AUTOMATIC GENERATION OF MUSICAL INSTRUMENT DETECTOR BY USING EVOLUTIONARY LEARNING METHOD

Yoshiyuki Kobayashi
SONY Corporation, Japan
Yoshiyuki.Kobayashi@jp.sony.com

ABSTRACT

This paper presents a novel way of generating information extractors that obtain high-level information from recorded music such as the presence of a certain musical instrument. Our information extractor is comprised of a feature set and a discrimination or regression formula. We introduce a scheme to generate the entire information extractor given only a large amount of labeled dataset. For example, data could be waveform, and label could be the presence of musical instruments in them. We propose a very flexible description of features that allows various kinds of data other than waveform. Our proposal also includes a modified evolutionary learning method to optimize the feature set. We applied our scheme to automatically generate musical instrument detectors for mixed-down music in stereo. The experiment showed that our scheme could find a suitable set of features for the objective and could generate good detectors.

1. INTRODUCTION

Musical information extraction technology has been extensively studied for various kinds of applications. Generally speaking, it extracts some features from input data, and then applies discriminant or regression analysis to estimate an objective variable from the features. There are some popular feature sets like MFCC (Mel-frequency cepstrum coefficient) [1] and features defined in Mpeg-7 standard [2], along with many other proposed features designed by heuristics. Popular discriminant analyses, which estimate objective variable from given feature set, include SVM, AdaBoost, GMM, HMM and so on. For example, Soo-Chang Pei et al. introduced instrumentation analysis and identification method with MFCC, Mpeg-7 features, and SVM [3]. T.Kitahara et al. introduced instrument identification method which can estimate the note-by-note presence probability of musical instruments by using linear discriminant analysis and some features other than MFCC or Mpeg-7 [4]. In these studies, feature sets are designed by human.

Meanwhile, there are some studies on Feature Generation [5]. Typically, a feature is obtained with a feature extractor composed of some basic functions. Genetic programming (GP) is used to design a feature that gives optimum objective variable. However, only a single feature could be designed, rather than an effective set of features for multivariate analysis. As a result the generated extractor is not accurate enough compared to popular methods with discriminant and multi-dimensional feature set designed by human. Also the description of feature is specialized to waveforms. As such, we could not apply this method to other kinds of data such as log-frequency spectrum.

It would appear that we can realize more accurate information extractor if we could automatically generate a set of effective features specialized for the objective. The work presented here is an approach to automatically generate an information extractor from dataset. The resulting extractor includes a set of effective features to estimate the objective variable. It also supports various types of data as input. First, we introduce the structure of the information extractor that our proposal generates. Next, the modified evolutionary learning method to optimize the feature set is presented. And finally as an application of this approach, we introduce our experiment of designing musical instrument detectors.

2. STRUCTURE OF INFORMATION EXTRACTOR

Figure 1 shows the structure of information extractor.

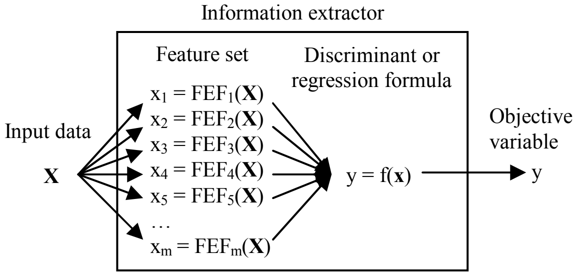

Information extractor
Feature set
Discriminant or regression formula
Input data
X
$x_1 = FEF_1(\mathbf{X})$
$x_2 = FEF_2(\mathbf{X})$
$x_3 = FEF_3(\mathbf{X})$
$x_4 = FEF_4(\mathbf{X})$
$x_5 = FEF_5(\mathbf{X})$
…
$x_m = FEF_m(\mathbf{X})$
$y = f(\mathbf{x})$
Objective variable
y

Figure 1. Structure of information extractor. **X** represents input data itself such as waveform. FEF represents a feature extraction function which extracts a single feature

from the input data. x_j represents the feature extracted by FEF_j, and **x** represents the feature vector consisting of x_j. f represents discriminant or regression formula which estimates the objective variable y based on the feature vector **x**.

First, the information extractor calculates multiple features from input data in accordance with the feature extraction functions (FEFs). The discriminant or regression formula estimates the objective variable from the extracted features. This structure itself is the same as the traditional information extractors. The difference is that our approach optimizes the entire information extractor, i.e. not only the discrimination or regression formula, but also the feature set.

2.1 Structure of input data

In our scheme, input data is expressed as a multi-dimensional matrix. For example, we can express stereo waveform as a two-dimensional matrix with channel and time dimensions (Figure 2). In this example, each element in two-dimensional matrix contains amplitude of the waveform in the channel at the time.

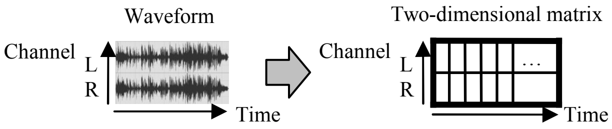


Figure 2. Example of input data of waveform.

Also we can express an image in RGB representation as a three-dimensional matrix with color, X, and Y axes (Figure 3). In this example, each element in three-dimensional matrix contains the brightness in RGB color space at the coordinate.

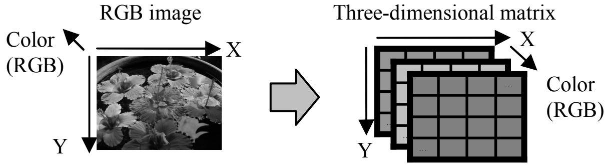


Figure 3. Example of input data of RGB image.

To express video data in this fashion, we would use four-dimensional matrix obtained just by adding one more dimension for time to the matrix for image. With this matrix based representation, we can flexibly handle various kinds of data as input data.

2.2 Description method for FEF

To support wide variety of input data and features, we propose a very flexible description of FEF. In our approach, FEF is formed as a cascade of basic functions (BFs) like a short computer program to reduce the input data matrix to a scalar. We prepared 51 BFs listed in Table 1.

Normalize
NormalizeAvg
NormalizeEach
NormalizeEachAvg
StandardizeEach
Abs
Sign
Add
Multiply
InverseSign
Sin
Cos
Tan
ASin
ACos
ATan
Differential

DiagonalDifferential
Integrate
LPF_1
HPF_1
DCCut
Order
Window_Hanning
Window_Gauss
MovingAverage
Extract1
Cut
LogAxis
LogAxisOctH
Mean
RMS
StdDev
MMean

MStdDev
MaxIndex
Max
Min
MaximumNum
MinimumNum
ZCR
TCR
ZCP
TCP
Difference
XDifference
Histogram_1D
Histogram01
Histogram_2D
Histogram2D01
DownSampling_To

Table1. List of basic functions.

The list includes four arithmetic operations, exponent functions, trigonometric functions, normalization algorithms, statistical functions, digital filters, etc. Figure 4 shows an example of FEF. And Figure 5 shows the calculation of the example FEF.

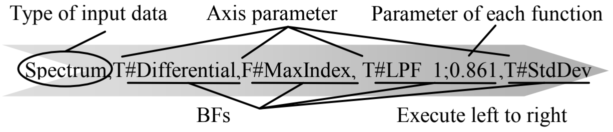
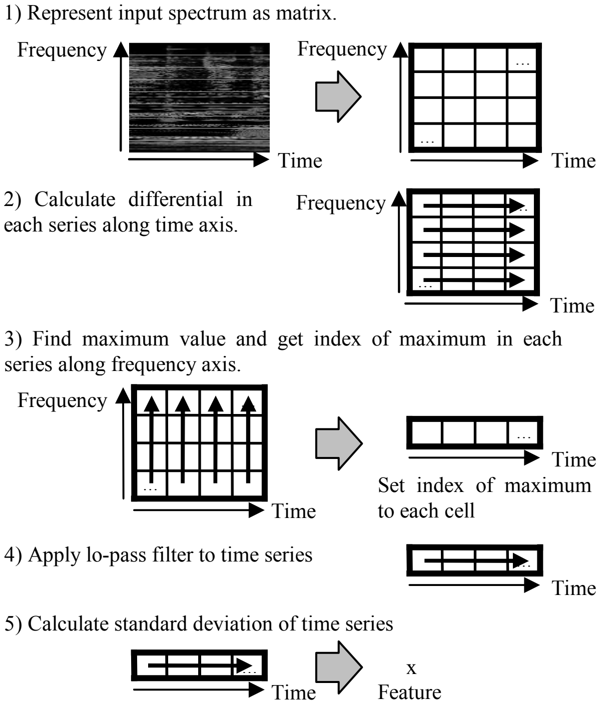

Type of input data
Axis parameter
Parameter of each function
Spectrum,T#Differential,F#MaxIndex, T#LPF_1;0.861,T#StdDev
BFs
Execute left to right

Figure 4. Example of FEF.


1) Represent input spectrum as matrix.

2) Calculate differential in each series along time axis.

3) Find maximum value and get index of maximum in each series along frequency axis.

Set index of maximum to each cell

4) Apply lo-pass filter to time series

5) Calculate standard deviation of time series

Figure 5. Calculation of the example FEF.

First, FEF represents the input spectrum as two-dimensional matrix with time and frequency axes, then it calculates differential along time axis, finds maximal value and gets the position of maximal value along frequency axis, applies lo-pass filter along time axis, and calculates standard deviation along time axis. With this formula, it extracts a single feature from input data of two-dimensional matrix. F and T before # represent frequency and time axes, and these are the axis parameters representing the axis along which the given matrix is processed. As Figure 4 shows, it executes several processes to the matrix of input data by following the FEF from left to right. The number of dimensions of the matrix was reduced in the course of processing, and eventually, a single value is extracted from input data. Some BFs have parameters. There are two kinds of parameter, one is axis parameter that represents which axis to process, and the other is the specific parameter for each BF such as the coefficient of lo-pass filter.

2.3 Discriminant or regression formula

We use linear discriminant or regression analysis with feature selection to estimate the objective variable from the feature set as below.

$$y = f(\mathbf{x}) = \Sigma_j b_j x_j + b_0 \qquad (1)$$

b_j represents linear combination coefficients, and b_0 represents intercept coefficient. We use linear procedure here because we can easily calculate contribution ratio which we later use to optimize the information extractor as a whole. Also it would appear that we can obtain a measure of accuracy without non-linear procedure because FEF can express various non-linear conversions.

3. MODIFIED EVOLUTIONARY LEARNING METHOD

Information extractor is optimized over training dataset which is a list of input data with label information. Table 2 shows an example of dataset. The label can be 0 or 1 for two-class discriminant analysis, or a numeric value for regression analysis.

Input data	1.wav	2.wav	3.wav	4.wav	5.wav	…
Vocal presence	0	0	1	0	1	

Table 2. Example of dataset to generate a vocal presence detector which accepts a segment of waveform and estimates the presence of vocal in the waveform. 0 signifies no vocal present in the waveform, and 1 signifies vocal present.

As previously described, each FEF in the information extractor has immense flexibility, so we used evolutionary learning method to search for a good feature set from the infinite set of possibilities. One generation of our evolutionary learning method executes the following steps.

1. Feature set generation
2. Feature extraction
3. Linear discriminant or regression analysis with feature selection
4. Calculation of contribution ratio of each feature

These steps are repeated until the learning is stopped by a user.

3.1 Feature set generation

In the first generation, the method synthesizes the feature set which is a list of fixed number of FEFs by combining BFs randomly. To generate the FEF, first, it chooses a BF randomly from the prepared BFs. If the chosen BF has parameters, they are set also randomly. Then this process is repeated to append more BFs until the matrix of input data is reduced to a single value by the FEF.

In the second and later generations, the method generates a new feature set based on the feature set from the previous generation by evolutionary learning process. It uses the contribution ratio of each feature calculated in the fourth step of the previous generation as the evaluation of that feature. Figure 6 shows the schematic of feature set generation in the second and later generations. First, it selects features in the order of contribution ratio and adds them to the feature set of next generation unmodified until cumulative contribution ratio becomes 99%. Next, it generates some features by randomly selecting from highly contributing features and mutating them by inserting, deleting BFs or modifying parameters. Finally, it generates remaining features randomly as done in the first generation. Figure 7 shows an example of the mutation of FEF.

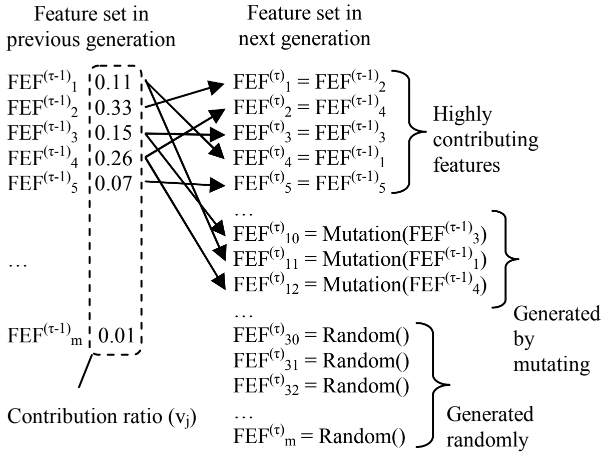

Feature set in previous generation
Feature set in next generation
$FEF^{(\tau-1)}_1$ 0.11
$FEF^{(\tau-1)}_2$ 0.33
$FEF^{(\tau-1)}_3$ 0.15
$FEF^{(\tau-1)}_4$ 0.26
$FEF^{(\tau-1)}_5$ 0.07
…
$FEF^{(\tau-1)}_m$ 0.01
Contribution ratio (v_j)
$FEF^{(\tau)}_1 = FEF^{(\tau-1)}_2$
$FEF^{(\tau)}_2 = FEF^{(\tau-1)}_4$
$FEF^{(\tau)}_3 = FEF^{(\tau-1)}_3$
$FEF^{(\tau)}_4 = FEF^{(\tau-1)}_1$
$FEF^{(\tau)}_5 = FEF^{(\tau-1)}_5$
Highly contributing features
…
$FEF^{(\tau)}_{10} = Mutation(FEF^{(\tau-1)}_3)$
$FEF^{(\tau)}_{11} = Mutation(FEF^{(\tau-1)}_1)$
$FEF^{(\tau)}_{12} = Mutation(FEF^{(\tau-1)}_4)$
Generated by mutating
…
$FEF^{(\tau)}_{30} = Random()$
$FEF^{(\tau)}_{31} = Random()$
$FEF^{(\tau)}_{32} = Random()$
…
$FEF^{(\tau)}_m = Random()$
Generated randomly

Figure 6. Example of feature set generation. τ represents generation in evolutionary learning process. Feature set in next generation contains highly contributing features in the previous generation, features generated by mutating the highly contributing features in the previous generation, and those randomly generated. All features in the first generation are generated randomly.

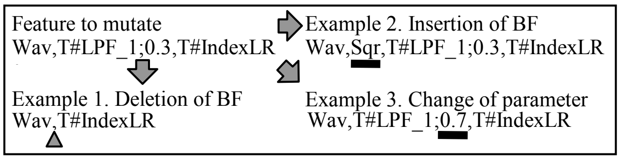

Feature to mutate
Wav,T#LPF_1;0.3,T#IndexLR
Example 2. Insertion of BF
Wav,Sqr,T#LPF_1;0.3,T#IndexLR
Example 1. Deletion of BF
Wav,T#IndexLR
Example 3. Change of parameter
Wav,T#LPF_1;0.7,T#IndexLR

Figure 7. Example of mutation of feature. A feature is mutated by inserting, deleting BFs or modifying parameters randomly.

3.2 Feature extraction

In this step, FEF$_j$ extracts feature $x^{(i)}_j$ from input data with index i. At this point, we have dataset with its features.

3.3 Linear discriminant or regression analysis with feature selection

In this step, the method estimates parameters of discriminant or regression formula (**b**) in equation 1 with the dataset and the features calculated in step 2. Because some features are generated randomly, there are many meaningless or redundant ones in the generated feature set, particularly in the first generation. Feature selection is very important in keeping only the effective features to realize maximum generalization accuracy. It is also important for the calculation of fair contribution ratio of features from discriminant or regression formula. For the feature selection, we used local-search to search for a good combination of features from information criteria perspective. More precisely, first, it prepares parameter $u_j = \{1, 0\}$ which indicates whether the j-th feature is selected or not, and sets all bits to 0 at the beginning. Then, it tries inverting a single bit among u_j's one by one starting from the first one, estimates parameters **b** with the currently selected features by using least squares method, and calculates AIC [6] by comparing the estimated objective variable and the label in the dataset.

$$\mathrm{AIC} = n * \log(\mathrm{PMSE}) + 2 * (k+1) \qquad (2)$$

n represents the number of the input data in the dataset, PMSE represents the prediction mean square error, and k represents the number of the features selected in **u**. Among the possible m bit inversion positions, the one at which the AIC improved the most is selected and executed, and the local-search is continued. In case of no improvement, it finishes the local-search with the selected features and the computed **b** as the optimum with respect to AIC.

3.4 Calculation of contribution ratio of each feature

Contribution ratio of each feature is calculated by the following formula.

$$v_j = b_j / \mathrm{StDev}(\mathbf{x}_j) * \mathrm{StDev}(\mathbf{t}) * \mathrm{Correl}(\mathbf{x}_j, \mathbf{t}) \qquad (3)$$

v_j represents the contribution ratio of the feature with index j. **t** represents objective variable which is the label in the dataset. StDev($\mathbf{x}_j$) represents the standard deviation of the feature with index j in the dataset. StDev(**t**) represents the standard deviation of the objective variable in dataset. And Correl($\mathbf{x}_j$, **t**) represents the coefficient of correlation between $\mathbf{x}_j$ and **t**. If x_j is not selected in step 3, v_j becomes zero. If there are multiple objectives, we can just use mean contribution ratio from each formula for each objective. With step 1, highly contributing features will survive and prosper, and poorly contributing features will die. With iteration of steps 1 through 4, the feature set will improve with respect to the objective compared to the previous generation. While traditional GP methods can optimize only a single feature, our approach can optimize multiple features simultaneously to achieve better generalization accuracy. Moreover because we use contribution ratio to select features, we maintain the variety of features in the later generations, which alleviates the local optimum problem.

4. APPLICATION TO MUSICAL INSTRUMENT DETECTION

We used our scheme to automatically generate musical instrument detectors for mixed sound.

4.1 Dataset

We prepared about 100 commercially available music files which are sampled at 44.1 kHz in stereo. They cover variety of genres such as pops, rock, jazz, world, and so on, and various kinds of musical instruments appear in these music files. We labeled each 1-second interval according to the presence of 10 kinds of musical instruments which are vocal, harmonize, piano, clean guitar, distortion guitar, distortion guitar solo, strings, brass, bass and drums with true (1), false (0) or unclear (no label). If there is audible sound of the instrument in an interval, we labeled it 1, otherwise 0, and if we feel it is very difficult to determine the presence of the musical instrument from only 1-second of waveform even for human ear, we put no label. We decided that it was not necessary to label the whole music file because there are repetitions in music, so there are about 40% of unlabeled sections. Finally, we got 21,272 segments of 1-second waveform in total. Table 3 shows the number of correctly labeled segments for each musical instrument.

	Vocal	Harmonize	Piano	Brass	Strings
TRUE	3505	1655	3184	946	1810
FALSE	7884	12455	8748	12083	10643

	C. guitar	D. guitar	D.G. solo	Bass	Drums
TRUE	2684	1706	354	5675	5062
FALSE	10185	14836	16208	5078	4414

Table 3. Number of segments of waveform with correct label information.

Segments contain 3.2/10 musical instruments on average and 7.5/10 musical instruments at maximum if we treat non-labeled instrument as 0.5. And we shuffled these segments without keeping reference to the songs from which they were taken. We used the half for training, and the other half for testing.

4.2 Input data

Our scheme can handle waveform directly. However, we found that we can achieve better accuracy by applying suitable pre-processing that emphasizes the characteristics of the input data for the objective. So, we converted the waveforms into three kinds of input data whose names are "12TonesM", "12TonesF" and "12TonesB". Each data is two-dimensional matrix with dimensions of time and musical pitch. The difference among these three data will be shown later. Original waveform is converted to these matrices with the following steps.

4.2.1 Simplified sound source separation

We applied simplified form of the sound source separation algorithm described in [7] to obtain foreground and background sounds from the original stereo sound. Figure 8 shows the signal flow diagram of this sound source separation.

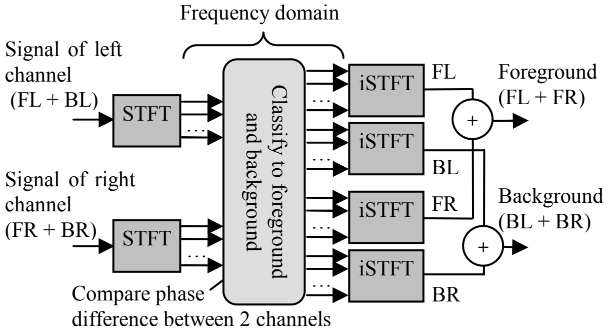

Frequency domain
Signal of left channel (FL + BL)
STFT
Signal of right channel (FR + BR)
STFT
Classify to foreground and background
Compare phase difference between 2 channels
iSTFT
iSTFT
iSTFT
iSTFT
FL
BL
FR
BR
+
+
Foreground (FL + FR)
Background (BL + BR)

Figure 8. Signal flow diagram of the simplified sound source separation. FL, BL, FR and BR represent foreground-left, background-left, foreground-right and background-right, respectively.

Each channel is analyzed with short-time Fourier transform with rectangle window of 16k samples and overlap of 8k samples. This very long frame size is needed to maintain the quality of separated sound. Then the phase difference between stereo channels in each frequency is calculated. If there is a difference greater than 0.2 PI, the frequency component is labeled as background. Otherwise, it is labeled as foreground. Then, for each channel, two waveforms for foreground and background are synthesized with inverse short-time Fourier transform with triangle window. This results in four channels of waveforms. Then, the left and right foreground channels are mixed, and the same is done for the background channels. As a result, two waveforms of foreground and background sounds are obtained. With this sound source separation, monaurally recorded sounds such as vocal, bass, snare and kick drums will appear in the foreground channel. On the other hand, sound recorded in stereo like strings or brass section will appear in the background channel.

4.2.2 Wavelet transform

We applied wavelet transformation to convert single waveform into two-dimensional matrix with time and musical pitch dimensions. We used band-pass filter which passes only a single semi-tone, as the mother wavelet. The original waveform was decomposed into 108 sub-bands corresponding to 12 semi-tones over 9 octaves. Then the logarithm of energy in each 7.8ms in each semi-tone is calculated. Figure 9 and Figure 10 show the schematic diagram and an example result of this process.

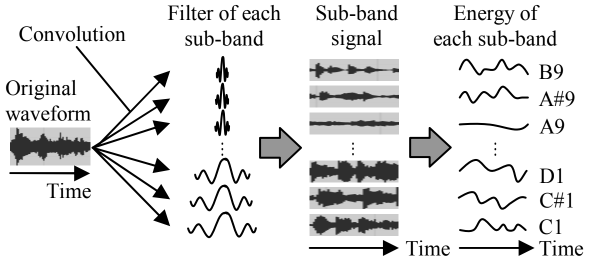


Figure 9. Schematic diagram of wavelet transform. It separates original waveform into 108 sub-bands, and calculates energy in 7.8ms in each band.

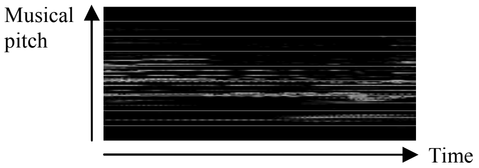


Figure 10. Example of result of wavelet transform. Brightness represents energy in each time and each musical-pitch.

We used the result of this process from foreground sound as "12TonesF", result from background sound as "12TonesB", and average of foreground and background as "12TonesM".

4.3 Result of learning

With our scheme and dataset, we generated musical instrument detection algorithms for mixed sound. Number of features is 1,000, and 165 generations were used in our evolutionary learning method. Figure 11 shows the learning curve. For comparison, it also shows the result for extractors with single feature. They are optimized with GP by selecting 3% of features most correlated with the label information in each generation.

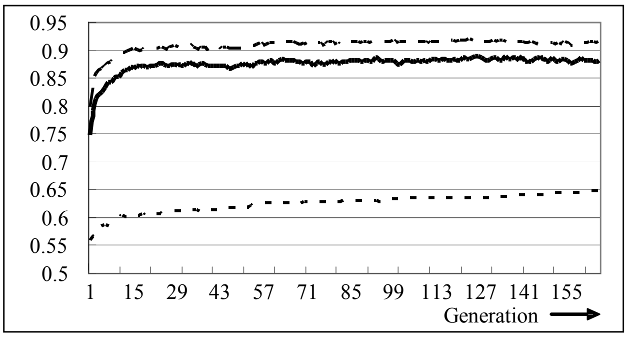


Figure 11. Learning curve. Dashed line represents the F-measure on training dataset averaged over all musical instrument detectors, and solid line represents the F-measure on testing dataset. Dotted line represents the F-measure of the detector with single feature optimized with GP on testing dataset.

As the learning curve shows, in the first generation, our detector realized average F-measures of 0.75 on testing dataset with features selected from 1,000 randomly generated features of various sorts. In the final generation, it realized 0.88 with the feature set optimized with our scheme. There is very clear advantage over the result of extractor with single feature optimized with GP. And Table 4 shows the F-measures for each musical instrument in the final generation on testing set.

Vocal	0.912	Brass	0.751	D. Guitar	0.956	Drums	0.991
Harmonize	0.852	Strings	0.794	D.G.Sol	0.762		
Piano	0.92	C. Guitar	0.897	Bass	0.987	Avg.	0.882

Table 4. F-measures of each musical instrument detector in the final generation on testing set.

```
12TonesF,T#Differential,T#StdDev,F#Window_Hanning,F#Mean
12TonesF,Add;0.223798,T#Differential,F#Window_Gauss;0.210
475;0.532910,T#StdDev,F#Mean
12TonesB,T#StdDev,ACos,F#Difference;0.633319
12TonesF,T#MaximumNum,F#Difference;0.689421
12TonesB,T#StdDev,F#Difference;-0.623785
12TonesB,T#Mean,F#Difference;-0.703600
```

Table 5. Part of highly-contributing features found in final generation.

Finally, table 5 shows some examples of generated FEF. The first feature in table 5 takes log-frequency spectrum of foreground as input, calculates differential in each series along time axis, calculates standard deviation in each series along time axis, processes Hanning window to frequency series and calculates average from frequency series. "Difference" function in table 5 splits the input in two at the boundary specified by the parameter, computes the sums for the two parts, and outputs the difference of the sums. It is not easy to understand what is going on in these generated features explicitly. However, it looks like it found variety of features, not only ones like MFCC and Mpeg-7 but also unique features with alien concept.

5. CONCLUSION

We presented a novel method to automatically design a information extractors. We introduced a very flexible description of features which supports various kinds of data types, and a modified evolutionary learning method to optimize multiple features given a partially labeled dataset. The method generated complete musical instrument detectors for mixed sound with various undiscovered and specialized features. The detectors realized either equal or superior performance compared to other methods even though the feature set is designed automatically given only the dataset without human intervention. Now we are applying the method to build various kinds of detection or recognition algorithms such as beat detection, attribute estimation, melody line estimation and more, not just for music recognition but for image recognition. We would like to report these results in the future.

ACKNOWLEDGEMENTS

The author would like to thank Francois Pachet, Pierre Roy, Gabriele Barbieri, Hideki Shimomura, Tatsuki Kashitani and Kei Tateno.

6. REFERENCES

[1] Beth Logan. Mel Frequency Cepstral Coefficients for music modelling. In International Symposium on Music Information Retrieval, 2000.

[2] H.G. Kim, N. Moreau and T. Sikora. MPEG-7 Audio and Beyond. Audio Content Indexing and Retrieval. John Wiley & Sons Ltd. 2005.

[3] Soo-Chang Pei, Nien-Teh Hsu. Instrumentation analysis and identification of polyphonic music using beat-synchronous feature integration and fuzzy clustering. Pages 169 – 172, ICASSP 2009.

[4] T.Kitahara, et al. Instrument Identification in Polyphonic Music: Feature Weighting to Minimize Influence of Sound Overlaps. EURASIP Journal on Advances in Signal Processing, Volume 2007.

[5] Pachet, F. and Roy, P. Analytical Features: A Knowledge-Based Approach to Audio Feature Generation. Eurasip Journal on Audio Speech and Music Processing, February 2009.

[6] H. Akaike. Information theory and an extension of the maximum likelihood principle. 2nd International Symposium on Information Theory, pages 267-281, 1973.

[7] Dae-young Jang, et al. Center channel separation based on spatial analysis. 11th International Conference on Digital Audio Effects, 2008.