Proceedings of the Eighth IASTED International Conference
ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING
September 1-3, 2004, Marbella, Spain

A PROTOTYPE FOR CLASSIFICATION OF CLASSICAL MUSIC USING NEURAL NETWORKS

R. Malheiro R. P. Paiva A. J. Mendes T. Mendes A. Cardoso

CISUC – Centro de Informática e Sistemas da Universidade de Coimbra
Departamento de Engenharia Informática, PÓLO II da Universidade de Coimbra,
Pinhal de Marrocos, P 3030, Coimbra, Portugal
Email: {rsmal@netcabo.pt}, {ruipedro, toze, tmendes, amilcar}@dei.uc.pt

ABSTRACT
As a result of recent technological innovations, there has been a tremendous growth in the Electronic Music Distribution industry. In this way, tasks such us automatic music genre classification address new and exciting research challenges. Automatic music genre recognition involves issues like feature extraction and development of classifiers using the obtained features. As for feature extraction, we use features such as the number of zero crossings, loudness, spectral centroid, bandwidth and uniformity. These are statistically manipulated, making a total of 40 features. As for the task of genre modeling, we train a feedforward neural network (FFNN). A taxonomy of subgenres of classical music is used. We consider three classification problems: in the first one, we aim at discriminating between music for flute, piano and violin; in the second problem, we distinguish choral music from opera; finally, in the third one, we aim at discriminating between all five genres. Preliminary results are presented and discussed, which show that the presented methodology may be a good starting point for addressing more challenging tasks, such as using a broader range of musical categories.

KEY WORDS
Neural networks, music classification, music signal processing, music information retrieval

1. Introduction

Presently, whether it is the case of a digital music library, the Internet or any music database, search and retrieval is carried out mostly in a textual manner, based on categories such as author, title or genre. This approach leads to a certain number of difficulties for service providers, namely in what concerns music labeling. Real-world music databases from sites like AllMusicGuide or CDNOW grow larger and larger on a daily basis, which requires a tremendous amount of manual work for keeping them updated. Thus, simplifying the task of music database organization would be an important advance. This calls for automatic classification systems. Such systems should overcome the limitations resulting from manual song labeling, which may be a highly time-consuming and subjective task.

Some authors have addressed this problem recently. Tzanetakis and Cook [1] classify music in ten genres, namely, classical, country, disco, hip-hop, jazz, rock, blues, reggae, pop and metal. They further classify classical music into choir, orchestra, piano and string quartets. Features used encompass three classes: timbral, rhythmic and pitch-related features. The authors investigate the importance of the features is training statistical pattern recognition classifiers, particularly, Gaussian Mixture Models and k-nearest neighbors. 61% accuracy was achieved for discriminating between the ten classes. As for classical music classification, an average accuracy of 82.25% was achieved. Golub [2] uses seven classes of mixed similarity (a capella, celtic, classical, electronic, jazz, latin and pop-rock). The features used are loudness, spectral centroid, bandwidth and uniformity, as well as statistical features obtained from them. A generalized linear model, a multi-layer perceptron and a k-nearest classifier were used. The best of them achieved 67% accuracy. Kosina [3] classifies three highly dissimilar classes (metal, dance and classical) using k-nearest neighbors. The used features were mel-frequency cepstral coefficients, zero-crossing rate, energy and beat. 88% accuracy was achieved. Martin [4] addresses the problem of instrument identification. He proposes a set of features related to the physical properties of the instruments with the goal of identifying them in a complex auditory environment.

In our work we aim at classifying five subgenres of classical music, namely, opera, choral music and music for flute, piano and violin. This is due to the fact that there are not many studies regarding specifically classical music. Also, digital music libraries have a great diversity of taxonomies of classical music, which demonstrates its practical usefulness. Unlike other authors who use a broad range of generic classes, we chose to focus on specific set of related classes. This seems to be a more challenging problem since our classes show a higher similarity degree, leading to, we think, a more difficult classification problem. We chose a set of features based on those used in [5] and [2], encompassing especially timbre and pitch content, which seemed relevant for the task under analysis: the number of zero crossings, loudness, spectral centroid, bandwidth and uniformity. Rhythmic features were not used. An FFNN classifier is used, which is trained via the Levenberg-Marquardt algorithm. For validation purposes we obtained 76% accuracy, using 6s' extracts from each song. Our results, tough far from ideal, are satisfactory. Comparing to [1], we got a similar accuracy using one more category and a reduced feature set.

Additionally, we present a prototype system for automatic music classification of entire songs (not only extracts). We use 10 extracts of 6 seconds for each song, uniformly distributed throughout the song. Each song is classified according to the most representative genre in all extracts.

This paper is organized as follows. Section 2 describes the process of feature extraction and the features used. In Section 3, we give a short overview of FFNNs and their application to our music genre recognition problem. Experimental results are presented and discussed in Section 4. Section 5, describes the prototype system for classical music classification and analyzes the obtained results. Finally, in Section 6, conclusions are drawn and directions for future work are presented.

2. Feature Extraction

Based on the classification objectives referred, and taking into account the results obtained in similar works, we gave particular importance to features with some significance for timbral and pitch content analysis. We used no rhythmic features, since they did not seem very relevant for the type of music under analysis. However, we plan to use them in the future and evaluate their usefulness in this context.

We started by selecting 6 seconds' segments from each musical piece (22khz sampling, 16 bits quantization, monaural). Since for training issues the segment samples used should have little ambiguity regarding the category they belong to, we selected relevant segments from each piece. The purpose was not to use long training samples. Instead, short significant segments are used, mimicking the way humans classify music, i.e., short segments [6] using only music surface features without any higher-level theoretical descriptions [7].

After collecting a relevant segment for each piece, the process of feature extraction is started by dividing each 6s signal in frames of 23.22 with 50% overlap. This particular frame length was defined so that the number of samples in each frame is a power of 2, which is necessary for optimizing the efficiency of Fast Fourier Transform (FFT) calculations [8] (Section 2.2). This gives 512 samples per frame, in a total of 515 frames.

Both temporal and spectral features are used, as described below.

2.1 Time-Domain Features

As for temporal features, we use loudness and the number of zero crossings. Loudness is a perceptual feature that tries to capture the perception of sound intensity. Only the amplitude is directly calculated from the signal. Loudness, i.e., the perception of amplitude, can be approximated as follows [2] (1):

$$L(r) = \log_2\left(1 + \frac{1}{N}\sum_{n=1}^{N}\left|x(n)\right|\right) \tag{1}$$

where L denotes loudness, r refers to the frame number, N is the number of samples in each frame, n stands for the sample number in each frame and $x(n)$ stands for the amplitude n-th sample in the current frame.

The number of zero crossings simply counts the number of times the signal crosses the time axis, as follows [5] (2):

$$Z(r) = \frac{1}{2}\sum_{n=1}^{N}\left|\operatorname{sgn}(x(n)) - \operatorname{sgn}(x(n-1))\right| \tag{2}$$

where Z represents the number of zero crossings. This is a measure of the signal frequency content, which is frequently used in music/speech discrimination and for capturing the amount of noise in a signal [1].

2.2 Frequency-Domain Features

The spectral features used, computed in the frequency domain, are spectral centroid, bandwidth and uniformity. Therefore, the process starts by converting the signal into the frequency domain using the Short-Time Fourier Transform (STFT) [9]. In this way, the signal is divided in frames, as stated above. The signal for each frame is then multiplied by a Hanning window, which is characterized by a good trade-off between spectral resolution and leakage [8].

Spectral centroid is the magnitude-weighted mean of the frequencies [2] (3):

$$C(r) = \frac{1}{N}\frac{\sum_{k=1}^{N} M_r(k)\cdot\log_2 k}{\sum_{k=1}^{N} M_r(k)} \tag{3}$$

where $C(r)$ represents the value of the spectral centroid at frame r and $M_r(k)$ is the magnitude of the Fourier transform at frame r and frequency bin k. This is a measure of spectral brightness, important, for instance, in music/speech or musical instrument discrimination.

Bandwidth is the magnitude-weighted standard deviation of frequencies [2], as in (4). There, $B(r)$ represents the spectral bandwidth at frame r. This is a measure of spectral distribution: lower bandwidth values denote a concentration of frequencies close to the centroid (which is the energy-weighted mean of frequencies), i.e., a more narrow frequency range.

$$B(r) = \sqrt{\frac{\sum_{k=1}^{N} \left(C(r) - \log_2 k\right)^2 M_r(k)}{\sum_{k=1}^{N} M_r(k)}} \qquad (4)$$

Uniformity gives a measure of spectral shape. It measures the similarity of the magnitude levels in the spectrum and it is useful for discriminating between highly pitched signals (most of the energy concentrated in a narrow frequency range) and highly unpitched signals (energy distributed across more frequencies) [2]. Uniformity is computed as follows (5):

$$U(r) = -\sum_{k=1}^{N} \frac{M_r(k)}{\sum_{k=1}^{N} M_r(k)} \cdot \log_N \frac{M_r(k)}{\sum_{k=1}^{N} M_r(k)} \qquad (5)$$

For each frame, the five features described are extracted. Then, first-differences are calculated, based on the feature values in consecutive frames, e.g., *L*(*r*) - *L*(*r*-1). These five new features plus the five features described before constitute our set of 10 basis features.

Classical music is usually characterized by accentuated variations in the basis features throughout time. Therefore, statistical manipulations of the basis features are calculated in order to cope with this aspect.

The means and standard deviations for the ten basis features are calculated in 2 seconds' chunks, leading to 20 features. The final features that compose the signature correspond to the means and standard deviations of the 20 intermediate features computed previously. We get a total of 40 features (2×2×10).

3. Genre Modelling with FFNNs

Artificial Neural Networks (ANN) [10] 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.

Feedforward Neural Networks (FFNN) are 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 [11]) and the output layer, which sends data to the exterior environment (Figure 1).

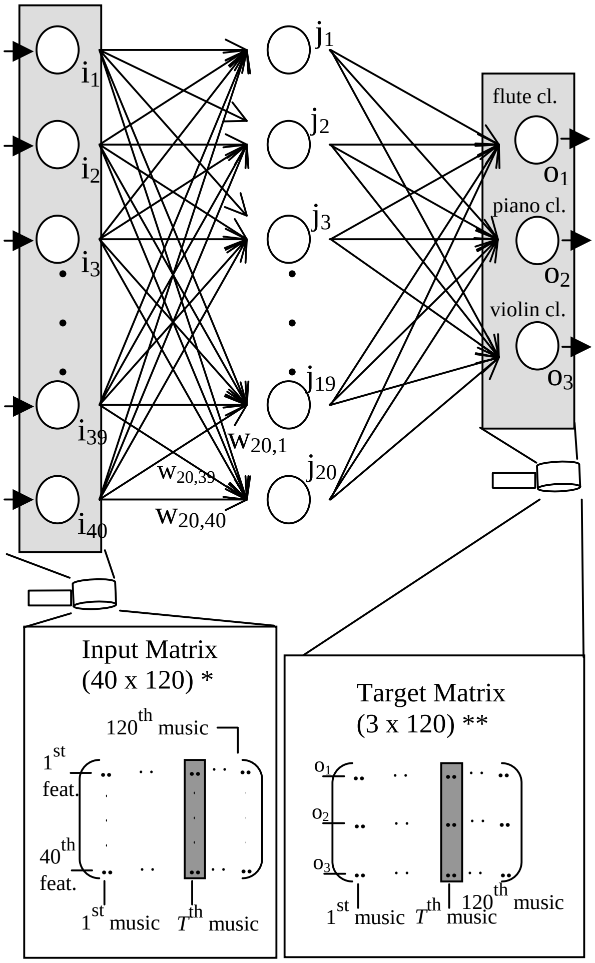

Figure 1. FFNN used on the classification of music in three musical genres (flute, piano and violin).

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 activation function. Usually, sigmoid functions are used [10].

The capability of the FFNN for mapping input values into output values depends on the link weights. Their optimal determination is still and 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.

ANNs are usually trained in a supervised manner, i.e., the weights are adjusted based on training samples (input-output pairs) that guide the optimization procedure towards an optimum. For instance, in the case of our music genre classification (Figure 1), each network input is a vector with the 40 extracted features and each target value has a value of 1 for the correct class and a value of

0 otherwise. Our FFNN is trained in batch mode, i.e., all the training pares are presented to the network, an error measure is computed and only then the weights are adjusted towards error reduction. In Figure 1, we have a 40×120 input matrix where each line corresponds to a particular feature and each column corresponds to each music feature-vector used for training the network. In the same figure, a 3×120 target output matrix is presented, where each column has information regarding the target class for the corresponding music feature-vector: all the lines have zero value, except for the line corresponding to the correct class, which has a value of one. For example, if the *T*th music signature denotes a piano piece, and the second output neuron was assigned to the piano category, then the *T*th ouput column would have a value of 1 in the second line, and zero for all other lines.

The most widely used training algorithm for FFNNs is backpropagation [10]. Here, there is a forward pass where inputs are presented to the network and output values are computed. The error between each target value and the corresponding output value is then calculated. Then, a backward pass is performed, where the weights are adjusted towards error reduction, using the gradient descent method. This process is repeated iteratively until the error is below a given threshold.

The gradient descent method has some limitations regarding convergence properties: the algorithm can get stuck in a local minimum and the selection of the learning rate is usually not trivial (if its value is too low, learning is slow; if it is too high, the network may diverge). Therefore, some variants are used, e.g., learning with a momentum coefficient or defining an adaptive learning rate [10].

Here, we use the Levenberg-Marquardt algorithm, which has the advantage of being significantly faster (10 to 100 times faster [12]) at the cost of higher memory consumption, due to the computation of a Jacobian matrix in each iteration. Also, this algorithm converges in situations where others do not [13].

After training, the neural network must be validated, i.e., its response to unknown data must be analyzed in order to evaluate its generalization capabilities. Thus, a forward pass is performed, with samples never presented before, and the same error measure used during training is computed. Typically, the available samples are divided in two sets, one for training and the other for validation, 2/3 for the former and 1/3 for the latter, respectively.

In order to avoid numerical problems, all the features were previously normalized to the [0, 1] interval [12].

4. Experimental Results

As we stated before, our aim is to build a prototype of a real system for classification of classical music. We defined a taxonomy of five sub-genres: pieces for flute, piano, violin, choral and opera. These can be organized in a hierarchical manner, grouping flute, piano and violin as instrumental music and choral and opera as vocal. The presented taxonomy is defined only for the sake of clarity: the practical classification performed was not hierarchical.

For evaluation purposes, we collected a database of 100 monaural classical pieces, 20 from each class, sampled at 22050 Hz, with 16 bits quantization. For each musical piece, 10 segments of 6 seconds each were extracted. Those segments were automatically created so as to uniformly cover the entire piece. Then, each piece was classified according to the most represented class in all its segments.

Before classifying entire songs, we evaluated our approach with a database of 300 monaural musical segments, 60 from each genre. The segments, each with duration of 6 seconds, were manually extracted from the initial database, based on their relevance for the genre in cause, as stated in Section 2. The difference to the segments in entire-song classification is that, in the prototype, the segments were automatically extracted, whereas in the segment classification task “well-behaved” samples were selected.

Our first goal was to discriminate between three genres of instrumental music: music for flute, piano and violin. The 6s’ segments extracted were chosen so as to include soles from each instrument by single or several players in unison, in isolation (monophonic segment) or with an orchestra in the background (polyphonic segment).

In our second goal, we wanted to discriminate between genres of vocal music: chorals and opera. Typically, the musical pieces used for opera were vocal soles, essentially performed by tenors, sopranos and mezzo-sopranos (Callas, Pavarotti, etc.), whereas for choral music segments of simultaneous distinct voices were used without many of the stylistic effects used in opera (vibrato, tremolo). Many of the used pieces were also a cappela, i.e., only human voices, no instruments.

Finally, our third goal was to discriminate between all of the five genres referred above.

For the three problems addressed we used three-layered FFNNs, trained in batch mode via the Levenberg-Marquardt algorithm. Each network consists of 40 input neurons (one for each extracted feature) a variable number of hidden neurons (described below) and 2, 3 or 5 output neurons, according to problem under analysis. Both hidden and output neurons use sigmoid activation functions. For training purposes, we used 40 pieces from each genre, whereas for validation the remaining 20 were used (a total of 200 pieces for training and 100 for validation). Special care was taken so that the training samples for each genre were diverse enough.

Validation, i.e., classification of unknown segments, was carried out under two different perspectives that we designate as percentage calculus rule 1 (PCR1) and percentage calculus rule 2 (PCR2).

Under the PCR1 perspective, a musical piece from a particular genre is well classified when the highest network output corresponds to that genre and its value is above or equal 0.7 (recall that the network outputs values between 0 and 1). In this situation, the piece considered is correctly classified, without any ambiguities.

When all output values are under 0.7, it is concluded that this particular musical piece does not belong to any of the defined categories. The highest value is not high enough to avoid possible ambiguities.

As for PCR2 In this case, a musical piece from a particular genre is well classified if the highest network output value corresponds to the right genre, regardless of its amplitude.

Regarding segment classification, 85%, 90% and 76% average accuracy was obtained for the three, two and five genre classification tasks, respectively [14]. These results, tough not accurate enough for real applications, are encouraging. Therefore, we decided to evaluate our approach in the classification of entire songs. Below we present the results for each of the classification tasks addressed, regarding the entire-song classification problem.

4.1. First Classification: Three Genres

In this case, musical pieces were classified into flute, piano and violin pieces. A database of 60 songs, 20 per class, was used. As referred before, each song is represented by 10 segments of 6 seconds each, and the final classification corresponds to the most represented genre.

For the determination of the most adequate number of neurons in the hidden layer, we tested several values in the range [10, 30]. The best classification results were obtained for 20 neurons in the hidden layer.

We obtained as average accuracy of 78% (75% for flute, 59.1% for piano and 100% for violin) both for PCR1 (Table 1) and PCR2 (Table 2). Analyzing the results for flute pieces, we also notice that 5% of them were erroneously classified as piano and 20%. Furthermore, no songs remained unclassified.

PCR1 78%	Flute	Piano	Violin
Flute	75	9	-
Piano	5	59.1	-
Violin	20	31.9	100
unclassif.	-	-	-

Table 1. Instrumental music confusion matrix: PCR1.

PCR2 85%	Flute	Piano	Violin
Flute	75	9	-
Piano	5	59.1	-
Violin	20	31.9	100

Table 2. Instrumental music confusion matrix: PCR2.

It is interesting to see the excellent results obtained for the violin class, showing that the network correctly captured is characteristics, particularly its timbre.

As for the piano class, the results were somewhat disappointing, with only 59.1% accuracy and 31.9% of songs misclassified as violin. We could no find any reasonable explanation for that.

Anyway, we think these results are positive, since the average results based on automatically extracted segments (78%) were close the ones obtained for segment classification using "well-behaved" samples (85%)

4.2. Second Classification: Three Genres

In this situation, musical pieces were classified into opera and choral pieces. A database of 40 songs, 20 per class, was used. We obtained best classification results with 25 neurons in the hidden layer: an average classification accuracy of 73.5% (81.8% for choral pieces and 65.2% for opera) both for PCR1 (Table 3) and PCR2 (Table 4). We can see that no songs remained unclassified.

PCR1 73.5%	Choral	Opera
Choral	81.8	34.8
Opera	18.2	65.2
unclassif.	-	-

Table 3. Vocal music confusion matrix: PCR1.

PCR2 73.5%	Choral	Opera
Choral	81.8	34.8
Opera	18.2	65.2

Table 4. Vocal music confusion matrix: PCR2.

The obtained results fell notoriously below the ones for segment classification (90%). This drop follows directly from the percentage of opera songs that were misclassified as chorals: 34.8%. We analyzed some of those cases and observed that many operas have regions that could easily be mistaken as chorals, even for humans. Those regions are, especially, the quieter ones.

4.3. Third Classification: Five Genres

Here, musical pieces were classified into the five categories listed before: flute, piano, violin, opera and choral music. A database of 100 songs, 20 per class, was used. Best classification results were obtained with 20 neurons in the hidden layer for PCR1, with 57.3% average classification accuracy, and 30 neurons for PCR2, with 66.7% average classification accuracy, for the five genres used.

Regarding PCR1 analysis (Table 5), we obtained 59.2% classification accuracy for flute pieces, 42.3% for piano, 85% for violin, 59.2% for chorals and 40.9% for opera. 15% of the musical pieces remained unclassified.

As for PCR2 analysis (Table 6), the classification accuracy was 66.7% for flute pieces, 50% for piano, 100% for violin, 66.7% for chorals and 50% for opera.

Though interesting, the results obtained for this more complex classification problem are less satisfactory. They fell from an average of 76% accuracy in the segment classification to 66.7% for classification of entire musical pieces.

However, once again the violin class accomplished outstanding results: 100% accuracy for PCR2. As for PCR1, there are only three false negatives: two unclassified pieces and one piece misclassified as choral. Therefore, we can conclude that this classifier learned the best way to identify the characteristics of the violin.

PCR1 57.3%	Flute	Piano	Violin	Choral	Opera
Flute	59.2	3.9	0	18.2	0
Piano	4.5	42.3	0	4.5	0
Violin	13.6	7.7	85	0	9.1
Choral	4.5	19.2	5	59.2	31.8
Opera	0	11.5	0	4.5	40.9
unclassif.	18.2	15.4	10	13.6	18.2

Table 5. Mixed classification confusion matrix: PCR1.

PCR2 66.7%	Flute	Piano	Violin	Choral	Opera
Flute	66.7	4.2	0	19.1	5
Piano	4.8	50	0	4.7	0
Violin	14.2	4.2	100	0	15
Choral	9.5	29.1	0	66.7	30
Opera	4.8	12.5	0	9.5	50

Table 6. Mixed classification confusion matrix: PCR2.

Unlike the violin class, the results for piano and opera were not so good. From Table 6, we can see that 29.1% of the piano pieces were classified as chorals and 30% of the operas were classified also as chorals. This conflict between opera and choral music had already been detected in the two-class separation task and comes from the same reasons pointed out before. As for the conflict between piano and choral pieces, we could not any reasonable justification for it, except for the fact that they both are often rather quiet.

As a conclusion, we could say that the obtained results are encouraging in a perspective of future evolvement. However, it is clear that the used features could not separate the five classes in a totally unambiguous manner. Therefore, a deeper feature analysis seems fundamental in order to obtain better results.

5. Conclusion

The main goal of this paper was to present a methodology for the classification of classical music. Although the results obtained are not sufficient for real-world applications, they are promising.

In the most complex case, where we defined five categories, the classification results were less accurate. However, in our opinion a hierarchical classifier, following the structure in Figure 3, would lead to better results.

In the future, we will conduct a more thorough analysis of the feature space: detection and elimination of redundant features, as well as definition and utilization of other features, which may help to discriminate the more atypical cases. Another way to increase the classification accuracy would be to increase the number of segments used in each song. One other possibility would be to train the network with a higher number of training examples, containing more atypical cases for each genre.

Additionally, we plan to use a broader and deeper set of categories, i.e., more basis classes and subclasses. In case we use categories like waltz, rhythmic features, not used in the present work, will certainly be important.

6. Acknowledgements

This work was partially supported by the Portuguese Ministry of Science and Technology (MCT), under the program PRAXIS XXI.

References:

[1] G. Tzanetakis & P. Cook, Musical Genre Classification of Audio Signals, *IEEE Transactions on Speech and Audio Processing*, 10(5), 2002.

[2] S. Golub, *Classifying Recorded Music* (MSc Thesis: University of Edinburgh, 2000).

[3] K. Kosina, *Music Genre Recognition* (MSc Thesis: Hagenberg, 2002).

[4] K. Martin, Toward Automatic Sound Source Recognition: Identifying Musical Instruments, *NATO Computational Hearing Advanced Study Institute*, Il Ciocco, Italy, 1998.

[5] G. Tzanetakis, Essl G. & P. Cook, Automatic Musical Genre Classification of Audio Signals, *2nd International Conference on Music Information Retrieval*, ISMIR, 2001.

[6] D. Perrot & R. O. Gjerdigen, Scanning the dial: An exploration of factors in the identification of musical style, *Proceedings of the Society for Music Perception and Cognition*, 1999.

[7] K. D. Martin, E. D. Scheirer & B. L. Vercoe, Musical content analysis through models of audition, *ACM Multimedia Workshop on Content-Based Processing of Music*, 1998.

[8] S. Smith, The Scientist and Engineer's Guide to Digital Signal Processing (California Technical Publishing, 1997).

[9] R. Polikar, *The Wavelet Tutorial*, (http://engineering.rowan.edu/~polikar/WAVELETS /WTtutorial.html, available by July 2003).

[10] S. Haykin, *Neural Networks: A Comprehensive Foundation* (Macmillan College Publishing, 1994).

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

[12] H. Demuth & M. Beale, *Neural Network Toolbox User’s Guide - version 4* (Mathworks, 2001).

[13] M. Hagan & M. Menhaj, Training Feedforward Networks with the Marquardt Algorithm, *IEEE Transactions on Neural Networks*, 5(6), 1994.

[14] R. Malheiro, R. P. Paiva, A. Mendes, T. Mendes & A. Cardoso, Classification of Recorded Classical Music Using Neural Networks, *Proceedings of the ICSC Symposium on Engineering of Intelligent Systems, EIS'2004*, Portugal, 2004.