Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP 2007)

Classification of Recorded Musical Instruments Sounds Based on Neural Networks

[1]Qian Ding and [2]Nian Zhang
South Dakota School of Mines and Technology
Department of Electrical and Computer Engineering
501 E. St. Joseph Street, Rapid City, SD 57701 USA
[1]dqian_daniel@hotmail.com, [2]Nian.Zhang@sdsmt.edu

ABSTRACT - **Neural networks have found profound success in the area of pattern recognition. The purpose of this paper is to classify automatically musical instrument sounds on the basis of a limited number of parameters. And this involves issues like feature extraction and development of classifier using the obtained features. As for feature extraction, a 5 second audio file stored in WAVE format is passed to a feature extraction function. The feature extraction function calculates more than 20 numerical features both in time-domain and frequency-domain that characterize the sample. Regarding the task of classification, we designed a two-layer Feed-Forward Neural Network (FFNN) using back-propagation training algorithm. The FFNN is trained in a supervised manner – the weights are adjusted based on training samples (input-output pairs) that guide the optimization procedure towards an optimum. After training, the neural network is validated by analyzing its response to unknown data in order to evaluate its generalization capabilities. Then, the sequential forward selection method is adopted to choose the best feature set to achieve high classification accuracy. Our goal is mainly to classify the sound into five different musical instrument families, such as the Strings, the Woodwinds and the Brass.**

Keywords: Artificial Neural Networks (ANN), Feature Extraction, Feed-Forward Neural Network (FFNN), Back-Propagation Training Algorithm, Mean-Square Error (MSE), Sequential Forward Selection (SFS)

1. INTRODUCTION

The collection of musical instrument sounds is an obligatory part of comprehensive music digital libraries. Automatic musical instrument classification can be very helpful for indexing the database as well as for annotation and transcription. In [2], four instruments, guitar, piano, marimba, and accordion, could be identified using an artificial neural network or nearest neighbor classifier. The results of this preliminary work achieved were encouraging although only temporal features were utilized. In [3], polyphonic music was separated into each monophonic one using comb filters (A comb filter adds a delayed version of a signal to itself, causing constructive and destructive interference. The frequency response of a comb filter consists of a series of regularly-spaced spikes, giving the appearance of a comb) and musical instruments were estimated by frequency analysis. More recently, a system for musical instrument recognition was presented that used a wide set of features to model the temporal and spectral characteristics of sounds [1].

In our work we aim at classifying the musical instrument sounds into five families, namely, brass, keyboard, percussion, string and woodwind. For the purpose of this study a small database containing musical sounds was constructed. For this project, we recorded the musical instrument sounds in standard audio format - WAV format. The WAV format for digital audio is simply the left and right stereo signal samples. It can be sampled at any rate. Typical sampling rates for WAVE files are 11025Hz, 22050Hz or 44100Hz. One important reason to use WAV file is that it is pretty handy to use MATLAB functions for processing WAV files of arbitrary size. In order to classify musical instruments properly several stages are needed: preprocessing, feature extraction (parameterization), and the actual classification process. Fig. 1 below is a block diagram of the classification system. The preprocessing stage consisted among others in pitch tracking procedures. A 5 second audio file stored in WAV format is passed to a feature extraction function. The feature extraction function calculates 23 numerical features that characterize the sample. When training the system, this feature extraction process is performed on many different input WAV files to create a matrix of column feature vectors. This matrix is then preprocessed to reduce the number of inputs to the neural network and then sent to the neural network for training. After training, single column vectors can be fed to the preprocessing block, which processes them in the same manner as the training vectors, and then classified by the neural network. A two-layer FFNN (Feed-Forward Neural Network) is used, which is trained via the back-propagation algorithm. FFNN is an artificial neural network where connections between the units do not form a directed cycle.

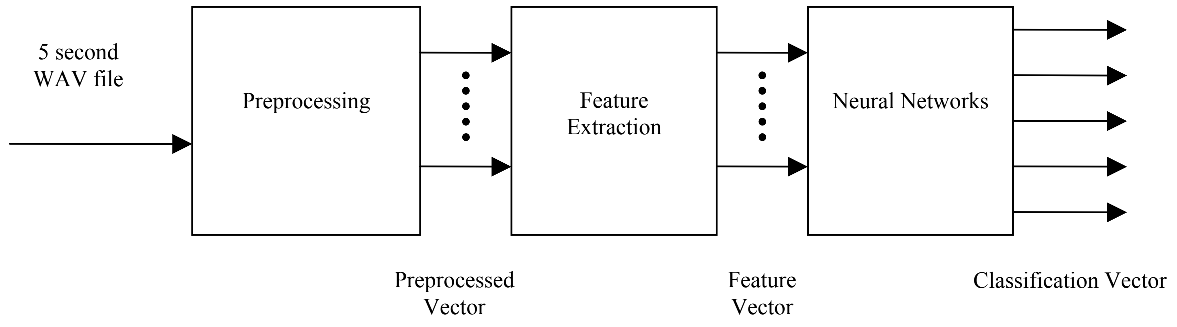

Fig. 1. Block diagram of the digital audio classification system

Due to a large number of audio features available, how to choose or combine them to achieve higher classification accuracy is studied in this paper. Simply choosing all the features available often doesn't yield the best performance, because some features give poor separability among different classes and some are highly correlated. These bad features have a negative effect when added into the feature vector. Therefore, a sequential forward selection method is adopted to select the so-called best feature set. Experiments of classifying the musical instruments into the right families have been conducted using Neural Network classifier based on the selected best features. The following sections will discuss each of these blocks in more detail.

2. THE DATABASE

The musical instruments are commonly sorted into five families according to their vibration nature, which are string, brass, percussion, woodwind, and keyboard. Currently, our small musical instrument database has 13 WAVE files. A brief description of the sound files is given in Table 1.

Table 1. The musical instrument collection

Families	Instruments
Brass	Trombone, Trumpet
Keyboard	Piano
Percussion	Crash Cymbal, Kick Drum, Ride Cymbal, Snare Drum
String	Bass, Cello, Violin
Woodwind	Alto Saxophone, Flute, Tenor Saxophone

3. PREPROCESSING

The data were preprocessed before going into the feature extraction block. The preprocessing is simply a kind of normalization by scaling the sampled sound file data to fall within the range of –1 to 1.

4. FEATURE EXTRACTION

Feature extraction is the process of computing a compact numerical representation that can be used to characterize a segment of audio. The design of descriptive features for a specific application is the main challenge in building pattern recognition systems. Once the features are extracted standard machine learning techniques which are independent of the specific application area can be used.

All the samples are 5 seconds long, monophonic and sampled in 44.1 KHz with 16 bit resolution. Every audio file is divided into frames of 256 samples, with 50% overlap at the two adjacent frames. Each frame is hamming-windowed and 23 features are extracted for each frame. We choose to extract the 23 features because they give a good representation of time and frequency characteristics of the musical sound. The 23 features from three categories are shown in Table 2. The features 1–3 are temporal features, 4–10 are spectral features, and 11–23 are coefficient features.

Table 2. Feature description

Feature Number	Descriptions
1	Energy
2	Zero-Crossing Rate
3	Periodicity
4	Amplitude
5	Bass
6	Spectral Centroid (Brightness)
7	Spectral Range (Bandwidth)
8	Spectral Rolloff
9	Spectral Flux
10	Pitch
11-23	First 13 Mel-frequency Cepstral Coefficients

4.1 TEMPORAL FEATURES

ENERGY

Energy is simply the sum of the amplitudes present in a frame, and is defined as:

$$\text{Energy} = \sum_{n}^{N-1} (x[n])^2$$

Where *x*[n] is the amplitude of the n^{th} sample.

ZERO-CROSSING RATE

This is the number of times the signal crosses zero amplitude during the frame, and can be used as a measure of the noisiness of the signal. It is defined as:

$$\text{ZeroCrossingRate} = \frac{1}{N}\sum_{n=1}^{N} | sign(x[n]) - sign(x[n-1]) |$$

Where *sign* = 1 for positive arguments and 0 for negative arguments [8].

PERIODICITY

The dominant periodicity of a signal is detected using a technique called Autocorrelation. The technique is to multiply the frame by a time-lagged copy of itself, then to measure the amplitude of the new signal. Where the amplitude reaches its peak will be where the peak(s) of the original signal are multiplied by the peak(s) of its copy, i.e. where the first period of the signal has been completed. The value of the time-lag where this peak occurs can then be considered the periodicity of the signal. The autocorrelation function is defined as:

$$\text{Autocorrelation (k)} = \sum_{t=1}^{N} x(t)x(t-k)$$

i.e. the signal *x*(*t*) multiplied by a time-lagged copy of itself x(t-*k*).

4.2 SPECTRAL FEATURES

AMPLITUDE

This feature relates to the loudness of the sound. The average amplitude of a frame is found by taking the square root of the sum of the squares of the first half of the magnitude spectrum [9]:

$$\text{Amplitude} = \sqrt{(\sum mag[i]^2)}\ , \quad i = 0\ldots \mathit{framesize/2}$$

Where *mag* = the magnitude spectrum.

BASS

This is the bassiness of the sound, i.e. how heavily the spectrum of a frame is weighted towards the lower frequencies [9]. One way in which this could be estimated is by applying a ramp function to the spectrum. A ramp function is simply linear from 1 to 0 across its length.

As with the amplitude, the square root of the sum of the squares is found having been normalized by the sum of the ramp function. The whole function is thus:

$$\text{Bass} = \sqrt{\left(\frac{(\sum ramp[i] \times mag[i]^2)}{\sum ramp[i]}\right)}, \quad i = 0\ldots\text{framesize/2}$$

Where *ramp* = ramp function from 1 to 0.

SPECTRAL CENTROID (BRIGHTNESS)

This is the amplitude-weighted average, or centroid, of the frequency spectrum, which can be related to a human perception of 'brightness' [9]. It is calculated by multiplying the value of each frequency by its magnitude in the spectrum, then taking the sum of all these. The value is then normalized by dividing it by the sum of all the magnitudes:

$$\text{Brightness} = \left(\frac{(\sum mag[i] \times freq[i])}{\sum mag[i]}\right), \quad i = 0\ldots\text{framesize/2}$$

Where: *mag* = the magnitude spectrum.
freq = the frequency corresponding to each magnitude element.

SPECTRAL RANGE (BANDWIDTH)

This is an amplitude weighted average of the differences between each frequency magnitude and the brightness [9], i.e. a representation of the range of frequencies that are present in a certain frame. We compute this in the same way as we would any range, by subtracting the mean value (in this case the brightness) from each data value:

$$\text{Bandwidth} = \left(\frac{(\sum mag[i] \times | (freq[i] - brightness) |)}{\sum mag[i]}\right), \quad i = 0\ldots\text{framesize/2}$$

Where; *mag* = the magnitude spectrum.
freq = the frequency corresponding to each magnitude element.

SPECTRAL ROLLOFF

This is a measure of spectral shape, which could be used instead of bandwidth [8]. It is defined as the frequency below which 85% of the magnitude distribution is concentrated. i.e.

$$\text{MIN(R) where } \sum_{i=1}^{R} mag[i] \geq 0.85 \times \sum_{i=1}^{N} mag[i]$$

Where *N* is the length of the signal.

SPECTRAL FLUX

This is a measure of the amount of local spectral change [8]. This is defined as the squared difference between the normalized magnitude spectra of successive frames.

$$\text{Flux} = \sum (norm_f[i] - norm_{f-1}[i])^2$$

Where $norm_f$ is the magnitude spectrum of the current frame scaled to the range 0...1, and $norm_{f-1}$ is the normalized magnitude spectrum of the previous frame.

PITCH

The perceived pitch of a sound is a very useful feature in musical content analysis, but estimating the pitch of a complex sound is a difficult problem, and one for which a number of methods have been proposed.

One method for pitch estimation [9] involves estimating the fundamental frequency of each frame of the signal. This is estimated as the best-fitting frequency whose harmonics match the spectrum's magnitude peaks. For each potential fundamental frequency we compute the first few harmonics, if these correspond to the other peaks in the spectrum we assume we have found our fundamental frequency. The fundamental frequency can then be used to calculate a pitch feature, converting to a MIDI note value by [8]:

$$n = 12\log_2 \frac{f}{440} + 69$$

4.3 MEL-FREQUENCY CEPSTRAL COEFFICIENTS (MFCCs)

Mel-Frequency Cepstral Coefficients (MFCCs) have been used very successfully in the field of speech recognition as classification features for speech audio signals. This is a measure of the perceived harmonic structure of the sound. Cepstral analysis provides a representation of spectral characteristics. A mel is a psychoacoustic unit of frequency which relates to human perception, the melscale can be approximated from a Hz value by the formula [9]:

$$\textit{Melfrequency} = 2595 \times \log_{10}\left(\frac{1+x}{700}\right)$$

Where x is the frequency in Hz.

The Mel-filter is a row of triangular windows overlapping at Mel-spaced intervals (Fig. 2). Applying this to a normalized (ranging from 0 to 1) spectrum results in a spectrum emphasized at Mel intervals. If we then compute the cepstrum of the filtered spectrum we can take some of the cepstral coefficients and use them as features in our system. Traditionally for speech processing the first 13 MFCCs are used. Cepstral Coefficients have been used for musical instrument recognition with a great deal of success [7].

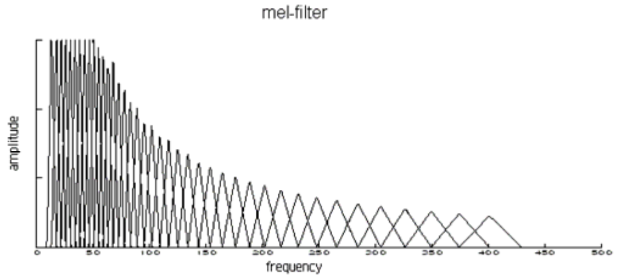

Fig. 2. The Mel-filter Bank

5. RESULTS OF AUTOMATIC CLASSIFICATION

Artificial Neural Networks (ANN) are computational models that try to emulate the behavior of the human brain. They are based on a set of simple processing elements, highly interconnected, and with a massive parallel structure. ANNs are characterized by their learning, adapting and generalization capabilities, which make them particularly suited for tasks such as function approximation.

Feed-Forward Neural Networks (FFNN) is a special class of ANNs, in which all the nodes in some layer l are connected to all the nodes in layer l-1. Each neuron receives information from all the nodes in the previous layer and sends information to all the nodes in the following layer. A FFNN is composed of the input layer, which receives data from the exterior environment, typically one hidden layer (though more layers may be used [10]) and the output layer, which sends data to the exterior environment.

The links connecting each pair of neurons are given some weight, w. This attribution of weights to links is the job of any training algorithm, as described below. Each neuron computes an output value based on the input values received the weights of the links from the neurons in the previous layer and the neuron's transfer function. Usually, sigmoid functions are used. The capability of the FFNN for mapping input values into output values depends on the link weights. Their optimal determination is still an open problem. Therefore, iterative hill-climbing algorithms are used. Their main limitation comes from the fact that only local optima are obtained: only occasionally the global optimum can be found. In the context of ANNs, these iterative optimization algorithms are called training algorithms.

The goal of the automatic classification experiments was to study a possibility of identifying selected classes of instruments by the neural network in order to verify the applicability of extracted sound parameters. A two-layer neural network of the *Feed-Forward* type was used in the experiments. The number of neurons in the input layer was equal to the number of elements of the feature vector. In turn, each neuron in the output layer was matched to a different class of the instrument and so their number was equal to the number of classes of instruments used in the experiment. In most such experiments the NN structure consisted in one hidden layer built of 18 neurons. This structure was first investigated in the pruning process. The pruning process was used for removing redundant parameters. The mechanism of searching redundant neurons or parameters consisted in observation of the *Mean-Square Error* (MSE). For example subsequent neurons were cut off up to the moment when the error increases significantly. Then the pruning process was interrupted and the last removed neuron was applied back to the structure and the optimization was finished. The EBP (error back-propagation) method based on the delta learning rule was used in the experiment [4]. EBP method is a training algorithm that uses backpropagation to calculate the gradient (This gradient is always used in a simple stochastic gradient descent algorithm to find weights that minimize the error) of the error of the network with respect to the network's modifiable weights. As the algorithm's name implies, the errors (and therefore the learning) propagate backwards from the output nodes to the inner nodes. In order to accelerate the convergence of the EBP training process, a momentum method is often applied by supplementing the current weight adjustment with a fraction of the most recent weight adjustment [4]. The momentum term (*MT*) in the *k*+1*th* iteration is expressed by the relationship:

$$MT^{k+1} = \alpha \cdot \Delta w^k$$

where:

- α - user-defined positive momentum constant, typically from the range 0.1 to 0.8
- Δw^k - increment of weights in the k^{th} step.

TRAINING PHASE

The training of the neural network was carried out several times using the EBP method. Each time different initial conditions were adopted as well as training parameters: the training process constant (η), that determines the rate of learning, and the momentum term (α) were changed dynamically in the course of the training. They were used later to evaluate the progress of the training process. Additionally the number of iterations was observed necessary to make the value of the 25 cumulative error dropping below the assumed threshold value. Such a scheme of training was adopted by the authors in previous studies and proved to be effective [5] [6]. The training of the network and its testing was carried out on the basis of the feature vectors described previously that were contained in databases. In experiments both feature vectors in time-domain and frequency-domain were used. Several configurations of musical instruments were selected. Each configuration contained five classes of musical instruments. They were as follows:

- Trombone, Piano, Snare Drum, Bass, Flute;
- Trumpet, Piano, Crash Cymbal, Violin, Alto Saxophone;
- Trombone, Piano, Kick Drum, Cello, Flute;
- Trumpet, Piano, Ride Cymbal, Violin, Tenor Saxophone;
- Trombone, Piano, Crash Cymbal, Cello, Flute;
- Trumpet, Piano, Snare Drum, Bass, Alto Saxophone.

To train the neural network, parameter vectors belonging to the corresponding five classes of musical instruments were used. The training proceeded up to the moment when the value of the cumulative error dropped below 0.01. This value was adopted arbitrarily in order to observe a possible case of network over-training. Several training processes were conducted for each musical instrument class configuration and for both types of the test set. The matrices of network weights were initiated at random, and unipolar activation function of neurons and training with the momentum method was applied (η = 0.05, and α = 0.45).

TESTING PHASE

To test the performance of the music classification system, the training data was divided further into two groups, one for training and one for validation. A validation data set was needed to ensure that the neural network did not overfit the data. Fig. 3 below shows the MSE versus training epoch plot – both the training data MSE and validation data MSE curves are shown. The MSE reached 0.0228 before a validation stop occurred.

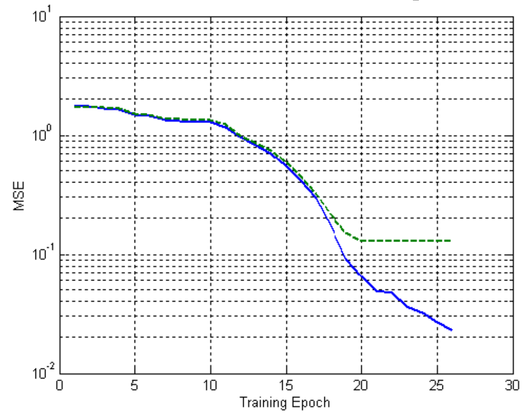

Fig. 3. MSE versus training epoch
(Training data – solid line, Validation data – dashed line)

6. FEATURE VECTOR SELECTION

The extracted features are normalized by their means and standard deviations. Then, a sequential forward selection (SFS) method is used to select the best feature subset of M features from the D originally available so that

1.we reduce the dimensionality of the feature vector (*M* < *D*)

2.we optimize the desired class separability criterion

Firstly, the best single feature is selected based on classification accuracy it can provide. Next, a new feature, in combination with the already selected features, is added in from the rest of features to minimize the classification error rate. This process proceeds until all the features are selected. The SFS method can quickly provide a suboptimized set of features in comparison with the exhaustive searching approach which is not practical due to exorbitant computation time involved in the concerned applications.

Experiments using single temporal, spectral, coefficient feature or any combination of them have the similar phenomenon. The classification accuracies of the best feature set and the corresponding feature numbers are listed in the Table 3. The best feature sets for different classifiers are different. Among all the experiments, the classifier using 18 features, in which 2 are temporal, 6 are spectral, and 10 are coefficients, achieves the highest accuracy of 93%, which is much better than 73%. Although the results obtained are not sufficient for real-world applications, they are promising.

Table 3 Classification performance

Features	**Accuracy (Total Number of Features)**
Time	73% (3)
Frequency	80% (7)
Coefficient	85% (13)
Time and Coefficient	90% (14)
Frequency and Coefficient	91% (16)
Time and Frequency	83% (10)
Time, Frequency and Coefficient	93% (18)

CONCLUSIONS

In this paper, we use a sequential forward feature selection scheme to pick up the best feature set in single or any combination of temporal, spectral, and coefficient space for classifying musical instruments into five families. Simple classifier using small set of features can achieve a satisfactory result. Since the number of features is reduced, less computation time is required for classifying the music instrument sounds. This will be beneficial to real-time applications such as sound retrieval from large databases.

A system for musical instrument recognition was presented that uses a limited number of features to model the temporal and spectral characteristics of sounds. Signal processing algorithms were designed to measure these features in acoustic signals. Using this input data, a classifier was constructed and the usefulness of the features was verified. Furthermore, experiments were carried out to investigate the potential advantage of a hierarchically structured classifier. The achieved performance and comparison to earlier results demonstrates that combining the different types of features succeeded in capturing some extra knowledge about the instrument properties. Hierarchical structure could not bring further benefits, but its full potential should be reconsidered when a wider data set including more instruments, as well as different examples from a particular instrument class is available. Future work will concentrate on these areas, and on integrating the recognizer into a system that is able to process more complex sound mixtures.

ACKNOWLEDGEMENTS

The authors want to thank Jason Ward for recording the musical instrument sound files to construct the database.

REFERENCES

[1] Eronen, A.; Klapuri, A. Musical instrument recognition using cepstral coefficients and temporal features. IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2, pp. 63-71, 2000.

[2] Kaminsky, I.; Materka, A. Automatic source identification of monophonic musical instrument sounds. IEEE International Conference on Neural Networks, Vol. 1, pp. 189-194, 1995.

[3] Miiva, T.; Tadokoro, Y. Musical pitch estimation and discrimination of musical instruments using comb filters for transcription. 42nd Midwest Symposium on Circuits and Systems, Vol. 1, pp. 105-108, 1999.

[4] Zurada J., "Introduction to Artificial Neural Systems", West Publishing Comp., St. Paul 1992.

[5] Kostek B., "Soft Computing in Acoustics, Applications of Neural Networks, Fuzzy Logic and Rough Sets to Musical Acoustics", Studies in Fuzziness and Soft Computing, Physica Verlag, Heilderberg, New York 1999.

[6] Kostek B., CZYZEWSKI A., "Automatic Classification of Musical Sounds, " *108th Audio Eng. Soc. Conv*., Preprint No. 2198, Paris, Feb. 19-22, 2000.

[7] Brown, J.C. (1999), Computer Identification of Musical Instruments Using Pattern Recognition with Cepstral Coefficients as Features, Journal of the Acoustical Society of America 105(3), March 1999.

[8] Tzanetakis, G. and Cook, P. (2002), Musical Genre Classification of Audio Signals, IEEE Transactions on Speech and Audio Processing, Vol. 10, No. 5, July 2002.

[9] Blum, T., Keislar, D.F., Wheaton, J.A., and Wold, E.H. (1999), Method and Article of Manufacture for Content-Based Analysis, Storage, Retrieval, and Segmentation of Audio Information, US Patent no. 5,918,223.

[10] Sarle W. (maintainer), “Neural Nets FAQ”, ftp://ftp.sas.com/pub/neural/FAQ3.html, 2001