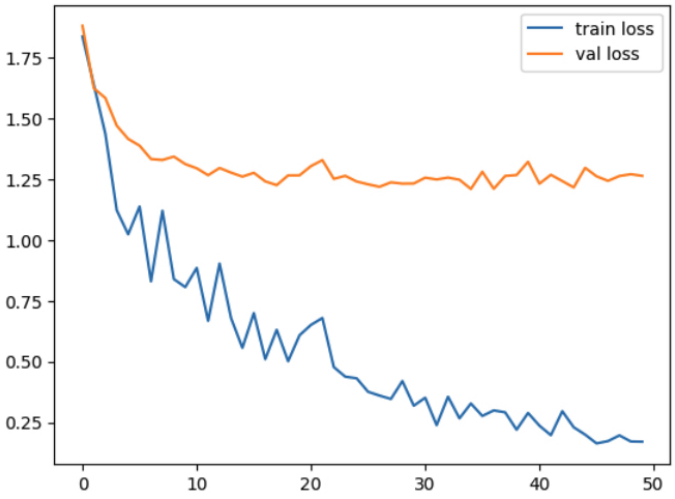

Figure 7: Loss of ResNeXt Model

Table 6: ResNext Classification Report

Class	Precision	Recall	F1-score
0	0.71	0.48	0.57
1	0.75	1.00	0.86
2	0.57	0.54	0.55
3	0.45	0.45	0.45
4	0.65	0.87	0.74
5	0.67	0.81	0.73
6	0.82	0.78	0.80
7	0.59	0.68	0.63
8	0.58	0.50	0.54
9	0.36	0.25	0.29
Macro avg	0.62	0.64	0.62
Weighted avg	0.61	0.61	0.60

4.3.3 SqueezeNet

- **Loss**: Different from the ResNeXt model and VGG model, the difference between the training loss and validation loss of the SqueezeNet model is not very large, which indicates that the model is not overfitting. The loss curve for validation set become stabe after around 30 epochs, which indicates that the model has converged.
- **Evaluation**: The SqueezeNet model achieved an average accuracy of 67% on the test set, which is better than the ResNeXt model and similar to baseline. The F1 score of the model is 0.66, which is also better than the ResNeXt model and similar to baseline.

5 Conclusion

In this study, I conducted a comprehensive evaluation of various deep learning models on the GTZAN dataset, encompassing MLP, CNN, LSTM, and GRU models trained from scratch, as well as fine-tuned VGG19, ResNeXt, and SqueezeNet models. Among these, the CNN model trained from scratch emerged as the top performer, achieving an impressive average accuracy of 79% and an F1 score of 0.76. Following closely, the fine-tuned VGG model delivered respectable results, with an average accuracy of 73% and an F1 score of 0.72. Notably, RNN models such as LSTM and GRU exhibited inferior performance compared to the CNN model, likely attributable to the dataset's preference for capturing short-term dependencies over long-term ones. Additionally, the performance of fine-tuned models fell short of expectations, possibly due to the mismatch between the pre-training and target

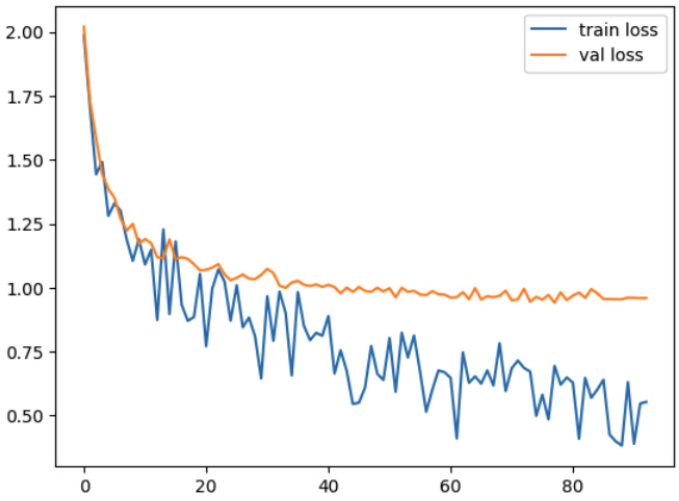

Figure 8: Loss of ResNeXt Model

Table 7: SqueezeNet Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.61	0.67	0.64	21
1	0.80	1.00	0.89	12
2	0.60	0.50	0.55	24
3	0.65	0.68	0.67	22
4	0.71	0.80	0.75	15
5	0.81	0.78	0.79	27
6	0.94	0.94	0.94	18
7	0.53	0.47	0.50	19
8	0.59	0.59	0.59	22
9	0.42	0.40	0.41	20
Accuracy			0.67	200
Macro Avg	0.67	0.68	0.67	200
Weighted Avg	0.66	0.67	0.66	200

datasets. Consequently, I recommend the CNN model trained from scratch as the preferred choice for music genre classification tasks on the GTZAN dataset. Furthermore, our findings underscore the importance of aligning the characteristics of the pre-training dataset with those of the target dataset when fine-tuning pre-trained models to achieve optimal performance.

6 Future Work

The study has identified several avenues for further exploration and improvement. Firstly, the adoption of more advanced deep learning architectures, such as Transformer models, holds promise for enhancing performance across various tasks. Secondly, fine-tuning the hyperparameters of existing models can potentially unlock further performance gains. Given the suboptimal performance of finetuned pre-trained models in this study, exploring alternative pre-trained models tailored to the characteristics of the GTZAN dataset is warranted. Additionally, investigating alternative fine-tuning techniques, such as transfer learning, may yield insights into improving model performance. Furthermore, exploring the impact of different audio features on model performance could uncover valuable insights. Lastly, investigating the generalizability of models to other music genre classification datasets represents an intriguing avenue for future research. Combining insights from this study with ongoing research efforts can inform the development of bespoke deep learning models optimized for short audio classification tasks.

References

[1] Hershey, Shawn and Chaudhuri, Sourish and Ellis, Daniel P. W. and Gemmeke, Jort F. and Jansen, Aren and Moore, R. Channing and Plakal, Manoj and Platt, Devin and Saurous, Rif A. and Seybold, Bryan and Slaney, Malcolm and Weiss, Ron J. and Wilson, Kevin. (2017) CNN architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131-135.

[2] Banuroopa, K. and Shanmuga Priyaa, D. (2021) MFCC based hybrid fingerprinting method for audio classification through LSTM. *International Journal of Nonlinear Analysis and Applications*, 12(Special Issue), pp. 2125-2136.

[3] Shubham Khandelwal, Benjamin Lecouteux, Laurent Besacier. COMPARING GRU AND LSTM FOR AUTOMATIC SPEECH RECOGNITION. [Research Report] LIG. 2016. ⟨hal-01633254⟩

[4] Simonyan, Karen, and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014. https://doi.org/10.48550/ARXIV.1409.1556.

[5] Orhan, A. Emin. "Robustness Properties of Facebook's ResNeXt WSL Models." arXiv, December 9, 2019. http://arxiv.org/abs/1907.07640.

[6] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. "SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <0.5MB Model Size." arXiv, November 4, 2016. http://arxiv.org/abs/1602.07360.