Universiteit Leiden
The Netherlands

Opleiding Informatica

Scalability of Music Genre Classification Algorithms

Luca Goemans

Supervisors:
E.M. Bakker & M.S. Lew

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

19/06/2023

Abstract

Since their introduction, Convolutional Neural Networks have been used for image classification and object detection. Over the past few years, they have also been used to classify music genres, where research has focused on relatively small datasets. In this thesis, an implementation of MusicRecNet [1] is created. It is trained and validated on the GTZAN dataset [2], a dataset that has been used heavily in the field of music genre classification. MusicRecNet's performance on GTZAN is compared to two other network architectures for music genre classification with VGG-16 [3] and EfficientNetV2B0 [4] as backbones. Two subsets of the Free Music Archive dataset [5] to evaluate the scalability of the three methods. Experimental results show lower accuracies for the datasets with more samples or more genres. The addition of an SVM as proposed by the MusicRecNet paper does not lead to the expected performance gains in our scalability experiments, but finetuning does have a significant impact on model performance. The EfficientNetV2B0 model performs best in terms of scalability, when used in combination with an SVM classifier and finetuning the network.

Contents

1 Introduction **1**

2 Related Work **2**

3 Fundamentals **3**
- 3.1 Mel-Spectrograms . . . 3
- 3.2 Convolutional Neural Network . . . 3
- 3.3 Transfer Learning . . . 4
- 3.4 Evaluation metrics . . . 4
  - 3.4.1 Accuracy . . . 4
  - 3.4.2 Confusion Matrix . . . 4

4 Baseline: MusicRecNet **5**

5 Method **6**
- 5.1 VGG-16 . . . 6
- 5.2 EfficientNetV2B0 . . . 7

6 Datasets **7**
- 6.1 Preprocessing . . . 8
- 6.2 GTZAN . . . 8
- 6.3 FMA . . . 8

7 Experimental Setup **9**
- 7.1 Baseline experiments . . . 10
- 7.2 Scalability experiments . . . 10

8 Experimental Results **11**
- 8.1 Baseline experiments . . . 12
- 8.2 Scalability Experiments . . . 12

9 Conclusions and discussion **17**

References **19**

A Network Architectures **20**

B GTZAN Baseline verification experiment results **22**

C FMA Baseline network Experiment results **24**

D Confusion matrices for GTZAN, input size 128x128 **26**

E Confusion matrices for GTZAN, input size 224x224 **29**

F Confusion matrices for FMA_8, input size 128x128 **30**

G Confusion matrices for FMA_8, input size 224x224 **33**

H Confusion matrices for FMA_14, input size 224x224 **34**

I Train/test accuracy curves **37**

J Precision, recall and F1 scores for the best performing models per dataset. **42**

1 Introduction

Neural networks have been transforming the way we solve problems since their introduction. Due to their context-independent nature, they can be used to solve a wide range of problems One of the areas in which they are being deployed is music genre classification.

One of the main ways of categorizing songs is by specifying their genre. It is a common task for a human expert within the field of music to determine the genre of a song. With major streaming platforms like Spotify having over 100 million songs [6] and over 100.000 tracks being added each day to streaming platforms [7], the amount of music that has to be categorized is growing rapidly. As a result of that growth, the process of classifying the song genres by human expertise at this scale can be time-consuming and challenging.

To automate music genre classification, this thesis will focus on the use of a particular type of Neural Network, the Convolutional Neural Network(CNN). This type of network performs well at image classification tasks, as shown by architectures like XCeption [8] and YOLO [9]. These CNNs are used as feature extractors to detect certain characteristics and even entire objects in images.

Over the years researchers have also attempted to train CNNs for music genre classification tasks. Most of the research focused on creating a neural network that is able to classify genres of the songs in the GTZAN dataset, created by G. Tzanetakis [2]. In order to compare performance between newer networks and existing solutions this dataset has been used extensively as a benchmark. Accuracy scores of up to 97.6% have been achieved on this dataset. The problem with the GTZAN dataset however is that it is relatively small and not that diverse: it contains 1000 songs, across 10 genres. This small size makes it relatively easy for the network to learn how to recognize certain genres of music that are inside the dataset.

Bigger and more diverse datasets exist, such as the Free Music Archive (FMA) [5] dataset. This set not only contains more songs, but the songs also span across more genres. Training a network on such a dataset is expected to make it able to generalize better across a variety of songs.

In this paper we use a baseline model proposed by Elbir, A & Aydin, N. [1]. Its performance is validated on the GTZAN dataset. The goal of this thesis is to see how such a network would handle scaling of a dataset. In our study, the network is trained on larger subsets created from the FMA dataset. These subsets allow for testing scalability of the models in two ways: by using a dataset that uses more samples per genre and a subset that uses more different classes/genres. After training and comparing its performance on those subsets to GTZAN, the model is compared to two other networks, that use VGG-16 [3] and EfficientNetV2B0 [4] as backbone networks. Their accuracies are compared and analyzed. Using a bigger dataset is expected to be beneficial to the network's capacity to generalize, but it will also create performance challenges, as the network will need more time to train and might perform worse when genres have overlapping distinctive features.

The rest of the paper is organized as follows: Section 2 contains related work in the field of music genre classification that is the foundation of this thesis. The main definitions of used techniques and measures will be discussed in Section 3. The aforementioned baseline network MusicRecNet will be fully explained in Section 4. The two other network architectures are defined in Section 5. The datasets that were created and used for the experiments are discussed in Section 6. The baseline and scalability experiments and their results can be found in Sections 7 and 8. Finally, our conclusions can be found in Section 9.

2 Related Work

In this section, the relevant information found in the field that contributes to our approach is highlighted. Human performance regarding genre classification, methods used on the GTZAN dataset, and methods used on the FMA dataset are also mentioned.

Human performance

In order to see if a network is a feasible solution for the genre classification problem a baseline accuracy level we expect the network to achieve has to be established, before comparing it to other models. In order for a model to be useful for classification problems we think it has to be at least as good or better than humans at genre classification. Robert O. Gjerdingen & David Perrott [10] showed that humans are able to classify music genres about 70% of the time. In their study they let participants listen to song excerpts of up to 3 seconds and asked the participants to classify the genre, being able to choose from 10 genres.

GTZAN performance

Researchers have generated models that have exceeded the aforementioned human performance of classification. Zhang et al. [11] were able to achieve an accuracy of 87.4% on the GTZAN dataset using their 10-layer CNN. They found that the classification accuracy improved after the used songs were cut into smaller 3-second clips, similar to the work by Robert O. Gjerdingen & David Perrott.

Others such as Liu et al. [12] have improved on this score, showing accuracy levels of up to 93.9%. This score was achieved by using a relatively small network that uses Bottom-up Broadcasting. The idea is that the network does not simply use convolution operations to detect features but that it utilizes the time-frequency information that is contained in mel-spectrograms.

Elbir, A & Aydin, N. [1] created a CNN architecture and reported the currently leading accuracy on the GTZAN dataset, 97.6%. This accuracy was achieved using a CNN architecture they called "MusicRecNet". Their network consists of three sets of layers: a convolution layer, a MaxPooling2D layer, and a dropout layer. These three sets are then flattened and fed into a dense layer with dropout and fed into another dense layer. The distinctive feature for MusicRecNet is that it uses this output as a feature vector for a Support Vector Machine (SVM) to classify the songs. A lot of research on genre classification uses the GTZAN dataset, which is why it is also used in this thesis as a baseline to test the networks' performance.

FMA performance

The FMA dataset has been less popular in the field of genre classification. Zhang et al. [13] achieved an accuracy of 65.2% on the FMA dataset, using a Convolutional Recurrent Neural Network approach. Similar to previously mentioned methods they also used spectrograms as input for the network. S. Chillara et al.[14] used the FMA dataset to compare three network architectures: CNNs, Convolutional Recurrent Neural Networks (CRNN), and CNN-RNNs. Their configuration of a CNN achieved the state-of-the-art accuracy of 88.5% Not only do they show that from the three tested architectures the CNN has the highest accuracy, but also that using spectrograms as input yields a higher accuracy when compared to feature-based models, that achieved an accuracy of 64.1%.

In this thesis the FMA dataset is used over other datasets like the Million Song Dataset [15], because it has the actual audio files in it, similar to the GTZAN dataset. This makes comparing the baseline network to other solutions possible. The accuracies achieved by Zhang et al. and S.

Chillara et al. on this dataset also leave room for improvement.

3 Fundamentals

In this chapter, the fundamentals of this thesis are explained. Melspectrograms are first introduced, as they form the input for the networks that are used. Convolutional Neural Networks are then briefly explained. After that, the evaluation metrics that will be used to assess model performance are highlighted.

3.1 Mel-Spectrograms

Audio can be represented as a time series of amplitude values. These audio signals can be converted, using a Fourier transform, to a spectrogram. Spectrograms show the energy levels of an audio signal for certain frequency bands. The energy levels are shown by plotting the intensity of different frequencies in the signal over time. In order to plot the frequency intensities over time, the signal has to be decomposed into different frequency bands at each time step. This is done by applying a Fast-Fourier-Transform [16] on a window of the audio signal around that time step. After this transformation, the different frequencies can be plotted per time step. This allows for a visualization of the frequencies and their intensity. A mel-spectrogram is a version of a spectrogram that contains frequencies that are mapped to the mel-scale [17] followed by a cosine deconvolution step. The mel-scale is a perceptual scale where the pitches are perceived as equally distanced from each other by humans. An example of a mel-spectrogram can be found in Figure 1.

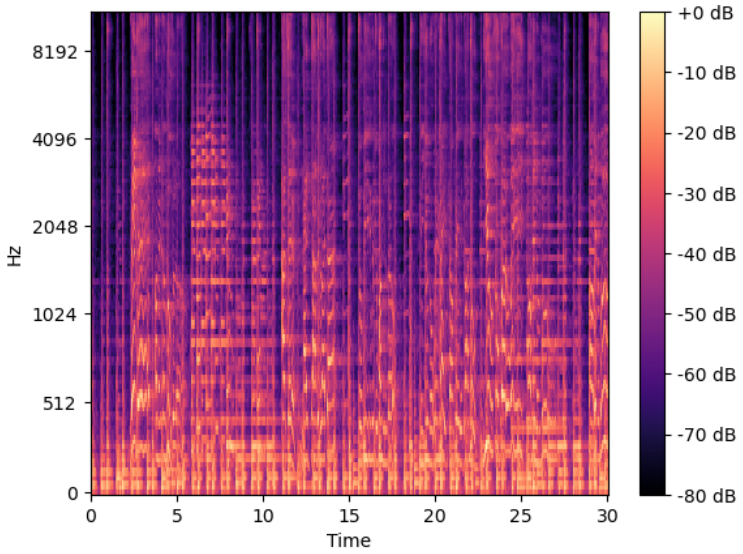


Figure 1: A melspectrogram image generated from the GTZAN dataset, for a piece of Blues music.

3.2 Convolutional Neural Network

Since their introduction in 1989 by LeCun[18], Convolutional Neural Networks have been used for a range of image recognition tasks. The convolutional layers in the network contain convolutional

kernels: matrices of weights that can be trained. These kernels slide over the input image to convert it to a feature map. This feature extraction can be used to extract features such as edges, but also more abstract features such as faces, text, etc. CNNs are especially suited for images because images are in essence matrices of pixel values, making it easier to multiply the kernel weights with the values. The network also utilizes other operations apart from convolution. Pooling is used for dimension reduction. For pooling a window size is defined, say 2x2. Depending on the type of pooling (max-, min-, or average-pooling) either the maximum, minimum, or average values of those 4 selected values are picked or calculated. This window then slides over the feature space in order to reduce its dimension.
To prevent overfitting our dataset, which is more common for smaller datasets, dropout layers are used. These layers ensure that some weights are reset to 0 at random. This helps the network to generalize better, instead of learning to recognize the dataset.

3.3 Transfer Learning

Training a neural network from scratch can be resource-heavy. Transfer learning is a technique used in machine learning, where instead of training a neural network from scratch, a pre-trained network is used. The pre-trained network has been trained to detect powerful features in images, for classifying images from big datasets such as ImageNet [19]. This feature extraction can be used for other datasets as well. In order to make sure the network can be applied to the new dataset, the pre-trained network can be fitted with a new, suitable classifier for the new dataset. If needed, a number of layers of the original pre-trained network can be "unfrozen" so they can also be trained along with the classifier. This can be beneficial if the used dataset differs from the original dataset.

3.4 Evaluation metrics

The neural network performance will be assessed and evaluated using two metrics: accuracy and confusion matrices. Accuracy is widely used to see how well a method is performing, which enables us to compare our method against other methods in the field. A confusion matrix will give us more detailed insight into the classifications the network is deciding on.

3.4.1 Accuracy

Accuracy is generally defined as the number of True Positives (TP) and True Negatives (TN), divided by the total number of predictions, or the True Positives, True Negatives, False Positives (FP), and False Negatives (FN) combined (see Equation 1). In this thesis, it is defined as the number of correctly classified genres divided by the number of total classifications, i.e. TN = 0.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

3.4.2 Confusion Matrix

Confusion matrices can be used to investigate how the network is behaving in terms of classifications and misclassifications. It shows what the predicted classes were and what the actual classes are. This can give a more detailed indication of where the network is classifying data incorrectly. This

information can then be used to make specific changes to the network based on those particular misclassifications, further improving network performance, for example by altering hyperparameters. When it comes to music, genres that are relatively similar to each other might be harder to classify. This would be clearly represented in a confusion matrix. See for example the confusion matrix for the genres Jazz and Rock, as presented in Table 1.

	Predicted label			
Actual label	Jazz	Rock	Total	
Jazz	5	7	12	
Rock	3	3	6	

Table 1: Confusion matrix for example genres Jazz and Rock. The first row shows that 5 out of 12 Jazz classifications were correctly classified, but in many cases, Jazz and Rock are confused with each other.

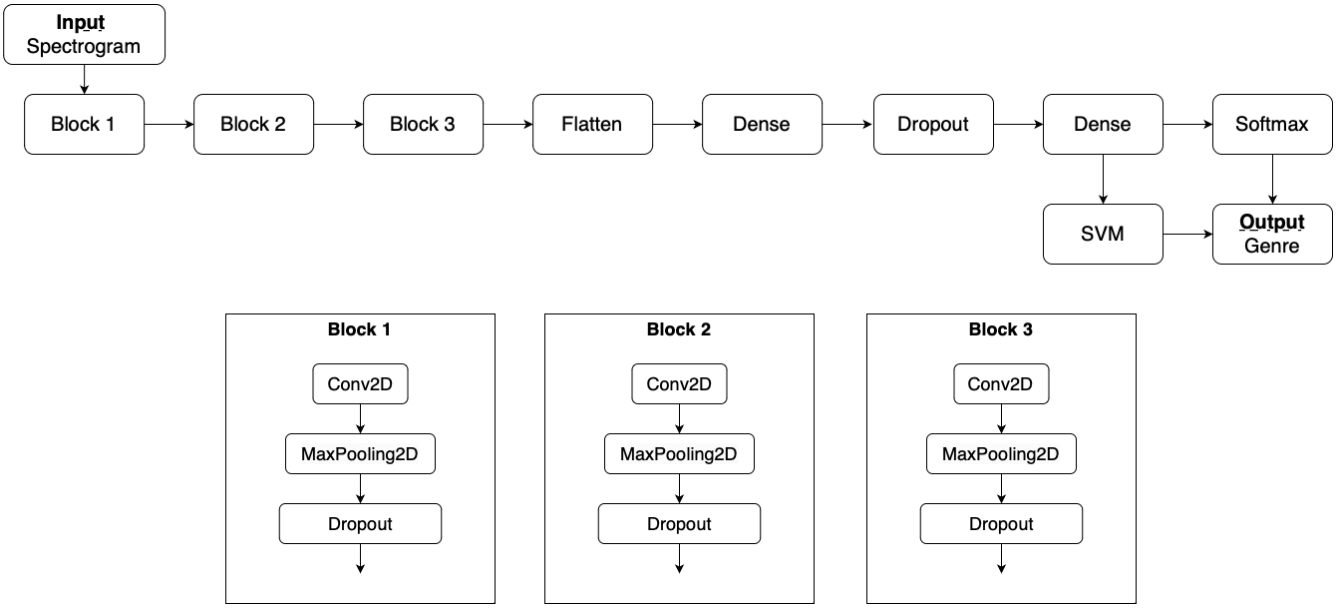


Figure 2: Visualization of the MusicRecNet [1] architecture. Output genres are either defined by using softmax probability scores or the SVM classifier.

4 Baseline: MusicRecNet

The network by Elbir, A & Aydin, N, called "MusicRecNet" [1] is not publicly available, but we have implemented our own network that has the same structure as described in [1] to be used as a

baseline network in our studies. We opt for this approach because [1] reported for MusicRecNet the highest accuracy for the GTZAN dataset. The network structure can be found in Figure 2. The network starts with a 2D convolutional layer, followed by a MaxPooling layer and a dropout layer. This set of 3 layers is repeated two more times, after which a flatten operation is performed. This flattened output is fed into a dense layer of size 128, with an additional dropout layer. The final layer is a dense layer for classification. This dense layer has a number of output nodes equal to the number of different labels in the dataset. The network processes input images and each output node receives a probability score. This score shows the probability that an image belongs to a certain class. The final prediction is given by choosing the node with the highest probability. The original parameter settings can be found in Table 2.

Parameter	**Value**
Input size	128x128
Batch size	64
Kernel size	3x3
Number of filters	32, 64, 128
Loss function	Categorical Cross Entropy
Optimizer	RMSProp
Trainable parameters	3,797,578

Table 2: Parameter configurations for our baseline network.

Elbir, A & Aydin, N also use MusicRecNet as a feature extractor and use its features as input for an SVM classifier. The output weights of the Dense(128) layer are used as sample features and fed into the SVM, as depicted in Figure 2. The SVM then classifies each sample based on those features.

5 Method

As mentioned earlier the goal of this thesis is to research the scalability of music genre classification algorithms The studied Neural Networks (NN) for music genre classification are the baseline MusicRecNet with and without SVM, and two NNs based on VGG-16 and EfficientNetV2B0, respectively. For our proposed NNs we use transfer learning to obtain the highest possible accuracies for the different music genre datasets described in Section 6. The NNs VGG-16 and EfficientNetV2B0 are used as a backbone feature extractor network in combination with a classifier. A visualization of this classifier can be seen in Figure 3. In this section, the model architectures are highlighted and a description of their implementation for this thesis is given.

5.1 VGG-16

The VGG-16 architecture by K. Simonyan & A. Zisserman [3] consists of 21 layers, of which 16 contain trainable weights. It has 138.4M parameters. The network is visualized in Appendix A, Figure 9. The layout of the network can be viewed as 5 blocks of layers. Each block consists of a

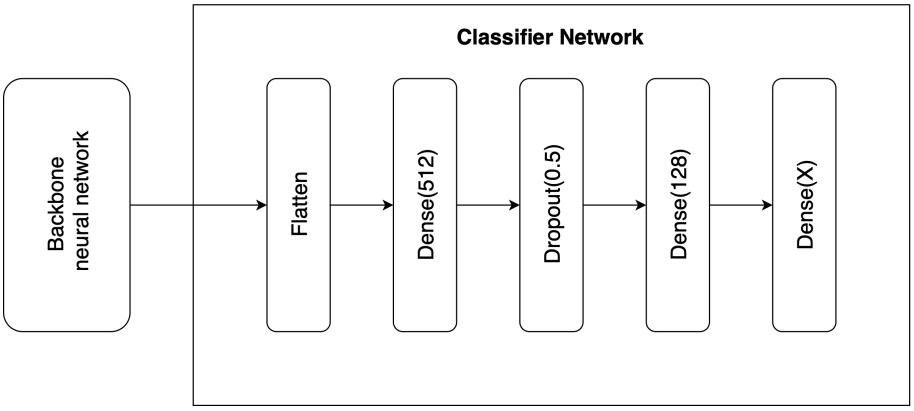


Figure 3: Visualization of the combination of a backbone neural network, e.g. VGG-16 or EfficientNetV2B0, and the classifier network. X denotes the number of unique classes in the dataset that is used. Images move through the network first, after which the output features are flattened and used in a classifier block. The chosen networks act as feature extractors.

set of two or three convolutional layers and a MaxPooling2D layer. The network is initiated with the weights that resulted after training on the ImageNet dataset. To use the network as a feature extractor for the music genre classification we freeze the first four blocks. These four blocks act as a feature extractor. The final block is set to trainable. This allows the network to train more features that are more catered to the spectrogram datasets.

5.2 EfficientNetV2B0

EfficientNetV2 is a family of CNNs by M. Tan and Q. V. Le [4]. The EfficientNetV2B0 configuration has 7.2M parameters, which is significantly less than VGG-16. This allows for more efficient training. The architecture can also be divided into 6 blocks of repeating layers. The network is visualized in Appendix A, Figure 10. In the described experiments for the EfficientNetV2B0 architecture the first 3 blocks will be used for feature extraction and the last 3 blocks will be trained on the dataset. The accuracy on the ImageNet [19] benchmark is 83.9%, which is higher than the VGG-16 architecture, which achieved an accuracy of 74.4%. The higher accuracy and lower amount of parameters are a reason to use this relatively recent and good-performing network, and see if it is capable to handle the GTZAN and different-sized FMA datasets better than the older VGG-16 model.

6 Datasets

To test the performance of the three different models, two datasets were used: the GTZAN dataset and the FMA dataset. The GTZAN dataset is widely used in the field of music genre classification. Its uniformly distributed genres make it a dataset that is well-suited for experiments, as each genre has the same number of examples. The FMA dataset was chosen because it contains more songs and more genres, which allows us to address the problem of scalability.

6.1 Preprocessing

The audio files are in .wav format and preprocessed using Librosa [20], a Python library. Similar to the work of Elbir, A & Aydin, N each audio file is split up into 5-second segments. After this, the Librosa library is used to convert the audio files to mel-spectrogram images. The images are resized to the desired input sizes of 128x128 and 224x224. This preprocessed image data is saved to a numpy array, so the data can be used to train and test the network. This preprocessing step is depicted in Figure 4.

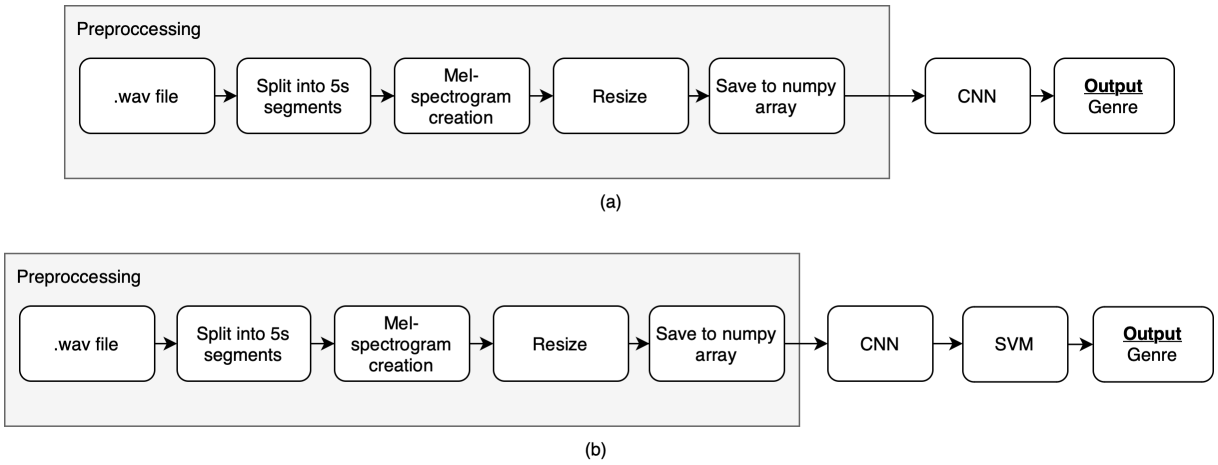


Figure 4: Pipeline for our baseline network. The images in the dataset are first preprocessed to make them suitable for use by the networks, which expect images as inputs. After this, the networks are trained on these images, in order to predict a genre for a given input image by outputting a probability for each of the possible genres. The output genre is equal to the class with the highest probability, by softmax (a) or SVM (b) output.

6.2 GTZAN

The GTZAN dataset was created by G. Tzanetakis and P. Cook [2], to be used within the field of Music Information Retrieval (MIR). The dataset consists of 1000 songs, each belonging to one of 10 genres: classical, country, metal, hip-hop, jazz, disco, pop, blues, reggae and rock. For each genre the dataset contains 100 songs that are classified as that genre, making it a balanced dataset. The dataset contains audio files that are 30 seconds long and also has feature data for each song. Feature data contains meta-information such as the calculated Mel-frequency cepstral coefficients and spectral roll-off information. For this study, we only use the generated Mel-spectrogram images for genre classification.

6.3 FMA

In 2017, M. Defferrard et al. [5] published another dataset suited for the field of MIR, called the Free Music Archive (FMA) dataset. This dataset was built to overcome the issue that audio datasets up to that time were relatively small. The FMA dataset contains 106,574 tracks, distributed over 161 genres. Metadata for the audio files is also provided, expanding on the GTZAN dataset, offering

not only data about the raw audio file, but also artist information. The set can be used in different sizes: small, medium, large, and full. The small version contains 8000 songs, uniformly distributed over 8 genres. The medium version contains 25000 songs, of which each belongs to 1 of 16 genres. It is important to note that the distribution of those songs across the genres is not uniform. In order to compare the performance of FMA to GTZAN we created a subset of the FMA medium dataset that contains 14 genres. These 14 genres each have 100 songs, similar to GTZAN. The genres "blues" and "easy-listening" were omitted from the original set as those contain less than 100 songs per genre. The FMA subsets will be referred to as FMA_8 and FMA_14, for their respective amount of genres. Genre composition and dataset size across the GTZAN dataset and the FMA subsets can be found in Tables 3 and 4.

Dataset	**GTZAN**	**FMA_8**	**FMA_14**	**FMA medium**
Genre				
Electronic		x	x	x
Experimental		x	x	x
Folk		x	x	x
Hip-hop	x	x	x	x
Instrumental		x	x	x
International		x	x	x
Pop	x	x	x	x
Rock	x	x	x	x
Jazz	x	x	x	x
Old-Time / Historic			x	x
Soul-RnB			x	x
Spoken			x	x
Classical	x		x	x
Country	x		x	x
Blues	x			x
Easy-listening				x
Metal	x			
Reggae	x			

Table 3: Overview of the genre composition of the different datasets used in this study.

7 Experimental Setup

In this paper, the influence of dataset scaling on the performance of neural network models for genre classification is researched. There are two sets of experiments. The first set of experiments has been conducted using the baseline network and the GTZAN and FMA datasets, to establish

Dataset	GTZAN	FMA_8	FMA_14	FMA medium
Number of songs per genre	100	1000	100	21-7103
Total number of songs	1000	8000	1400	25000

Table 4: Overview of the songs per genre and the total amount of songs in the different datasets used in this study.

the performance on those datasets. The second set of experiments tries to answer the following three research questions:

- How does increasing the amount of samples affect model performance?
- How does increasing the amount of genres affect model performance?
- How does adding an SVM affect model performance?

The experiments are conducted using Google Colab Pro, using their GPU runtime, and run for 100 epochs unless differently specified.

7.1 Baseline experiments

The goal of the baseline experiments was to find the optimal selection of values for a set of parameters of the baseline network. The parameters that have been explored are: optimizer, batch size, number of epochs, and learning rate. The different configurations can be found in Table 5. For each experiment the initial default optimizer was RMSprop, the default batch size was 64, the default learning rate was 0.001 and the default number of epochs was 30.

Parameter	Values
Optimizer	[Adam, Adadelta, RMSprop]
Batch size	[16, 32, 64]
Number of epochs	[10, 25, 50]
Learning Rate	[0.0001, 0.001, 0.01]

Table 5: Parameters and their explored values in the parameter optimization experiments.

7.2 Scalability experiments

To determine the effects of scaling a dataset on the accuracy of models we use a set of architectures on the GTZAN and FMA datasets. The initial focus is on scaling depth-wise: increasing the amount of samples a certain genre has. The FMA dataset offers more samples per genre. The MusicRecNet-inspired network is trained and used on both datasets, to establish a baseline performance for a state-of-the-art model. In this experiment, the accuracies of GTZAN and FMA are compared.
As an alternative approach two pre-trained models, VGG-16 [3] and EfficientNetV2B0 [4] are used. The two networks are used as feature extractors in a set of configurations and combinations:

1. Network as feature extractor
2. Network as feature extractor, with an SVM after the last classifier layer
3. Network as feature extractor, with an SVM after the second to last classifier layer
4. Network as feature extractor, but with the last block(s) unfrozen
5. Network as feature extractor, but with the last block(s) unfrozen and an SVM after the last classifier layer
6. Network as feature extractor, but with the last block(s) unfrozen and an SVM after the second to last classifier layer

The addition of a classifier such as SVM was inspired by [1], as their paper shows an increase from 81.8% to 97.6% accuracy. Using the SVM after the second to last classifier layer, a Dense(128) layer, gives the SVM more data points to work with. The different implementations of the SVM are shown in Figure 5. The effect of this is examined. Setting the last block(s) of a model to trainable is suggested by [21], as an accuracy-improving measure. Allowing the network weights to train on the given dataset should help with classification. Input size is also varied for a set of experiments to see if this affects performance as well.
Finally, the effect of increasing the amount of genres is examined. The subset of FMA using 14 genres is used. A comparison is made between performance on the GTZAN dataset and the FMA dataset, using the best-performing model.

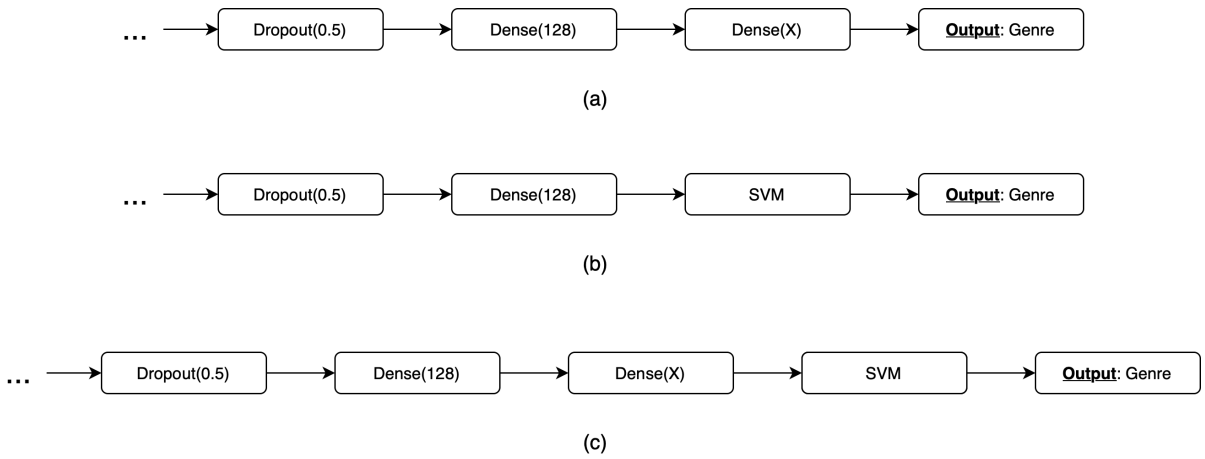


Figure 5: Visualization of the described ways of classification with and without SVM. The genre can be classified by using a Dense layer that has a number of nodes equal to the amount of classes (a). Adding an SVM to the second to last Dense(128) layer results in more features per sample (b). The final implementation is using an SVM at the end of the Dense layers (c).

8 Experimental Results

In this section, the results of the described experiments are given. We try to validate the baseline method and compare our performance to the original paper and other models. In Section 8.2

performance results are given for the baseline, VGG-16, and EfficientNetV2B0 models across the three datasets GTZAN, FMA_8, and FMA_14.

8.1 Baseline experiments

The goal of these experiments was to find the optimal hyperparameters for the baseline network and establish its performance on the GTZAN dataset. The goal is to validate the baseline architecture and its original performance. The accuracies for our implementation of MusicRecNet and other models can be found in Table 6.

Model	*GTZAN Accuracy*
Zhang et al. [11]	87.4%
Liu et al. [12]	93.9%
Elbir, A & Aydin, N. [1]	81.8%
Elbir, A & Aydin, N. with SVM [1]	**97.6%**
Our Baseline Implementation	81.0%
Our Baseline Implementation + SVM	81.6%

Table 6: Accuracies on GTZAN across state-of-the-art models and our own baseline MusicRecNet implementation.

After conducting a set of experiments to find the optimal hyperparameter values described in Table 5 we found the optimal optimizer to be RMSProp, with a batch size of 32 when using GTZAN and 64 when using FMA, for 100 epochs and a learning rate of 0.01. Using these hyperparameter values our implementation of the MusicRecNet architecture was able to achieve a similar level of accuracy as the described method by Elbir, A. & Aydin, N.[1]. Our model without SVM achieved 81.0%, whereas Elbir, A & Aydin, N achieved 81.6%. Their model with SVM got significantly higher results, whereas our model only improved to 81.6%. Our implementation was created based on information from the original paper and a meeting with one of the authors, Elbir, A. The cause of the difference in performance is unclear.

8.2 Scalability Experiments

In this section the results for the experiments as described in Section 7.2 are given.

The goal of these experiments is to measure the effect of scaling a dataset on model accuracy performance. The accuracies for all combinations of datasets and models can be found in Table 7.

Dataset	GTZAN	GTZAN 224	FMA_8	FMA_8 224	FMA_14
Method					
Baseline	81.0		68.6		42.0
Baseline-SVM-Output	81.5		70.7		42.0
Baseline-SVM-D128	81.6		72.1		42.6
VGG	73.1		53.4		53.6
VGG-SVM-Output	73.0		53.9		53.6
VGG-SVM-D128	76.5		54.4		54.8
VGG-FT	81.6		60.7		57.3
VGG-SVM-Output-FT	81.6		61.0		57.3
VGG-SVM-D128-FT	83.0		61.2		56.9
EfficientNet	80.0	82.1	59.6	62.0	56.8
EfficientNet-SVM-Output	80.6	82.5	60.5	63.0	56.5
EfficientNet-SVM-D128	83.0	87.5	61.4	63.1	60.8
EfficientNet-FT	90.0	90.5	76.9	73.8	60.4
EfficientNet-SVM-Output-FT	89.8	90.5	76.8	73.7	60.4
EfficientNet-SVM-D128-FT	**90.3**	**90.8**	**77.4**	**73.9**	**61.1**

Table 7: Accuracies for the different models across the different datasets. Models have been used as-is, or with finetuning (denoted as FT). The 224 suffix indicates that the used dataset consists of images that are of size 224x224. SVM accuracies were achieved using an "rbf" kernel. The SVM has been applied in two ways: after the last output Dense layer (denoted as SVM-Output) or after the second to last Dense(128) layer (denoted as SVM-D128). Models that use finetuning have been denoted with the -FT suffix.

It can be seen that accuracies for the FMA_8 dataset are lower than those of GTZAN, across the different models used. The accuracies for FMA_14 are lower than those of FMA_8 and GTZAN.

The effect of increasing the amount of samples on model performance can be measured by comparing the different datasets. The FMA_8 subset has fewer genres but more samples per genre than GTZAN (see Table 4). Accuracies for the FMA_8 dataset are lower for all models tested.

The effect of increasing the amount of genres can be measured by comparing the FMA_14 dataset to the other datasets. Moving from 8 or 10 to 14 different genres results in lower accuracies across all models. In our experiments increasing the amount of different genres seems to have a bigger effect on accuracies than increasing the amount of samples.

Table 7 shows that for all models, adding an SVM leads to higher accuracies, with improvements of up to 5.4%. In most cases, the accuracy gains were highest when using the SVM after the Dense(128) layer. This was to be expected, as the SVM will have more data points per sample to work with. Performance gains through the use of an SVM are highest when used in combination with the VGG-16 and EfficientNetV2B0 models. For the baseline network, the use of an SVM has the largest impact when used with the FMA_8 dataset.

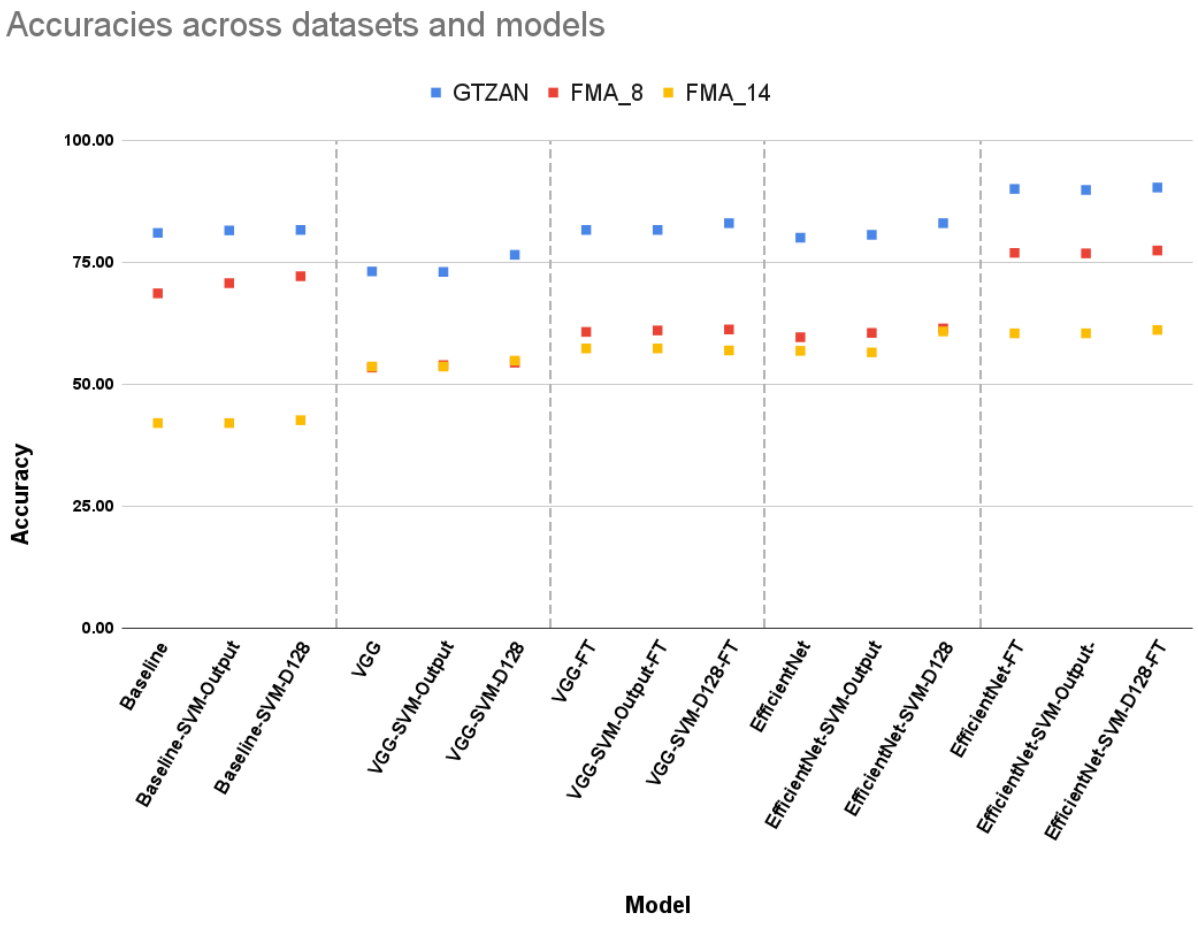
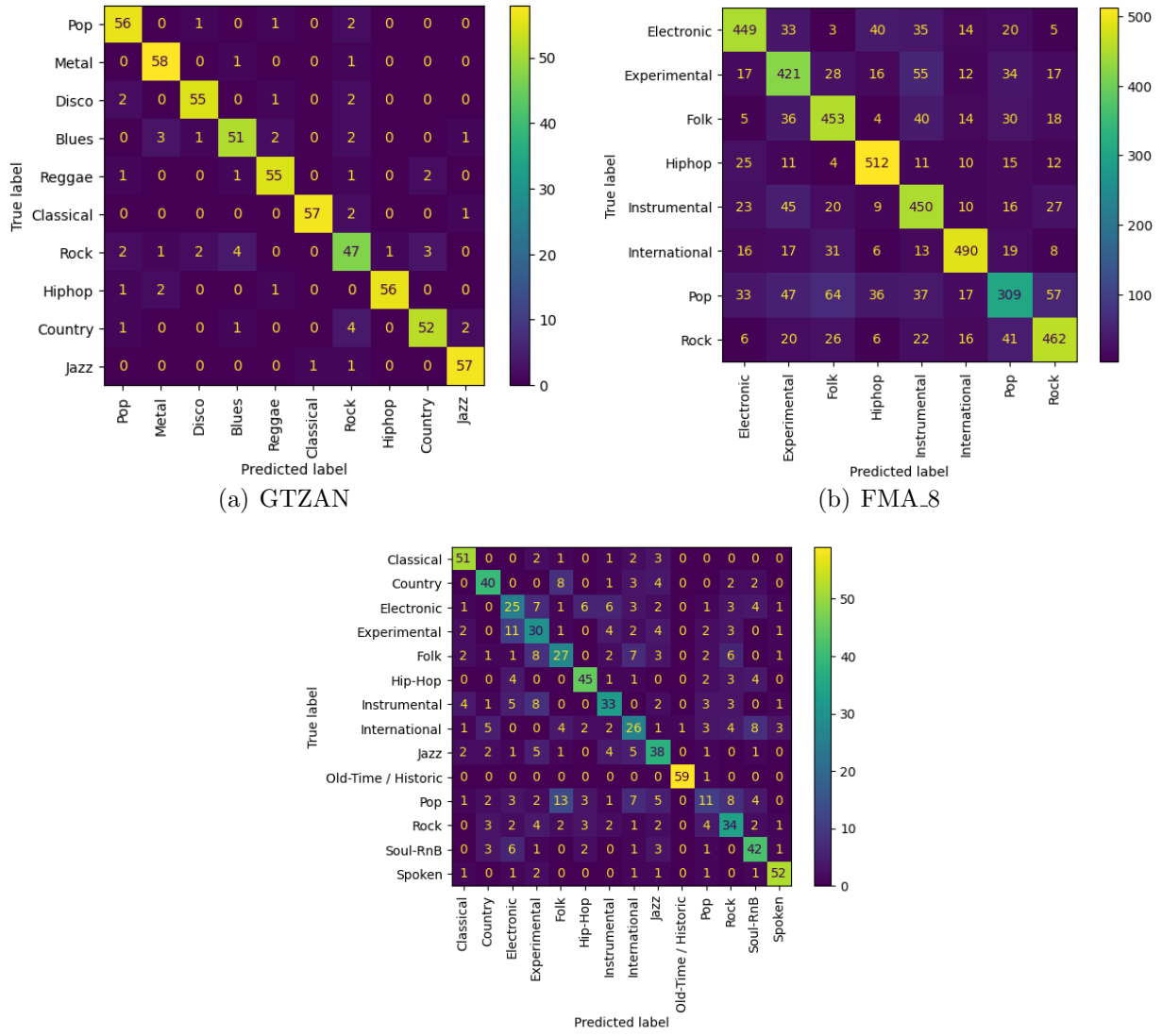


Figure 6: Accuracies from Table 7 visualized.

Finetuning however has a bigger effect on accuracies across the models tested. VGG-16 performance on the GTZAN dataset goes up from 73.0% to 81.6% and EfficientNetV2B0 performance on the GTZAN dataset goes up from 80.0% to 90.0%. Similar accuracy gains are shown across models when used on the FMA_8 dataset.

Figure 6 shows a graph of the values from Table 7. This figure is used to illustrate the scalability of the different models. The accuracy difference between the GTZAN and FMA_8 datasets is the smallest for the baseline network using an SVM on the second to last Dense(128) layer. In terms of scalability however, the EfficientNetV2B0 model in combination with an SVM on the second to last Dense(128) shows a similar gap, but with higher accuracies for both GTZAN and FMA_8. This makes this the best scalable method when it comes to a dataset that uses more samples per genre.



(a) GTZAN (b) FMA_8

(c) FMA_14

Figure 7: Confusion matrices for the EfficientNet-SVM-D128-FT model across the GTZAN, FMA_8 and FMA_14 datasets.

A comparison between the GTZAN and FMA_14 datasets shows the scalability in terms of increasing the amount of genres. The difference is smallest for the VGG-16 network here, but EfficientNetV2B0 again has higher accuracies for both GTZAN and FMA_14. This makes it the better method for scaling in terms of using a dataset with more genres.

Accuracies can be further explored by looking at confusion matrices for the three datasets. Figure 7 shows the confusion matrices for the best-performing model across the datasets: EfficientNet-SVM-D128-FT. Model performance on the GTZAN dataset is between 78.3% and 96.6% for all genres, with Rock being the worst-performing genre and Metal being the best-performing genre. FMA_8 performance shows accuracies in the range 51.5%-85.3%. The lowest-scoring genre is "Pop". After listening to some of the pop samples in the dataset this low accuracy can be explained: the pop songs are not distinctly pop songs, especially in terms of their frequencies, and can probably more easily be mistaken for other genres. We see that it often gets mistaken for Folk and Rock. The best-performing genre in this set is hip-hop.

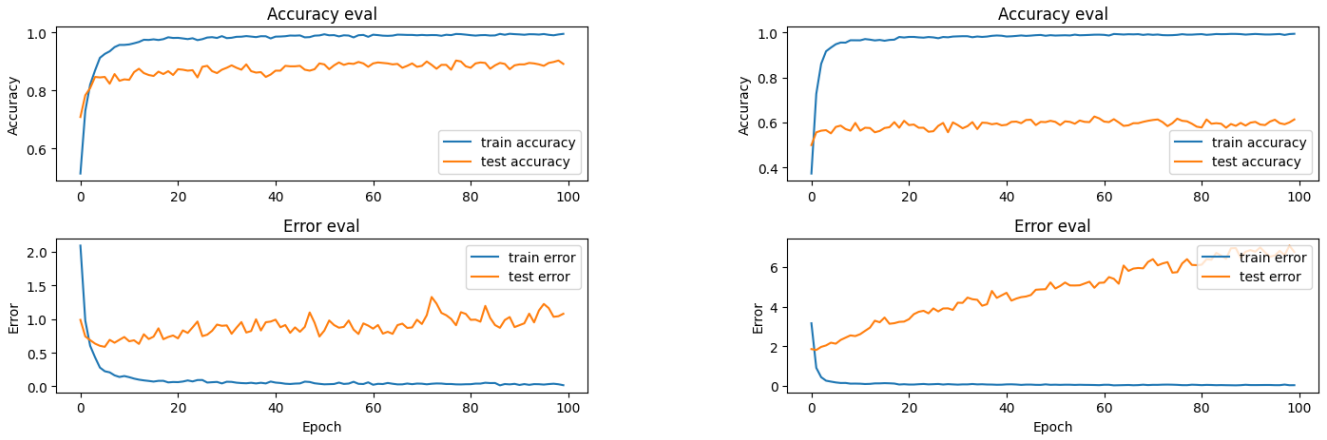


Figure 8: EfficientNetV2B0 training curves for GTZAN (left) and FMA_14 (right) datasets.

A similar trend can be seen when looking at the FMA_14 set. Accuracies range from 18.3% for Pop to 98.3% for Old-Time/Historic. The Pop class is again often mistaken for Folk. The average accuracy is lower across the set. A possible explanation for this is that the network is overfitting: it memorizes the training data and is less good at generalizing, causing the network to be worse at recognizing the test data, which it has never seen before. Figure 8 shows a comparison between the train and test accuracy and loss functions during training for the EfficientNetV2B0 architecture, for the GTZAN and FMA_14 datasets. The GTZAN curves show regular behavior: train and test accuracies are relatively close to each other and test loss is around 1. The FMA_14 curves however show signs of overfitting: the test accuracy is much lower than the training accuracy and the loss function keeps increasing over time.

Additional precision, recall, and F1 scores for the best-performing variations of the used models are available in Appendix J. A quick analysis shows no irregularities and the scores are in line with the achieved accuracies.

9 Conclusions and discussion

In this paper, we have looked at the scalability of different music genre classification algorithms. We compared our baseline implementation of MusicRecNet [1] to music genre classification models using VGG-16 [3] and EfficientNetV2B0 [4] as feature extraction backbones on the GTZAN [2] and FMA_8 and FMA_14 [5] datasets. Our implementation of MusicRecNet achieved similar levels of performance on the GTZAN set without using an SVM classifier: 81.0%. We were not able to reproduce the 97.6% given for use with an SVM in the original paper, as our baseline implementation, based on the paper and a conversation with one of the paper authors, Elbir, A., achieved an accuracy of 81.6%. Finetuning seems to have a bigger effect on accuracies than the addition of an SVM classifier.
The baseline network was also trained on the FMA_8 and FMA_14 datasets to illustrate the effect of scaling on model accuracies. A larger number of samples resulted in a lower accuracy across the three models. A larger number of genres also resulted in a lower accuracy across the three models. The best-performing model is the EfficientNetV2B0 model when used with a form of fine-tuning by unfreezing the last three blocks of layers and in combination with an SVM classifier that used the output of the second-to-last Dense(128) layer. In our experiments this model achieved the highest accuracy on the GTZAN dataset: 90.3%. It also showed the best scalability, with its accuracies being the highest and not having the biggest difference in those accuracies between datasets. Its accuracy levels also exceeded expected human performance [10] for the GTZAN and FMA_8 datasets.

Accuracies for the FMA_14 are lower than those for GTZAN and FMA_8. As mentioned before this could be due to overfitting and the fact that the "pop" class contains music that is not distinctive enough to be classified within one genre. The FMA_14 subset was created by taking a subset of the FMA Medium subset, that contains 16 genres and 25000 samples. Due to the non-uniform distribution of samples per genre, the first 100 songs were picked per genre. This method might have affected model performance, as the first 100 songs might have been less distinctively classifiable as "pop".
It is also worth mentioning that the GTZAN dataset is a widely used benchmark dataset in this field, but it might be less suited to conduct research on scalability, as it is a relatively small dataset both in terms of samples and classes.

The FMA_14 subset that we created is similar to GTZAN because it also has 100 samples per genre. In order to improve performance for future research it would be beneficial if a bigger dataset existed, with at least 1000 samples per genre, similar to the FMA_8 subset. Future research could also look into preventing overfitting, by applying more or stronger dropout layers.

A final recommendation for future research is that the effect of sample length can be investigated and used more. Songs often belong to a particular genre because of repetitive parts in the song that make it a distinctive genre. A combination of different parts of songs might be beneficial to classification. For this, a dataset that contains complete songs would be ideal.

References

[1] A. Elbir and N. Aydin, "Music genre classification and music recommendation by using deep learning," *Electronics Letters*, vol. 56, no. 12, pp. 627–629, 2020.

[2] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," ICLR, 2015.

[4] M. Tan and Q. Le, "Efficientnetv2: Smaller models and faster training," in *International conference on machine learning*, pp. 10096–10106, PMLR, 2021.

[5] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.

[6] "About spotify." https://investors.spotify.com/about/, Accessed 24-01-2023.

[7] T. Ingham, "100,000 tracks being uploaded to spotify and other services daily," Oct 2022. https://www.musicbusinessworldwide.com/its-happened-100000-tracks-are-now-being-uploaded/ , Accessed 24-01-2023.

[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[10] R. O. Gjerdingen and D. Perrott, "Scanning the dial: The rapid recognition of music genres," *Journal of new music research*, vol. 37, no. 2, pp. 93–100, 2008.

[11] W. Zhang, W. Lei, X. Xu, and X. Xing, "Improved music genre classification with convolutional neural networks.," in *Interspeech*, pp. 3304–3308, 2016.

[12] C. Liu, L. Feng, G. Liu, H. Wang, and S. Liu, "Bottom-up broadcast neural network for music genre classification," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 7313–7331, 2021.

[13] C. Zhang, Y. Zhang, and C. Chen, "Songnet: Real-time music classification," 2019.

[14] S. Chillara, A. Kavitha, S. A. Neginhal, S. Haldia, and K. Vidyullatha, "Music genre classification using machine learning algorithms: a comparison," *International Research Journal of Engineering and Technology*, vol. 6, no. 5, pp. 851–858, 2019.

[15] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011.

[16] W. Cochran, J. Cooley, D. Favin, H. Helms, R. Kaenel, W. Lang, G. Maling, D. Nelson, C. Rader, and P. Welch, "What is the fast fourier transform?," *Proceedings of the IEEE*, vol. 55, no. 10, pp. 1664–1674, 1967.

[17] S. S. Stevens and J. Volkmann, "The relation of pitch to frequency: A revised scale," *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940.

[18] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[20] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th Python in Science Conference*, vol. 8, pp. 18–25, 2015.

[21] F. Chollet, *Deep Learning with Python*. Manning, Nov. 2017.

Appendix

A Network Architectures

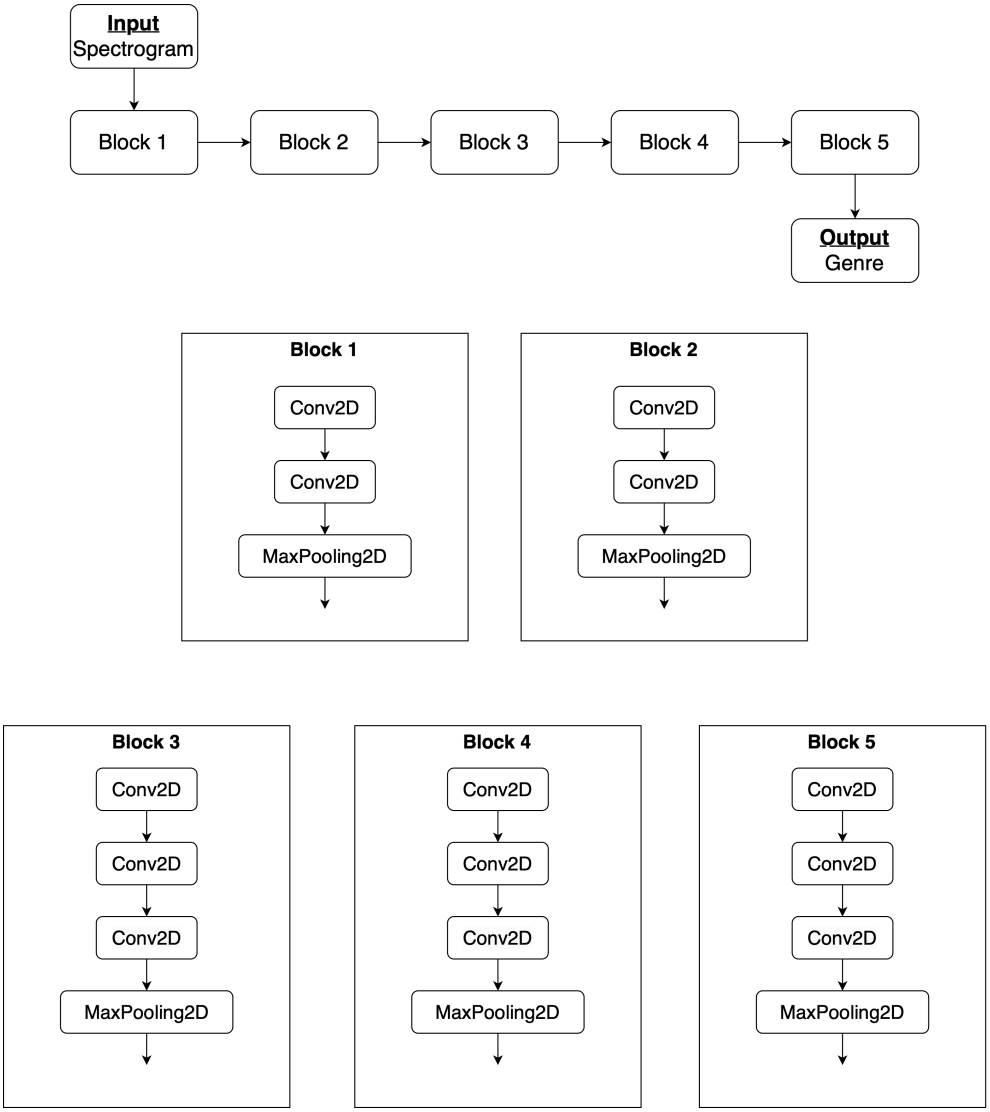


Figure 9: Visualization of the VGG-16 [3] architecture.

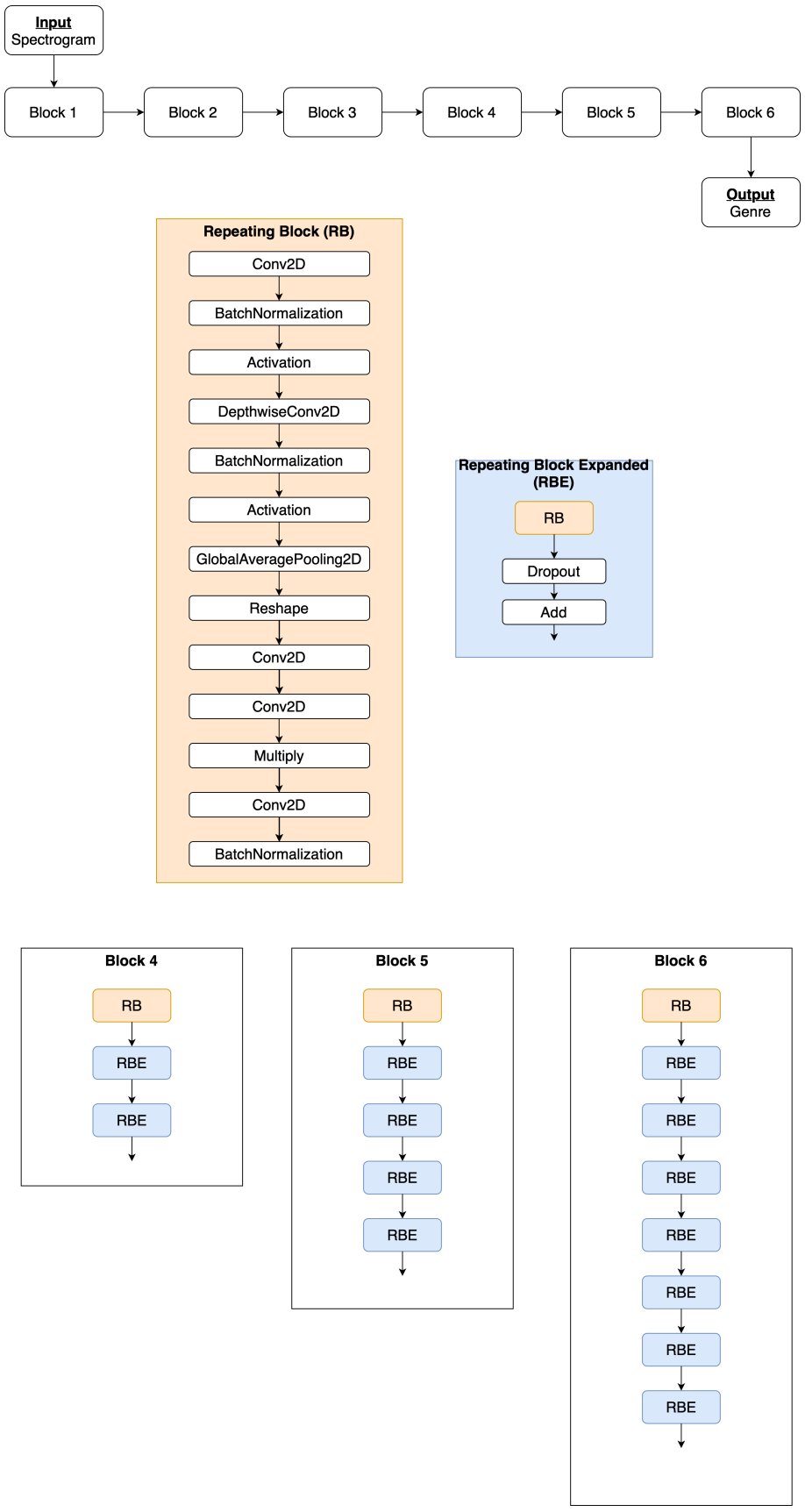


Figure 10: Visualization of the EfficientNetV2B0 [4] architecture.

B GTZAN Baseline verification experiment results

This appendix contains the accuracy and loss values for the exploration of different hyperparameters. Experiments were conducted using our implementation of the MusicRecNet network using the GTZAN dataset.

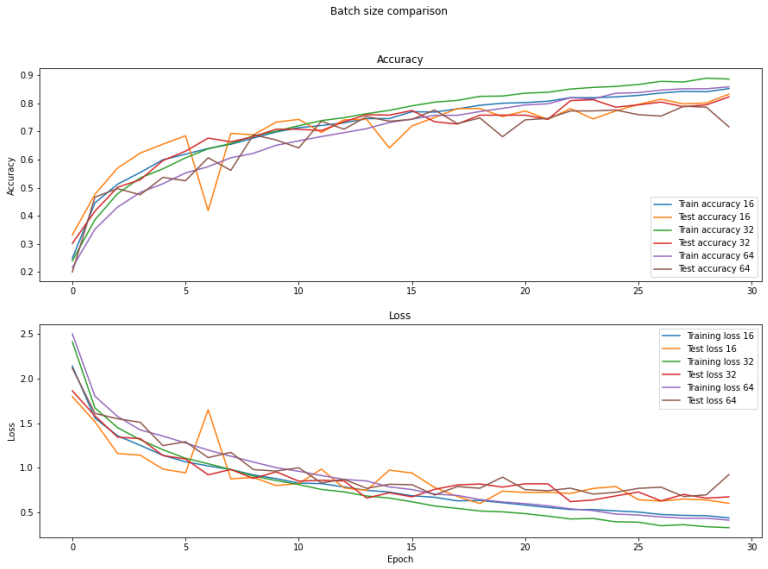


Figure 11: Accuracy and loss across different values for batch size, on the baseline network. A batch size of 32 yields the most stable high accuracy.

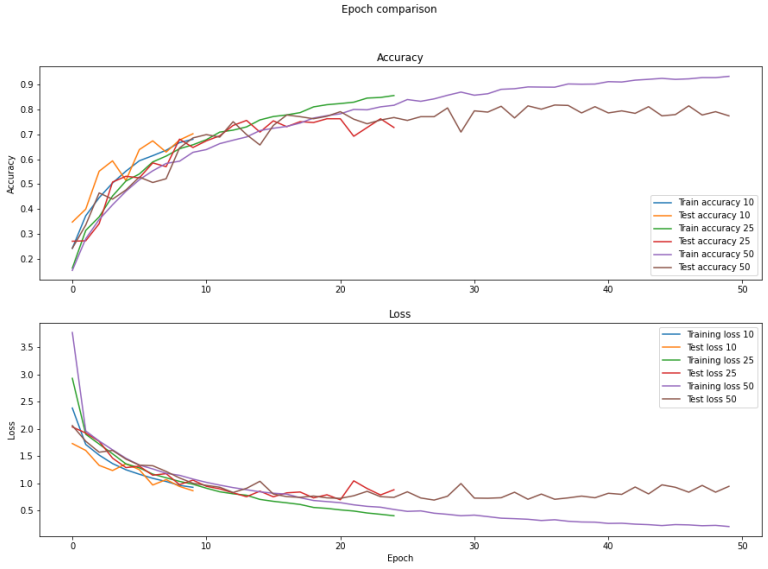


Figure 12: Accuracy and loss across different values for the amount of epochs, on the baseline network. In this set of experiments the highest amount of epochs, 50, yielded the highest accuracy.

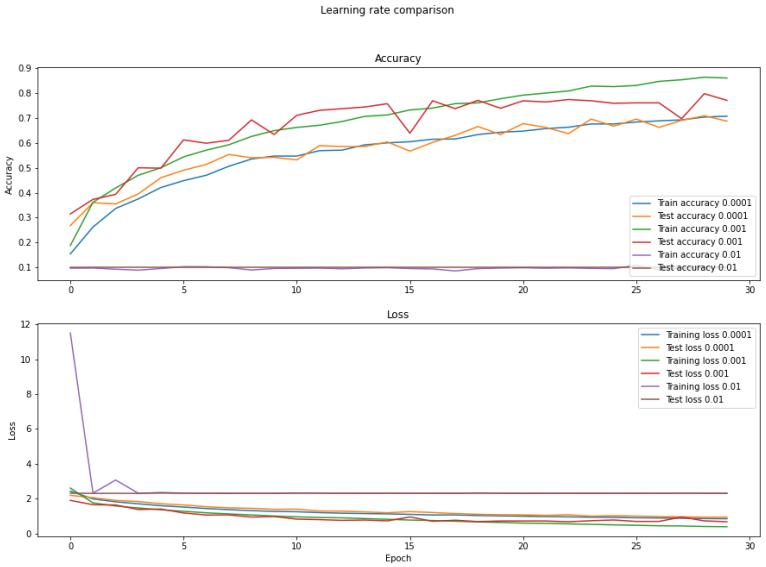


Figure 13: Accuracy and loss across different values for learning rate, on the baseline network. The default learning rate of 0.001 yielded the highest accuracy.

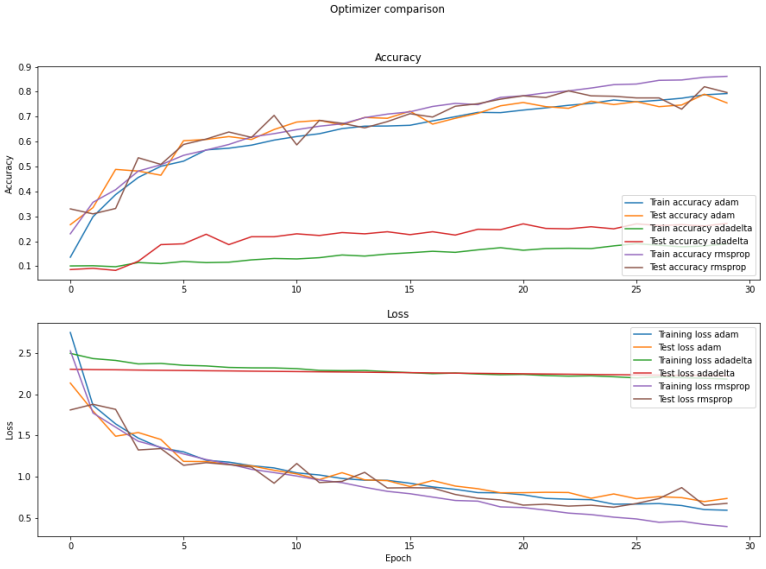


Figure 14: Accuracy and loss across different types of optimizers, on the baseline network. The RMSProp optimizer yielded the highest accuracy.

C FMA Baseline network Experiment results

This appendix contains the accuracy and loss values for the exploration of different hyperparameters. Experiments were conducted using our implementation of the MusicRecNet network using the FMA dataset.

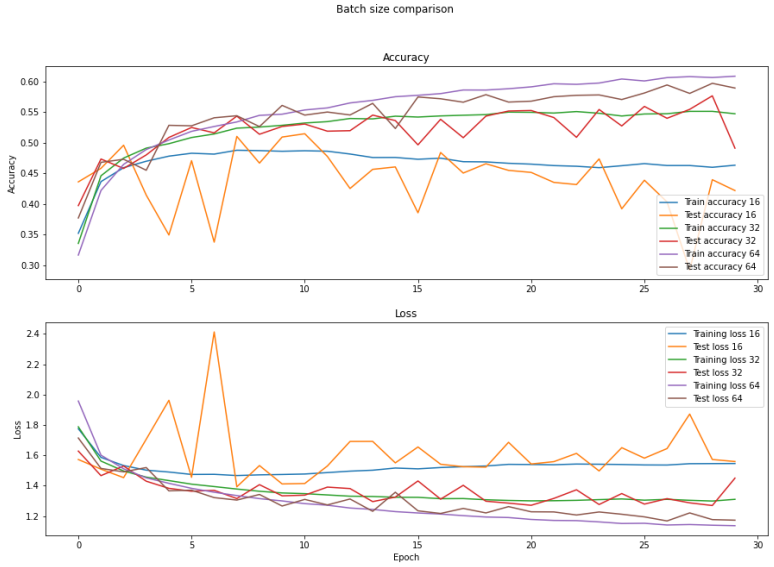


Figure 15: Accuracy and loss across different values for batch size, on the baseline network. A batch size of 64 yields the highest accuracy.

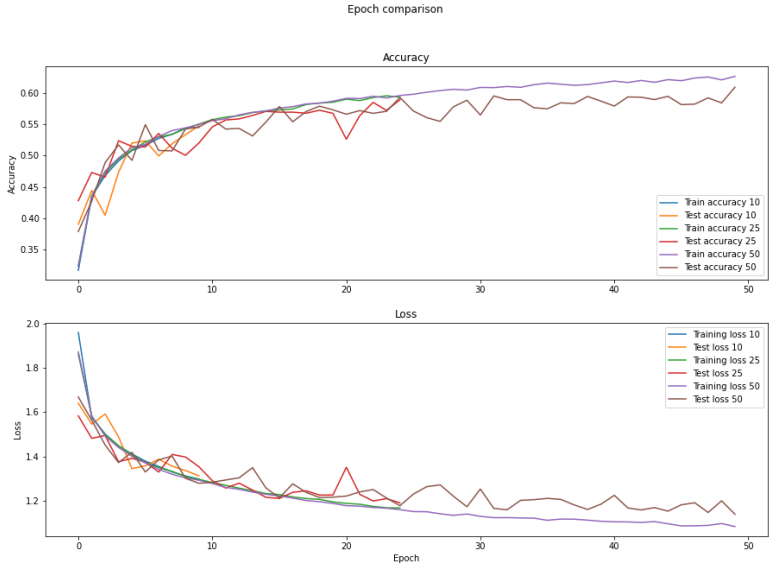


Figure 16: Accuracy and loss across different values for the amount of epochs, on the baseline network. In this set of experiments the highest amount of epochs, 50, yielded the highest accuracy.

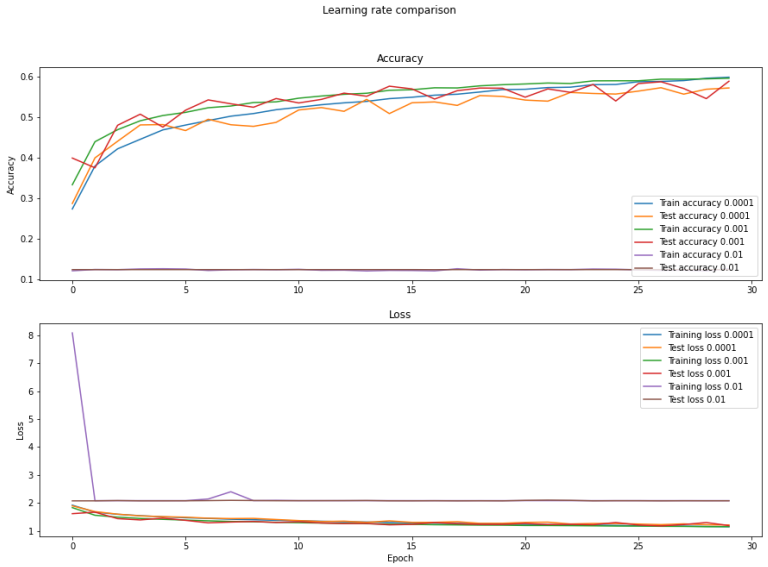


Figure 17: Accuracy and loss across different values for learning rate, on the baseline network. The default learning rate of 0.001 yielded the highest accuracy.

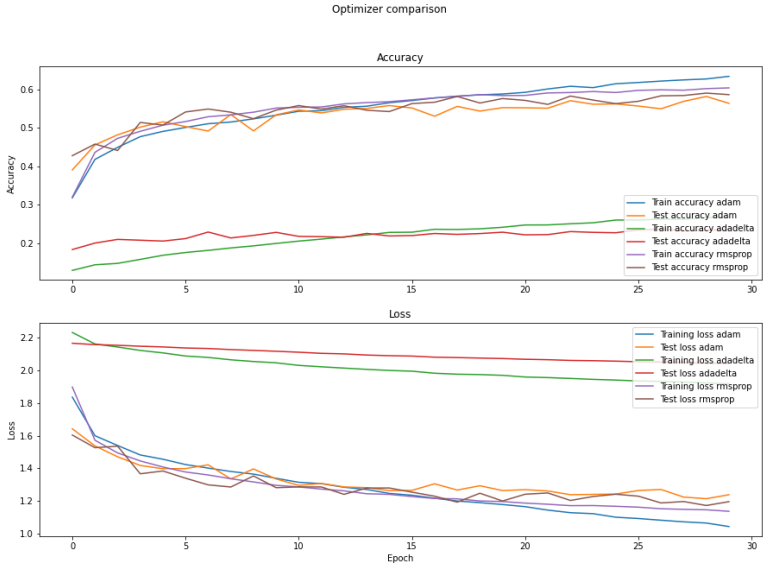
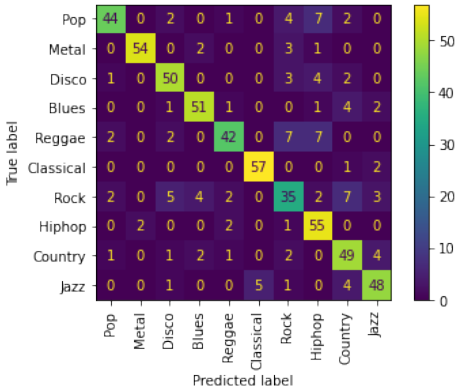


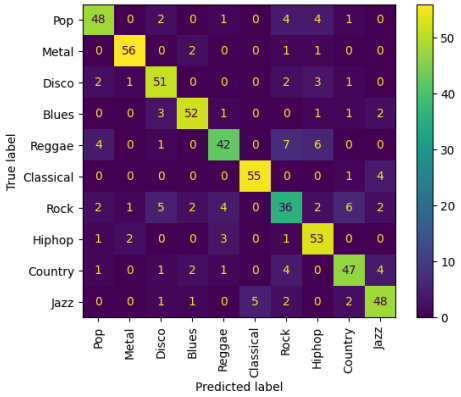
Figure 18: Accuracy and loss across different types of optimizers, on the baseline network. The RMSProp optimizer yielded the highest accuracy.

D Confusion matrices for GTZAN, input size 128x128

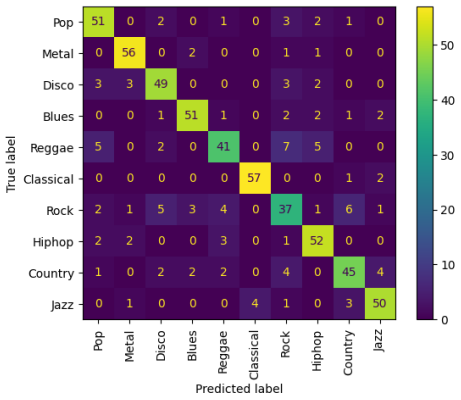
This appendix contains confusion matrices for the models used on the GTZAN (128x128 input) dataset.



(a) Baseline

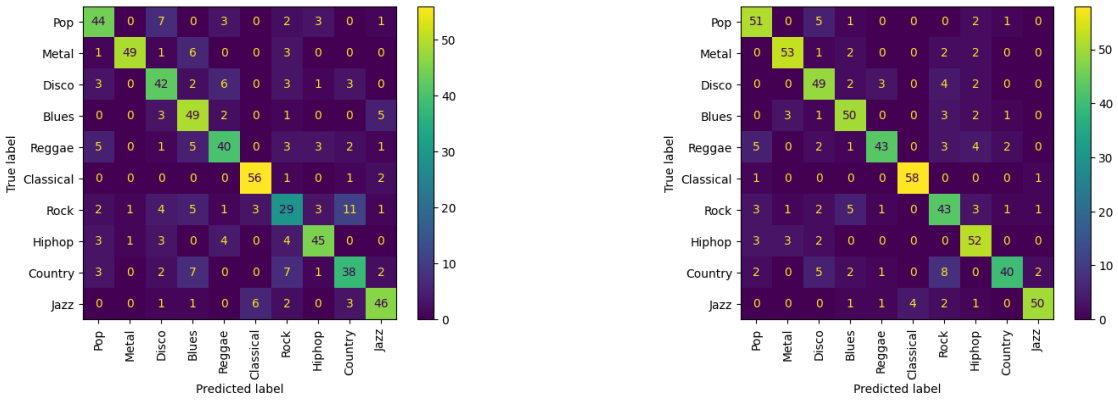


(b) Baseline + SVM Dense(10)



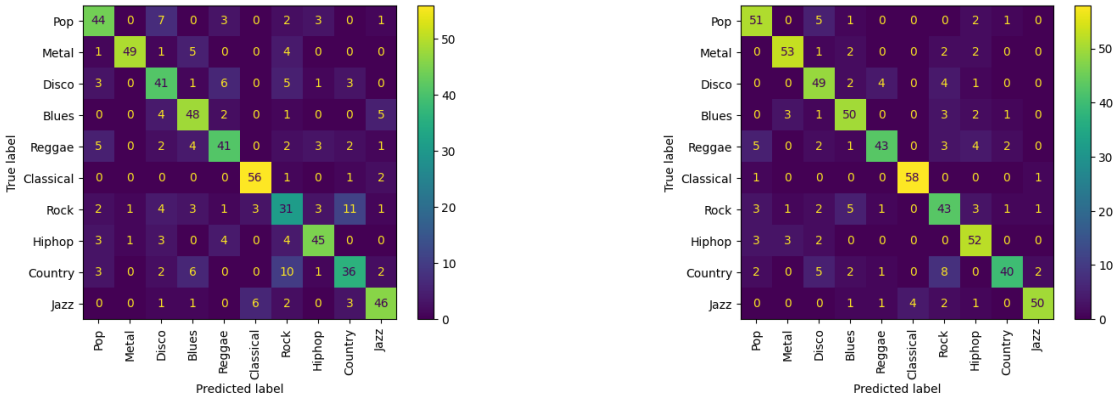
(c) Baseline + SVM Dense(128)

Figure 19: Confusion matrices on the GTZAN dataset, for 128x128 input images, for the Baseline network



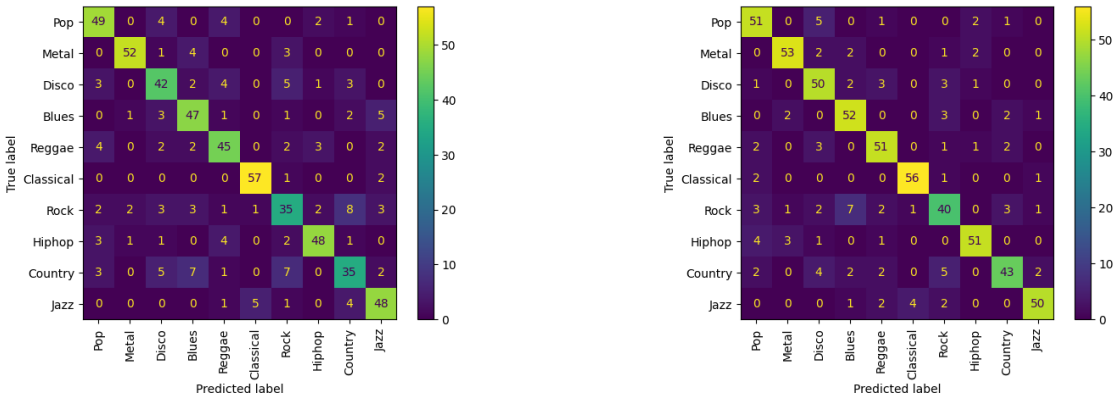
(a) VGG-16

(b) VGG-16_FT



(c) VGG-16 + SVM Dense(10)

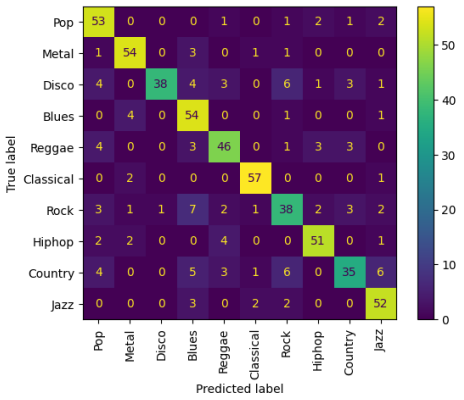
(d) VGG-16_FT + SVM Dense(10)



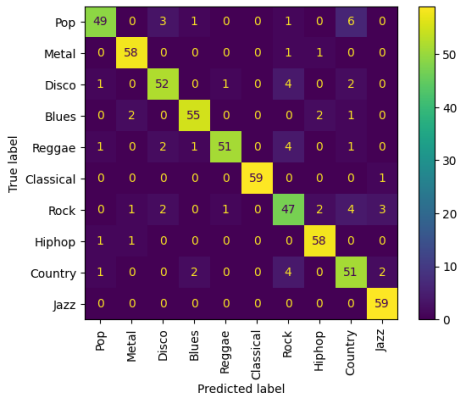
(e) VGG-16 + SVM Dense(128)

(f) VGG-16_FT + SVM Dense(128)

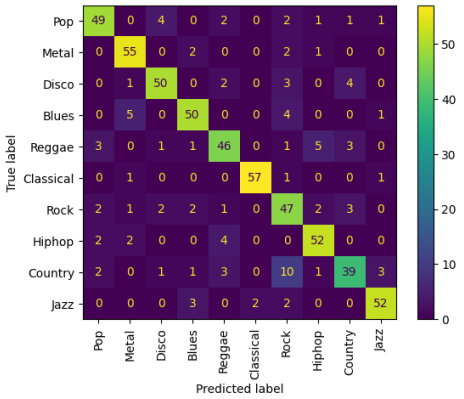
Figure 20: Confusion matrices on the GTZAN dataset, for 128x128 input images, for the VGG-16 and VGG-16(with finetuning, denoted as VGG-16_FT) networks.



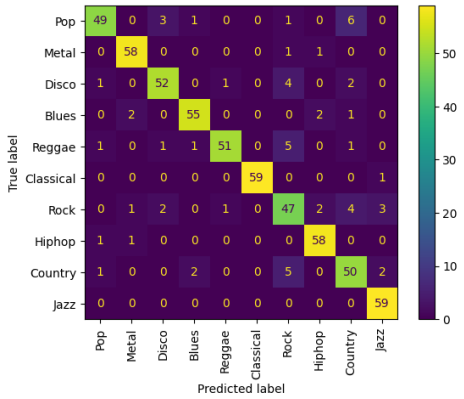
(a) EfficientNetV2B0



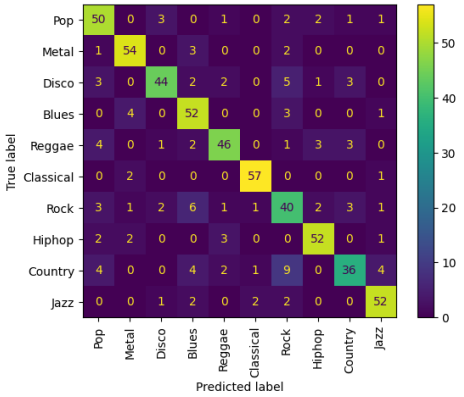
(b) EfficientNetV2B0_FT



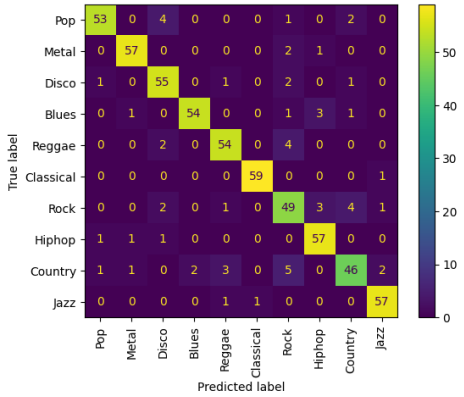
(c) EfficientNetV2B0 + SVM Dense(10)



(d) EfficientNetV2B0_FT + SVM Dense(10)



(e) EfficientNetV2B0 + SVM Dense(128)

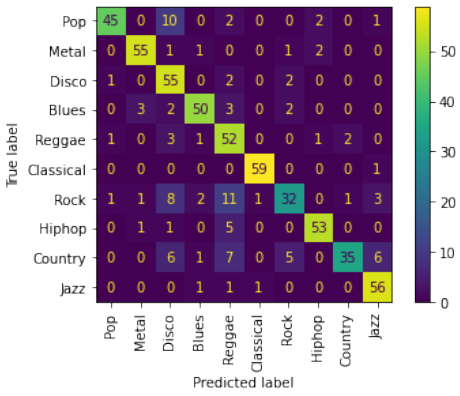


(f) EfficientNetV2B0_FT + SVM Dense(128)

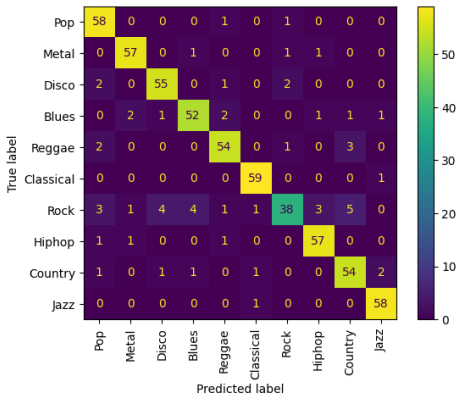
Figure 21: Confusion matrices on the GTZAN dataset, for 128x128 input images, for the EfficientNetV2B0 and EfficientNetV2B0(with finetuning, denoted as EfficientNetV2B0_FT) networks.

E Confusion matrices for GTZAN, input size 224x224

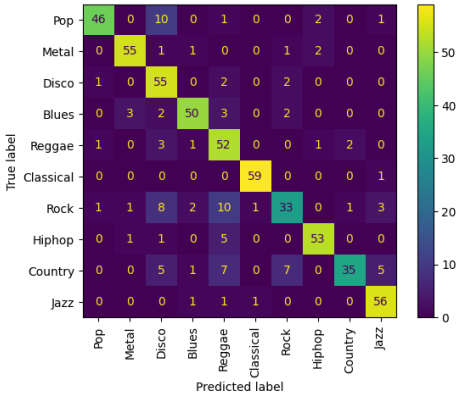
This appendix contains confusion matrices for the models used on the GTZAN (224x224 input) dataset.



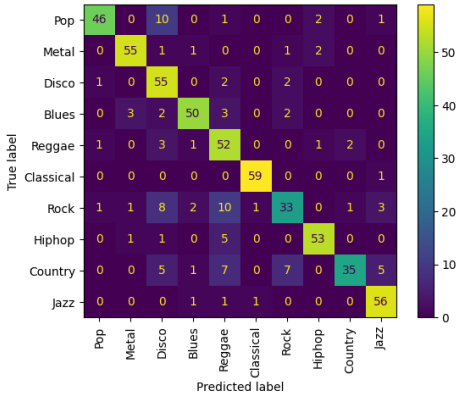
(a) EfficientNetV2B0



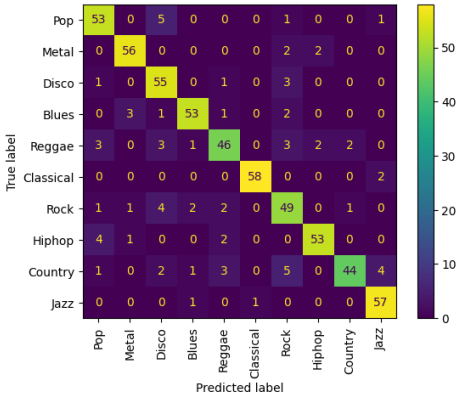
(b) EfficientNetV2B0_FT



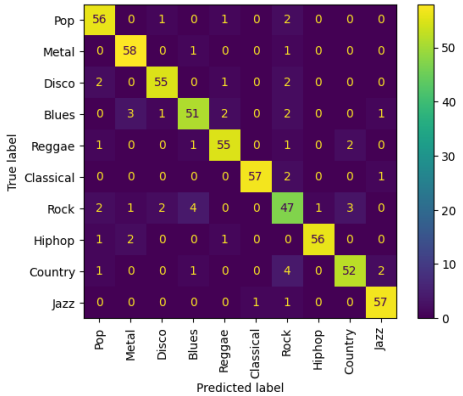
(c) EfficientNetV2B0 + SVM Dense(10)



(d) EfficientNetV2B0_FT + SVM Dense(10)



(e) EfficientNetV2B0 + SVM Dense(128)

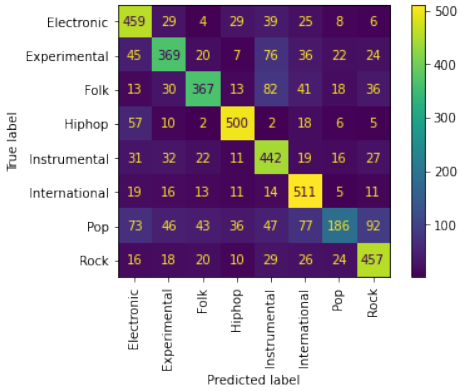


(f) EfficientNetV2B0_FT + SVM Dense(128)

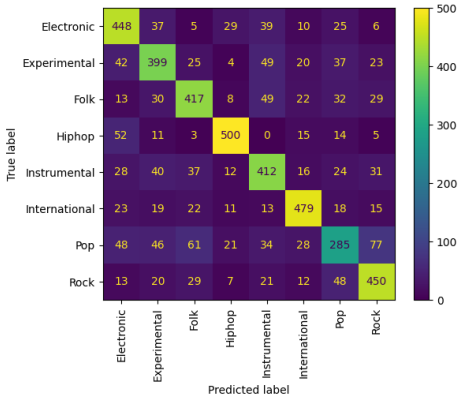
Figure 22: Confusion matrices on the GTZAN dataset, for 224x224 input images, for the EfficientNetV2B0 and EfficientNetV2B0(with finetuning, denoted as EfficientNetV2B0_FT) networks.

F Confusion matrices for FMA_8, input size 128x128

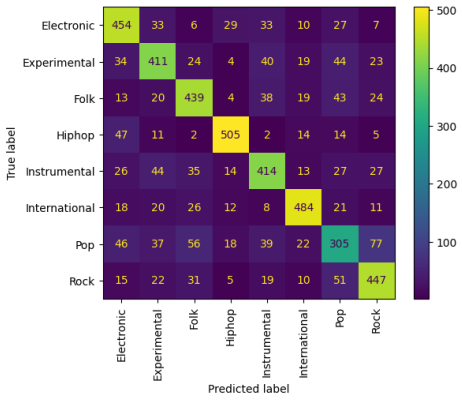
This appendix contains confusion matrices for the models used on the FMA_8 (128x128 input) dataset.



(a) Baseline

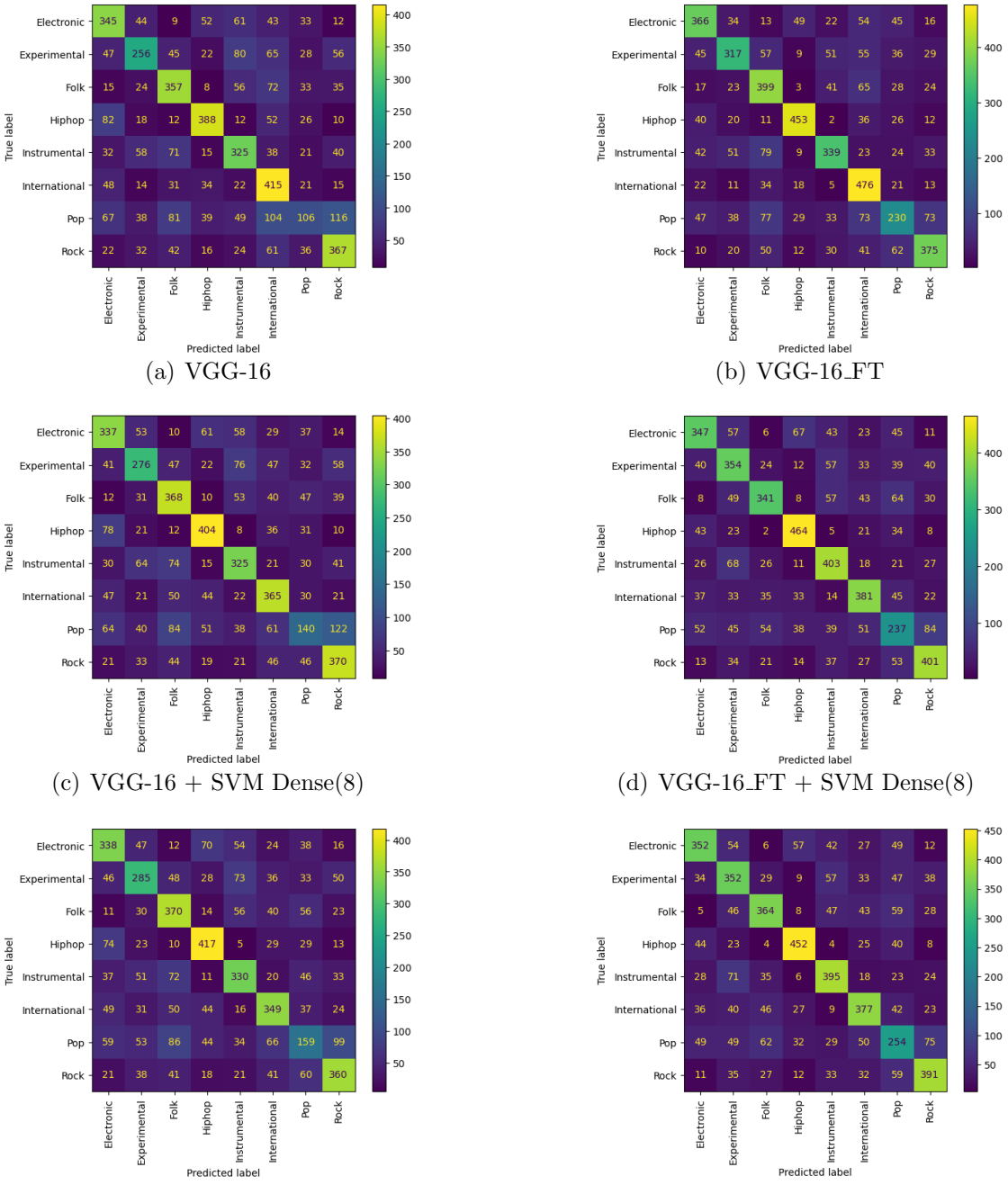


(b) Baseline + SVM Dense(8)



(c) Baseline + SVM Dense(128)

Figure 23: Confusion matrices on the FMA_8 dataset, for 128x128 input images, for the Baseline network

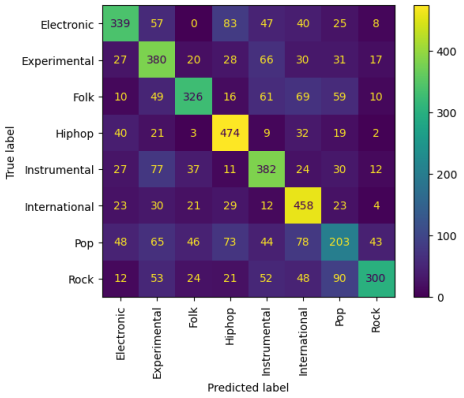


(a) VGG-16 (b) VGG-16_FT

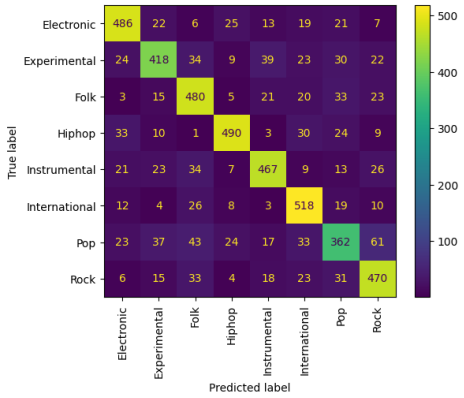
(c) VGG-16 + SVM Dense(8) (d) VGG-16_FT + SVM Dense(8)

(e) VGG-16 + SVM Dense(128) (f) VGG-16_FT + SVM Dense(128)

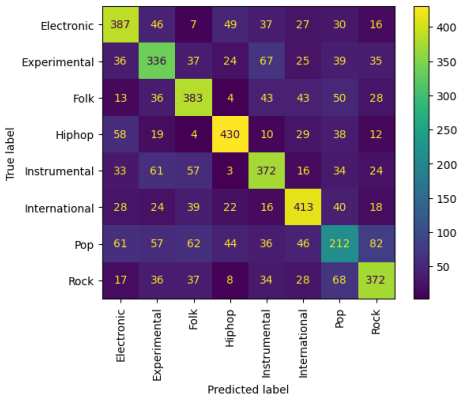
Figure 24: Confusion matrices on the FMA_8 dataset, for 128x128 input images, for the VGG-16 and VGG-16(with finetuning, denoted as VGG-16_FT) networks.



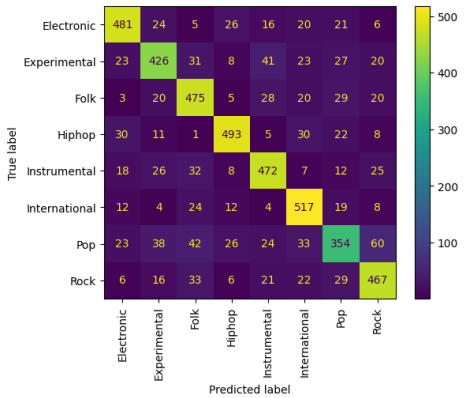
(a) EfficientNetV2B0



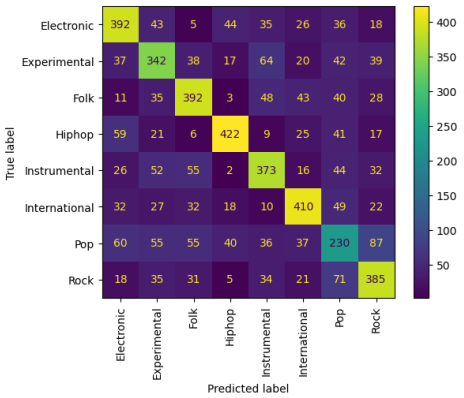
(b) EfficientNetV2B0_FT



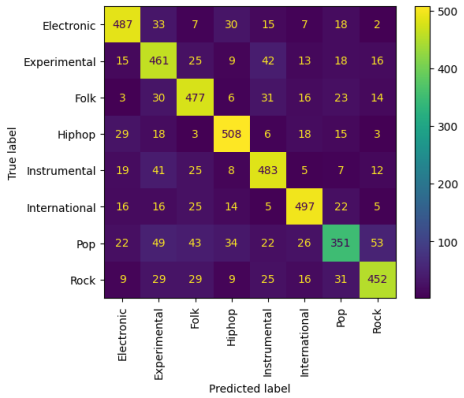
(c) EfficientNetV2B0 + SVM Dense(8)



(d) EfficientNetV2B0_FT + SVM Dense(8)



(e) EfficientNetV2B0 + SVM Dense(128)

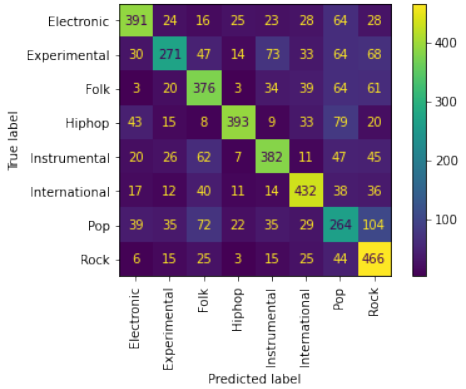


(f) EfficientNetV2B0_FT + SVM Dense(128)

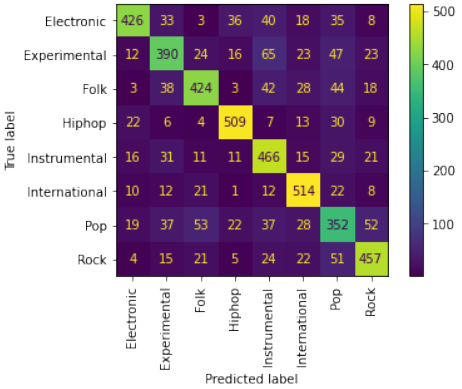
Figure 25: Confusion matrices on the FMA_8 dataset, for 128x128 input images, for the EfficientNetV2B0 and EfficientNetV2B0(with finetuning, denoted as EfficientNetV2B0_FT) networks.

G Confusion matrices for FMA_8, input size 224x224

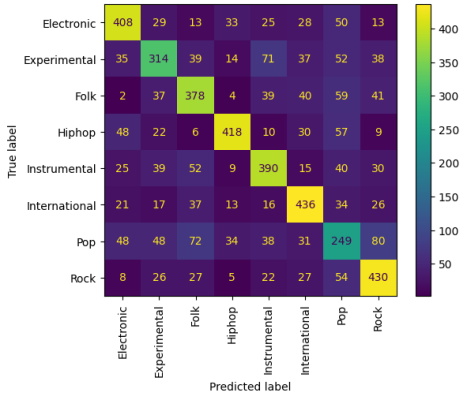
This appendix contains confusion matrices for the models used on the FMA_8 (224x224 input) dataset.



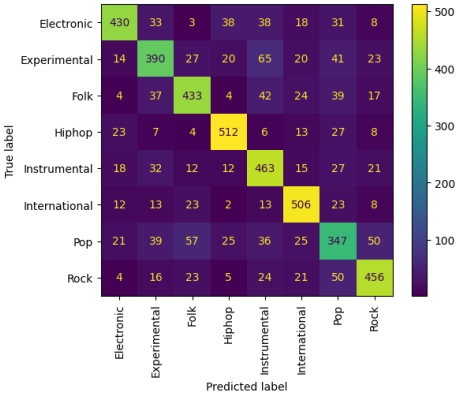
(a) EfficientNetV2B0



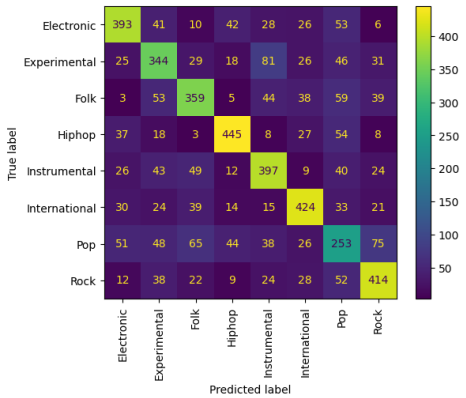
(b) EfficientNetV2B0_FT



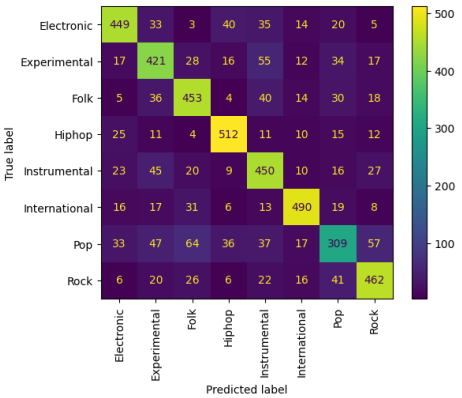
(c) EfficientNetV2B0 + SVM Dense(10)



(d) EfficientNetV2B0_FT + SVM Dense(10)



(e) EfficientNetV2B0 + SVM Dense(128)

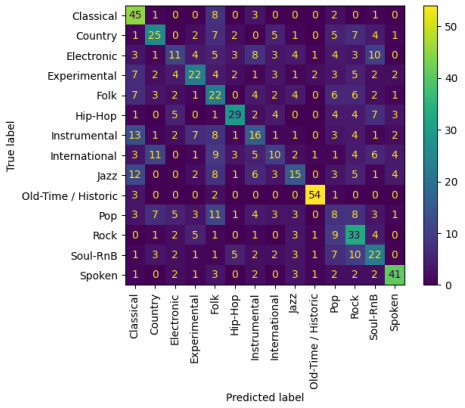


(f) EfficientNetV2B0_FT + SVM Dense(128)

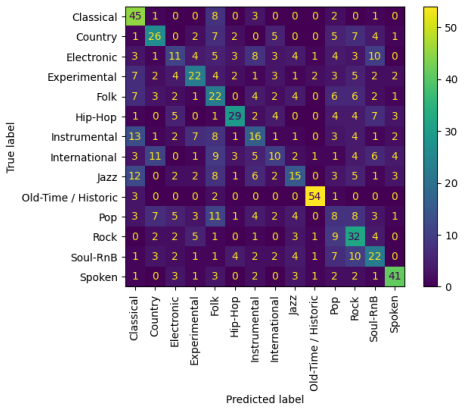
Figure 26: Confusion matrices on the FMA_8 dataset, for 224x224 input images, for the EfficientNetV2B0 and EfficientNetV2B0(with finetuning, denoted as EfficientNetV2B0_FT) networks.

H Confusion matrices for FMA_14, input size 224x224

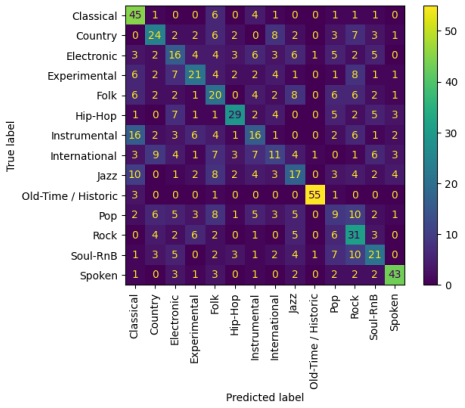
This appendix contains confusion matrices for the models used on the FMA_14 (224x224 input) dataset.



(a) Baseline

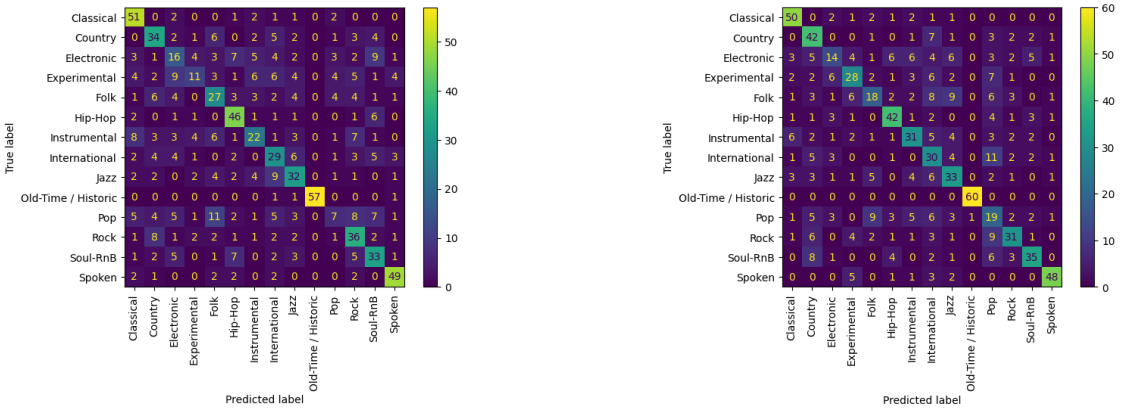


(b) Baseline + SVM Dense(14)



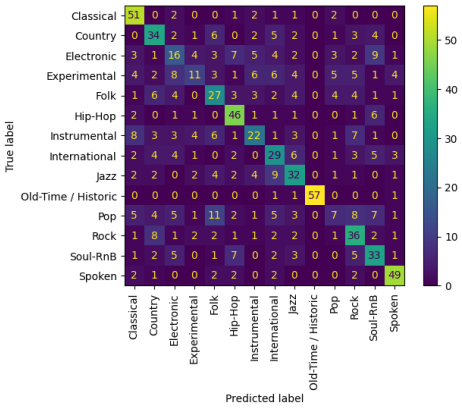
(c) Baseline + SVM Dense(128)

Figure 27: Confusion matrices on the FMA_14 dataset, for 224x224 input images, for the Baseline network

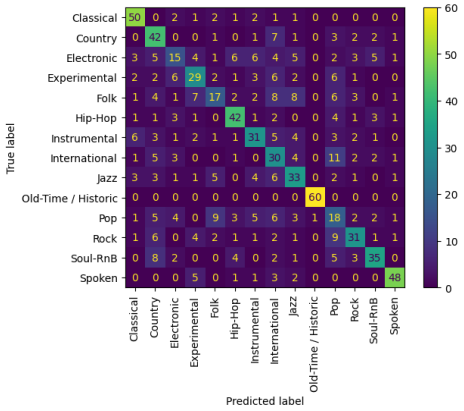


(a) VGG-16

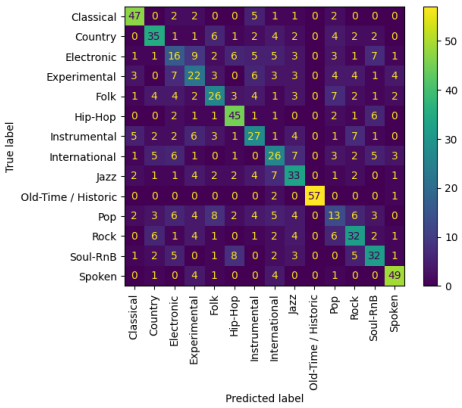
(b) VGG-16_FT



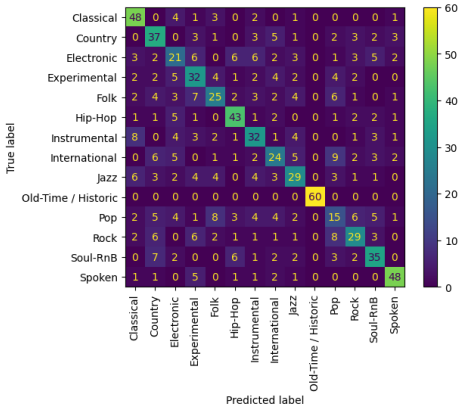
(c) VGG-16 + SVM Dense(14)



(d) VGG-16_FT + SVM Dense(14)

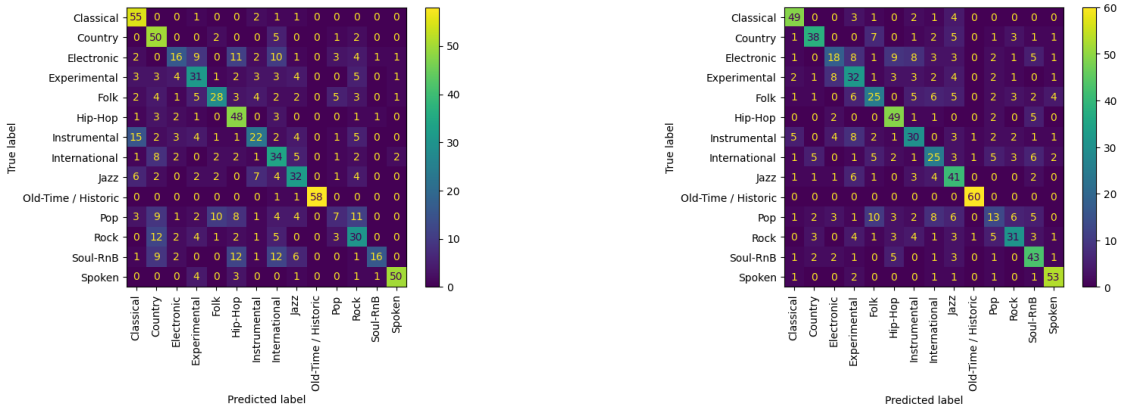


(e) VGG-16 + SVM Dense(128)



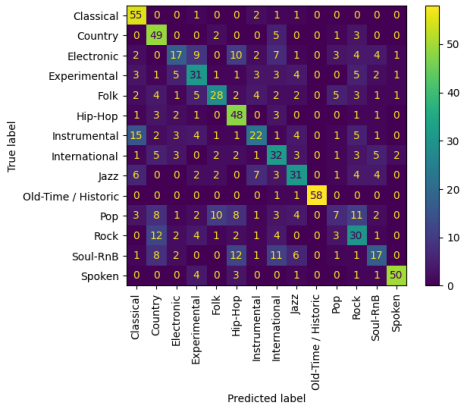
(f) VGG-16_FT + SVM Dense(128)

Figure 28: Confusion matrices on the FMA_14 dataset, for 224x224 input images, for the VGG-16 and VGG-16(with finetuning, denoted as VGG-16_FT) networks.

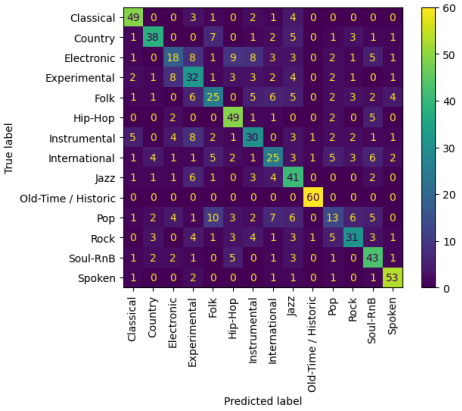


(a) EfficientNetV2B0

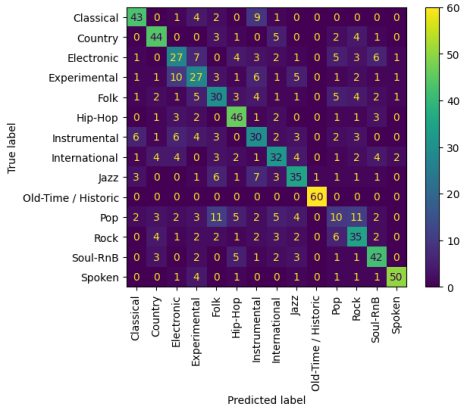
(b) EfficientNetV2B0_FT



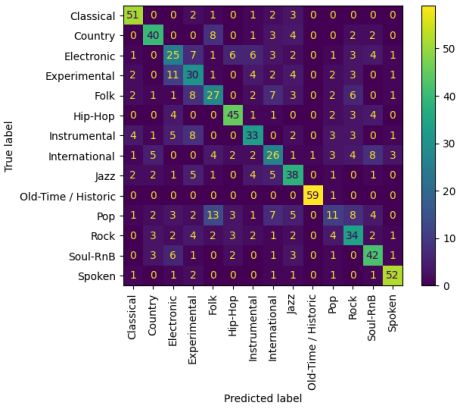
(c) EfficientNetV2B0 + SVM Dense(14)



(d) EfficientNetV2B0_FT + SVM Dense(14)



(e) EfficientNetV2B0 + SVM Dense(128)

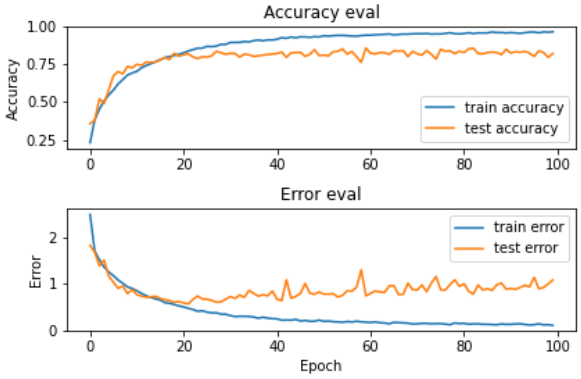


(f) EfficientNetV2B0_FT + SVM Dense(128)

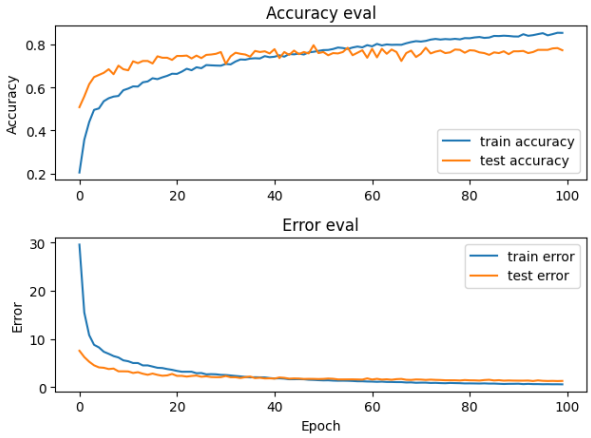
Figure 29: Confusion matrices on the FMA_14 dataset, for 224x224 input images, for the EfficientNetV2B0 and EfficientNetV2B0(with finetuning, denoted as EfficientNetV2B0_FT) networks.

I Train/test accuracy curves

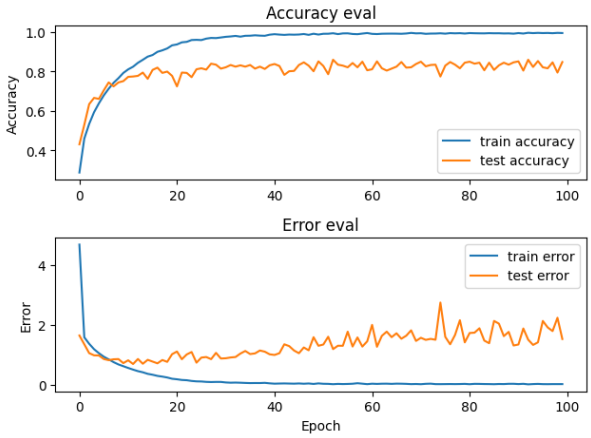
This appendix contains train/test accuracy curves for the models used on the different datasets and their respective error curves.



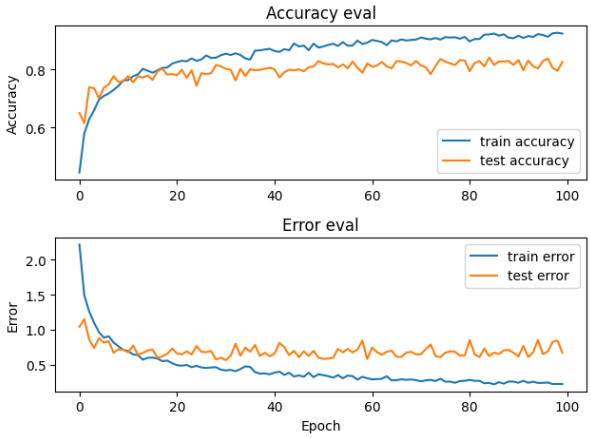
(a) Baseline network



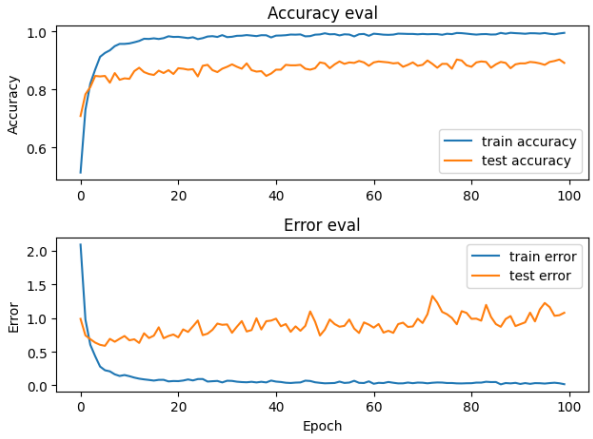
(b) VGG-16



(c) VGG-16_FT

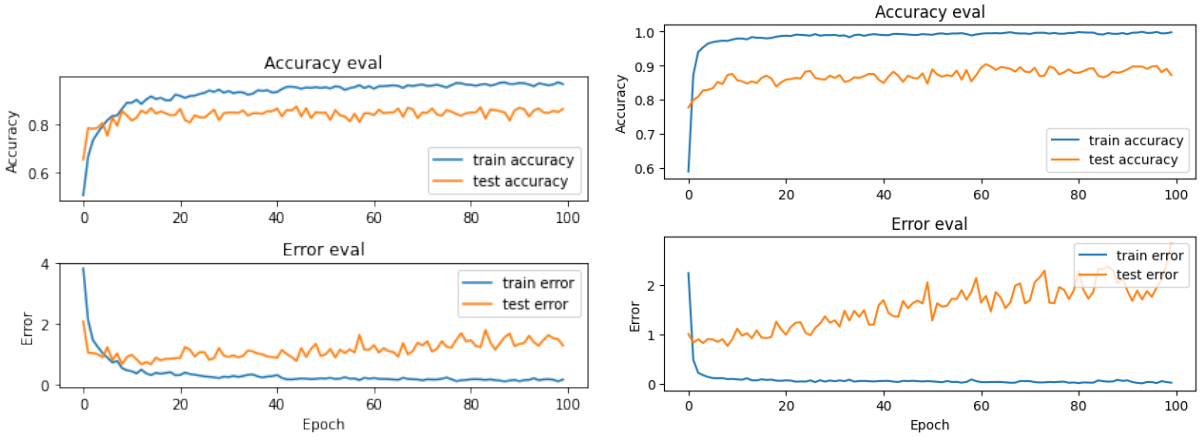


(d) EfficientNetV2B0



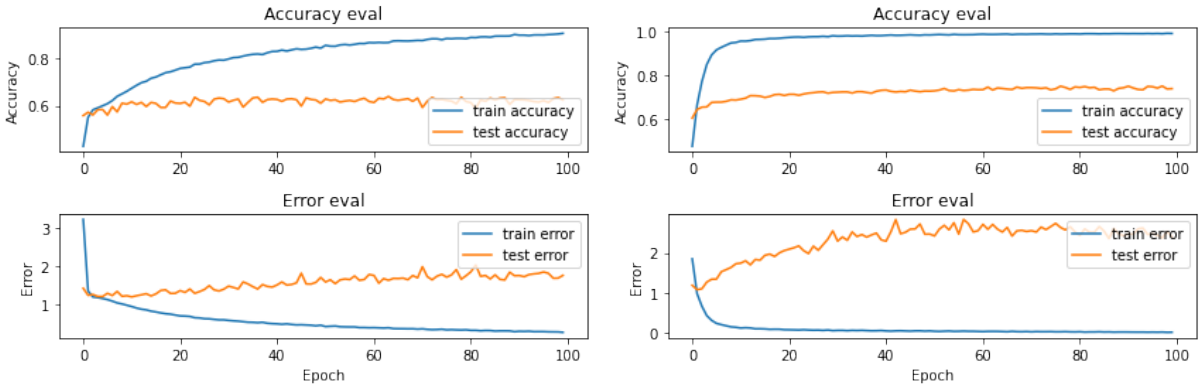
(e) EfficientNetV2B0_FT

Figure 30: Train/test accuracy curves and train/test error curves for the baseline, VGG-16 and EfficientNetV2B0 models on the GTZAN(128) dataset, for 128x128 input. Networks are also used in combination with finetuning, denoted with a "_FT" suffix.



(a) EfficientNetV2B0 (b) EfficientNetV2B0_FT

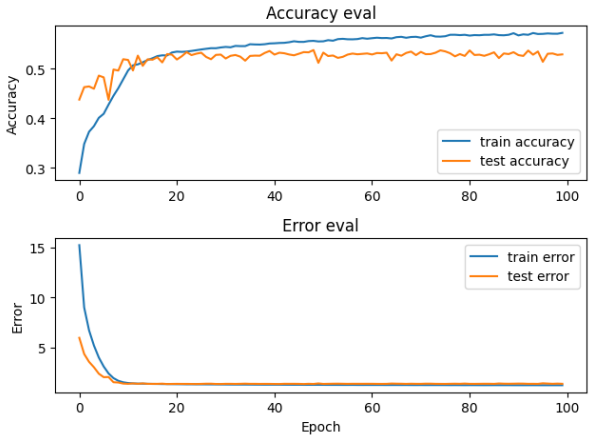
Figure 31: Train and test accuracy curves and train and test error curves for the EfficientNetV2B0 models on the GTZAN(224) dataset, for 224x224 input. The network is also used in combination with finetuning, denoted with a "_FT" suffix.



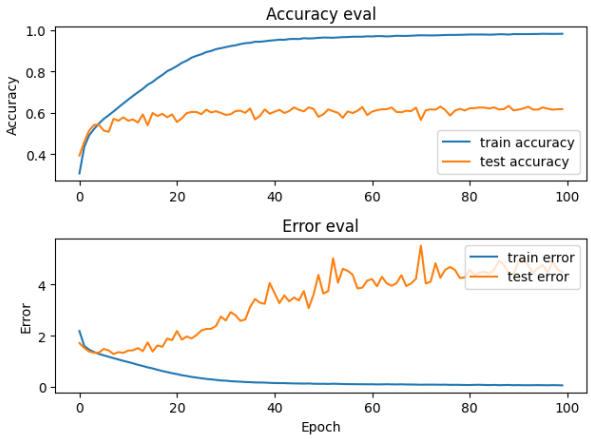
(a) EfficientNetV2B0 (b) EfficientNetV2B0_FT

Figure 32: Train and test accuracy curves and train and test error curves for EfficientNetV2B0 models on the FMA_8(224) dataset, for 224x224 input. The network is also used in combination with finetuning, denoted with a "_FT" suffix.

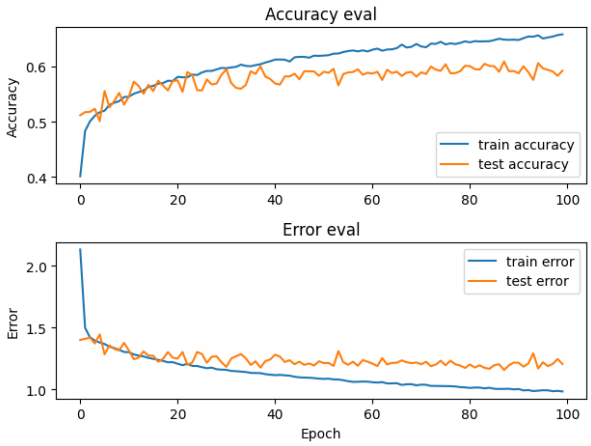
(a) Baseline network



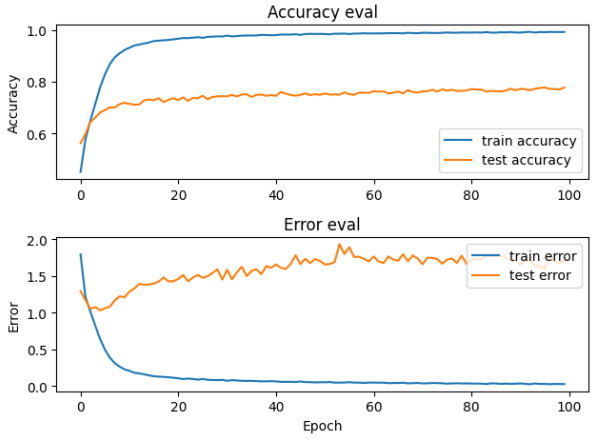
(b) VGG-16



(c) VGG-16_FT

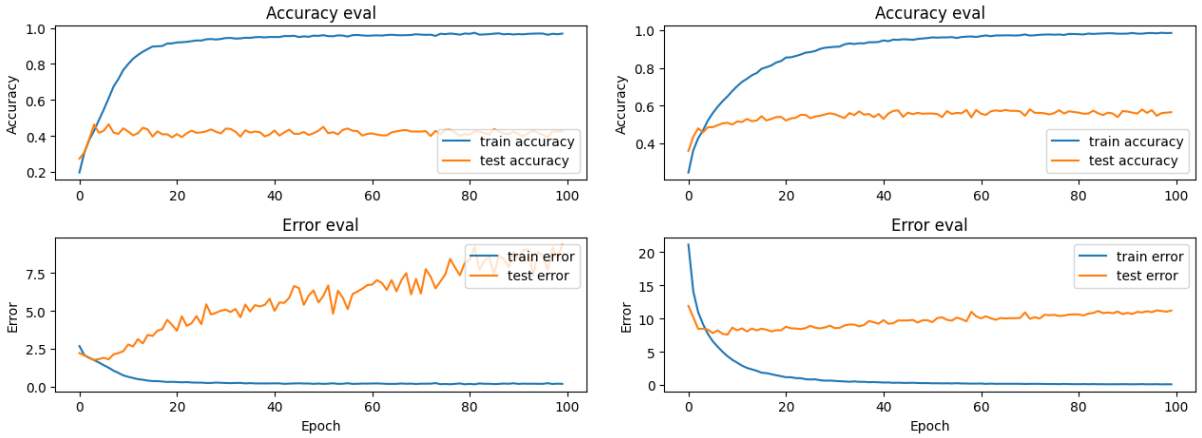


(d) EfficientNetV2B0



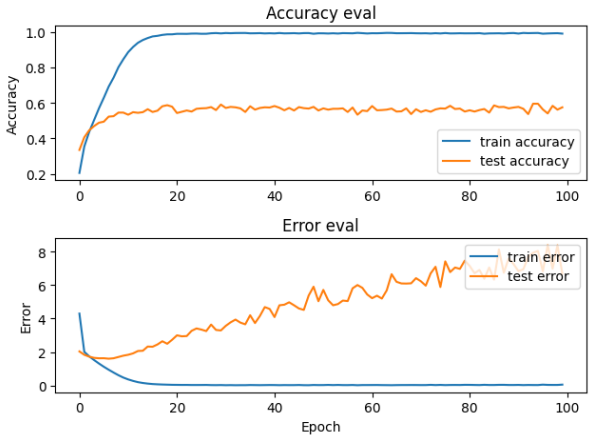
(e) EfficientNetV2B0_FT

Figure 33: Train and test accuracy curves and train and test error curves for the baseline, VGG-16 and EfficientNetV2B0 models on the FMA_8(128) dataset, for 128x128 input. Networks are also used in combination with finetuning, denoted with a "_FT" suffix.

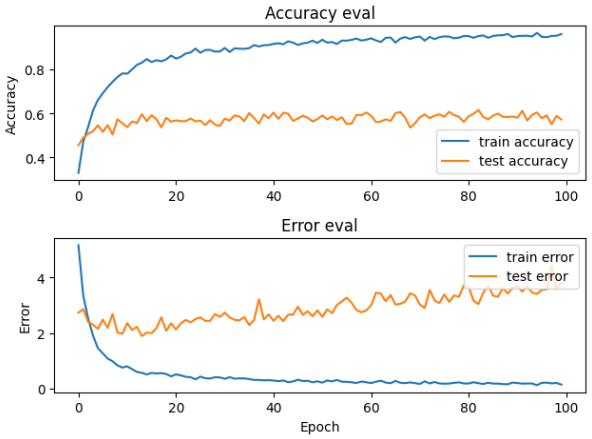


(a) Baseline network

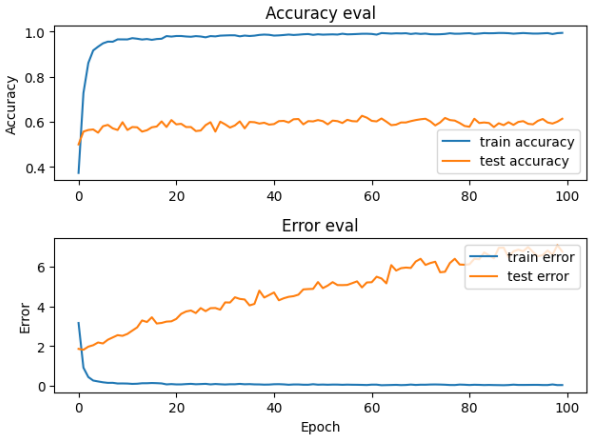
(b) VGG-16



(c) VGG-16_FT



(d) EfficientNetV2B0



(e) EfficientNetV2B0_FT

Figure 34: Train and test accuracy curves and train and test error curves for the baseline, VGG-16 and EfficientNetV2B0 models on the FMA_14(224) dataset, for 224x224 input. Networks are also used in combination with finetuning, denoted with a "_FT" suffix.

J Precision, recall and F1 scores for the best performing models per dataset.

This appendix contains precision, recall and F1 scores for the best performing variants of the three tested models, per genre in each dataset.

Method	Baseline-SVM-D128			VGG-SVM-D128			EfficientNet-SVM-D128		
Genre	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Pop	0.797	0.850	0.823	0.785	0.850	0.816	0.946	0.883	0.914
Metal	0.889	0.933	0.911	0.898	0.883	0.891	0.950	0.950	0.950
Disco	0.803	0.817	0.810	0.746	0.833	0.787	0.859	0.917	0.887
Blues	0.879	0.850	0.864	0.788	0.867	0.825	0.964	0.900	0.931
Reggae	0.788	0.683	0.732	0.823	0.850	0.836	0.900	0.900	0.900
Classical	0.934	0.950	0.942	0.918	0.933	0.926	0.983	0.983	0.983
Rock	0.627	0.617	0.622	0.714	0.667	0.690	0.766	0.817	0.790
Hiphop	0.800	0.867	0.832	0.895	0.850	0.872	0.891	0.950	0.919
Country	0.789	0.750	0.769	0.843	0.717	0.775	0.852	0.767	0.807
Jazz	0.847	0.847	0.847	0.909	0.847	0.877	0.934	0.966	0.950

Figure 35: Precision, recall and F1 scores for the GTZAN dataset.

Method	Baseline-SVM-D128			VGG-SVM-D128			EfficientNet-SVM-D128		
Genre	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Electronic	0.695	0.758	0.725	0.630	0.588	0.608	0.812	0.813	0.812
Experimental	0.687	0.686	0.687	0.525	0.588	0.555	0.681	0.770	0.723
Folk	0.709	0.732	0.720	0.635	0.607	0.621	0.752	0.795	0.773
Hiphop	0.854	0.842	0.848	0.750	0.753	0.751	0.822	0.847	0.834
Instrumental	0.698	0.690	0.694	0.641	0.658	0.650	0.768	0.805	0.786
International	0.819	0.807	0.813	0.623	0.628	0.626	0.831	0.828	0.830
Pop	0.573	0.508	0.539	0.443	0.423	0.433	0.724	0.585	0.647
Rock	0.720	0.745	0.732	0.653	0.652	0.652	0.811	0.753	0.781

Figure 36: Precision, recall and F1 scores for the FMA_8 dataset.

Method	Baseline-SVM-D128			VGG-SVM-D128			EfficientNet-SVM-D128		
Genre	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Classical	0.464	0.750	0.573	0.746	0.783	0.764	0.741	0.717	0.729
Country	0.436	0.400	0.417	0.583	0.583	0.583	0.698	0.733	0.715
Electronic	0.281	0.267	0.274	0.302	0.267	0.283	0.474	0.450	0.462
Experimental	0.438	0.350	0.389	0.367	0.367	0.367	0.443	0.450	0.446
Folk	0.263	0.333	0.294	0.481	0.433	0.456	0.484	0.500	0.492
Hiphop	0.630	0.483	0.547	0.652	0.750	0.698	0.657	0.767	0.708
Instrumental	0.302	0.267	0.283	0.458	0.450	0.454	0.455	0.500	0.476
International	0.262	0.183	0.216	0.406	0.433	0.419	0.542	0.533	0.538
Jazz	0.315	0.283	0.298	0.493	0.550	0.520	0.600	0.600	0.600
Old-Time / Hist	0.948	0.917	0.932	1.000	0.950	0.974	0.984	1.000	0.992
Pop	0.176	0.150	0.162	0.277	0.217	0.243	0.278	0.167	0.208
Rock	0.344	0.517	0.413	0.500	0.533	0.516	0.515	0.583	0.547
Soul-RnB	0.389	0.350	0.368	0.533	0.533	0.533	0.646	0.700	0.672
Spoken	0.729	0.717	0.723	0.778	0.817	0.797	0.926	0.833	0.877

Figure 37: Precision, recall and F1 scores for the FMA_14 dataset.