MUSIC GENRE CLASSIFIER WITH DEEP NEURAL NETWORKS

Chenyu Xi[1]

***Abstract*— Deep neural network has been widely used in recognition and classification tasks. Traditional convolutional networks such as VGG16 and InceptionV3 achieved high score in image processing applications. In this paper, we discuss the possibility to implement similar learning techniques in musical genre classification task. We compare three different models in this paper, one convolutional neural network(CNN) model and two convolutional recurrent neural network(CRNN) models. We train each model on GTZAN dataset enhanced by adding more audio samples. Model performance is tested with full songs.**

I. INTRODUCTION

Music genres represent categories identifying music pieces with similar tradition and form. Music genre classification aims to tag audio signal with genre labels, which is largely used in building music search or recommendation engine.

Extracting audio signal features is crucial for music genre classification. On common method for this task is using deep neural network(DNN). It's unnecessary for us to figure out what features of input signal are meaningful for classification task, because DNN selects them automatically during training stage. The usage of DNN simplifies the feature extraction stage.

There are many existing works on automatic music genre recognition. A four-layer CNN network is used as classifier in [1] and pretrained VGG16 and AlexNet are adopted as extractors in [2]. In this paper, we discuss a CNN model inspired by [1] in the first stage, and we add GRU layer to build CRNN models in the second stage.

II. RELATED WORK

A. *Convolutional Neural Network*

Convolutional neural network(CNN) was firstly applied to extract image features[3], and has been explored for other feature recognition tasks, such as semantic parsing and audio feature extraction. CNN extract hierarchical features of input signal with convolutional kernels.

B. *Convolutional Recurrent Neural Network*

Recurrent neural network(RNN) is designed to solve sequential input problems. RNN models are frequently used in the field of natural language processing and audio signal modeling.

Gated recurrent unit(GRU) is a gating mechanism in RNN, which is similar with LSTM while has fewer parameters by eliminating the output gate. In our works, we build hybrid model combining convolutional layers and GRU layers together[4]. This architecture is called convolutional recurrent neural network(CRNN). The additional GRU layers play the role of temporal feature extractor.

[1]cx2219@columbia.edu, Department of electrical engineering, Columbia University

III. DATA PREPOSSESSING

A. *GTZAN Dataset*

GTZAN is a famous dataset used for music genre classification tasks. The dataset consists of 10 genres with one hundred 30s audio tracks each. The ten genres are pop, metal, disco, blues, reggae, classical, rock, hip-hop, country and jazz. All tracks are 22050Hz Mono 16-bit audio files in .wav format. In this paper, we use 80% tracks for training and use 10% tracks for test and validation each.

B. *Handmade Dataset*

Due to the small size and bias of GTZAN dataset, merely training on GTZAN may lead to an biased model which might not be enough for training precise prediction models. Thus, handmade dataset is used in both training stage and test stage. We enhance original GTZAN training set by adding more pieces for each genre, 72 full songs in total.

We use both full-length songs grasped from internet (3 songs for each genre, they are top songs recommended by Google Music) and GTZAN test set for model performance evaluation.

C. *Multiframe*

Multiframe strategy[4] is adopted to augment the training data and speed up the training process. We firstly convert original audio signal into log scale mel-spectrogram with 128 mel bands and 16000 sample rate, and divide the original spectrogram into small chunks to create 128×128 feature maps. Each chunk stands for approximately 3s audio sequence. We also use this strategy during recognition process. Test songs are divided into smaller chunks and the model predicts the genre of each chunk. We summarize results of all chunks to improve the confidence of classification.

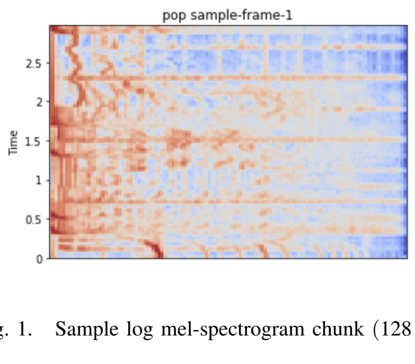

Fig. 1. Sample log mel-spectrogram chunk (128×128)

On the second stage, we tried 256 mel bands to create 256×256 feature maps to extend the input sequence length. The results of these two multiframe strategies are compared over CRNN models.

When dealing with audio files in GTZAN dataset, we use the full track for multiframe generation. However, when we process full songs in handmade dataset, the first and last 10 seconds of each song are cut off as they usually contribute little to the genre features.

IV. MODEL DESCRIPTION

A. CNN Model

The first model contains four convolutional layers following with three fully connected layers. All convolutional layers have fixed 3×3 kernel size. Model structure is illustrated in Fig 2. Convolutional layers reduced the input feature map from 128×128 to 2×2, while increase the feature map channel number from 1 to 512. The output of CNN block is then resized to a 2048-d vector and feed into 3 fully connected layers. The model outputs 10-d vector with sigmoid activation. Each element of the output vector stands for a prediction score for one genre class.

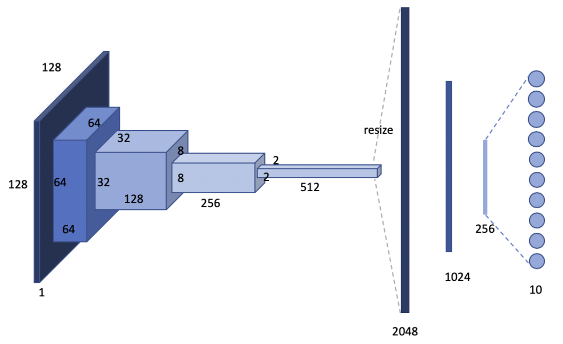

Fig. 2. CNN model structure

We use xavier uniform initializer for each convolutional layer to speed up the training process, and add 0.5 drop out to each fully connected layer to reduce model overfitting.

B. CRNN Models

We remove the last convolutional layer in the CNN model and add one GRU layer with 256 hidden units to create the first CRNN model. The model structure is shown in Fig 3.

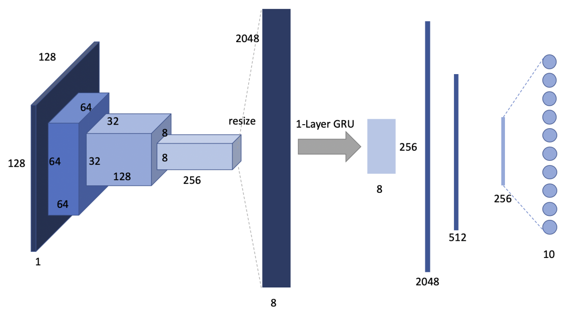

Fig. 3. CRNN model-I structure

Meanwhile, we design another similar structure which takes 256×256 feature map as input instead of 128×128. We found that keeping four convolutional layers and adding one GRU layer before fully connected layers offers the best result. Fig 4 illustrates the model structure.

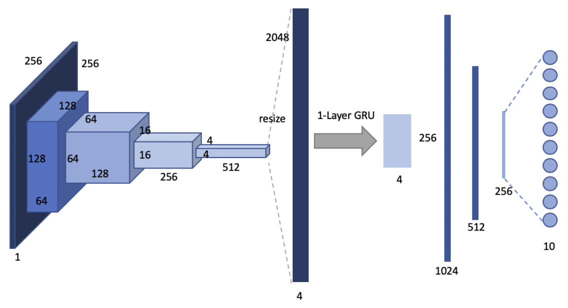

Fig. 4. CRNN model-II structure

We also built another model which has GRU layer in front of CNN blocks. This RCNN model has similar performance with previous models while takes longer time for training. Thus we are not going to discuss this one in the paper.

V. EXPERIMENT

A. Training

During the training stage, we found that setting the learning rate to e^{-5} and batch size to 20 offers acceptable results. We use binary cross-entropy(BCE) loss function (1) for back propagation, which is a common method used for classification task. RMSprop optimizer is used to adjust model parameters. Meanwhile, the temporary prediction accuracy for each epoch is recorded for learning rate adjustment, and we save the model with highest accuracy during the training stage.

$$\begin{gathered}\ell(x,y) = mean(L), L = \{l_1, \ldots, l_N\}^\top, \\ l_n = -w_n \left[y_n \cdot \log x_n + (1-y_n) \cdot \log(1-x_n)\right]\end{gathered} \tag{1}$$

To calculate the accuracy score, we count successful prediction cases and divide the count by total experiment time. We trained 50 and 60 epochs for CNN and CRNN models respectively.

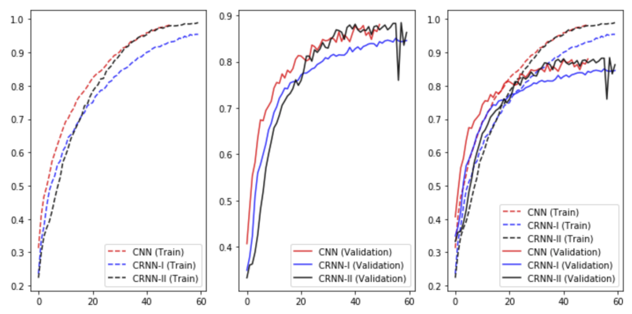

Fig. 5. Accuracy vs Epoch

In Fig 5 and Fig 6, we compare the training process among these three models. In general, all model architectures eventually reach accuracy of over 95% on training data. However, all models get overfitted on validation set after accuracy reaching 80%.

Although models have close performance on current test set, we can tell from the figures that CNN model and the second CRNN model have slightly better prediction result. Meanwhile, CNN model converges faster compared to model architectures containing RNN layers.

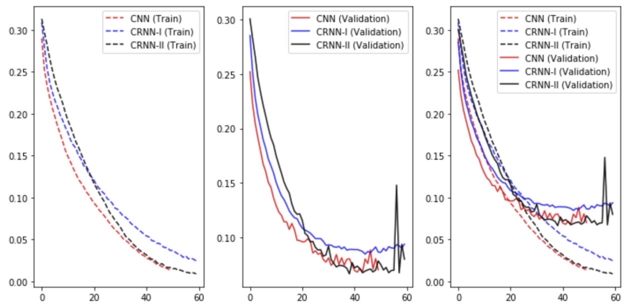

Fig. 6. BCE loss vs Epoch

B. Results

We firstly test model performance on audio frames, which have the same format as our training data. We compared model performance on both test and validation set in the following table. As shown in the table, the CNN model has overall best result. All models have over 80% accuracy in both set. The overall performance is better than reference work in [4]. We also generate normalized confusion matrices, and they are shown in appendix part.

TABLE I
MODEL ACCURACY

	CNN	CRNN-I	CRNN-II
validation set	88.05%	85.08%	88.45%
test set	86.89%	83.05%	82.67%

TABLE II
MODEL LOSS

	CNN	CRNN-I	CRNN-II
validation set	0.0701	0.0899	0.0674
test set	0.0696	0.0961	0.0955

In order to make our works more practical, We also test our models on 30 songs grasped from the Internet. We tag these songs according to Google Music genre classification. Since music pieces usually contain components of different genres in real life, we propose a simple algorithm to analyze the genre composition of one audio file. We firstly remove the first and last 20 seconds of the test song and convert remaining part into small log mel-spectrogram chunks. Each chunk's genre will be predicted separately by model. We define genre proportion for test song as the number of chunks classified as one genre divided by the count of chunks generated in the first step. Sample prediction results are shown in Fig 7.

```
Test on Offset - Clout ft. Cardi B.mp3
Genre pop: 58.2%
Genre hiphop: 27.11%
Genre reggae: 9.12%
Genre blues: 5.57%
*****************************************************
*******************************************
Test on Wu-Tang Clan - C.R.E.A.M. (Official Music Vide
o).mp3
Genre hiphop: 71.01%
Genre blues: 18.96%
Genre reggae: 4.7%
Genre jazz: 3.89%
Genre pop: 1.44%
```

Fig. 7. sample prediction results

To evaluate the overall performance of models with regard to different genres, we design a score function: Firstly, the function takes the three genres with highest proportion for one test audio track. Then we compared the ground truth genre tag of this track with the three top genre labels. If any of them matches the tag, we add the proportion of the label to the model score of this genre class. In ideal case, we should get 3 for each genre as we have three songs to test for one genre. We normalize scores by dividing them with 3. The prediction result are shown in Fig 8.

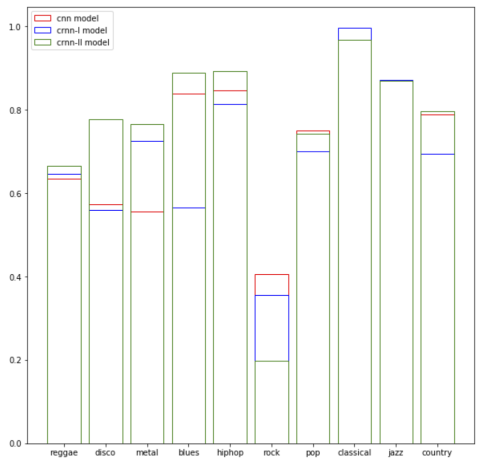

Fig. 8. sample prediction on full songs

The full song estimation implies that the second CRNN model has best performance on full songs. One interesting fact is that genre prediction models tend to have better performance on music genres with explicit features. For example, all models have almost 100% accuracy on classical and hip-hop music. In general, all genres are well-predicted by models except rock genre.

The main reason that models fail to recognize rock musics well might be an ambiguous classification standard over this genre. When we take a look at the predicted genre components of the rock songs, the rock songs which sounds not very 'typical' might be classified as pop or metal music.

VI. CONCLUSION

In real word, one song usually belong to more than one music genres, so it's not proper to train models on full songs when we treat this task as a multiclass classification problem. We use multiframe strategy to recognize genre proportion of songs by predict genre tag of smaller audio chunks, as we believe 3s or 5s length audio is enough to contain music genre features. We discuss three different models in this paper, and all of them have acceptable performance for genre classification task.

The second CRNN model has best full song prediction result among all models. However, it's not proper to view our current work as a solid prove that the addition of GRU layers could bring performance improvement, because we cannot find a obvious gap between performances of CNN model and CRNN models on our current test set. The variances in model architecture may differ the model performance, but how well model parameters are tuned should also be an crucial factor.

Our propose of adding extra GRU layers is to extract the temporal features of input signal. Nevertheless, we should notice that convolutional layers captures features of mel-spectrogram, which also contains temporal feature. Meanwhile, the second CRNN model takes larger frame as input, which also offers richer temporal information. Further works are needed to figure out whether the additional GRU layer is necessary for genre prediction. From the perspective of our work, convolutional neural network is efficient enough for genre recognition problem.

A better dataset would be inevitable to improve our work. All models overfit on current training set and the performance differences are very small among different model architectures. Though we already enhance the GTZAN dataset with more songs, the current dataset is still a relevant small set for training model. On the other hand, for the test stage, 3 songs for each genre might not be enough to generate a convincible result. More songs should be tested in further works.

APPENDIX

GitHub Link: https://github.com/XiplusChenyu/Musical-Genre-Classification

ACKNOWLEDGMENT

We would like to express our gratitude here to Prof. Mesgarani and TA Luo Yi in the Electrical Engineering Department of Columbia University, who provided essential suggestions and guidance for us.

REFERENCES

[1] Choi, K., Fazekas, G., Sandler, M., Cho, K. (2017). Transfer learning for music classification and regression tasks. arXiv preprint arXiv:1703.09179
[2] https://github.com/Hguimaraes/gtzan.keras
[3] L. Liu et al. (2018). *Deep learning for generic object detection: A survey.* [Online]. Available: https://arxiv.org/abs/1809.02165.
[4] https://github.com/jsalbert/Music-Genre-Classification-with-Deep-Learning.

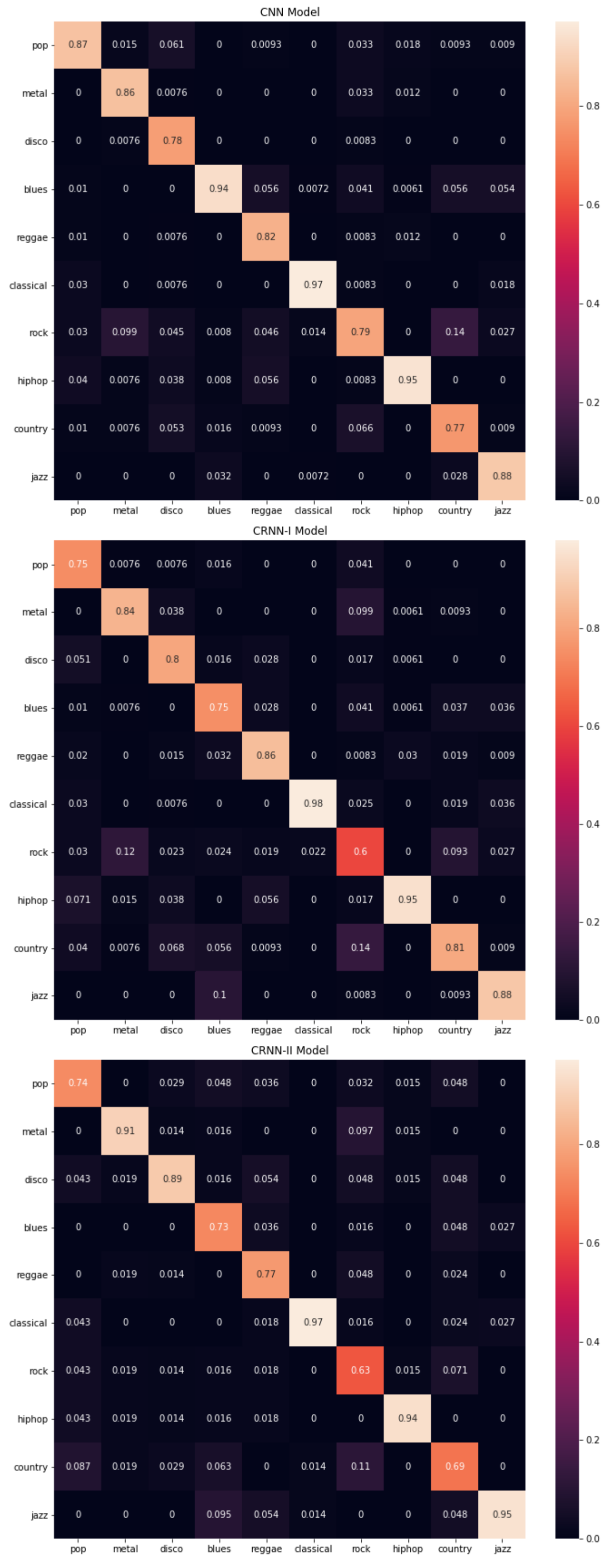

Fig. 9. Confusion matrices: up: CNN model; middle: CRNN-I model; down: CRNN-II model