A Brief Survey on Automatic Music Genre Classification

1921335
Peilin Zhou
yo19143@bristol.ac.uk

1842678
Siyu Zhou
oj18065@bristol.ac.uk

1855948
Tianyue Lyu
sv18072@bristol.ac.uk

I. SIGNATURE

We agree that all members have contributed to this project (both code and report) in an approximately equal manner

II. INTRODUCTION

Music are everywhere nowadays. With the rapid development of Internet, a tremendous number of new digital audios are created, uploaded, and spread everyday. However, this also brings side effects, such as making music recommendation or music search a difficult task. To handle these problems, a wide range of methods aiming at solving the music information retrieval (MIR) problem have been proposed by researchers, among which music genre classification is a basic but import subject. Music genre classification aims at predicting the genre, such as classical, disco, pop, etc., of a specific audio clip. Recently, neural networks, such as convolutional neural networks (CNNs), have been widely used to solve the music genre classification problem, as they can avoid sophisticated hand-crafted features and make predictions in an end-to-end manner, such as in [1]. Besides, [2] proposed that low-level features tend to be more contributed for improving the genre classification than high-level semantic features. Based on this conclusion, they further design a network structure to capture both the abstract high-level information and lower-level features at the same time. Other model structures are also widely investigated by researchers. [3] proposed that music structure or recurrent harmonies, which are significant for music classification, can be captured by recurrent neural networks (RNNs). Thus, they combine CNNs and RNNs together to capture both spatial features and temporal frame orders of music data. They claim that the prediction performance shows a great improvement than simple CNNs. [4] proposed a "divide-and-conquer" to solve this problem. Specifically, they split the spectrogram of the music signal into consecutive 3-second segments, then make predictions for each segment by convolutional deep belief networks, and finally combine the predictions of each segment to give the final prediction of the music.

In this report, we try to replicate the results given by [1].

III. METHOD

A. Dataset

The GTZAN dataset is used in this report. The dataset is split into a training set with 11250 samples and a validation set with 3750 samples. Each sample is of size $1 \times 80 \times 80$, with a label in one of blues, classical, country, disco, hiphop, pop, jazz, metal, reggae, and rock. Here, the labels are encoded accordingly in sequence from 0 to 9.

B. Model Structure

We follow the shallow CNN model structure designed in [1]. Given an input $x \in \mathbb{R}^{1\times80\times80}$, we first pass it to two substructures. The first substructure starts with a convolutional layer with 16 kernels, where the kernel size is 10×23, stride is 1 and padding is used to guarantee that the output has the same shape with the input. Following the convolutional layer are a leaky ReLU activation layer with slope 0.3, and a max pooling layer with kernel size 1×20. This pipeline maps the input into shape $x_1 \in \mathbb{R}^{16\times80\times4}$. Another pipeline is almost the same, except that the kernel size of the convolutional layer is 21×20 and the kernel size of the max pooling layer is 20×1. The output of the second pipeline is $x_2 \in \mathbb{R}^{16\times4\times80}$. Then, these two outputs are flattened and concatenated to form the input to the following linear layers, i.e. $x_3 = concat(flatten(x_1), flatten(x_2)) \in \mathbb{R}^{1\times10240}$. x_3 is then passed to a linear layer with 200 units, a leaky ReLU layer with slope 0.3, a dropout layer with probability 0.1, a linear layer with 10 units and then a softmax layer in sequence, to give the final output $o \in \mathbb{R}^{1\times10}$.

C. Implementation Details

To investigate the model stated above, we use the cross entropy loss to compute the error between predictions and ground-truths. Besides, the ℓ_1 norm of all trainable weights in the model with weight 0.0001 is added to the cross entropy loss to formulate the final loss for back propagation. The parameters of the model are updated using the Adam optimizer, with learning rate $lr = 0.00005$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. We conduct 3 experiments to find out the best batch size in $\{64, 128, 256\}$. In each experiment, we initialize the model parameters with default initializers, and then train the model for 200 epochs on the training dataset. After each epoch ends, we record the value of the loss function, and then evaluate the accuracy of the model on both the training and

validation dataset. The validation accuracies of the models are evaluated after training for 100-th epoch and 200-th epoch, and then used as the metric of the models' performance.

D. Results

The results of the model trained with different batch sizes, i.e. 64, 128 and 256, are plotted in from Fig. 1 to Fig. 6, and summarized in TABLE I. As can be seen, the model with batch size 128 trained for 200 epochs has the best test accuracy among all models. From the training and validation accuracies of each model, i.e. Fig. 1, Fig. 3 and Fig. 5, we can conclude that none of the models shows sign for overfitting, as the validation accuracy is growing during the whole training processes. However, we observe that the validation accuracy with batch size 256 is growing sharply when the training ends, which might suggest that the model is still not converged, and the validation accuracy might keep growing if training for more epochs, see Fig. 5.

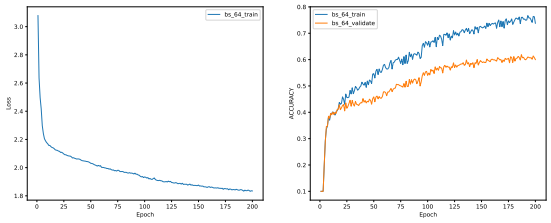


Fig. 1: The training curves of the model with batch size 64. (Left) The training losses, (right) the training and validation accuracies.

Fig. 2: The confusion matrix of the model trained with batch size 64 for 200 epochs.

From the confusion matrix, see Fig 2, Fig. 4 and Fig. 6, we can observe that classical and jazz are always the easiest genres for the model to recognize. Besides, the model with

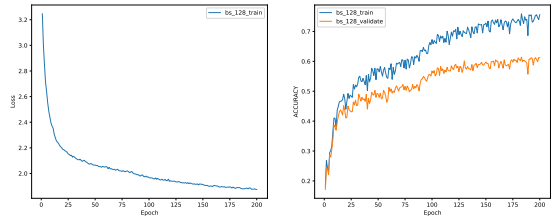


Fig. 3: The training curves of the model with batch size 128. (Left) The training losses, (right) the training and validation accuracies.

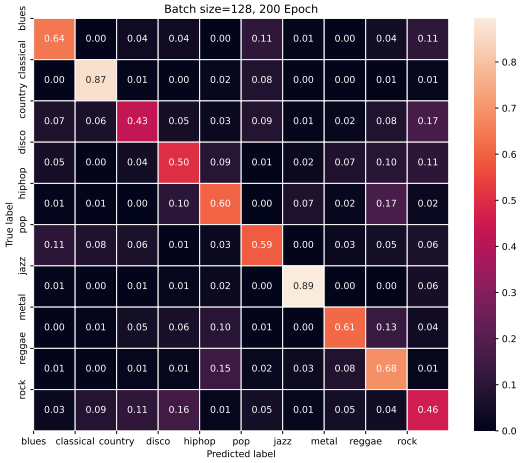


Fig. 4: The confusion matrix of the model trained with batch size 128 for 200 epochs.

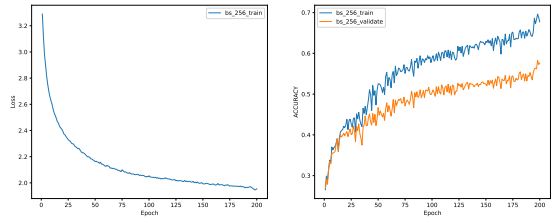


Fig. 5: The training curves of the model with batch size 256. (Left) The training losses, (right) the training and validation accuracies.

TABLE I: The test results of all experiments.

Model	***Batch size***	***Accuracy(%)***	***Epoch***
shallow	64	54.03	100
shallow	64	60.00	200
shallow	128	56.80	100
shallow	128	**61.23**	200
shallow	256	50.37	100
shallow	256	57.63	200

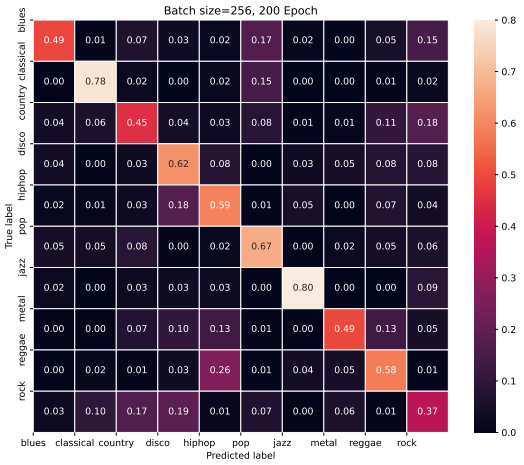
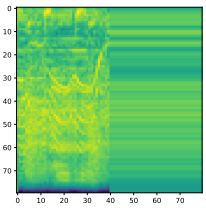
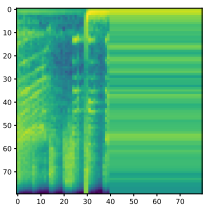


Fig. 6: The confusion matrix of the model trained with batch size 256 for 200 epochs.

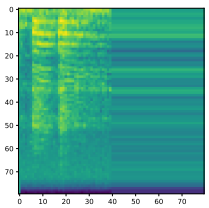
batch size 64 fails to classify more than half of the disco and rock music, the model with batch size 128 fails to classify more than half of the country and rock music, while the model with batch size 256 fails to classify more than half of the blues, country, metal, and rock music. Some correctly and incorrectly classified samples of the model with batch size 64 are plotted in Fig. 7.



(a) The spectrogram of jazz predicted as jazz



(b) The spectrogram of hiphop predicted as reggae



(c) The spectrogram of blues predicted as pop

Fig. 7: The spectrogram of some correctly and incorrectly classified samples.

IV. Conclusions and Future Work

In this report, we have conducted several experiment on the shallow CNN structure proposed by [1]. In addition to the replication of the model structure, we run the model several times to find out the best batch size during training. The results show that the model can generalize with the best validation accuracy when training with batch size 128 for 200 epochs. However, there are more works can be done in future works. For example, by training the model for more epochs, we can find where the model starts to overfit the data, and apply early stopping to get even better validation results. Besides, the hyper parameters, such as learning rate, optimizer, ways of weight regularization, etc., can be tunes with more efforts.

References

[1] A. Schindler, T. Lidy, and A. Rauber, "Comparing shallow versus deep neural network architectures for automatic music genre classification." in *FMT*, 2016, pp. 17–21.

[2] 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.

[3] L. Feng, S. Liu, and J. Yao, "Music genre classification with paralleling recurrent convolutional neural network," *arXiv preprint arXiv:1712.08370*, 2017.

[4] M. Dong, "Convolutional neural network achieves human-level accuracy in music genre classification," *arXiv preprint arXiv:1802.09697*, 2018.