Semi-supervised Learning of Bottleneck Feature for Music Genre Classification

Jia Dai[1], Wenju Liu[1]*, Hao Zheng[1], Wei Xue[1], and Chongjia Ni[2]

[1]NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[1]University of Chinese Academy of Sciences, Beijing, China
[1]{jia.dai,lwj,hzheng,wxue}@nlpr.ia.ac.cn
[2]School of Mathematic and Quantitative Economics, Shandong University of Finance and Economics, shandong, China
[2]cjni_sd@sdufe.edu.cn

Abstract. A good representation of the audio is important for music genre classification. Deep neural networks (DNN) enable a better approach to learn the representation of audio. The representation learned from DNN, which is known as bottleneck feature, is widely used for speech and audio related application. However, in general, it needs a large amount of transcribed data to learn an effective bottleneck feature extractor. While, in reality, the amount of transcribed data is often limited. In this paper, we investigate a semi-supervised learning to train the bottleneck feature for music data. Then, the bottleneck feature is used for music genre classification. Since the target dataset contains few data, which cannot be used train a reliable bottleneck DNN, we train the DNN bottleneck extractor on a large out-of-domain un-transcribed dataset in semi-supervised way. Experimental results show that with the learned bottleneck feature, the proposed system can perform better than the state-of-the-art best methods on GTZAN dataset.

Keywords: Bottleneck, DNN, Semi-supervised, Multilingual, Cross-lingual

1 Introduction

For music genres classification tasks, most conventional methods use a common procedure which consists of two stages: feature extraction and classification. A discriminative feature and a good classifier can often lead to a better classification result [1].

Recently, Deep Neural networks (DNN) have become increasingly popular in many fields [2] [3]. Classification systems based on DNN usually have a better performance than others, which also includes audio classification. DNN is widely used especially for the big data cases, in which the amount of available data is large. Music genre classification is a special kind of audio classification. Generally

* Corresponding author.

there are two ways to incorporate the DNN into the classification system: using DNN as the classifier [4] [5], or training a DNN to learn the feature of audio [6]. The deep architecture enables DNN to learn more invariant and discriminative features. There are many techniques to use a deep architecture to learn the representation of audio data. For example, Sigtia [7] improves feature learning for audio data using DNN. This method provides a significant improvement in training time, and the features learned are better than state-of-art handcrafted features. Andén [8] proposes the deep scattering transform to extract feature from a deep convolution network. This deep scattering feature can perform better when using long frame window and improves the music genre classification performance.

For automatic speech recognition (ASR), the most popular feature learned by using DNN is bottleneck feature. It uses a small hidden layer (bottleneck layer) which is smaller than other hidden layers in DNN to train a feature extractor. Bottleneck feature is used widely and can achieve good results [9] [10]. However, there is a limit that we don't always have enough data to train a bottleneck DNN, and little data makes the extractor become over-fitting easily. In order to solve this problem, multilingual bottleneck and cross-lingual bottleneck are proposed [11] [12]. The multilingual bottleneck and cross-lingual bottleneck extractors are trained on other languages (rich-resource datasets), and then the trained bottleneck DNN is used to extract bottleneck feature for target dataset with limited amount of data. Experiments show that the bottleneck system trained on other languages outperform the system trained on limited data in ASR [13] [14].

Inspired by this, the semi-supervised training for bottleneck feature (ST-BN system) is proposed for music genre classification. Since there is no "other language" for bottleneck DNN training like ASR, we use semi-supervised method to train a bottleneck DNN on a large similar out-of-domain dataset instead. Semi-supervised is an important topic, especially when target dataset is small while large un-transcribed audio data is available [15]. After the low-level feature is extracted, we first use some of the limit target dataset to train an artificial neural network (ANN) to label the large out-of-domain dataset, which is used for semi-supervised training. The out-of-domain dataset is a dataset which is similar to the target dataset. Then the labeled out-of-domain dataset is used to train a bottleneck DNN. At last, the trained bottleneck DNN is used to extract bottleneck feature for target dataset, and the bottleneck feature is used for training and testing the classification system. The experiment results show that our bottleneck feature perform better than the low-level feature.

To the best of our knowledge, in this paper, it is the first work to apply the bottleneck feature to music genre classification. This ameliorates the over-fitting problem of DNN caused by the lacking of training target data. The proposed ST-BN system achieves 94% average classification result, which is the best classification result on GTZAN dataset [16] compared with other methods.

The rest of this paper is organized as follows. Section 2 describes the prior work related to this paper. Section 3 and section 4 is the feature extraction and proposed ST-BN system respectively. Experiment and results analysis are given in Section 5, followed by a conclusion section.

2 Prior Work

The multilingual DNN for music genre classification has been explored in our previous work [17]. It uses an out-of-domain dataset to train a DNN classification model, then transfer it to target dataset. The DNN model trained on out-of-domain dataset can be seen as a pre-training of the target DNN classification model, and the pre-training on out-of-domain dataset improves the performance. In this work, the out-of-domain dataset is used to train a bottleneck extractor in semi-supervised way, and this is different from the pre-training in the previous work. The feature extracted from the trained bottleneck extractor are more discriminative and robust, which can improve the accuracy of "dissimilar" music track. This model is easy to implement as there are a lot of un-transcribed music data available on the Internet. Also, the trained bottleneck extractor can be used to extract music features for other music systems. So, this model can be used more widely than the previous work.

3 DATA AND FEATURE

3.1 Data

The GTZAN database is used here as the target dataset to evaluate the proposed system. This dataset has been used by many authors [7] [18] [19] for music genre classification. GTZAN dataset is divided into ten genres: classical, jazz, blues, metal, pop, rock, country, disco, hip-hop, reggae. Each genre has 100 tracks with 30 seconds duration for every track. The detail of the the database is described as Table 1.

Table 1. Database Description

GTZAN		ISMIR	
genre	tracks	genre	tracks(train/test)
classical	100	Classical	320/320
jazz	100	Electronic	115/114
blues	100	Jazz/Blue	26/26
metal	100	Metal/Punk	45/45
pop	100	Rock/Pop	101/102
rock	100	World	122/122
country	100		
disco	100		
hiphop	100		
reggae	100		
total	1000	total	729/729

The ISMIR dataset [20] is used as the large out-of-domain dataset for the semi-supervised training of a bottleneck DNN. It is a large dataset which contains 100 hours of music data with 1458 music tracks. The time length of each track is different, ranging from 8 seconds to 42 minutes. Those tracks contain six genres: classical, electronic, jazz/blue, metal/punk, rock/pop, and world. In this paper, we do not use these genre labels, and the genre labels of ISMIR dataset is re-labeled by the ANN trained on GTZAN dataset and then used for semi-supervised training. The detail of the the database is described as Table 1.

3.2 Scattering Feature

The Mel-Frequency Cepstral Coefficient (MFCC) is widely used in many fields. The scattering feature is an extension of MFCC. It is computed by scattering the signal information along multiple paths, with a cascade of wavelet modulus operators implemented in a deep convolution neural network (CNN). For music, rhythm or beat is an important information to distinguish different music tracks, and short frame features cannot represent this information well. Therefore, long time representation can represent music data better for music genre classification, while MFCC may lose information by averaging spectrogram of long time window. The scattering moments can recover the lost information, which has been proved by [8] [21].

Before feature extraction, each audio file in ISMIR and GTZAN has been converted into a 22050Hz, 16 bit, and single channel WAV file. We extract the scattering feature for GTZAN and ISMIR using ScatNet [22]. In our work, we calculate first-order and second-order time scattering coefficients using a window of 370 ms with half overlap. The parameters for scattering transform are just the same as our previous work in [17]. After that, we stack the 3 consecutive frame features into a long frame feature vector with context information.

4 ST-BN System

4.1 Framework

The proposed ST-BN system framework is illustrated in Fig. 1. The DNN and ANN used in ST-BN are feed-forward neural networks. Karel's DNN in the Kaldi Toolkit[23] is used for DNN or ANN training. The DNN has one input layer, some hidden layers and one output layer. Within each hidden layer, the sigmoid function is used as the active function. For output layer, the soft-max function is used to compute the posterior probability. The learning rate is $8 * 10^{-6}$, and no pre-training is used for DNN training. The cross-entropy function is used as the objective function to optimize the ANN and DNN.

4.2 Semi-supervised Training of the Bottleneck Feature Extractor

A bottleneck DNN extractor is a special DNN which has a narrower hidden layer compared to other hidden layers. The narrow hidden layer is called bottleneck

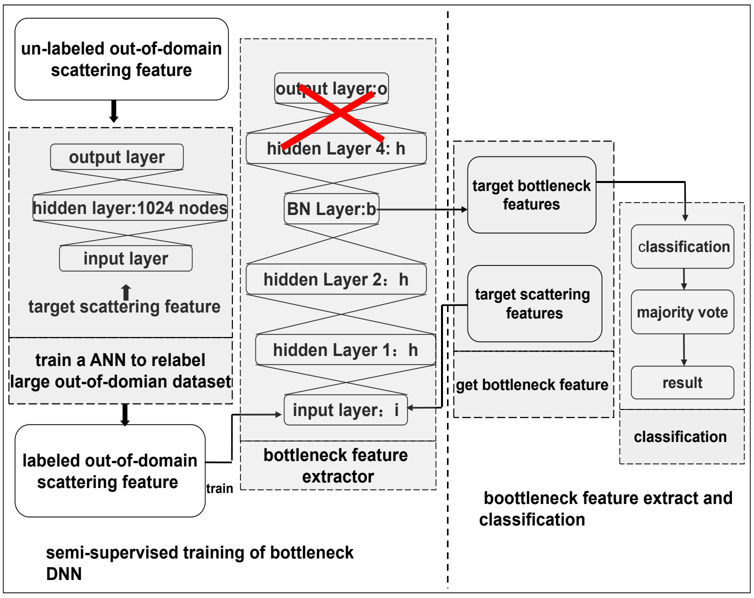
un-labeled out-of-domain scattering feature
output layer
hidden layer:1024 nodes
input layer
target scattering feature
train a ANN to relabel large out-of-domian dataset
labeled out-of-domain scattering feature
train
output layer:o
hidden Layer 4: h
BN Layer:b
hidden Layer 2：h
hidden Layer 1：h
input layer：i
bottleneck feature extractor
semi-supervised training of bottleneck DNN
target bottleneck features
target scattering features
get bottleneck feature
classification
majority vote
result
classification
boottleneck feature extract and classification

Fig. 1. Architecture of ST-BN System: "BN layer" in the figure is the bottleneck layer.

layer, and the bottleneck feature is generated from this narrow bottleneck layer. After we get the scattering feature for target dataset and large out-of-domain dataset, the frame scattering feature of out-of-domain dataset is used for semi-supervised training of the bottleneck DNN. It mainly contains two key stages.

In the first stage, we use scattering feature of target dataset (low-resource data) to label the large out-of-domain dataset. First, we randomly select 70% (we suggest equal or greater than 70% since little data cannot train a reliable ANN) of target scattering feature for training an ANN. The structure of ANN is described in section 4.1, and it has one hidden layer with 1024 nodes. We train the ANN for 200 epochs. Then, the scattering feature of out-of-domain dataset is put into the trained ANN. At last, the label of out-of-domain dataset is decided by the posterior probability output of the ANN: the genre index of the max probability is the label.

In the second stage, we train a bottleneck extractor. As in Fig. 1, the bottleneck DNN has one input layer, two hidden layers followed by a bottleneck layer, a hidden layer and one output layer. First, the frame feature of the labeled out-of-domain dataset is used as the input to train a bottleneck DNN. The training steps is 200. Then, we get the final bottleneck extractor by removing the hidden layer 4 and the output layer (in Fig. 1) of bottleneck DNN.

4.3 BN Feature extraction and Classification

After we get the bottleneck extractor, we extract the bottleneck features and then do the classification. The scattering feature of target dataset is used as the input of bottleneck extractor, then the output is bottleneck feature. After we get the bottleneck feature of the target dataset, we divide these bottleneck feature into training feature set and testing feature set.

In this paper, support vector machine (SVM) and ANN are used as the classifiers of ST-BN system. The SVM used here is a linear SVM, the training set of bottleneck feature is used as the input to train a liner SVM, and the testing set of bottleneck feature is used to get the frame test predict genre labels. The structure of ANN for classification is the same as the ANN in previous subsection. We use the training set of bottleneck feature to train the ANN, and then use testing set of bottleneck feature to get the soft-max output. After training and testing, we got the posterior probability for each frame feature. Each predict frame label is determined by the genre which has the maximum probability.

After classification, we get the predicted frame genre labels. However, people are always interested in the genre label of the whole music track (a clip of music), rather than the genres of the internal frames. So the majority voting is performed on the frame predict labels of the testing sets to get the label of each music track. The genre label of the whole music clip is decided by the majority of the genre labels on the internal frames. After that we get the genre label of each test music track, and the classification accuracy is computed on the track labels.

5 EXPERIMENT AND ANALYSIS

5.1 Evaluation Method and Baseline System

In this paper, we use 10-fold cross-validation to evaluate the performance. The features of target database are divided into 10 folds, 9 folds of which are used for training and the remaining one is used to test the classification accuracy. The classification accuracy is evaluated 10 times on the 10 different combinations of training/testing sets. The overall classification accuracy is calculated as the average of 10 independent 10-fold cross-validations.

SVM and DNN are used to build baseline systems: Baseline-SVM and baseline-DNN. The GTZAN dataset is used as target dataset. Frame scattering feature of target dataset is used as the input of baseline systems. Baseline-SVM use radial basis function as kernel function, and the parameters (the “cost” and “gama”) for kernel function are obtained by grid search algorithm on the GTZAN training feature sets. This baseline-DNN use the Karel’s DNN in Kaldi Toolkit[23], and other parameters are the same as DNN or ANN described in section 4.1. The baseline-DNN are three kinds of DNN with one, two and three hidden layers. Each hidden layer has 1024 nodes, and the training epochs are 1000. After testing, the genre which has the maximum probability is regarded as the frame genre labels. After getting frame labels from the baseline systems, the majority voting is used to get the track labels. The result is reported in Table 2.

Table 2. Classification Results of Baseline Systems

model name	classifier(hidden layers)	average accuracy
baseline-SVM	SVM	88.5%
baseline-DNN1	DNN(1024)	89.4%
baseline-DNN2	DNN(1024-1024)	89.7%
baseline-DNN3	DNN(1024-1024-1024)	90.1%

5.2 Experiment of ST-BN System

After getting scattering feature of target dataset and large out-of-domain dataset, we start to build the ST-BN system for music genre classification. In this paper, the ISMIR dataset is used as the large out-of-domain dataset and GTZAN dataset is used as target dataset. We first train 200 epochs to get an ANN to label ISMIR dataset. Then use labeled ISMIR dataset to train 200 epochs to get the bottleneck extractor. At last we use the bottleneck feature as the input of classifier to test the performance. In Table 3, we compare the proposed ST-BN system and other works on GTZAN dataset. The hidden layer structure of bottleneck DNN is "2048-2048-100-2048", and the classifier is linear SVM. From the result we can see that our model can achieve a good result and outperform other approaches.

Table 3. The Comparison of different works

model	average accuracy
ST-BN system (proposed system)	**94%**
Dai's work [17]	93.4%
Pan2009music [24]	92.4%
And2013deep [8]	91.9%
Lee2009auto [25]	90.6%
Eckleafeat [18]	84.3%
Hena2011uns [19]	83.4%

5.3 Experiment Analysis

First, we analyze the effectiveness of different classifiers and different structures of bottleneck DNN. The structure of bottleneck DNN can be represented as "i-h-h-b-h-o", just as in Fig. 1. The 'i','h','b','o' respectively represent the number of nodes in input layer, hidden layer, bottleneck layer and output layer. Fig. 2 shows the performance of ST-BN system with different 'h', different 'b' and different classifiers. From the figure, we can see that more nodes in bottleneck layer cannot always increase the performance, and more nodes in hidden layer can increase the result in some degree. The number of nodes in bottleneck layer cannot be too large, but also cannot be too small. The best number of node in bottleneck layer for GTZAN feature sets in this paper is 200. SVM perform better than ANN in ST-BN (in Fig. 2), but perform worse than DNN in the baseline (in Table 2).

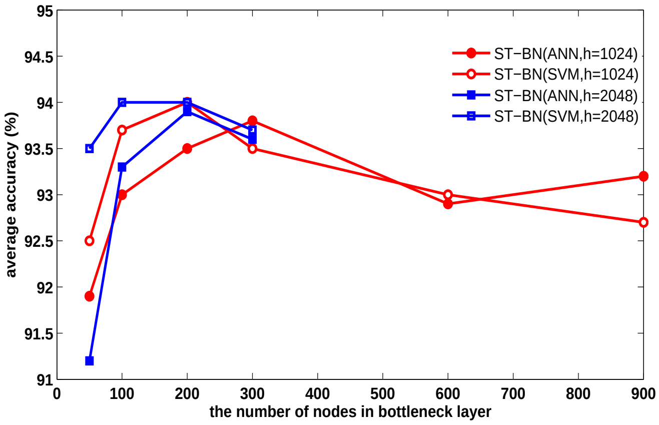

Fig. 2. the classification results of different bottleneck DNN. "SVM" and "ANN" in the legend indicate the classifier used for bottleneck feature classification. "h=1024" and "h=2048" represent the nodes in hidden layers.

That's maybe because SVM is more suitable for low dimensional features (The scattering feature has 525*3 dimension, while the bottleneck feature has a much lower dimension). Another reason is that the feature tends to be linear after the processing of bottleneck extractor, therefore a linear SVM is more powerful here.

Then, we analyze how our ST-BN system improves the performance. Fig. 3 shows the detailed frame results before majority voting, and they are two fold results chosen from the total ten folds. It indicates the frame rate for each test music track. The total tracks for testing sets is 100, so the track id in the figure is from 1 to 100. In Fig. 3, blue stars represent baseline frame rates in each track, and red triangles represent ST-BN frame rates in each track. A line connect baseline rate and ST-BN rate in a same track. A higher frame rate in a track may led to a correct classification label of the track when perform majority voting. In baseline system, we can see that some tracks have very high frame rate, and some tracks have very low rate. This "un-balanced" situation appears because the limited of training data. It makes the classification model easier to over-fitting. For the tracks which are similar to training tracks, the algorithm will have a good result, but for a "dissimilar" track, it will have a bad result. So in this paper, we use a similar large dataset to training the bottleneck DNN to make the model more stable. It improves the performance by increasing the rate of those low frame rate tracks. From the figure, we can see that many low frame rates are improved. Though some high rates in some tracks are decreased, these rates are still higher enough to produce a correct label for these tracks after majority voting. So the ST-BN system can achieve a better classification performance.

Fig. 4 shows the comparison between the baseline systems and ST-BN system. The figure shows the mean accuracy and variance of 10-fold cross-

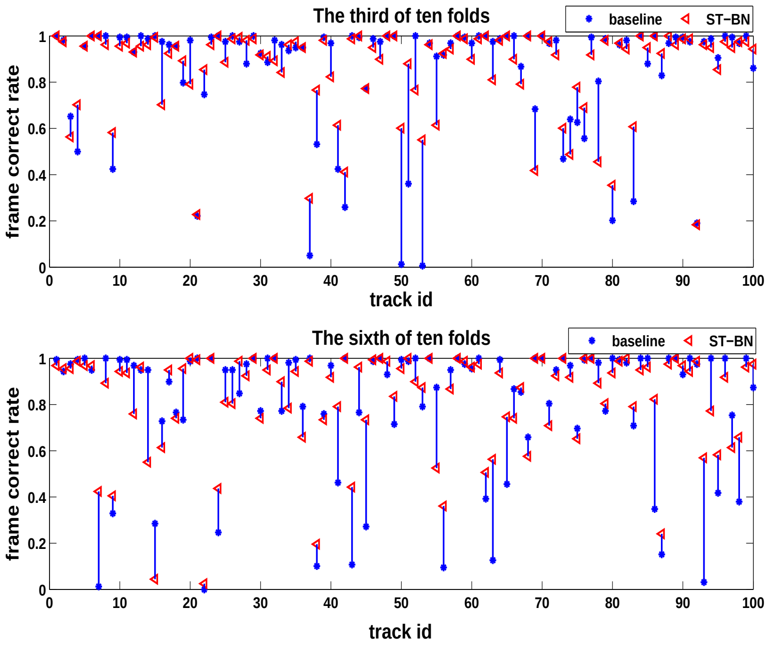

Fig. 3. The comparison of frame correct rate in test music tracks between baseline system and ST-BN system. The baseline system used here is baseline-DNN1 system. The structure of hidden layers in bottleneck DNN used in ST-BN is "1024-1024-100-1024", and the classifier is linear SVM.

validation. The accuracy used here is the accuracy after majority voting (track accuracy). From the result, we can see that our model outperforms baseline models.

At last, we analyze the limit of ST-BN system, which uses a large out-of-domain data to improve the performance of target dataset. The limit is that if the target dataset is also a large dataset (e.g. the same dataset as out-of-domain dataset), the result can not be improved so much. That's because that if the target dataset is a large dataset, there will be little "un-balanced" situation or a "dissimilar" tracks.

6 Conclusions

In this paper, we propose a music genre classification approach using bottleneck feature, which is learned at semi-supervised learning framework. This approach can help to learn a more stable feature representation, and the learned bottleneck features using semi-supervised learning from the out of domain data can be used to improve the performance of music genre classification. Experimental

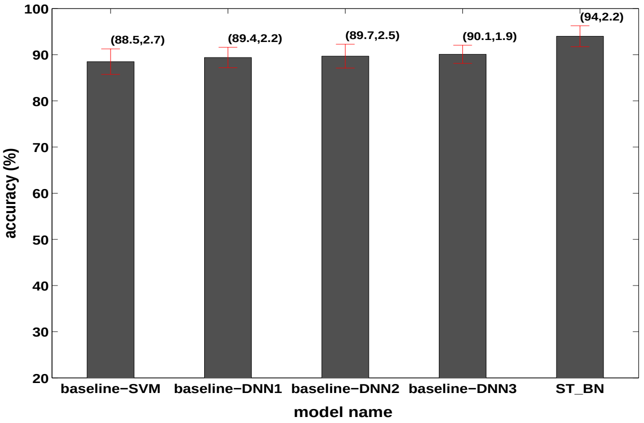

Fig. 4. The comparison of different models. The number in the figure is the mean accuracy and the standard deviation. The hidden layer structure of bottleneck DNN used in ST-BN system is "2048-2048-100-2048".

results show that the proposed system outperforms the current state-of-the-art best system on GTZAN dataset, and experimental analysis further shows the bottleneck features can improve the classification accuracy of those music tracks which are "dissimilar" to training tracks. Although the proposed approach is similar with "Multilingual bottleneck" and "Cross-lingual bottleneck" in ASR, but to our best knowledge, it is the first work to try to apply the bottleneck feature to music genre classification or audio classification.

7 Acknowledgements

This research was supported by following two parts: The China National Nature Science Foundation (No. 61573357, No. 61503382No. 61403370, No. 61273267 and No. 91120303, No. 61305027); Technical development project of state grid corporation of China entitled "machine learning based Research and application of key technology for multi-media recognition and stream processing".

References

1. T. Li, M. Ogihara, and Q. Li, "A comparative study on content-based music genre classification," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, ser. SIGIR '03. New York, NY, USA: ACM, 2003, pp. 282–289.
2. G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

3. T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *SLT*, 2012, pp. 234–239.
4. I. Mcloughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao, "Robust sound event classification using deep neural networks," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 23, no. 3, pp. 540 – 552, 2015.
5. X. Yang, Q. Chen, S. Zhou, and X. Wang, "Deep belief networks for automatic music genre classification," *ntM*, vol. 92, no. 11, pp. 2433–2436, 2011.
6. L. Lu and S. Renals, "Probabilistic linear discriminant analysis with bottleneck feature for speech recognition," in *Interspeech*, 2014.
7. S. Sigtia and S. Dixon, "Improved music feature learning with deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6959 – 6963.
8. J. Andén and S. Mallat, "Deep scattering spectrum," *CoRR*, vol. abs/1304.6763, 2013.
9. Q. B. Nguyen, J. Gehring, K. Kilgour, and A. Waibel, "Optimizing deep bottleneck feature extraction," in *IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, 2013, pp. 152–156.
10. D. Liu, S. Wei, W. Guo, Y. Bao, S. Xiong, and L. Dai, "Lattice based optimization of bottleneck feature extractor with linear transformation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5617 – 5621.
11. N. T. Vu, J. Weiner, and T. Schultz, "Investigating the learning effect of multilingual bottle-neck feature for asr," in *Interspeech*, 2014.
12. J. Li, R. Zheng, and B. Xu, "Investigation of cross-lingual bottleneck feature in hybrid asr system," in *Interspeech*, 2014.
13. V. H. Do, x. Xiao, E. S. Chng, and H. Li, "Kernel density-based acoustic model with cross-lingual bottleneck features for resource limited lvcsr," in *Interspeech*, 2014.
14. Y. Zhang, E. Chuangsuwanich, and J. Glass, "Language id-based training of multilingual stacked bottleneck features," in *Interspeech*, 2014.
15. H. Xu, H. Su, E.-S. Chng, and L. Haizhou, "Semi-supervised training for bottleneck feature based dnn-hmm hybrid system," in *Interspeech*, 2014.
16. 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.
17. J. Dai, W. Liu, C. Ni, L. Dong, and H. Yang, "Multilingual deep neural network for music genre classification," in *Interspeech*, 2015.
18. D. Eck and U. D. Montréal, "Learning features from music audio with deep belief networks," in *International Society for Music Information Retrieval Conference(ISMIR)*, 2010.
19. M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. Lecun, "Unsupervised learning of sparse features for scalable audio classification," *International Society for Music Information Retrieval Conference(ISMIR)*, 2011.
20. "Ismir," http://ismir2004.ismir.net/genre_contest/index.html.
21. J. Anden and S. Mallat, "Multiscale scattering for audio classification," in *International Society for Music Information Retrieval Conference(ISMIR)*, 2011, pp. 657–662.
22. "scattering," http://www.di.ens.fr/data/scattering/.
23. "Kaldi," http://kaldi.sourceforge.net.

24. Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Music genre classification using locality preserving non-negative tensor factorization and sparse representations," in *International Society for Music Information Retrieval Conference(ISMIR)*, 2009, pp. 249–254.
25. C. H. Lee, J. L. Shih, K. M. Yu, and H. S. Lin, "Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features," *IEEE Transactions on Multimedia*, vol. 11, no. 4, pp. 670–682, 2009.