Automatic Piano Music Transcription

Jianyu Fan Qiuhan Wang Xin Li

Jianyu.Fan.Gr@dartmouth.edu Qiuhan.Wang.Gr@dartmouth.edu Xi.Li.Gr@dartmouth.edu

1. Introduction

Writing down the score while listening to the music can be done by very experienced musicians. However, it is time consuming and a painful task.

Our aim is to convert piano music recording to MusicXML [1] files. The music recording file contains all the information about audio signals that can be used to extract audio features directly. We transcript the wave files into MusicXML file which has simple representation of music information such as pitch, duration, rhythm, and dynamics. This work has many applications such as score following, interactive performance. There are many research has been done to solve this problem. People used Non-Negative Matrix Factorization [2],PCA [3],HMM [4], and reached promising results. There are competitions of MIREX every year judging which new method can acquire the best accuracy.

2. Proposed Solution

We proposed to use Extreme Learning Machine [5][6] (as the learning model)model to do the note determination . And we compared the result with Naïve Bayes classification and K means cluster. Out proposed solution is described below.

First, we read wave files that are recorded by playing the electronic piano and we did time segmentation to get the start time of each note. Second, we extracted audio feature such as Harmonic feature [7] and Chroma feature [8] from each note. Third, we used our features to train the Extreme Learning Machine model. Fourth, we predicted the note from out test data. Finally, we parse the MusicXML files to show the score of our predicted notes. The wave files we used are monophonic piano music recorded by playing the electronic piano. Out training data contains a 60 pieces and our test data contains about 10 pieces. The total number of note is 2760.

3. Approach

Time Segmentation (Onset detection)

We have run the onset detection algorithm [9] [10] to get the starting point of each note in each piece. The algorithm uses the edge detecting filter, doing fast convolution and finding where exactly the peak occurs. For several pieces, the note overlap too much and cannot be segmented well, we remove those pieces.

Figure1 shows the result:

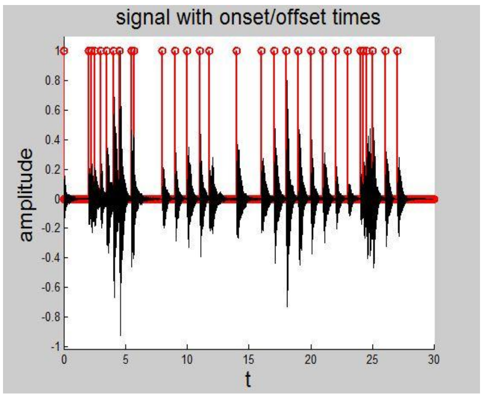


Figure 1. Time segmentation of training data #18

Feature extraction

In each segment, we've extracted audio features. We've chosen two features and decide to test on them separately.

Harmonic feature:

We framed on each segment first. In each frame, we have computed the energy. If the energy is lower than a threshold then we regard this frame as non-note frame and throw this frame away [4]. Then we have computed each point through frequency axis and got the result of the position of harmonics. If there is harmonic in one point then the value of this point will be 1, if not. Then the value will be 0. Therefore the harmonic feature is a binary feature. The feature vector will then be$(y^1 \ldots, y^K)$. We assumed that the components of the vector are conditionally independent. $y^1 \ldots y^k$ are harmonic features.[4][7]. As for harmonic feature, each frame is a training data. We have chosen window size to be 2048. Therefore, we have a really large dataset the number of samples in which is 162104.

Chroma Feature:

Chroma features consist of a twelve-element vector with each dimension representing the intensity associated with a particular semitone, regardless of octave [8]. In this case, we also compute 10 values of Chroma in each semitone; therefore we have 120 dimensions Chroma features. Chroma feature is continuous feature. We extracted Chroma feature on each note there, therefore, we have a smaller dataset the number of samples in which is 2700.

Extreme Learning Machine

We used Extreme Learning Machine [5] to do note determinations. Extreme Learning Machine model has been used for signal processing a lot. It's an improvement of Singular Hidden Layer Feedforward Neural Network.

The traditional feedforward neural network has several disadvantages. First, it takes long because gradient descent method takes many time iterations to get the appropriate weight value and bias. Second, we have to tune parameters of the networks iteratively. Third, the result might be local minimum but not global minimum [5] [6].

The Extreme Learning Machine is faster and is able to acquire global minimum. The parameter of hidden nodes is independent of each other and is independent of training data. In this model, (the model) we could generate parameter of hidden layer nodes before input the training data.

The Algorithm of Extreme Learning Machine model [5] [6].

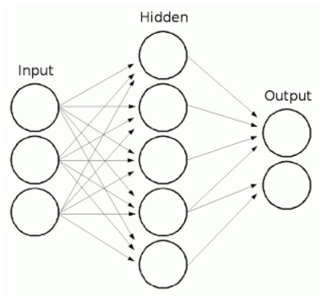


Figure 2. Neural network[5]

The figure of conventional neural network is like Figure 2.[5] [6].

For training set with N different data, $Z = \{(x_i, t_i) | i = 1, \dots, N\}$, in this Z, $x_i = [x_{i1}, x_{i2}, \dots, x_{in}] \epsilon R^n$, $t_i = [t_{i1}, t_{i2}, \dots, t_{in}] \epsilon R^m$. There are L nodes in hidden layer. The activate function is $g(x)$. For any a_j and b_j, we can acquire result that is close to N samples without error. It can be showed as:

$$O_i = \sum_{j=1}^{L} B_j g(a_j, b_j, x_i) = t_i$$

a_j is the input weight value, b_j bias of hidden layer, x_i is input vector, O_i is output vector, B_j is the output weight value.

The equation above can be written as

$$HB = T$$

In this equation, $B_{L*m} = [B_1, B_2, \dots, B_L]^T$, $T_{N*m} = [T_1, T_2, \dots, T_N]^T$, H is the hidden layer,

$H(a_1, \dots, a_L;\ b_1, \dots, b_L;\ x_1, \dots, x_N) = \begin{bmatrix} g(a_1, b_1, x_1) & \cdots & g(a_L, b_L, x_1) \\ \vdots & \ddots & \vdots \\ g(a_1, b_1, x_N) & \cdots & g(a_L, b_L, x_N) \end{bmatrix}_{N*L}$, the jth column in H is the output of $x_1, x_2, \dots, x_N$ of jth hidden node.

Then we can get:

$$\hat{B} = (H^T H)^{-1} H^T T$$

4. Result and Analysis

Average Error Rate by using Harmonic Feature

Model	Extreme Learning Machine	Na ïve Bayes
Average Error Rate (%)	22.1	12.3

Table 1.

Average Error Rate by using Chroma Feature

Model	Extreme Learning Machine	K means
Average Error Rate (%)	17.9	69.4

Table 2.

Table1 and Table2 show the results of using different models on different features. As for harmonic feature which is a binary feature. We used 150000 data as training data and 12104 data as test data. The Extreme learning Machine model has gotten the average error rate of 22.1% while the Na ïve Bayes model get better result of 12.3%. In harmonic feature, we computed each point through frequency axis and got the result of the position of harmonics. Harmonic feature is suitable for the independent assumption of Na ïve Bayes Model.

Apart from that,

The Harmonic Feature has a large training dataset. We have 150,000 training samples. According to the EML theory, when the number of hidden layer node approximates the number of samples, the model would produce very good result. However, limited by the computer, we can only set up to 300 hidden layer nodes. After testing, we found that 100 nodes have not much difference with the result of 300 nodes. Compare to the huge sample number, the node number is quiet small, which produced a bad result.

When using Chroma feature, we compare the result of Extreme Learning Machine Mode with K means cluster. The dataset of Chrome feature is smaller and we used 2760 training samples and 368 testing samples. We choose the number of nodes in hidden layer to be 700 and reach a promising result. The Kmeans cluster is working but gets lower accuracy due to the fact of unsupervising model and less training data.

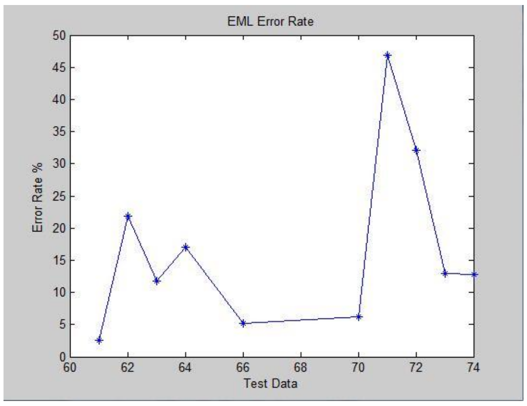


Figure 3. ELM test Error Rate of all test data

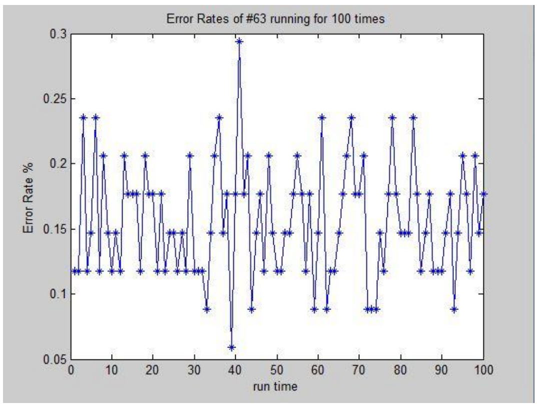


Figure 4. ELM test Error Rate of #63 run 100 times

We have tested 10 pieces of music recordings. The Figure3 shows that error rate of #71 and #72 test data are quite high. We found these two test sets having some notes overlapped which becomes a polyphonic problem. Figure 4 shows the result of run ELM on 63# for 100 times.

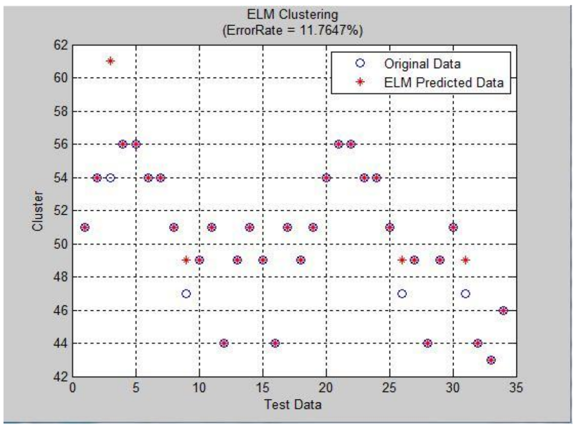
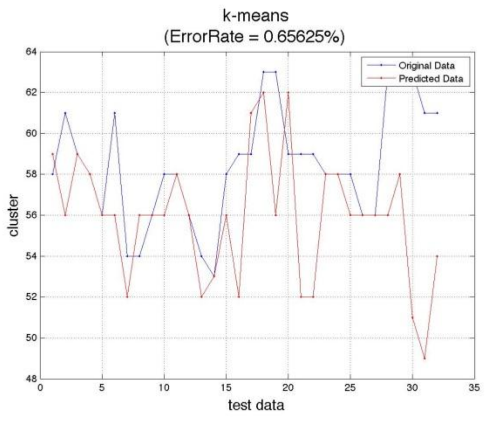


Figure 5. The Cluster situation of #63

Figure 5 is the detail cluster situation of piece #63. Every time run the EML will have different results but all in all, the results seem promising.



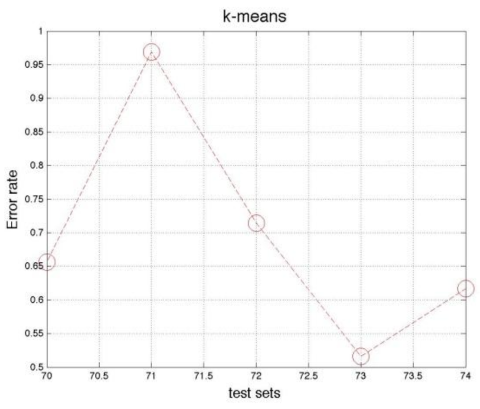
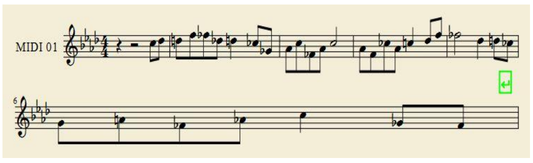


Figure 6. The Error rate of test file #70

Figure 7. Error rate of all the test data

We have parsed the MusicXML files and inputted the predicted note information. Then we loaded the MusicXML file into software "Muse Score" and get the result as below



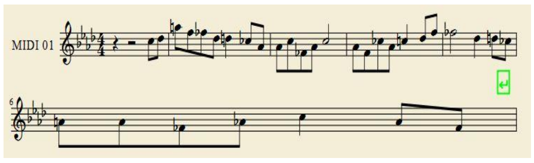
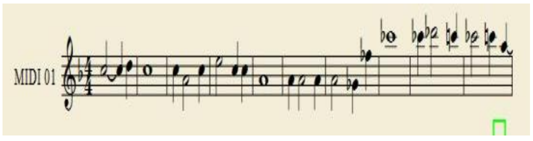


Figure 8.Original Score of #63

Figure 9. #63 predicted by ELM Model



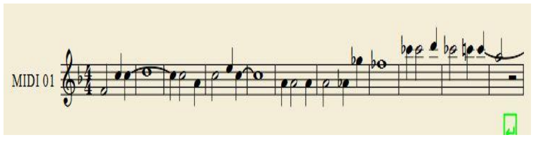


Figure 10.Original Score of #66

Figure 11. #66 predicted by ELM Model

5. Compare with baseline

There are 55 different notes in our training data, Therefore the accurate rate of random guess is 1.818%, our models work well on this problem. Apart from that, we have compared our result with commercial software 'IntelliScore [11]'. The result is: 10.9%

Reference

[1]. MusicXML: a digital sheet music interchange and distribution format

[2]. P. Smaragdis and J. C. Brown, “Non-negative matrix factorization for polyphonic music transcription,” in IEEE Workshop on Applications of Signal Process. Audio Acoustic, New Paltz, NY, pp. 177–180, 2003.

[3]. A Combined Mathematical Treatment for a Special Automatic Music Transcription System, Journal of Abstract and Applied Analysis, 2012

[4]. C. Raphael, “Automatic transcription of piano music,” in Proc. Int. Conf. Music Information Retrieval (ISMIR), Paris, France, 2002.

[5]. G. B. Huang, Q. Y. Zhu, and C. K. Siew, “Extreme learning machine: Theory and applications,” Neurocomputing, vol. 70, no. 1–3, pp. 489–501, Dec. 2006

[6]. G.-B. Huang, L. Chen, and C.-K. Siew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” IEEE Trans. Neural Netw., vol. 17, no. 4, pp. 879–892, Jul. 200

[7]. Spectral Audio Signal Processing, W3K Publishing, 2011

[8]. D. Ellis. Classifying music audio with timbral and chroma features. In International Symposium on Music Information Retrieval (ISMIR2007), pages 339–340, 2007

[9]. Bello, J.P.; Daudet, L.; Abdallah, S.; Duxbury, C.; Davies, M.; Sandler, M.B.; , "A Tutorial on Onset Detection in Music Signals," Speech and Audio Processing, IEEE Transactions on , vol.13, no.5, pp. 1035- 1047, Sept. 2005`

[10]. Piano note onset detection, from http://cnx.org/content/m14196/latest/

[11]. IntelliScore , Innovative Music Systems, Inc.

Code: (Readme)

1. In the " feature" file:

The main_harmonic and main_chroma is written by us.

(Because our wave files are so many and very large. we put one wav file 'fantase1.wav'here to show how it works)

The main_harmonic.m could extract harmonic feature from wav file.

The main_chroma.m could extract Chroma feature from wav file.

The variable "feature" is the feature we get

The chromagram.m is from

http://www.cs.northwestern.edu/~pardo/courses/old_courses/eecs352/mpm06/useful_files/chromagram.m

Others are from http://cnx.org/content/m14196/latest/

2. In the "Extreme learning machine" file:

dataMessage is written by us we used the value of error rate of each test data and index of test data to plot the figure of errorate of each test data.

file1: The file "ELM1" using the harmonic feature.

main.m is written by us.

ELM_Training.m is written by us (line 31-line 49 cited from Mathwork)

ELM_Testing.m is written by us

ELM_Training.m train the ELM model

ELM_Testing.m test the ELM model and show the error rate

The feature.mat is the harmonic feature data . It contains the data after removing non-note frames. It contains 162104 data. From 1 – 150000 is training data. From 150001:162104 is test data.

The state.mat is the labels of all the data. From 1-150000 is the label of training data. From 150001-162104 is the label of test data.

file2: The file “ELM1” using the Chroma feature.

main.m is written by us.

ELM_Training.m is written by us (line 31-line 49 cited from Mathwork)

ELM_Testing.m is written by us

ELM_Training.m train the ELM model

ELM_Testing.m test the ELM model and show the error rate

TrainsetX.mat contains Chroma feature data of training data, trainsetY.mat contains label of training data. f61-f74.mat contains Chroma feature data of test data. (We found that we can’t do time segmentation on some pieces, so we removed those pieces such as 65 68 69). s61-s74.mat are label of test data.

3. In the “Naïve Bayese” file:

bayes_main.m is written by us.

Run bayes_main.m will train the model, do prediction and count the error rate.

The feature.mat is the harmonic feature data . It contains the data after removing non-note frames. It contains 162104 data. From 1 – 150000 is training data. From 150001:162104 is test data.

The state.mat is the labels of all the data. From 1-150000 is the label of training data. From 150001-162104 is the label of test data.

4. In the “K Means” file:

test_120.m, train_120.m, x_kmeans.m are written by us.
train_120.m train the k-means model, and count the error rate of training set.
test_120.m test the model, compute the error rate
x_kmeans.m is the function for k-means clustering.
trainsetX.mat contain Chroma feature data of 2760 training data
trainsetY.mat contain label of all the training data.
f61-f74.mat contains Chroma feature data of test data (We found that we can’t do time segmentation on some pieces, so we removed those pieces such as 65 68 69)
s61-s74.mat contains label of test data.

5. In the “parseMusicXML” file: the code is Java

Createsongwithnewnote.java is written by us.

This code reads the note information from txt file which contain predicted note information; reads the original MusicXML; replace the note from original MusicXML file with the note that we predicted,

file1: The file “original note testdata” contains the original note information of test data

file2: The file”predicted note testdata” contains the predicted note information of test data

file3: The file “original MusicXML of testdata” contains the original MusicXML of test data

file4: The file “new MusicXML with predicted note” contains the new MusicXML with predicted note information of test data

Step1: In line 82 you need to put the txtfile in a certain directory. The textfile contain the note information. If you load a txtfile from file1, after these three steps, you will get the same original MusicXML, if you load a txtfile from file2, after these three steps, you will get the new MusicXML file with predicted note.

Step2: In line 136 line you put an original MusicXML file from file3 in certain directory. The MusicXML should correspond to the txtfile.

Step3: In line 773 you should set a certain directory to contain new MusicXML file that are generated by this code.

To see the score, you need to have software “MuseScore”which is really easy to be downloaded. Any music software that can read MusicXML is fine. `