Instrument Classification Using SVM and Other Machine Learning Algorithms

Zeting Luan
Department of Electrical and Computer Engineering
UC San Diego, CA
zluan@ucsd.edu

Zheng Yang
Department of Electrical and Computer Engineering
UC San Diego, CA
zhy013@ucsd.edu

Abstract

This project is aiming to classify different musical instruments from live concert or orchestra performance using Machine Learning algorithms. By doing so, the audiences can understand music performance better, and music creators can also benefit from it to improve their production by analyzing how each instrument is played during the performance. To figure out which Machine Learning algorithms are more suitable for the instrument classification problem, we compare different algorithms, and evaluate their performances. We transform the audio classification problem to image classification problem by processing the audio signal with Mel-Frequency Cepstral Coefficients (MFCC). When it comes to image classification, our hypothesis is using Support-Vector Machines (SVM) for its effectiveness in high dimensional spaces and the memory efficiency. Then we also want to compare how traditional Machine Learning algorithms such as Multi-layer Perceptions (MLP) and Random Forest perform against SVM. Finally, we take a step further, and compare how well the traditional Machine Learning algorithms perform against Deep Learning algorithms such as Convolutional Neural Network (CNN).

1. Introduction

With the development of machine learning, data scientists increasingly try to come up with models to deal with tasks in other fields. One of such tasks is instrument classification, where a model tries to detect and classify the involved musical instruments from audio excerpts. Unlike the case for image data, audio data usually has a high sampling rate, making it hard to process directly. Also, audio signals are more subject to environment factors such as noises. Therefore, it is essential to properly pre-process audio data.

In our project, we follow the step and apply an audio processing method called Mel-Frequency Cepstral Coefficients (MFCC) to the input audio signal like [1] and [2]. MFCC aims to reduce the dimensionality of audio data while extracting its characteristic features. It is constructed in a way based on the human perception, which means that the extracted features serves as an intuitive representation of the audio signal. MFCC outputs a spectrogram of the audio signal, which also serves as a feature map representing the signal. The output of the MFCC is shown in Figure 1.

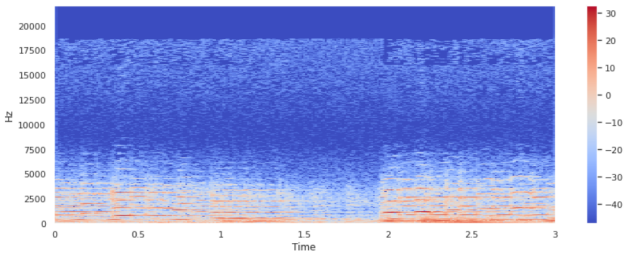

Figure 1. MFCC Output: A Time-Cepstrum Spectrogram.

After processing the audio signal, we apply Machine Learning algorithms to implement instrument classification based on extracted features. Since different algorithms have their own advantages in different scenarios, we choose several representative algorithms to experiment on and compare their performances in the instrument classification task. Apart from traditional Machine Learning algorithms, we also train a CNN as a comparison between traditional Machine Learning and Deep Learning. The overall process flow of instrument classification is demonstrated in Figure 2. Besides, for a better comparison, we combine traditional Machine Learning algorithm and Deep Learning algorithm for this task. We replaced the fully-connected layers of the

CNN with SVM and Random Forest[3] for classification.

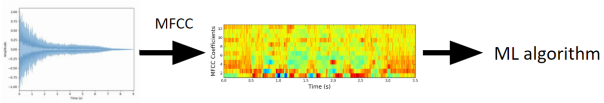

Figure 2. The Overall Process Flow.

2. Motivation

Over the years, music enthusiasts and professionals have grown interested in studying the instruments separately in sheet music or performance from an orchestra. For example, Violin players might be more interested in listening to only the violin part from a concert to figure out how should they improve their own performances. Although music performers and producers can listen to only a certain part of the music they produced in the record studio, it is not the case for live concert. During live performances, all instrumental and environmental sounds are mixed together and sampled at a high rate, yielding high-dimensional data with considerable noise. Such data can hardly be used without certain processing techniques, making it a difficult task to have better understanding of different instruments from a live performance separately. To approach this problem, we explore in this work the possibility of training an instrument classifier on MFCC outputs and audio labels, using different neural network architectures.

3. Related Works

In a 1976 paper on speech recognition, P. Mermelstein introduced the idea of MFCC as a novel representation of speech sounds[4]. He found in his experiments that MFCC has desirable properties for distance measurement, which can benefit recognition results significantly.

Four years later, S. Davis and P. Mermelstein compared several different parametric representations of audio signals, among which MFCC turns out to have the best performance[5]. They concluded that MFCC extracts more perceptually relevant information from the audio signal than other forms of representation.

In 2009, H. Lei and E. Lopez modified the filterbank configurations of MFCC for telephone speaker recognition[6]. The new coefficients, which they called a-MFCC, achieved better results for this specific task. Their work showed the flexibility of the MFCC methodology.

Shortly afterwards, M. A. Hossan et al. proposed a new MFCC implementation method based on distributed Discrete Cosine Transform (DCT-II)[2]. They performed speaker verification tests on different approaches and found that distributed DCT-II based MFCC outperforms all former feature extraction methods.

4. Methods

4.1. Mel-Frequency Cepstral Coefficients (MFCC)

MFCC is one of the most important methods we are using in this work, since feature extraction is essential for the instrument classification problem. The coefficients are collected through the following steps of calculation.

Step I. Framing

Frame the signal into small frames with length of 25 ms.

Step II. Discrete Fourier Transform (DFT) and windowing

Apply DFT with Hamming window to the frame. Step II is concluded by eq.1.

$$X(k) = \sum_{n=1}^{K} X(n)w(n) \exp\left(-\frac{j2\pi kn}{N}\right) \tag{1}$$

Specifically, the Hamming window function is as follow in eq.2.

$$W(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \tag{2}$$

The periodogram-based power spectral estimate for the frame X(n) is given by eq.3.

$$P(k) = \frac{1}{N}(|X(n)|)^2 \tag{3}$$

Notation	Meaning
N	Number of samples in each frame
$X(n)$	Signal frame
$W(n)$	Window function (Hamming window)
K	DFT length
$P(k)$	Periodogram-based power spectral estimate

Table 1. Notations

Step III. Mel filter bank processing A set of triangular filters are applied to periodogram from step 2, which is showed in Figure 3 The filters are used to compute the summation of filter spectral components and output the process to a Mel scale. For a given frequency f in Hz, the Mel is calculated by eq.4.

$$M = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{4}$$

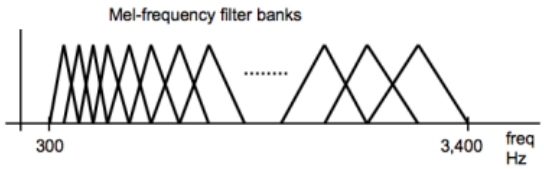

Figure 3. Mel Scale Filter Bank[6]

Step IV. Discrete Cosine Transform Discrete Cosine Transform (DCT) is a finite sequence of data points that is a combination of a summation of cosine signal. Use DCT in this process to convert the log Mel spectrum into time domain. After the conversion, the result is called Mel-Frequency Cepstrum Coefficient.

Step V. Delta coefficients The MFCC feature vector represents the power spectral of a single frame, but we need to consider the information in the dynamics when it comes to audio signal. We calculate the delta coefficients using the following equation in eq.5. We compute the delta coefficient d_t from the t-th frame and the static coefficients we get from step IV.

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2} \tag{5}$$

Notation	Meaning
d_t	Delta coefficient
t	Frame number
c	Static coefficients

Table 2. Notations

4.2. Multi-Layer Perceptrons (MLP)

Multi-Layer Perceptron (MLP) is a type of artificial neural network (ANN), which usually consists of at least three layers of nodes. Since our object was to compare how traditional Machine Learning algorithms compete with Deep Learning algorithms, we designed the MLP to have 6 dense layers with ReLU activation function.

4.3. Support Vector Machine (SVM)

Support Vector Machine (SVM)[7] is a supervised learning algorithm specializing in classification and regression task. SVM is one of the most robust traditional Machine Learning algorithm, so we are interested to find out how well it performs on instrument classification problem. Since there are 11 classes in IRMAS dataset[8], we aim to use SVM for multi-class classification. The basic idea for multi-class SVM is transform the multi-class problem into multiple binary classification problem. We tried out different kernels and found out that the radial basis function kernel performs the best.

4.4. Random Forest

Random Forest is an ensemble learning method based on decision tree learning[3]. While maintaining the advantage of high training and processing speed, random forest overcomes the over-fitting problem for decision trees, thus becoming a competitive approach to training classifiers over unbalanced data.

4.5. XGBoost

Extreme Gradient Boosting, or XGBoost, is a machine learning library based on gradient boosting decision tree (GBDT)[9]. XGBoost utilizes gradient boosting methods to achieve higher accuracy than a single decision tree, but in the mean time it over-fits much easier than random forest.

4.6. Adaboost

Adaptive Boosting, or AdaBoost in short, is a boosting algorithm which adjusts adaptively to loss by predicting the distribution of weak learners[10]. AdaBoost can be generalized to learn any function with arbitrary finite range, which is desirable for most classification problems.

4.7. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is one of the most common network architectures for deep learning. It is particularly useful in image processing, having the advantage of learning implicit features directly from data. The architecture of our CNN model is shown in Table 3. The activation functions for the convolutional and fully-connected layers are all selected to be ReLU.

Layer	Filter	Output	Kernel	Stride	Padding
conv1	4	25	3	1	1
conv2	16	25	3	1	1
conv3	64	25	3	1	1
pool	64	13	2	2	1
flatten	1	832			
dense1	1	128			
dense2	1	64			
dense3	1	11			
softmax	1	11			

Table 3. CNN Architecture

Considering the fact that CNN models might be too complex for our problem, we add dropout layers before each fully-connected layer to prevent over-fitting, using a dropout rate of 0.3 for hidden layers and 0.2 for the output layer. For the same reason, we also introduce L2-regularization with penalty $\lambda = 0.001$ to our CNN model.

4.8. CNN-SVM

CNN is constructed with convolutional layers and fully-connected layers. Fully-connected layers serve the purpose of classification. We replaced the fully-connected layers with SVM to test if it can be a better classifier.

4.9. CNN-Random Forest

Similiar to what we did in CNN-SVM, we replaced the fully-connected layers with Random forest as a classifier.

4.10. Evaluation

4.10.1 Accuracy

The accuracy score is calculated using eq.6. The accuracy score is the fraction between the number of true results and the total number of cases.

$$\text{Accuracy} = \frac{T_p + T_N}{T_p + F_p + T_n + F_n} \tag{6}$$

4.10.2 Precision

The precision score is calculated using eq.7. The precision score evaluate the proportion of the predicted positive cases is truly positive.

$$\text{Precision} = \frac{T_p}{T_p + F_p} \tag{7}$$

4.10.3 Recall

The recall score is calculated using eq.8. The recall score evaluate the proportion of the predicted positive cases is classified correctly.

$$\text{Recall} = \frac{T_p}{T_p + F_n} \tag{8}$$

4.10.4 F1 score

F1 score is calculated using eq.9. The value of F1 score is between 0 and 1, which can be viewed as the harmonic mean of the precision and recall score. For a model, we want it to have both decent precision and recall score.

$$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{9}$$

Notation	Meaning
T_p	True positive
T_n	True negative
F_p	False positive
F_n	False negative

Table 4. Notations

4.10.5 Confusion Matrix

A confusion matrix visually describes the prediction results on a classification problem. The values on the matrix represent either the correct or the incorrect predictions.

5. Experiments

5.1. Dataset

We used the Instrument Recognition in Musical Audio Signals (IRMAS) dataset[8] for the instrument classification problem. IRMAS is widely used not only in musical instrument recognition and classification problem, but also musical genre classification problem. The training data is consists of 6705 audio files in 16 bit stereo wav format sampled at 44.1kHz. They are excerpts of 3 seconds from more than 2000 distinct recordings. The audio files are also labeled with 11 different classes including cello, clarinet, flute, acoustic guitar, electric guitar, organ, piano, saxophone, trumpet, violin, and human singing voice. The testing data is consists of 2874 excerpts in 16 bit stereo wav format sampled at 44.1kHz. The length of the audio files in testing data is between 5 and 20 seconds. Figure 4 shows the distribution of audio files among different classes.

Instrument	Training data
Cello	388
Clarinet	505
Flute	451
Acoustic guitar	637
Electric guitar	760
Organ	682
Piano	721
Saxophone	626
Trumpet	577
Violin	580
Vocal	778

Figure 4. IRMAS Dataset Training Data

5.2. Experiment results

5.2.1 MLP

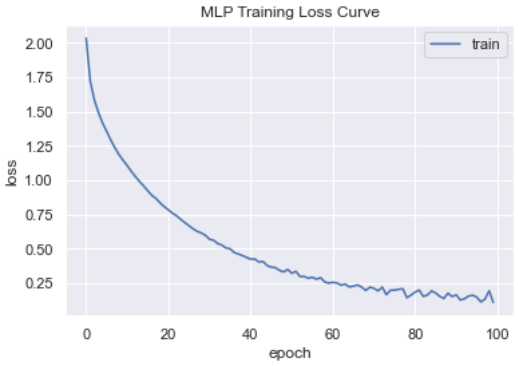

Figure 5. MLP Training Loss Curve

Figure 6. MLP Confusion Matrix

5.2.2 SVM

Figure 7. SVM Confusion Matrix, 'rbf', C = 10, γ = 0.1

Figure 8. SVM Confusion Matrix, 'linear', C = 10, γ = 0.1

Figure 9. SVM Confusion Matrix, 'sigmoid', C = 10, γ = 0.1

5.2.3 Random Forest

Figure 10. Random Forest Confusion Matrix

5.2.4 XGBoost

Figure 11. XGBoost Confusion Matrix

5.2.5 AdaBoost

Figure 12. AdaBoost Confusion Matrix

5.2.6 CNN

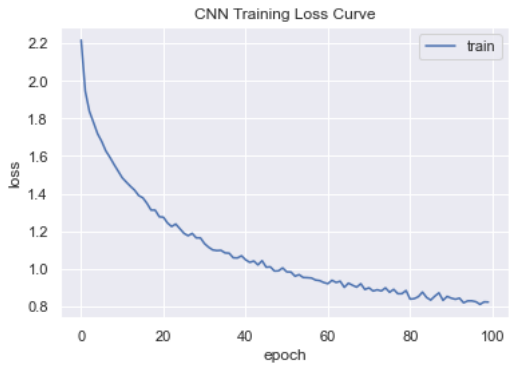

Figure 13. CNN Training Loss Curve

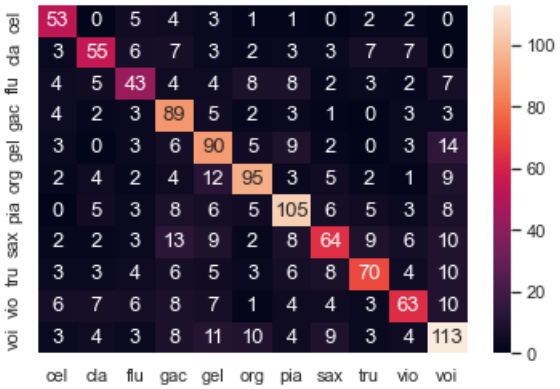

Figure 14. CNN Confusion Matrix

5.2.7 CNN-SVM

Figure 15. CNN-SVM Confusion Matrix

5.2.8 CNN-Random Forest

Figure 16. CNN-Random Forest Confusion Matrix

5.3. Results evaluation

Algorithm	Accuracy	Precision	Recall	F1-Score
MLP	0.58	0.55	0.56	0.55
SVM	0.70	0.71	0.70	0.70
RF	0.61	0.64	0.61	0.61
XGBoost	0.59	0.58	0.58	0.58
AdaBoost	0.33	0.34	0.34	0.32
CNN	0.62	0.62	0.62	0.62
CNN-SVM	0.60	0.60	0.60	0.60
CNN-RF	0.61	0.60	0.61	0.60

Table 5. Results evaluation

The results evaluation for different algorithm are shown in table 5. The precision, recall, and F1-Score are weighted based on support(the number of true instances for each label). SVM has the best performance among all the algorithms we tested, and even surpass the CNN-based algorithm. As a result, CNN does not have a significantly better accuracy compared to other traditional Machine Learning algorithms. Besides, by replacing the fully-connected layers with SVM or Random Forest as classifiers do not improve the performance. This is especially obvious in the SVM case, for its accuracy drop from 0.70 to 0.60.

In regards to the good performance of SVM in this task, this could be due to the features among different instruments are more distinguished compared to other classification tasks. It benefits SVM a lot since different classes have a relatively clearer margin of separation. Our hypothesis that CNN and CNN-based models would have a better performance is based on SVM does not do well in large dataset, and sensitive to noise. But IRMAS is a relatively small dataset, and even though it has some degree of noise, the process of MFCC helped it eliminate a large amount of noise. Therefore, SVM maximized its advantages in the instrument classification task.

6. Future work and improvement

Music instrument classification benefits not only the audience but also the music creators, so our future works aim to further improve the their experience. Since IRMAS dataset also has the music genre labels, we can modify the MFCC process accordingly so that the features extracted represent each class. Compared to instrument classification, people who are interested in music theory might be more interested in studying different music genres.

Apart from music genre classification, we would also want to separate different instrument from a live concert or orchestra. Zhao *et al.*[11] proposed U-Net for sound source separation by converting the sound signal into pixels, which converts the audio problem into image problem.

For this project, we can further make improvement regarding instrument classification. IRMAS is a great dataset because it provides a decent number of audio files with 11 classes, however with a large dataset, Deep Learning algorithm might prevail and give better result. As we mentioned before, MFCC helped reducing the background noise from the input. We can intentionally add some noises to the input and test if it significantly lower the performance. By doing so, we can further improve the robustness of the models.

7. Conclusion

In this project, we compared how traditional Machine Learning algorithms and Deep Learning algorithms perform in classifying musical instrument. Since classifying audio signal directly is difficult, we processed the audio signal using MFCC, and extracted the audio features in to spectrogram. Then, we applied different Machine Learning algorithms and compare their performance. Our initial hypothesis was that CNN and CNN-based model would outperformed traditional Machine Learning model, but the results showed that SVM had the best performance among all the models. Our speculation is that the dataset we used has a relatively small amount of data. Therefore, in the situation that the training data is sparse, traditional Machine Learning algorithm like SVM has a outstanding performance.

References

[1] L. Muda, M. Begam, and I. Elamvazuthi, "Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques," *arXiv preprint arXiv:1003.4083*, 2010.

[2] M. A. Hossan, S. Memon, and M. A. Gregory, "A novel approach for mfcc feature extraction," in *2010 4th International Conference on Signal Processing and Communication Systems*, pp. 1–5, IEEE, 2010.

[3] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1, pp. 278–282, IEEE, 1995.

[4] P. Mermelstein, "Distance measures for speech recognition, psychological and instrumental," *Pattern recognition and artificial intelligence*, vol. 116, pp. 374–388, 1976.

[5] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[6] H. Lei and E. Lopez, "Mel, linear, and antimel frequency cepstral coefficients in broad phonetic regions for telephone speaker recognition," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, "A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals.," in *ISMIR*, pp. 559–564, Citeseer, 2012.

[9] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[10] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[11] H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba, "The sound of pixels," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 570–586, 2018.