# Machine learning in environmental sound classification

Zengxiang Zhao

***Abstract*—**One of the main research in machine learning is environmental sound classification. This project tries to classify the raw audio signals from the open database ESC-50. We propose to use convolutional neural networks(CNN) to learn raw audio recordings. In this project, we will use LibROSA to convert audio recordings to log-scaled mel-spectrogram imagines. We will build three different CNN models to learn the patterns in the imagines and compare them by accuracy. As a result, Adam optimizer is better than SGD optimizer in this project, and Model-1 has the highest accuracy of 0.5192 among the three models. Augmentation of the dataset by flipping the spectrogram imagines horizontally and vertically doesn't improve the accuracy. Thus by changing the spectrogram pictures to augment the size of the database is not suitable for audio classification.

***Index Terms*—**machine learning, sound classification, CNN, ESC-50.

## I. INTRODUCTION

Signal processing can be divided into two categories: analog signal processing and digital signal processing[1]. Analog signal processing means the signal input is continuous and will be handed directly as shown in figure 1. While the digital signal processing will convert the analog signal input into a digital signal by A/D converter and be fed into the digital signal processor as shown in figure 2 [2]. As the advent of the digital computer, Digital Signal Processing increasing play an important role in signal processing. The term 'signal' here includes audio, video, speech, image, communication, geophysical, sonar, radar, medical, musical, and other signals…[3]

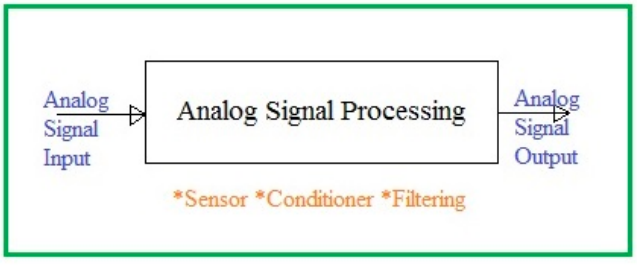


Figure 1

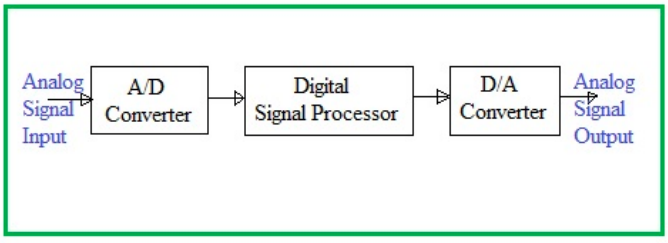


Figure 2

Machine learning emerged as shallow architecture, such as conventional hidden Markov models (HMMs) and support vector machines (SVMs), with one layer of nonlinear feature transformation, which is not good enough to apply to signal to process, as this process needs deep architecture to find the internal pattern of input[10].

Convolutional Neural Network(CNN) emerged from the human vision process and is widely used in image processing. Thanks to the computer power and increasing amount of available data, CNN can apply to voice recognition or natural language processing (NLP). This project will use CNN to classify the audio recordings. The main idea of machine learning is to learn patterns from the datasets. The signal audio recordings, such as speech and music, have to be converted to pictures that the machine can learn from. In some sense, each audio recording can be regarded as a picture. So the first step we should do before feed into the CNN is to convert the signals into imagines.

LibROSA is a python package concerning music and audio analysis[4]. We will use this library to convert the audio signal to log-scaled mel-spectrogram imagines. Then we can use the CNN architecture to classify these pictures. Thus the whole process becomes the traditional picture classification problems.

ESC-50 is an open dataset, which is a collection of 2000 environmental audio recordings with targets. The dataset loosely consists of 50 semantical classes loosely arranged into 5 major categories. And each class has 40 5-second-long recordings. The disadvantage of this dataset is that each class has only 40 recordings. The code we will implement is in Jupyter notebook " AudioTOSpectrum_Convertion". In the report, we will augment the dataset by flipping the spectrogram imagines horizontally and vertically[5]. The code is in Jupyter notebook "AgumentData".

In this project, we will firstly illustrate the CNN architectures. Then the outcomes of different CNN architectures and the datasets augmentation will be discussed.

## II. CONVOLUTIONAL NEURAL NETWORK(CNN)

CNN was inspired by the research of visual cortex and wildly used in image recognition[6,7]. CNN is a stack of different layers including convolutional layers, pooling layers, dropout layers, fully connected layers. Obviously, you must have an input layer to feed the network and output layers to produce the outcomes. We will use keras.models.Sequential to implement the CNN architecture.

### 1. Input layer

The shape of the input layer will be different from a different application. So the shape of the original image should be resized to the shape of the input layer. In some situations, the image is colorful, and you should pay attention to the position of the RGB channels of the image, cause there are the channels_first and channels_last. In this project, we will use the resize function( from skimage.transform import resize) to reshape the original images.

## 2. Convolutional Layers

Convolutional layers are the most important layers in CNN, cause the role of convolutional layers is to extract the features of the maps. Each convolutional layer consists of a different number of feature maps, which are obtained from the same number of filters. Every feature map in the layer is corresponding to one filters, different feature maps maybe use different filters. All neurons in one feature map use the same filters, so the feature map can memorize the simple pattern extracted by filters in every location of the image. Thus the CNN can recognize the pattern in the image, which is not limited in a specific location of the image. Through recognizing some simple patterns by feature maps in one convolutional layer, this convolutional layer can recognize a complex pattern contained in the imagine illustrated by figure 3 [8].

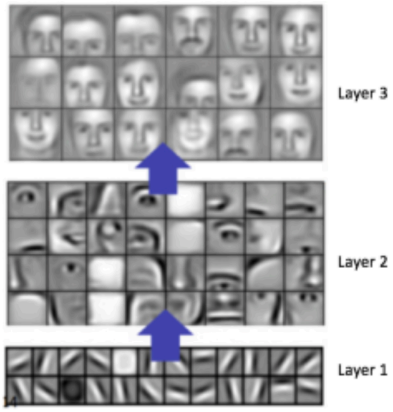

Figure 3

## 3. Pooling Layers

Pooling layers can shrink the size of the input image by just take the specific value of the limited neurons of the previous layers. Thus the pooling will reduce the computational load and prevent overfitting of the architecture. There are mainly two kinds of pooling layers: max-pooling layers and mean pooling layers. Max pooling layers will take the maximum of the area in the previous layer. While mean pooling layers will compute the average of the values of the area in the previous layer and act as the output of the next layer.

## 4. Fully connected layers

Fully connected layers are at the top of the stack of CNN architecture, which acts as the regular forward network to compute the output. Usually the last fully connected layer is the output layer with the softmax activation to produce the probability distribution according to classes.

## 5. CNN Architecture in the project

Figure 4 shows the main idea of CNN architecture.

**Model-1:**

The CNN architecture of Model-1 comes from the class homework. We run this architecture and find the learning rate is the highest but very time consuming since there are total 11,732,394 parameters in this architecture.

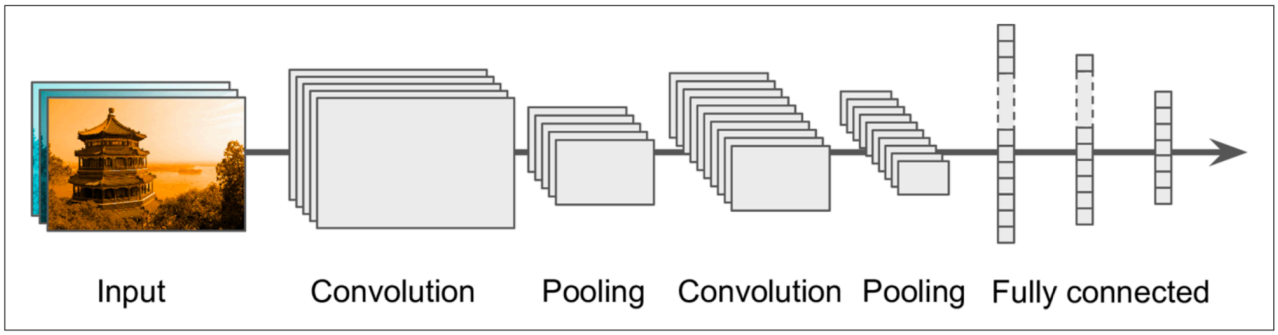


Figure 4

Table 1:Model-1 Architecture

| Layer | Type | Maps | Size | Kernel size | Stride | Padding | Activation |
|---|---|---|---|---|---|---|---|
| In | Input | 3(RGB) | 40×140 | – | – | – | – |
| C1 | Convolution | 32 | 38×138 | 3×3 | 1 | Valid | Relu |
| C2 | Convolution | 64 | 36×136 | 3×3 | 1 | Valid | Relu |
| C3 | Convolution | 80 | 34×134 | 3×3 | 1 | Valid | Relu |
| S4 | Max pooling | 80 | 17×67 | 2×2 | None | Valid | – |
| D5 | Dropout | 80 | 17×68 | 0.25 | – | – | – |
| F6 | Flatten | 91120 | – | – | – | – | – |
| F7 | Fully Connected | 128 | – | – | – | – | Relu |
| D8 | Dropout | 128 | – | 0.5 | – | – | – |
| Out | Fully Connected | 26 | – | – | – | – | Softmax |

## Model-2:

Based on the most widely known CNN architecture LeNet-5 created by Yann LeCun in 1998[7], we create the model-2 with some modification. The activation of "tanh" is replaced with "relu", and dropout layers are added in the architecture to shrink the parameters and reduce the computational load. This architecture has half the number of parameters( 5,071,826) compared with the Model-1.

Table 2:Model-2 Architecture

| Layer | Type | Maps | Size | Kernel size | Stride | Padding | Activation |
|---|---|---|---|---|---|---|---|
| In | Input | 3(RGB) | 40×140 | – | – | – | – |
| C1 | Convolution | 12 | 38×138 | 3×3 | 1 | Valid | Relu |
| C2 | Convolution | 24 | 34×134 | 5×5 | 1 | Valid | Relu |
| S3 | Max pooling | 24 | 17×67 | 2×2 | 2 | Valid | – |

| Layer | Type | Maps | Size | Kernel size | Stride | Padding | Activation |
|---|---|---|---|---|---|---|---|
| C4 | Convolution | 48 | 13×63 | 5×5 | 1 | Valid | Relu |
| D5 | Dropout | 48 | 13×63 | 0.5 | – | – | – |
| F6 | Flatten | 39312 | – | – | – | – | – |
| F7 | Fully Connected | 128 | – | – | – | – | Relu |
| D8 | Dropout | 128 | – | 0.5 | – | – | – |
| Out | Fully Connected | 26 | – | – | – | – | Softmax |

## Model-3:

The architecture of Model-3 comes from [9]. In this paper, the classification of audio and picture are fused to produce a 2-way classification output. In this project, Model-3 is a modification of audio architecture classification in paper[9], and the total number of parameters is 107,798 which is about 100 times less than the total number parameters of Model-1.

Table 3:Model-3 Architecture

| Layer | Type | Maps | Size | Kernel size | Stride | Padding | Activation |
|---|---|---|---|---|---|---|---|
| In | Input | 3(RGB) | 40×140 | – | – | – | – |
| C1 | Convolution | 12 | 38×138 | 3×3 | 1 | Valid | Relu |
| C2 | Convolution | 12 | 36×136 | 3×3 | 1 | Valid | Relu |
| S3 | Max pooling | 12 | 18×68 | 2×2 | None | Valid | – |
| C4 | Convolution | 24 | 16×66 | 3×3 | 1 | Valid | Relu |
| C5 | Convolution | 24 | 14×64 | 3×3 | 1 | Valid | Relu |
| S6 | Max pooling | 24 | 7×32 | 2×2 | None | Valid | – |
| C7 | Convolution | 48 | 5×30 | 3×3 | 1 | Valid | Relu |
| C8 | Convolution | 48 | 3×28 | 3×3 | 1 | Valid | Relu |
| S9 | Max pooling | 48 | 1×14 | 2×2 | None | Valid | – |
| F10 | Flatten | 762 | – | – | – | – | – |
| F7 | Fully Connected | 96 | – | – | – | – | Relu |
| D8 | Dropout | – | – | 0.5 | – | – | – |
| Out | Fully Connected | 26 | – | – | – | – | Softmax |

## III. RESULTS

We modify the code posted in ESC-50 Github[8] to produce the spectrum pictures, which is stored in jupyter-notebook " AudioTOSpectrum_Convertion ". As the computational ability of a personal computer, we only choose the first 26 classes of 50 classes in alphabet ascending order. The pictures are stored on the local computer. So the next step is to use the pictures dataset to train the tree models. Two optimizers are compared in Model-1: one is SGD optimizer and another is Adam optimizer. We found that Adam optimizer is better than the SGD optimizer. Even though we run the SGD optimizer only 3 epochs, we could realize that the growth of accuracy is not quickly compared with Adam optimizer. So the next two models will be compiled with Adam optimizer. In order to obtain higher accuracy, we enlarge the size of the database by flipping the spectrogram images horizontally and vertically. The total number of spectrogram images is increased to 2241. But the accuracy has not improved. The results are shown in table 4.

Table 4: Results

| | Model | Optimizer | Epochs | Accuracy |
|---|---|---|---|---|
| **Original Picture dataset(1040)** | Model-1 | SGD | 3 | 0.0385 |
| | Model-1 | Adam | 10 | 0.5192 |
| | Model-2 | Adam | 10 | 0.472 |
| | Model-3 | Adam | 10 | 0.329 |
| **After augmentation of dataset(2441)** | Model-1 | SGD | 10 | 0.2106 |
| | Model-1 | Adam | 10 | 0.4376 |
| | Model-2 | Adam | 10 | 0.4560 |
| | Model-3 | Adam | 10 | 0.3599 |

## IV. CONCLUSION

The highest accuracy among the three models is Model-1 with 0.5192, which compiled with Adam optimizer. Even though the audio recordings can be converted to pictures and learned by CNN. But we can't use the traditional augmentation method, flipping the spectrogram imagines horizontally and vertically, to enlarge the training dataset to improve the learning accuracy. So changing the original audio recordings to augment the dataset may be a better choice, such as time shifting, speed tuning, mix background noise and volume tuning[9].

**REFERENCES:**

[17] Richard G. Lyons, “Understanding Digital Signal Processing”, Pearson Education, 2004.
[2] “Analog Signal Processing vs Digital Signal” ,http://www.rfwireless-world.com/Terminology/analog-signal-processing-vs-digital-signal-processing.html.

[3] Li Deng, “Expanding the Scope of Signal Processing” , IEEE SIGNAL PROCESSING MAGAZINE, vol. 25, issue 3, page 2-4, 2008.

[4] LibROSA website: https://librosa.github.io/librosa/

[5] Bharath Raj, “Data Augmentation | How to use Deep Learning when you have Limited Data — Part 2”, https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced.

[6] G. Hinton, S. Osindero, and Y. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, vol. 18, pp. 1527–1554, 2006

[7] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.

[8] Karol J. Piczak, “ESC-50: Dataset for Environmental Sound Classification”, https://github.com/karoldvl/ESC-50.

[9] Qishen Ha, “Augmentation methods for audio”, https://www.kaggle.com/haqishen/augmentation-methods-for-audio

[10] Dong Yu and Li Deng, “Deep Learning and Its Applications to Signal and Information Processing”, IEEE SIGNAL PROCESSING MAGAZINE, vol. 28, issue 1, page 145-154, 2011.