Music Genre Classification Using MIDI and Audio Features

Zehra Cataltepe, Yusuf Yaslan, Abdullah Sonmez
Istanbul Technical University
Computer Engineering Department
Maslak, Sariyer, Istanbul, Turkey
{zehra, yyaslan, sonmezab}@cs.itu.edu.tr

Abstract
We report our findings on using MIDI files and audio features from MIDI, separately and combined together, for MIDI music genre classification. We use McKay's 3 root and 9 leaf genre data set. Although MIDI and audio from MIDI alone achieve much smaller accuracy than results reported by McKay, combining those classifiers improves accuracy. Successful classifier combination requires diversity of the base classifiers. We achieve diversity through using certain number of seconds of the MIDI file, different sample rates and sizes for the audio file and different classification algorithms.

1. Introduction
The increase of the musical databases on the internet and multimedia systems have brought a great demand for Music Information Retrieval (MIR) applications and especially automatic analysis of the musical databases. Most of the current databases are indexed based on song title or artist name, where improper indexing can result in an incorrect search. More effective systems extract important features from audio and classify the audio to its genre based on these features. This kind of music retrieval systems should also have the ability to find similar songs based on their extracted features. However there are not any strict distinguishing boundaries between audio genres and no complete agreement exists in their definition [1,2].

Generally audio signal can be represented in two ways on computers. The first one is symbolic representation based on musical scores. Examples of this representation are MIDI and Humdrum where for each note, pitch, duration (start time/end time) and strength is kept in the file. The second one is based on acoustic signals, recording the audio intensity as a function of time based on a certain frequency and can be in compressed or uncompressed domain [3]. Note that because of the difference of the representation of symbolic and acoustic data music retrieval algorithms that deal with these data are very different.

Previous work that deals with genre classification, uses symbolic representations of music such as MIDI files [2,4,5] or audio, such as wav or mp3 files [6,7]. Most of the proposed methods have two processing steps. The first one is frame based feature extraction step of acoustic data where feature vectors of low-level descriptors are computed from each frame. Most common features used for genre classification of acoustic data are; timbre related, rhythm related and pitch related features [8]. Timbre related features are; FFT coefficients, Cepstrum and Mel Frequency Cepstrum Coefficients (MFCC), Linear Prediction (LP) coefficients, MPEG filterbank components, Spectral Centroid, Spectral Flux, Zero Crossing Rate, Spectral Roll-Off, Low order statistics and Delta coefficients [6]. More detailed descriptions of the features can be found in [6,9,8]. In the second step pattern recognition algorithms are applied on the feature vectors to infer genre [6].

MIDI format developed as a standard to play music on instruments or computer. The sound quality of a MIDI music piece depends on the synthesizer (sound card) and MIDI has its other limitations, such as it can not store voice. On the other hand, this format takes a lot less space, hence it is much easier to store and communicate, is widely accepted and allows for better comparison between music pieces played on different instruments. Music retrieval problem for symbolic data generally becomes similar to a string matching problem and can be solved by several text searching algorithms [3].

Recently Vitanyi and his colleagues [10,11] have suggested using an approximation to Kolmogorov distance between two musical pieces as a means to compute clusters of music. They first process the MIDI representation of a music piece to turn it into a string from a finite alphabet. Then they compute the distance between two music pieces using their Normalized Compression Distance (NCD). NCD uses the compressed length of a string as an approximation to its Kolmogorov Complexity. Although the Kolmogorov Complexity of a string is not computable, the compressed length approximation seems to have given good results for musical genre and composer clustering and other data sets [12].

Previously Cory McKay [4] has reported very good root (98%) and leaf (90%) genre classification accuracy on his 3 root and 9 leaf genre dataset of 225 MIDI music pieces. We use the same data set in our experiments. We first train classifiers for MIDI genre classification. We produce audio files from MIDI files and then use audio to determine the genres. We combine MIDI and audio classifiers to achieve better accuracy.

We use our pre-processing method [13] of MIDI files, compute NCD distance between them using complearn software [www.complearn.org] and then k-Nearest Neighbour classifiers to predict root and leaf genre of MIDI files. In order to achieve classifier diversity, we train four different MIDI classifiers, using the first 30 seconds, 60 seconds, 120 seconds of the pieces only and also using the whole piece.

We convert the MIDI files to aiff files using QuickTime Player and Audio Hijack. Then, we use iTunes to obtain wav encoded mono files using 6 different sample rates and sample sizes (22.050 kHz, 8bit; 22.050 kHz, 16bit; 32 kHz, 8bit; 32 kHz, 16bit; 44.1 kHz, 8bit; 44.1 kHz, 16bit). We use the freely available Marsyas software [opihi.cs.uvic.ca/marsyas] by Tzanetakis [8] to extract the audio features.

2. Classifiers

Many classification techniques have been used for genre classification. Examples are: Gaussian mixture models [8], support vector machines[14,15], radial basis functions [16], linear discriminant analysis [15] and k-nearest neighbors [15]. In this study, we report our experiments with Linear Discriminant classifiers (LDC) which assume normal densities and k-nearest neighbor classifiers (KNN). We also have experimented with Quadratic Discriminant Classifiers (QDC), Fisher Linear Discriminant (Fisher), Naïve Bayes Classifier (NBC) and Parzen Density Based Classifier (PDC). However, since they gave as good results and for their simplicity, in this study, we report our experiments using LDC and KNN. We give brief descriptions of LDC and KNN classifiers below and refer the reader to [17] for more information.

Linear Bayes Classifier (LDC): The objective of the linear discriminant analysis is to find sets of hyperplanes separating classes. LDC is a linear classifier assuming normal densities with equal covariance matrices. Fisher's LDA performs dimensionality reduction while preserving the class discriminatory information.
k-Nearest Neighbor (KNN): is a well known nonparametric classifier. The training data is stored with their labels. A new input x is classified according to the labels of its closest (according to a distance metric) k neighbors in the training set. The value of k affects the complexity of the classifier. In our experiments we use k=10 (10NN).

3. Genre Classification Using Audio Features

Several feature extraction methods including low-level parameters such as zero-crossing rate, signal bandwidth, spectral centroid, root mean square level, band energy ratio, delta spectrum, psychoacoustic features, MFCC and Auditory filterbank temporal envelopes have been employed for audio classification [8]. We have obtained the following content based audio features using Tzanetakis's Marsyas software.

3.1. Timbral Features

Timbral features are generally used for music-speech discrimination and speech recognition. They differentiate mixture of sounds with the same or similar rhythmic content. In order to extract the timbral features audio signal is divided into small intervals that can be acceptable as stationary signal. The following timbral features are calculated for these small intervals:

- Spectral Centroid: Measures the spectral brightness and is defined as the center of the gravity of the magnitude spectrum of the STFT.
- Spectral Rolloff: Measures the spectral shape and is defined as the frequency value below which lies the 85% of the magnitude distribution.
- Spectral Flux: Measures the amount of local spectral change and is defined as the squared difference between the normalized magnitudes of successive spectral distributions.
- Time Domain Zero Crossing: Measures the noisiness of the signal and is defined as the number of time domain zero crossings of the signal.
- Low Energy: Measures the amplitude distribution of the signal and is defined as the percentage of the frames that have RMS energy less than the average RMS energy over the whole signal.
- Mel-Frequency Cepstral Coefficients (MFCC): MFCCs are well known for speech representation. They are calculated by taking the log-amplitude of the magnitude spectrum and then smoothing the grouped FFT bins according to the perceptually motivated Mel-frequency scaling.

Means and the variances of the spectral centroid, spectral rolloff, spectral flux, zero crossing and low energy [8] results in 9 dimensional feature vector and represented in experimental results as STFT label. Means and variances of the first five MFCC coefficients yield a 10 dimensional feature vector, which is represented as MFCC in the experiments.

3.2. Rhythmic Content Features

Rhythmic content features characterize the movement of music signals over time and contain such information as the regularity of the rhythm, the beat, the tempo, and the time signature [8,18]. The feature set for representing rhythm structure is based on detecting the most salient periodicities of the signal. Rhythmic content features are calculated by beat histogram calculation and yield a 6 dimensional feature vector which is represented using BEAT label.

3.3. Pitch Content Features

The melody and harmony information about the music signal is obtained by pitch detection techniques. Although musical genres by no means can be characterized fully by their pitch content, there are certain tendencies that can lead to useful feature vectors [8]. Pitch content features are calculated by pitch histogram calculation and yield a 5 dimensional feature vector which is represented as MPITCH in the experimental results.

The following is a list of audio features we use and their length:

- BEAT (6 features)
- STFT (9 features)
- MFCC (10 features)
- MPITCH (5 features)
- All (30 features)

3.4. Effect of Sample Rate and Size on Genre Classification

When an audio file is compressed under different settings, its features could change. In order to understand this, we used different sample rates (22.050 kHz, 32 kHz, 44.1 kHz), sample sizes (8 bit, 16 bit) to convert the audio file to wav format. As seen in Figure 1, we examined the normalized mean difference between features on all data points using one setting versus another settting. There is some variability on all the features, although features 6 (BEAT), 7, 8 and 10 (STFT) seem to vary more than others.

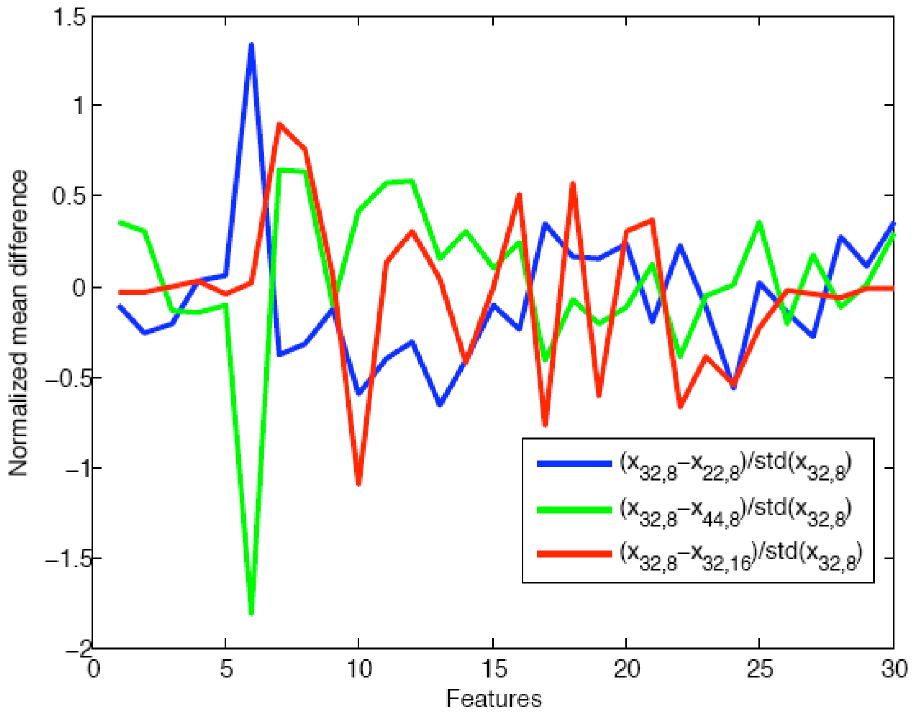

Figure1: The change of audio features when different sample rates and sample sizes are used.

In order to understand the effect of feature changes due to compression settings we trained different classifiers using diffferent feature sets (ALL, BEAT, MFCC, MPITCH, STFT) obtained under different compression settings. In Figures 2 and 3 the x axis shows different audio sampling rates and sizes: 1: 22.05kHz,8bit; 2:22.05kHz,16bit; 3: 32kHz,8bit; 4: 32kHz,16bit; 5: 44.1kHz,8bit, 6:44.1kHz,16 bit. Using ALL features almost always gave better performance than using one of the other specified feature sets. MFCC's performance was very close to that of ALL, though. MPITCH and BEAT usually gave the least classification accuracy. When ALL features were used, we found out that the expected performance did not change a lot between different sample rates and sizes.

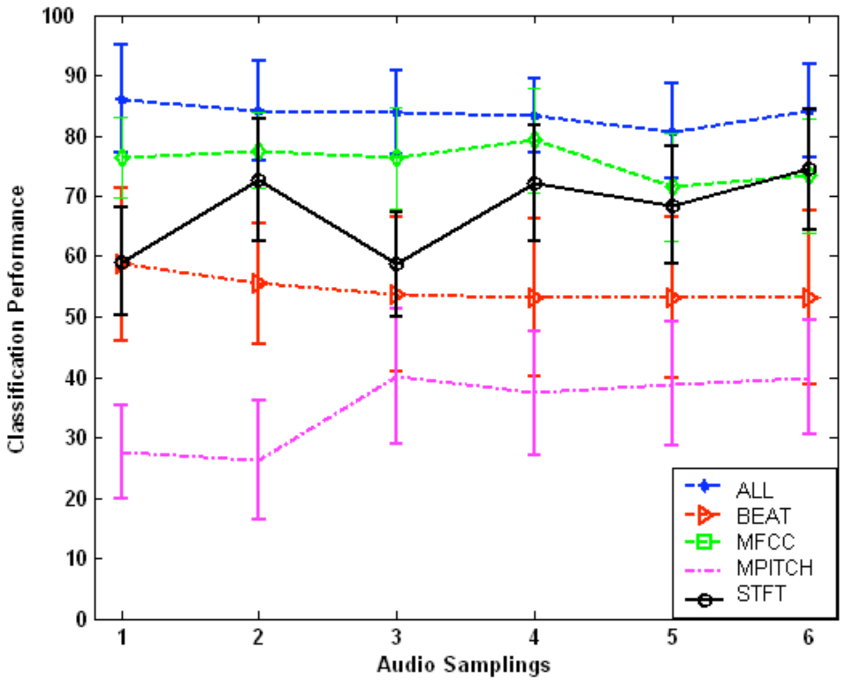

Figure 2: Root Genre classification using LDC classifier and different sets of features.

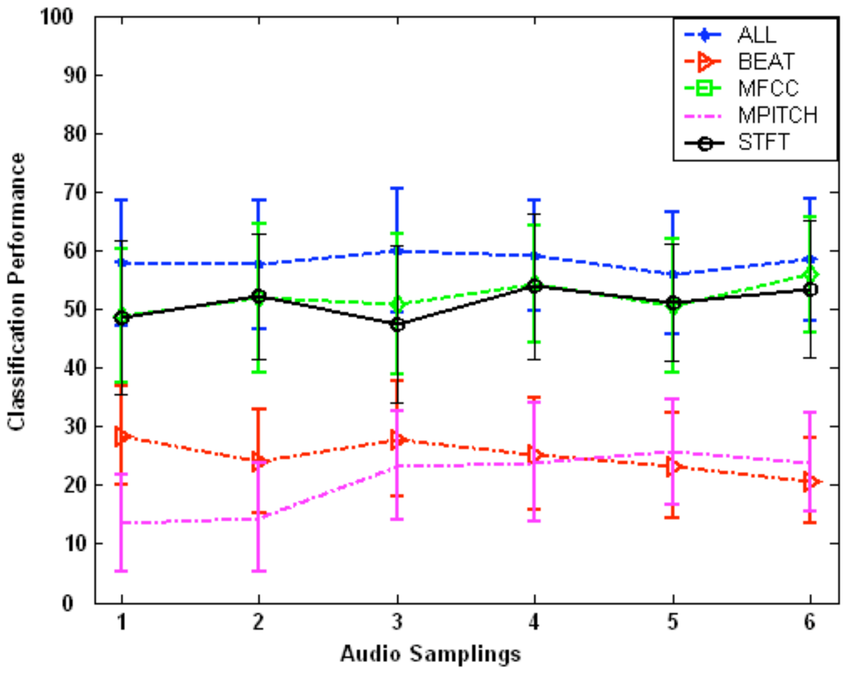

Figure 3: Leaf genre classification with LDC classifier using different sets of features.

Table 1 shows the root and leaf genre classification accuracies obtained using the first and last two (22.05Kbps or 44.1Kpbs and 8 or 16 bits) compression settings. LDC performs better than 10NN for both root and genre classification.

AUDIO	22.05Kbps, 8bits	22.05Kbps, 16bits	44Kbps, 8bits	44Kbps, 16bits
ROOT,10NN	0.52 ± 0.01	0.53 ± 0.01	0.54 ± 0.01	0.58 ± 0.01
ROOT, LDC	0.86 ± 0.01	0.84 ± 0.01	0.83 ± 0.01	0.86 ± 0.01
LEAF,10NN	0.19 ± 0.01	0.20 ± 0.01	0.23 ± 0.01	0.30 ± 0.01
LEAF, LDC	0.59 ± 0.01	0.63 ± 0.01	0.60 ± 0.01	0.63 ± 0.01

Table 1. Root and leaf genre test classification accuracies on audio data obtained from MIDI, using different compression settings and 10NN and LDC classifiers.

4. Genre Classification Using MIDI and NCD

One way to measure distance between two music pieces is to first extract features and then measure distance between feature vectors. For example, McKay [4] uses twenty features of musical information such as orchestration, number of instruments, adjacent fifths etc.. Once distances are available, a classification algorithm, such as k-Nearest Neighbor, can be used to predict genre of a music piece.

In this study, in order to measure the distance between two music pieces, we use Normalized Compression Distance (NCD). According to NCD, two objects are said to be close if the information contained in one of them can be compressed in the other. In other words, if two pieces are similar, then it is possible to describe one given the other. The compression is based on the ideal mathematical notion of Kolmogorov complexity, which unfortunately is not effectively computable. However it is possible to approximate the Kolmogorov complexity by using standard compression techniques. NCD uses no background knowledge about music, it is completely general and can, without change, be used in different areas like linguistic classification and genomics.

Vitanyi and his colleagues [10,11], first process the MIDI representation of a music piece to turn it into a string from a finite alphabet. Then they compute the distance between two music pieces x and y using their Normalized Compression Distance (NCD):

$$d(x,y)=\frac{\max\{K(x\mid y),K(y\mid x)\}}{\max\{K(x),K(y)\}} \tag{1}$$

In this formula K(x) denotes the Kolmogorov Complexity of x and $K(x\mid y)$ denotes the Kolmogorov Complexity of x given y. $K(x\mid y)$ is approximated using $K(x\mid y)\approx K(xy)-K(x)$. NCD uses the compressed length of a string as an approximation of its Kolmogorov Complexity. $K(xy)$ is computed simply as the compressed length of x and y concatenated together. This compressed length approximation to Kolmogorov Complexity seems to have given good results for musical genre and composer clustering and other problems in [12].

In this study, we use our preprocessor [13] on MIDI files to turn them into strings. The MIDI preprocessor samples the MIDI file at each 5 ms. and discovers the notes simultaneously played at each interval. It converts each note played in that interval to an integer between 0 and 127. Since all pieces used in experiments are polyphonic, like in most of the cases in the real world, polyphonic to monophonic conversion is needed. The note which is heard as the highest pitch [19] is taken as the representative of the interval. Then the difference between consecutive monophonic notes is taken and written to a binary file. Apart from Vitanyi and his colleagues work, tempo variations are taken into account and difference between consecutive monophonic notes is taken. Like them we use NCD as the distance metric between two pieces.

Table 2 shows the root and leaf genre classification accuracy of the 10NN classifier together with NCD distances. Distances are computed using the first 30, 60, 120 seconds and finally using all the available music piece. The accuracies shown are computed over 100 different train/test paritions of all the available data. Using the whole piece results in the best root genre classification performance, while using only the first 120 seconds results in the best leaf genre classification performance. Note that, as in the case of previous section, the root and leaf genre classification performances are quite below the results obtained by McKay.

MIDI	30 seconds	60 seconds	120 seconds	All
Root	0.67 ± 0.01	0.66 ± 0.01	0.67±0.01	0.75±0.01
Leaf	0.31±0.01	0.39±0.01	0.46±0.01	0.42±0.01

Table 2. Root and leaf genre test classification accuracies on MIDI data using NCD distance and 10-nearest neighbor classifier.

5. Genre Classification Using Both MIDI and Audio from MIDI

We explored the root and genre classification accuracy using MIDI and audio separately and found out that the accuracy varied between different feature sets and classifiers. However, the accuracies reached were far below the accuracies obtained by McKay. In this section, we investigate if we can get better results by combining MIDI and audio classifiers we obtained in the previous two sections.

According to Kuncheva [20], in order for classifier combination to be successful, classifiers need to be diverse. The probability that many classifiers, trained independently, will agree on the same wrong output is small. Therefore, majority voting could give the right answer for the many, independent and diverse classifiers case.

There are a number of methods to achieve diverse classifiers:

a) Use independent subsamples of data to train each classifier,
b) Use different set of features to train each classifier,
c) Use different algorithms to train each classifier.

Usually a) and b) are more effective than c) [21].

MIDI distances and audio features give us an initial base of different features. We get still more different features by using different initial portions of the MIDI file and different sample rates and sizes for the audio file. The k-nearest neighbor and LDC classifiers also help achieve more diversity. Therefore, we have a pool of different classifiers whose votes we can combine to achieve better accuracy (Figure 4).

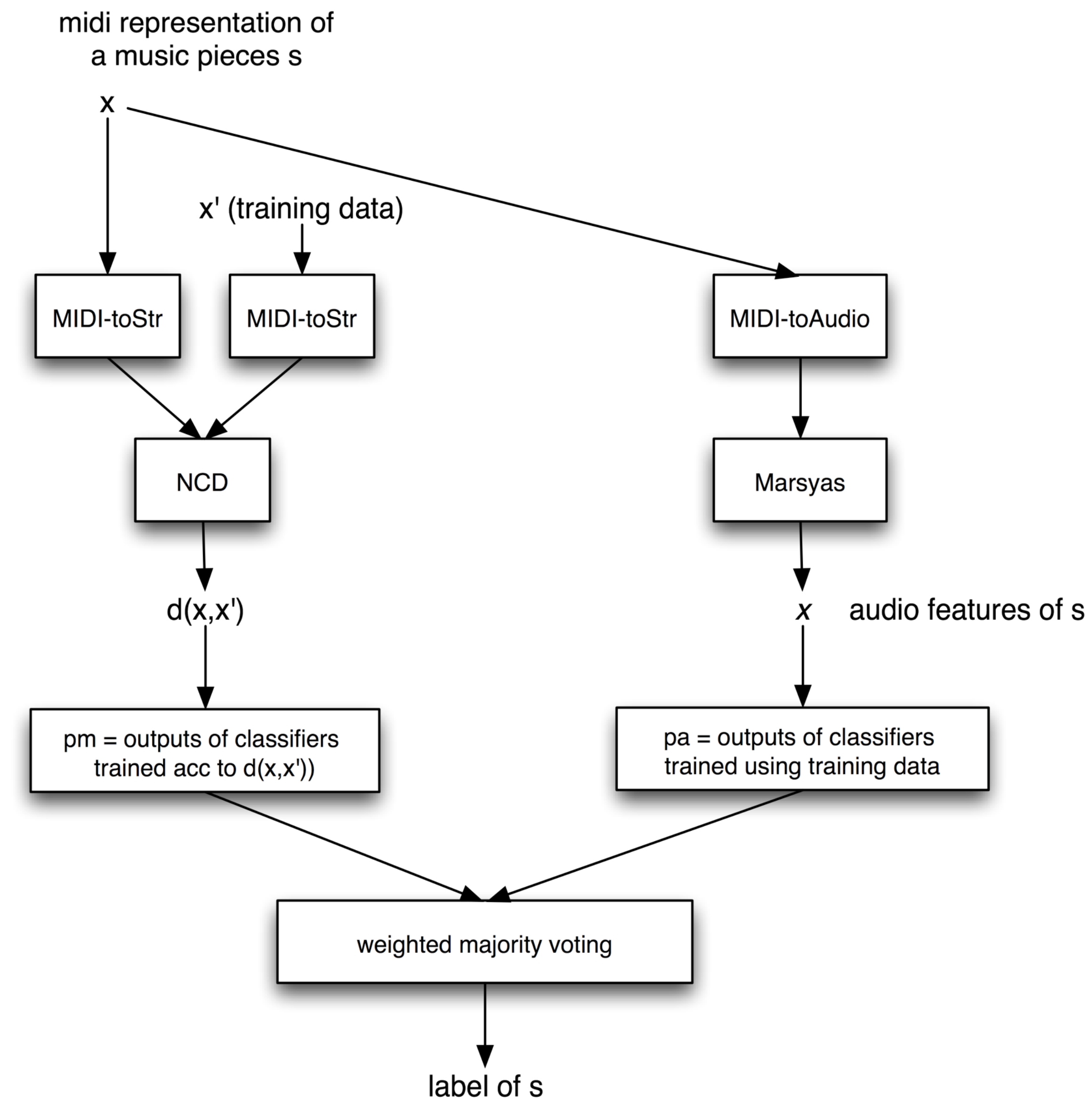
midi representation of a music pieces s
x
x' (training data)
MIDI-toStr
MIDI-toStr
MIDI-toAudio
NCD
Marsyas
d(x,x')
x audio features of s
pm = outputs of classifiers trained acc to d(x,x'))
pa = outputs of classifiers trained using training data
weighted majority voting
label of s

Figure 4. A method to combine MIDI and audio features to predict the genre of a MIDI music piece.

Let D_i, i=1,..,L indicate the different trained classifiers. In this paper, L=12 and i=1:4 corresponds to 10-nearest neighbor classifiers, trained using NCD distance between MIDI files. i=5:8 corresponds 10-nearest neighbor classifiers, trained using all 30 features. i=9:12 corresponds to linear discriminant classifiers trained, again, using all 30 features. Let $s_i(x)$ denote the class label produced for input x by classifier i. Let $d_{i,j}$ be 1 if classifier i labels x in class j and 0 otherwise. Let w_i denote the weight of classifier i. The weighted majority voting chooses class j* such that:

$$j^* = \arg\max_{j=1,..,L} \sum_{i=1,...,C} w_i d_{i,j} \tag{2}$$

We consider four different flavors of weighted majority voting described by the weights w_i given to each classifier:

- $w_i=1$: This voting scheme gives each classifier the same amount of vote.
- $w_i=2$ if $1 \le i \le 4$ or $9 \le i \le 12$ and $w_i=1$ if $5 \le i \le 8$: Inspired by the fact that audio-10NN gives the worst results, this method gives less weight to those classifiers.

- w_i proportial to accuracy of ith classifier: This method depends on the accuracy of each classifier and is not realizable in practice because accuracies are usually not known.
- w_i selected to maximize accuracy: This method exhaustively searches the w_i's in [0.2:1] interval and reports the w_i.that results in the best accuracy.

Table 3 shows the leaf and root genre classification accuracies of each classifier combination method. Comparison of tables 1, 2, and 3 show that root genre classification accuracy increases when classes are combined for all of the combination schemes.

MIDI and AUDIO	w_i=1	i=1-4: w_i=2
		i=5-8: w_i=1
		i=9-12: w_i=2
			$w_i \ \alpha \ acc_i$	w_i optimal

Root	0.88 ± 0.01	0.89 ± 0.01	0.89±0.01	0.93±0.01
Leaf	0.58±0.01	0.58±0.01	0.58±0.01	0.62±0.01

Table 3: Root and leaf genre classification accuracies when classifiers are combined.

Table 4 shows the confusion matrix entries for each of the base classsifiers. The entries are averaged over 100 train/test partitions and normalized to 100 per actual class. Each row corresponds to a classifier with a different feature and classification method. Second column shows whether the MIDI or Audio input is used and the type of classifier used. This column also shows the length of the used piece for MIDI and the sample rate and sample size for audio. Although the accuracies were similar, clearly the confusion matrices are different for each feature-classifier combination and this helped combination achieve better results. Another observation is Classic is recognized best when 30 seconds of MIDI file is used, whereas Pop benefits from longer files. While higher quality (i.e. more kbps and 16 bits) encoding usually helps Classic and Pop, the same is not true for Jazz.

.

No	Feature, Classifier	Actual=Classic			Actual=Jazz			Actual=Pop		
		Pred Class	Pred Jazz	Pred Pop	Pred Class	Pred Jazz	Pred Pop	Pred Class	PredJ azz	Pred Pop
1	MIDI, 30sec, 10NN	89	7	4	14	82	4	45	26	29
2	MIDI,60sec,10NN	69	13	18	6	86	8	25	33	42
3	MIDI,120sec,10NN	70	4	26	8	76	16	20	23	56
4	MIDI,All,10NN	75	4	21	6	84	10	13	22	66
5	AUDIO,22,8,10NN	72	13	15	10	48	41	21	42	37
6	AUDIO,22,16,10NN	71	6	23	12	41	47	19	34	47
7	AUDIO,44,8,10NN	63	15	22	8	53	39	14	39	47
8	AUDIO,44,16,10NN	69	12	19	14	46	40	10	30	60
9	AUDIO,22,8,LDC	94	3	2	6	88	6	12	13	75
10	AUDIO,22,16,LDC	96	0	4	3	87	10	12	20	68
11	AUDIO,44,8,LDC	98	1	2	2	82	16	9	21	70
12	AUDIO,44,16,LDC	97	0	3	4	82	14	4	18	78

Table 4. Root genre confusion matrices for 12 different base classifiers.

Table 5 shows the confusion matrices for the classifier combinations. Using audio and LDC usually gave the best results on Table 4, and Table 5's entries are better than that. Choosing classifier weights according to accuracies did not improve over the equal weighted majority voting. On the other hand, choosing the optimal weights according the specific set of samples being classified resulted in better performance.

	Actual=Classic			Actual=Jazz			Actual=Pop		
Combination method	Pred Class	Pred Jazz	Pred Pop	Pred Class	Pred Jazz	Pred Pop	Pred Class	Pred Jazz	Pred Pop
w_i=1	99	0	1	3	93	4	8	19	72
i=1-4: w_i=2 i=5-8: w_i=1 i=9-12: w_i=2	99	0	1	3	93	4	8	17	76
$w_i \alpha acc_i$	99	0	1	3	93	4	8	17	76
w_i optimal	100	0	0	2	94	3	5	10	86

Table 5. Root genre confusion matrices for four different combinations of base classifiers.

8. Conclusions

In this paper we first classified genres using MIDI files using Normalized Compression Distance (NCD) and 10-nearest neighbor classifier. We converted MIDI files to audio and did genre classification using features at different sample rates and sizes and LDC and KNN classifiers. Finally, we combined 12 different classifiers we obtained at the previous steps, using different schemes of majority voting. We found out that majority voting improved the classification accuracy. Although the classification accuracies for MIDI or audio only were much below the results obtained by McKay [4], classifier combination improved genre classification. The approach outlined does not require any background musical knowledge, hence could be used to improve MIDI genre accuracy.

Currently, the audio to MIDI conversion is not very successful, especially when multiple instruments are used in the piece. We hope that as technology gets better, a similar approach could be used to improve audio genre classification.

Acknowledgements

We would like to express our gratitude to George Tzanetakis and Cory McKay for generously sharing their data sets. We also would like to thank Tzanetakis for Marsyas, Vitanyi and colleagues for Complearn and Bob Duin and friends for PrTools, which was used in the LDC experiments. We thank the reviewers for helping us improve the quality of the paper.

References

1. S. Lippens, J.P Martens, M. Leman, B. Baets, H. Meyer, and G. Tzanetakis, "A comparison of human and automatic musical genre classification," in ICASSP, 2004.

2. R. Basili, A. Serafini, and A. Stellato, "Classification of musical genre: A machine learning approach," in Proceedings of the International Symposium on Music Information Retrieval (ISMIR), 2004.

3. C. Yang, "Efficient acoustic index for music retrieval with various degrees of similarity," in Proc. ACM Multimedia, 2002.

4. C.McKay and I.Fujinaga, “Automatic genre classification using large high-level musical features,” in Proceedings of the International Conference on Music Information Retrieval, 2004, pp. 525–530.

5. G. Tzanetakis, A. Ermolinskyi, and P. Cook, “Pitch histograms in symbolic and audio music information retrieval,” Journal of New Music Research, vol. 32, no. 2, pp. 143–152, 2003.

6. J. J. Aucouturier and F. Pachet, “Representing musical genre:a state of the art,” Journal of New Music Research, vol. 32, no. 1, pp. 83–93, 2003.

7. C. C. Liu and C.S. Huang, “A singer identification technique for content-based classification of mp3 music ob jects,” in Proceedings of the eleventh international conference on Information and knowledge management, Virginia, 2002, pp. 438–445.

8. G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” IEEE Transactions on Speech and Audio Processing, vol. 10, no. 5, pp. 293–302, 2002.

9. M. F. Kinney and J. Breebaart, “Features for audio and music classification,” in International Symposium on Music Information Retrieval (ISMIR), 2003.

10. R. Cilibrasi, P.M.B. Vitanyi, and R.Wolf, “Algorithmic clustering of music based on string compression,” Computer Music Journal, vol. 28, no. 4, pp. 49–67, 2004.

11. M. Li, X. Chen, X. Li, B. Ma, and P.M.B. Vitanyi, “The similarity metric,” IEEE Transactions on Information Theory, vol. 50, no. 12, pp. 3250– 3264, 2004.

12. E. Keogh, S. Lonardi, and C.A. Rtanamahatana, “Toward parameter free data mining,” in Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining Seattle, WA, August 2004, pp. 206–215.

13. Z. Cataltepe, A. Sonmez, and E. Adali, “Music classification using kolmogorov distance,” in Representation in Music/Musical Representation Congress, October 2005.

14. C. Xu, N.C. Maddage, X. Shao, F. Cao, and Q. Tian, “Musical genre classification using support vector machines,” in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), April 2003, vol. 5, pp. 429–432.

15. T. Li, M. Ogihara, and Q. Li, “A comparative study on content-based music genre classification,” in ACM SIGIR, 2003.

16. D. Turnbull and C. Elkan, “Fast recognition of musical genres using rbf networks,” IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 4, pp. 580–584, 2005.

17. Richard O. Duda, Peter E. Hart, and David G. Stork, Pattern Classification, Wiley, 2000.

18. T. Li and G. Tzanetakis, “Factors in automatic musical genre classification of audio signals,” in IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2003.

19. L. Uitdenbogerd and J. Zobel, “Music ranking techniques evaluated,” Australian Computer Science Communications, vol. 24, no. 1, January-February 2002.

20. L.I.Kuncheva, Combining Pattern Classifiers, Wiley, 2004.

21. R. Duin, “Prtools Seminar at Sabanci University, Turkey,” June 2005.