Tempo and Beat Tracking for Audio Signals with Music Genre Classification*

Mao-Yuan Kao, Shiue-Hung Shiau and Chang-Biau Yang†
Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung, Taiwan
†cbyang@cse.nsysu.edu.tw

Abstract

Most people follow the music to hum or follow the rhythm to tap sometimes. We may get different meanings of a music style if it is explained or felt by different people. Therefore we cannot obtain a very explicit answer if we can not have the notation of the music. We need some techniques and methods to analyze the music that we listen to, and obtain some information that we want. Tempo and beats are very important elements in the perceptual music. Therefore, tempo estimation and beat tracking are fundamental techniques in automatic audio processing, which are crucial to multimedia applications. We first develop an artificial neural network to classify the music excerpts into the evaluation preference. And then, with the preference classification, we can obtain accurate estimation for tempo and beats, by either Ellis's method or Dixon's method. We test our method with mixed data set which contains ten music genres from the "ballroom dancer" database. Our experimental results show that the accuracy of our method is higher than only one individual Ellis's method or Dixon's method.

Key words: tempo, beat, audio processing, neural network, classify.

1 Introduction

Tempo is one of the basic music elements. It indicates the performance speed of the music. We have to distinguish the *notation tempo* from the *perceptual tempo*. The notation tempo is a mark which is on the top of the general music staff. The perceptual tempo is the musical feeling tempo that is felt by the listener listening to the music. The listeners (even all of them are expert musicians) cannot come up with a common answer for excerpting the music annotation tempo. They will give you different answers if they are not familiar with the piece. They could tap in different metrical levels because the feelings of music melody for people may be different. Thus people may have different annotation tempos with an identical music.

A *beat* is a pulse which is a base unit for the sense of hearing. Thus a beat is a base time unit for a piece of music. It specifies every tick of the metronome, and each tick is a beat. A beat can be regarded as a pulse wave or an event occurrence in the audio signal.

Tempo [12] is the speed of playing a piece of music. It is usually represented by *beats per minute* (BPM). In brief, we calculate it by counting the number of beats for playing the music per minute.

Since tempo and beat are very important elements in the perceptual music, tempo estimation and beat track are fundamental techniques for the automatic audio processing. The research in tempo track in early years, the file structure with midi or other representative form is emphasized. Recently, researchers process CD audio recordings directly, so the *WAV files* or *Audio MPEG-1 Layer 3 files* (MP3) become more popular. The approaches for tracking tempo and beats are usually to analyze audio signal with time frequency [1, 13, 10] or subband [17, 15].

Our purpose is detecting the tempo and beat locations of the audio signal. Therefore we first develop an artificial neural network to classify the music excerpts into the evaluation preference. And then, with the preference classification, we can obtain accurate estimation for tempo and beats, by either Ellis's method or Dixon's method.

The organization of this paper is as follows: In Section 2, we will introduce the basic concept of the automatic audio process. In Section 3, we will introduce Ellis's [5] and Dixon's [4] methods. In Section 4, we will design a neural network to classify the evaluation preference of an input music excerpt. Then, either Ellis's method [5] or Dixon's [4] method is invoked for calculating tempo and beat tracking. In Section 5, our experimental results will be shown and some discussions will be given. At last, in Section 6,

*This research work was partially supported by the National Science Council of Taiwan under 95-2745-H-309-003-HPU.

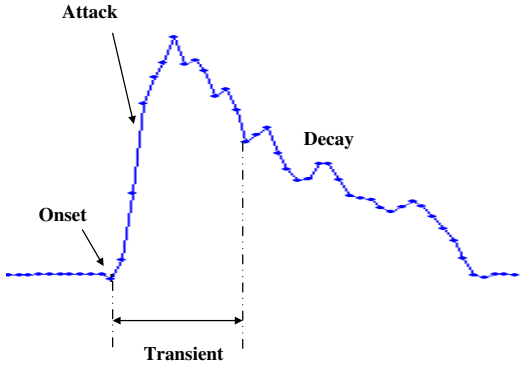

Figure 1: Four elements of a signal note: onset, attack, transient, and decay.

we will give the conclusion of this paper.

2 Preliminary

2.1 Onset Detection

The onset detection is a basic procedure and also the most important part for the automatic audio processing. So we will give a brief introduction for onset detection in this section. An ideal sound wave signal is constituted by four parts: *onset, attack, transient*, and *decay* which were presented by Bello *et al.* [2]

The onset detection is useful for analysis technology and can be applied to process the audio signal. It is usually used to look for transient regions on audio signal. For the realistic situation, getting an ideal audio wave is very difficult. In a real music, the signal is polyphonic, which means that many sound objects occur simultaneously in one time interval. In general, it is unable to detect the onset location directly with quantitative time-varying in the transient region. Therefore, the onset positions found by different algorithms may not be the same.

2.2 Construction of Automatic Audio Processing

The structures in most of automatic audio processing are very similar with each other. In 2004, Alonso *et al.* [1] presented the construction method for the automatic audio processing as follows.

Input: A audio signal (waveform).

Output: The tempo and the beat locations of the audio signal.

Step 1. Transform the input signal from time domain to frequency domain.

Step 2. Extract the significant events from music (onset detection).

Step 3. Estimate the periodicity.

Step 4. Detect the beat locations and marker.

In Step 1, we transform the input signal from time domain to frequency domain. The transformation can usually be done by using *Fourier transform* [3]. Its mathematical expression is given as follows.

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt.$$

Here, we use *short time Fourier transform* (STFT) [3] to obtain the relationship between the time and frequency domains. We can obtain the corresponding relation between position t and frequency f in the specific range with STFT. Its mathematical expression is given as follows.

$$STFT\{x()\} \equiv X(\tau, f) = \int_{-\infty}^{\infty} x(t)\omega(t-\tau)e^{-j2\pi ft}dt.$$

Step 2 is to extract the significant music event (onset detection). Using various kinds of onset detection algorithms, we can obtain the signal wave. And then by using the peak-picking algorithm [6], we can obtain all onset locations from the input signal.

In Step 3, period estimation is performed. In other words, it is to calculate the periodicity from the onset location. *Spectral product* [1] and *autocorrelation function* (ACF) [1, 7] are two methods for this purpose. The result calculated by ACF is better than the former. ACF is used frequently in signal processing. It is a signal multiplied by itself with time-shifted. In a more precise view, it can be regarded as a cross correlation of a signal with itself. ACF can help us to look for repeated patterns in a signal. Its mathematical expression is given as follows.

$$\Re(\tau) = \int_{-\infty}^{\infty} x(t)x(t-\tau)dt.$$

Step 4 is to detect and mark the beat locations. After we obtain the information of the onset locations and tempo, and we can make a beat tracking with the information. At certain time, the beat tracking algorithm will distinguish whether there is a beat at the time or not, and it will record its time information by which the beat location in an audio signal can be obtained. Finally, we can obtain the tempo and beat locations in the audio signal by the automatic audio processing.

3 Related Algorithms

3.1 Ellis's Method for Beat Tracking

Ellis [5] presented a beat tracking system with dynamic programming [11].

The first stage of the system processing is to detect the onset locations of the audio signal. The original signal is down-sample to 8 kHz and it is also converted to mono. In order to keep the onset information, the system uses the half-wave rectifier to process each frequency channel, then it sums all information of the frequency channels. The system finally uses the high-pass filter to remove d.c. offset.

The second stage of the system processing is a tempo estimation through autocorrelation function (ACF). In Ellis's method [5], Gaussian window with a log-time axis is used to capture the important information. It has the property that the largest position is the tempo (bpm) of the audio signal. Then we can obtain the global tempo, which is an important information for beat tracking processing.

The beat tracking system attempts to find out a sequence of the beat times that correspond to the large values in the onset locations of the audio signal. The best cumulative scores are found. They showed that the beat sequence consists of the ends of all possible time samples. The beat tracking system searches a range from 0.5 to 2 beat periods for each time point. And a dynamic programming approach is applied to the searching process. The best predecessor beats at the current time is obtained by choosing the largest value from the range. The value is the current onset signal value added to the best cumulative score. The time point is also stored. After the end of the audio signal, we can choose the best cumulative score, and then trace back to obtain the entire sequence of beats through all the time records.

3.2 Dixon's Method: Beatroot

Beatroot [4], proposed by Dixon, is a system using multiple agents' architecture. A clustering algorithm looks for the most meaningful metrical units. Based on the way, we can use the multiple agents' architecture to match the sequence of beats with the audio signal, where each agent represents a particular tempo and a sequence of beats with the audio signal.

The onset detector is to find out the peaks in the spectral flux. The spectral flux sums the magnitude changes for each frequency bin. $X(n,k)$ expresses the kth frequency bin of the nth frame, given as follows.

$$X(n,k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(hn+m)w(m)e^{-\frac{2j\pi mk}{N}},$$

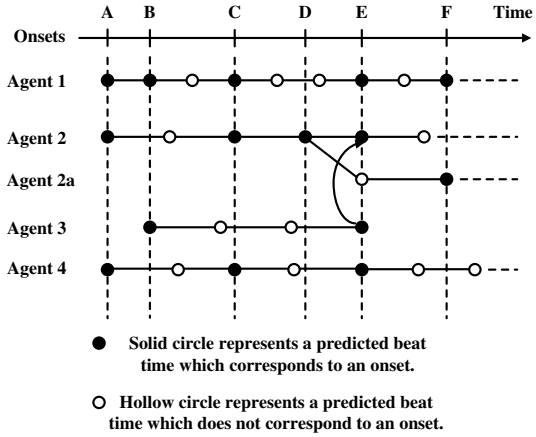

Figure 2: Beat tracking subsystem by multiple agents.

where $w(m)$ is a hamming windows. The spectral flux function (SF) is given as follows.

$$SF(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - |X(n-1,k)|).$$

With the onset information, the tempo estimation stage of the system will find out a suitable beat period. The tempo estimation is calculated by the inter-onset intervals (IOIs). The IOI is defined as an interval between any two onsets which are not necessarily consecutive. The tempo estimation stage of the system uses the clustering algorithm to find out the cluster of IOIs which represents many distinct musical units. With the information, the system combines the information between the clusters on the next stage and passes the combined information to the beat tracking subsystem.

The beat tracking subsystem uses the multiple agents' architecture to find out a sequence of the beat locations which matches the varied tempo (IOI). Each agent has a tempo which is from the tempo estimation subsystem and it also has an onset time at the beginning. Each agent predicts future beats according to the given tempo and the first onset time. Each agent uses a tolerance window to find out the beat times. If an onset falls in the inner window of the predicting beat times, this onset is the real beat time. If it falls in the outer window, it is the probable beat time.

Figure 2 illustrates the operation of the beat tracking subsystem with multiple agents, in which the time line with six onsets (A to F) are shown. The beat tracking of each agent uses a tempo and first onset time. In the figure, each solid circle represents a predicted beat time which corresponds to an onset. And each hollow circle represents a predicted

Table 1: The accuracy of tempo and beat prediction on ten music genres with Ellis' and Dixon's methods within 1% error tolerance.

Music Genre	Ellis	Dixon	Both	Number
ChaChaCha	78.38%	77.48%	*75.68%*	111
Jive	55%	*83.33%*	53.33%	60
QucikStep	*62.2%*	8.54%	7.32%	82
Rumba American	0%	28.57%	0%	7
Rumba Internation	*82.35%*	50.98%	49.02%	51
Rumba Music	*67.5%*	22.5%	22.5%	40
Samba	52.33%	47.67%	27.91%	86
Tango	44.19%	*72.09%*	40.7%	86
Viennese Waltz	35.38%	*61.38%*	30.77%	65
Waltz	17.27%	8.18%	2.73%	110
All Data	52.29%	47.56%	34.1%	698

beat time which does not correspond to an onset.

4 Our Method

Combining with Ellis's and Dixon's methods, we build a neural network model which can be used to detect the tempo and beat locations of the audio signal. The input of the network is the audio signal, and the output is its tempo and beat locations. Because the goal is to establish a mode of neural network, we need design the input-output pairs of the neural network. The input is the features of music genre, which can be served as the criteria of the evaluation preference classification. A binary bit will be output. It is the answer of the preference classification, and indicates that either Ellis's method or Dixon's method is the preferred method used for detecting the tempo and beat locations. If Ellis's method (Dixon's method) is the preferred one, then we think Ellis's method (Dixon's method) is an effective method.

Table 1 shows that the prediction accuracies of the two methods are various in different music genres. The fields of "Ellis" and "Dixon" indicate the accuracies of tempo prediction when Ellis's method and Dixon's method are applied, respectively. The field of "Both" illustrates the accuracies of both Ellis's and Dixon's methods, which means the predictions with both methods are correct. Therefore, the classification of the music genre may be a good clue for us to decide which prediction method should be applied. We use the public "ballroom dancer" music database as our training and testing data set.

4.1 Feature Extraction

These six kinds of features are the inputs of our classification neural network.

1. *Linear predictive coefficients* (LPC) [16]: The basic principle of LPC is that the current signal value can be represented with the linear combination of the former signal values. This assumption is reasonable because most audio signals have the periodicity property. The mathematical expression of LPC is given as follows.

$$x_{n+1} = w_0 * x_n + w_1 * x_{n-1} + \cdots + w_{p-1} * x_{n-p+1} + e_{n+1},$$

where e_{n+1} is the prediction error.

2. *Discrete Fourier transform* (DFT) [7]: We first get the information of the audio signal on time domain, and then we convert it into the frequency domain with fast Fourier transform (spectrum). Finally, we sum up the data obtained in various down sampling schemes. This method could enhance the data of the low frequency.

3. *Harmonic product spectrum* (HPS) [7]: Harmonic product spectrum is a method which detects the pitch in the frequency domain. HPS first converts the data into the frequency domain with fast Fourier transform. Then, the spectrum is down sampled continuously. Finally, HPS is the sum of all data in all downsampling steps.

4. *Cepstrum* [7]: We obtain the data of the signal calculated by the cepstrum method. The original definition of cepstrum is the *Fourier transform* (FT) of the *decibel spectrum*. Another definition is the *inverse Fourier transform* (IFT) of log of the decibel spectrum. The latter definition is more commonly used.

5. *Mel-scale frequency cepstral coefficients* (MFCC) [7]: This parameter considers human perception sensitivity in terms of different frequency. Therefore, it is usually used on speech recognition. The process of MFCC is given as follows:

Step 1: Get the spectrum of the signal (the data of frequency domain).

Step 2: Get the log energy of each triangular band-pass filter through the magnitude frequency response multiplied by a set of N triangular band-pass filters. In this paper, we set $N = 20$ which is the number of triangular band-pass filter.

Step 3: Take P mel-scale cepstral coefficients that applied *discrete cosine transform* (DCT) on log energy from Step 2. In this paper, we set $L = 12$ which is the number of mel-scale cepstral coefficients. The MFCC function was proposed by [7].

6. *Volume* [7]: Volume is the strength of sound. The volume can be calculated by the amplitude of signals in each frame. There are basically two methods for calculation:

$$Volume = \sum_{i=1}^{n} |x_i|,$$

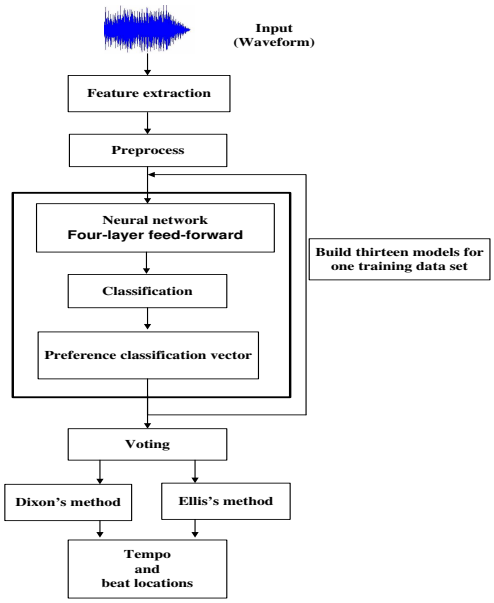

Figure 3: The flow chart of our prediction method.

$$Volume = 10 * log_{10}(\sum_{i=1}^{n} x_i^2),$$

where x_i is a sample of frame and n is a frame size.

4.2 Framework of Our Method

The flow chart of our prediction method is shown in Figure 3.

In Step 1, we extract the meaningful feature vectors of the audio signal to represent the audio signal. In Section 4.1, we have already introduced the feature extraction method. Thus, the feature vector is mainly described here.

We first divide an audio signal into N frames in the feature extraction processing. The size of each frame is 512 samples and the overlapping size of two neighboring frames is 128 samples. Each frame forms an operation unit in the feature extraction processing. We extract 32 LPC values from each frame independently. We form the ith group of LPC values by collecting the ith values of the 32 LPC values of all frames together. Thus, 32 groups are obtained. Then, in each group, we calculate its mean, sum, maximum, minimum and median values. Now, each group is represented by these five values. In order to extract the relationship among all groups, we first calculate the N differences between the corresponding pairs of two groups i and $i+1$, where $1 \leq i \leq 31$. Then the maximum of the N differences between groups i and $i+1$ is calculated as the difference of these two groups. Thus, we get 31 difference values between every two neighboring groups. In addition, we extract 32 LPC values from the whole audio signal. Counting the above values, we have 32*5+31+32=223 values of the linear predictive coefficients.

We also extract 60 values containing 50 low frequency data and 10 high frequency data in the DFT step, 64 values containing 32 low frequency data and 32 high frequency data in the HPS step, 60 values containing 10 low frequency data and 50 high frequency data in the Cepstrum step and 12 MFCC values from each frame. We also group the above values, and then use the five values (mean, sum, maximum, minimum and median) to represent each group. The maximum value among the N difference values between every two adjacent groups are also calculated. Thus we have the number of the discrete Fourier transform amplitude values is 60*5+59=359, the number of the harmonic product spectrum coefficients is 64*5+63=383, the number of the cepstrum coefficients is 60*5+59=359 and the number of the mel-scale frequency cepstral coefficients is 12*5+11=71. We also extract 2 volume values for each frame. Since we use two methods to calculate the maximum value, the number of the volume coefficients is 2*5+2=12. Summing the above numbers, 223+359+383+359+71+12, we have 1407 which is the total number of features in the six types extracted in Step 1.

The total number, 1407, is still huge for a neural network. Therefore, in Step 2, we have to reduce the *dimension* of the feature vector, where a dimension represents a feature. There are 698 music excerpts in the "ballroom dancer" music database. We divide the data into three disjoint sets, one for training, another for validation, and the other for testing. Then, we normalize the training and validation data sets by regulating their average values and the standard deviations. The normalization procedure is given as follows. The *meanp* is the average value of the input vector:

$$meanp = \bar{p} = \frac{1}{n}\sum_{i=1}^{n} p_i,$$

The *stdp* is the standard deviation of the input vector:

$$stdp = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - \bar{p})^2},$$

Then, the input vector can be normalized as follows:

$$\acute{p} = (p - meanp)/stdp,$$

where p is the input vector and $\acute{p}$ is the normalized result of p. The *meant* is the average value of the traget vector:

$$meant = \bar{t} = \frac{1}{n}\sum_{i=1}^{n} t_i.$$

The *stdt* is the standard deviation of the traget vector:

$$stdt = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(t_i - \bar{t})^2},$$

Then, the target vector can be normalized as follows:

$$\acute{t} = (t - meant)/stdt,$$

where t is the target vector and $\acute{t}$ is the normalized result of t.

We can also normalize the testing data set with *meanp*, *stdp*, *meant*, and *stdt* similarly. Then, we apply *principal components analysis* (PCA) [9] to the training and validation data sets for reducing their dimensions to lower dimensions. The principal components can be calculated as follows. First, the covariance matrix of input vector p is calculated by

$$S = \frac{1}{n-1}\sum_{i=1}^{n}(\acute{p}_i - \bar{\acute{p}})(\acute{p}_i - \bar{\acute{p}})^T.$$

Then, the eigenvectors U and eigenvalues V of the covariance matrix can be obtained by

$$S = UVU^T.$$

Finally, we get the principal components as follows.

$$c = U^T\acute{p},$$

where $\acute{p}$ is the normalized result of the input vector p. We use the principal components whose accumulated contribution exceeds 99.9% of the variation in the training data set to build the neural network model. Table 2 shows the dimension in each feature type remained after PCA. Matrix U^T is also used to get the principle components of the testing data set.

In Step 3, we use a four-layer *feed-forward back-propagation* neural network [8], as shown in Figure 4. The numbers of neurons in the input, second, third and output layers are 28, 19, 10 and 2, respectively. Before the endings of the input and second layers, *tangent sigmoid transfer function* [8] is used to convert the calculated element to fall within the range from -1 to 1.

The third layer applies the *log sigmoid transfer function* [8] to convert the result to fall within the range from 0 to 1. No transfer is done in the output layer. The *training function* we use is the *Levenberg-Marquardt back-propagation algorithm* [14, 8].

The *performance function* we use is the *mean squared error with regularization performance function* [8]. The calculation of mean squared error is given as follows:

$$\gamma\frac{1}{n}\sum_{i=1}^{n}(e_i)^2 + (1-\gamma)\frac{1}{n}\sum_{j=1}^{n}(w_j)^2,$$

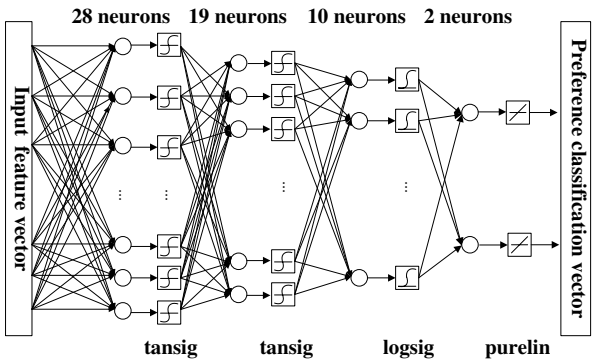

Figure 4: A four-layer neural network.

where γ is used to weight the two terms.

The *adaptive learning function* we use is the *gradient descent with momentum weight and bias learning function* [8].

In Step 4, we use two binary bits to represent the classification result predicted by the neural network.'0 1' means that Ellis's method is the better one for predicting tempo and beat locations for the input music signal, and '1 0' means that Dixon's method is the better one. Since each neural network model is not very reliable, to increase the classification accuracy, we build 13 neural network models. Then the classification result depends on the vote of the 13 models, with the majority rule. Thus, Step 5 performs the voting procedure. According to the voting result, Step 6 invokes either Ellis's method or Dixon's method to predict the tempo and beat locations of the input audio signal. Finally, our method outputs the estimated tempo and the predicted beat location by the result obtained from either Ellis's method or Dixon's method.

5 Experimental Results

We evaluate our method with a public "ballroom dancer" database. Suppose the tempo of a music excerpt is estimated by a method. If the absolute difference between the estimated tempo and the real tempo is not greater than P times the real tempo, we say that the estimation of the music excerpt is correct. Usually the error tolerance is $P = 0.01$. There are ten music genres (types) in the "ballroom dancer" database, which contains 698 music excerpts (music segments). Here, the training, validation and testing data sets are randomly chosen from in the "ballroom dancer" database with sizes about $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{4}$ of each music genre, respectively. For the experiment purpose, the genre of each excerpt in the training set is assumed to be known. But, the genre of each excerpt in the training or testing set is as-

Table 2: The dimension of each feature type after PCA with threshold 99.9% (average).

Feature type	LPC	DFT	HPS	Cepstrum	MFCC	Volume	All
Feature number (PCA)	31	67	65	99	31	10	80

Table 3: The mean and standard deviation of 14 experimental results with training, validation and testing data sets.

	Ellis's method (%)	Dixon's method (%)	Our method (All) (%)	Our method (Val)(%)	Our method (Some) (%)
Training data set (mean)	53.043	47.571	60.473	57.873	60.505
Testing data set (mean)	51.275	46.378	54.625	53.522	53.829
Validation data set (mean)	52.886	49.038	55.326	55.083	55.302
Training data set (standard deviation)	1.825	1.059	1.344	1.745	1.361
Testing data set (standard deviation)	2.370	2.197	2.624	3.101	3.218
Validation data set (standard deviation)	2.466	2.804	3.045	2.846	2.166

sumed to be unknown. When we build a neural network, we try to give three kinds of different parameter settings as follows.

All-parameter : All parameters included in library function for the neural network in Matlab are set as follows:

$max\_fail = 5$, the maximum occurrences of the validation failures.

$mem\_reduc = 1$, the factor for memory and speed trade-off.

$mu = 1$, the initial μ value.

$mu\_dec = 0.8$, the decreasing factor of μ.

$mu\_inc = 1.5$, the increasing factor of μ.

$mu\_max = 1e^{10}$, the maximum value of μ.

Validation convergence : In addition to the above parameters, the training program may terminate early due to the convergence of the validation data set.

Some-parameter : The settings in all-parameters and the validation data set are removed. Instead, default settings are used.

We build thirteen neural network models for each of the above three parameter settings. Then, we use a voting mechanism to get the final result. In the following tables, we use 'All', 'Val' and 'Some' to represent 'all-parameter', 'validation convergence' and 'some-parameter', respectively. In one experiment, we form the training, validation and testing data sets by randomly selected from the database. The evaluation preference is decided by the vote of 13 neural network models, each with the three types of parameter settings. In this paper, we perform the above experiment 14 times. The mean and standard deviation of the 14 experimental results is shown in Table 3. The table shows that the all-parameter setting gets more accurate and more stable solutions. The accuracy of our method is better than the other two methods. In fact, our method gets better solutions in all of the 14 experiments.

In addition, We perform another experiment in which the original input data set is changed. We change the classification from the ten music data sets (genres) of the "ballroom dancer" database to four data sets. The four data sets represent four types which are '01', '10', '00', and '11'respectively. '0 1' means that Ellis's method is the better one for predicting tempo and beat locations for the input music signal, and '1 0' means that Dixon's method is the better one. '0 0' means that neither of Ellis's or Dixon's methods can get correct prediction. '1 1' means that both Ellis's and Dixon's methods can get correct prediction.

We choose the training and testing data sets from the four data sets for insuring that the size of the training data set is enough. The training data set is constructed from the '0 1' and '1 0' data sets, and the testing data set is constructed from the four data sets. Here, the testing data set is randomly chosen from the four data sets with sizes from $\frac{1}{4}$ to $\frac{1}{2}$ of the each data set. The training way and the estimated method are the same as the above description. We also use a neural network to classify the music excerpts into the evaluation preference. And then, with the preference classification, we can obtain accurate estimation for tempo and beats, by either Ellis's method or Dixon's method. We set the evaluation preference to Ellis's method if the music excerpt is not classified by neural network. We perform the experiment 14 times. The mean and standard deviation of the 14 experimental results is shown in Table 4. We get better solutions in all of the 14 experiments by this way, too.

Table 4: The mean and standard deviation of 14 experimental results with testing data set.

	Ellis's method (%)	Dixon's method (%)	Our method (All) (%)
Testing data set (mean)	54.179	47.577	56.336
Testing data set (standard deviation)	4.727	5.069	4.345

6 Conclusion

We first build a neural network model to classify a given music excerpt (audio signal) into the preference of either Ellis's method [5] or Dixon's method [4]. As the experimental results described in Section 5, the accuracy ratio of our method is better than only one individual Ellis's method or Dixon's method. It is not easy to extremely increase the accuracy ratio for detecting the tempo and beat locations of the audio signal. Moreover, it is difficult to detect the tempo and beat locations of the audio signal for some music genres such as the classical music, jazz music, etc. Detecting the tempo and beat locations by using only one method is certainly unable to obtain a good result. Thus our method is to build a system combining the advantages of the two methods to obtain a good result for each music genre.

References

[1] M. A. Alonso, B. David, and G. Richard, "Tempo and beat estimation of musical signals," *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*, Barcelona, Spain, 2004.

[2] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, M. B. Sandler, and S.Member, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, Vol. 13, No. 5, pp. 1035–1047, 2005.

[3] L. Cohen, *Time-frequency analysis: theory and applications.* Prentice-Hall, Inc, 1995.

[4] S. Dixon, "Mirex 2006 audio beat tracking evaluation: Beatroot," *Proceedings of the 2th Music Information Retrieval Evaluation eXchange (MIREX 2006)*, 2006.

[5] D. P. Ellis, "Beat tracking with dynamic programming," *Proceedings of the 2th Music Information Retrieval Evaluation eXchange (MIREX 2006)*, 2006.

[6] L. Gu and K. Rose, "Perceptual harmonic cepstral coefficients as the front-end for speech recognition," *Proceedings of the 6th International Conference on Spoken Language Processing*, Vol. 1, Beijing, China, pp. 309–312, 2000.

[7] J. S. R. Jang, "Audio signal processing and recognition." (in Chinese) available at the links for on-line courses at the author's homepage at http://neural.cs.nthu.edu.tw/jang.

[8] S. Kumar, *Neural Networks: A Classroom Approach.* Tata Mcgraw Hill, 2004.

[9] L. I. Smith, "A tutorial on principal components analysis." http://csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf, 2002.

[10] A. Lacoste and D. Eck, "A supervised classification algorithm for note onset detection," *EURASIP Journal on Advances in Signal Processing*, Vol. 2007, No. 43745, p. 13, 2007.

[11] R. C. T. Lee, S. S. Tseng, R. C. Chang, and Y. T. Tsai, *Introduction to the Design and Analysis of Algorithms.* McGraw-Hill, 2005.

[12] M. F. McKinney and D. Moelants, "Deviations from the resonance theory of tempo induction," *Proceedings of the Conference on Interdisciplinary Musicology (CIM04)*, Graz, Austria, 2004.

[13] G. Peeters, "Time variable tempo detection and beat marking," *Proceedings of International Computer Music Conference (ICMC 2005)*, Barcelona, Spain, 2005.

[14] S. Roweis, "Levenberg-marquardt optimization." http://www.cs.toronto.edu/∼roweis/notes.html.

[15] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *The Journal of the Acoustical Society of America*, Vol. 103, pp. 588–601, 1998.

[16] P. Scott, "Music classification using neural networks," Stanford University, 2001.

[17] G. Tzanetakis, "Tempo extraction using beat histograms," *Proceedings of the 1th Music Information Retrieval Evaluation eXchange (MIREX 2005)*, 2005.