CIAIC-GATFC SYSTEM FOR DCASE2018 CHALLENGE TASK2

Technical Report

Xueyu Han[1]*, Di Li*[1]*, Qing Liu*[1]*, Mou Wang*[1]*,Dexin Li*[1]*,Qian Wang*[1]*,*
Jisheng Bai[2]*, Ru Wu*[2]*, Bolun Wang*[3]*, Zhonghua Fu*[3]*,*

[1] Northwestern Polytechnical University, Center of Intelligent Acoustics and Immersive Communications, Xi'an, China, hanxueyu@mail.nwpu.edu.cn
[2] Northwestern Polytechnical University, Xi'an, China, baijs@mail.nwpu.edu.cn
[3] Northwestern Polytechnical University, Audio Speech Language Processing Group, Xi'an, China, berlloon@gmail.com

ABSTRACT

In this report, we present our method to tackle the problem of general-purpose automatic audio tagging described in DCASE 2018 challenge task 2. Two convolutional neural networks (CNN) models with different inputs and different architectures were trained respectively. Outputs from the two CNN models were then fused together to give a final decision. In particular, the distribution of training samples among 41 categories were unequally. Therefore, we presented a data augmentation method, which guaranteed the number of training samples per category was equal. A relative 21.4% improvement over DCASE baseline system [1] is achieved on the public Kaggle leaderboard.

Index Terms— Audio tagging, Convolutional neural networks, Data augmentation

1. INTRODUCTION

The goal of audio tagging is to assign a given audio recording with one or several pre-defined tags. Research on automatic audio tagging has been becoming increasingly popular in recent years. A task in the IEEE AASP Challenge DCASE 2016 aimed to perform audio tagging on 4-second audio recordings made in a domestic environment. Among the submitted systems, neural networks were widely used including convolutional neural networks and deep neural networks (DNN) [2], [3] and gained a considerable improvement over challenge baseline system.

The DCASE 2018 challenge task 2 focuses on a more complex general-purpose audio tagging problem with an increased number of categories and training data with annotations of different reliability. All audio samples in the dataset used in the DCASE 2018 challenge task 2 are distributed among 41 categories, which encompass sound events captured in a wide range of real-world environments. Except for the diversity of audio category, unequally distribution of training samples among categories is also a challenge to be considered.

To tackle the general-purpose audio tagging problem described in the DCASE 2018 challenge task 2, we chose to use CNN models which performed quite well in previous research of audio tagging and other neighboring fields such as acoustic scene classification and music genre classification. Meanwhile, we presented a data augmentation method to increase the number of training samples per category to ensure all audio categories have an equal number of audio samples.

Detailed method is described in Section 2. Experiments we carried out and the results are given in Section 3. Finally, in Section 4, we summarize our work and a brief conclusion is presented.

2. PROPOSED METHODS

2.1. Models

We built two CNN models for automatic audio tagging. Before training, audio samples in the dataset are processed in two different ways.

For the first model, we use raw audio directly to train a 1-D convolutional neural network [4]. As the duration of the audio samples ranges from 30s to 0.3s, we first change all audio samples to 2-sceond long audio clips. To do this, we pick 2 consecutive seconds randomly in each audio sample that is longer than 2s and, for those shorter than 2s, we pad zeros. Then, we take all these 2-second audio clips as input to train the 1-D CNN model. As shown in Fig. 1, the 1-D CNN model consists of four convolutional blocks. The convolutional block is defined as two 1-D convolutional operations followed by a max pooling operation. In each convolutional block, we apply ReLU activation in both convolutional layers. We apply a 10% dropout operation between convolutional blocks in order to make the whole model generalize better.

The second model takes mel-frequency cepstral coefficients (MFCCs) as input. As the audio pre-processing method described above, all audio samples are first changed into 2-second long audio clips. Then, we generate MFCCs from audio clips and use them to train our 2-D CNN model. According to Fig. 2, we can see that a 2-D CNN model with 7 convolutional layers is built. Similarly, we apply a ReLU activation function after each convolutional layer. But before that, a batch normalization function is applied over the outputs of the convolutional layer. Two dropout operations with probability 30% are applied between the third and fourth convolutional layers and the fifth and sixth convolutional layers respectively. The 2-D neural network is trained with Adam [5] optimizer with cross entropy as loss function.

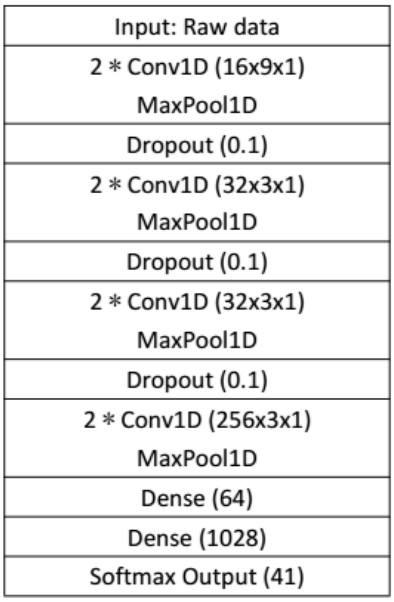

Figure 1: Structure of the 1-D CNN model.

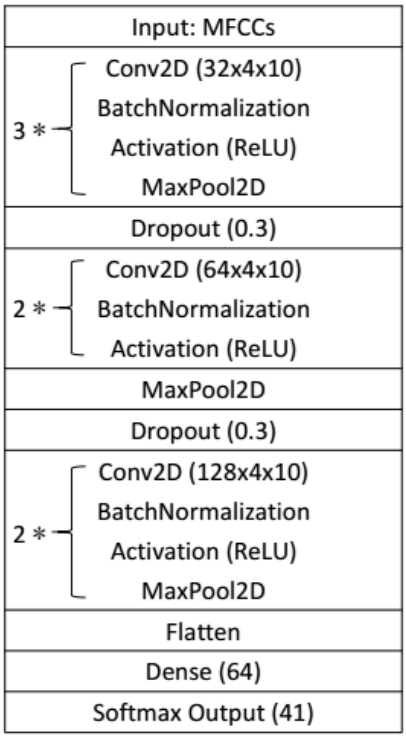

Figure 2: Structure of the 2-D CNN model.

2.2. Data augmentation

In general, audio samples in the train set are distributed equally among all categories, which means that the number of training samples per category is the same. As a result, training samples belong to different categories have an equal chance to be selected in a mini-batch when training CNN models. However, the audio data provided by the DCASE 2018 challenge task 2 is highly unequally distributed among 41 audio categories. The minimum number of training samples per category is 94. The maximum number of training samples per category is 300, which is more than 3 times the minimum. In this situation, when we are using mini-batch, it is likely that the model is biased to categories with larger training data. Meanwhile, the provided audio data in the train set have annotations of different reliability. Only less than 40% of the training data have manually-verified ground truth annotations while the left of the train set are annotated with a quality estimate of at least $65 - 70\%$.

To solve the problem of unequally distributed data and make full use of manually-verified data, we presented a data augmentation method. For each category that has training samples less than 300, we pick out a certain number of training samples from all manually-verified data of that category and then add them into the original train set to make sure that the number of training samples per category is same. By dong so, we actually augment the amount of training data as we take a randomly cut or padding operation described in Section 2.1 before we train models.

3. EXPERIMENTS

3.1. Dataset and experimental setup

The dataset provided for the DCASE 2018 challenge task 2 is a reduced subset of FSD [6]. The dataset contains 18,873 audio files with annotations from Google's AudioSet Ontology [7]. A total of 9,473 audio files are gathered in the train set. Another 9,400 audio files are in the test set while only about 1,600 samples with manually-verified annotations will be used to score systems. All audio samples provided in the dataset are 44.1 kHz, mono audio files.

In our experiments, 40 MFCCs were used in the audio preprocessing phase based on librosa [8]. Both of the two CNN models were implemented in Python using TensorFlow-based Keras deep learning package [9] and a 10-fold validation was applied.

Table 1: Performances of different models.

Model	LB Score
DCASE baseline	0.704
1-D CNN	0.811
2-D CNN	0.872
Fusion	0.918

3.2. Results

The outputs of the two CNN models were fused at last into one final decision by averaging them. In testing phase, each test file was assigned with top three categories. Performances of the 1-D CNN model, 2-D CNN model, fusion system and DCASE baseline system achieved on the public Kaggle leaderboard (LB) are all shown in Tab. 1.

It is noteworthy that the leaderboard is calculated with approx-

imately 19% of the test data. We can see that the 2-D CNN model performed better than the 1-D CNN model and a remarkable improvement of 0.214 LB score has been gained using the fusion system.

4. CONCLUSION

For the general-purpose audio tagging problem provide by the DCASE 2018 challenge task 2, we demonstrated our method to tackle it in this report. In our opinion, there are two challenges in this task. One is the diversity of audio category and the other is the unequally distribution of training data among different categories. To solve these two challenges, we built CNN models to extract higher level features based on raw data and MFCCs respectively and applied a data augmentation method to make full use of manually-verified data. Our system was evaluated on the public Kaggle leaderboard and a relative 21.4% improvement has been gained over the DCASE baseline system.

5. REFERENCES

[1] Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, Xavier Favory, Jordi Pons, and Xavier Serra. "General-purpose tagging of freesound audio with audioset labels: task description, dataset, and baseline." Submitted to DCASE2018 Workshop, 2018.URL: https://arxiv.org/abs/1807.09902, arXiv:1807.09902.

[2] Tomas L., and Alexander S. "CQT-BASED CONVOLUTIONAL NEURAL NETWORKS FOR AUDIO SCENE CLASSIFICATION." Detection and Classification of Acoustic Scenes and Events 2016 Workshop 2016.

[3] Qiuqiang K. and Iwnoa S. et al. "DEEP NEURAL NETWORK BASELINE FOR DCASE CHALLENGE 2016." Detection and Classification of Acoustic Scenes and Events 2016 Workshop 2016.

[4] `https://www.kaggle.com/fizzbuzz/beginner-s-guide-to-audio-data`.

[5] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[6] Eduardo Fonseca, Jordi Pons, Xavier Favory, Frederic Font, Dmitry Bogdanov, Andrs Ferraro, Sergio Oramas, Alastair Porter, and Xavier Serra. "Freesound datasets: a platform for the creation of open audio datasets." In Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017), pp 486-493. Suzhou, China, 2017.

[7] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. "Audio set: An ontology and human-labeled dartaset for audio events." Proceedings of the Acoustics, Speech and Signal Pro- cessing International Conference, 2017.

[8] `https://github.com/librosa`.

[9] F. Chollet, "Keras," https://github.com/fchollet/keras, 2016.