Aggregate features and ADABOOST for music classification

Article *in* Machine Learning · December 2006

DOI: 10.1007/s10994-006-9019-7 · Source: DBLP

James Bergstra
Harvard University
27 PUBLICATIONS **24,763** CITATIONS

Norman Casagrande
Google Inc.
20 PUBLICATIONS **2,082** CITATIONS

Dumitru Erhan
Google Inc.
58 PUBLICATIONS **156,808** CITATIONS

Balázs Kégl
French National Centre for Scientific Research
174 PUBLICATIONS **15,573** CITATIONS

Mach Learn
DOI 10.1007/s10994-006-9019-7

Aggregate features and AdaBoost for music classification

**James Bergstra · Norman Casagrande ·
Dumitru Erhan · Douglas Eck · Balázs Kégl**

Received: 11 October 2005 / Revised: 29 April 2006 / Accepted: 2 May 2006 / Published online: 30 June 2006
Springer Science + Business Media, LLC 2006

Abstract We present an algorithm that predicts musical genre and artist from an audio waveform. Our method uses the ensemble learner AdaBoost to select from a set of audio features that have been extracted from segmented audio and then aggregated. Our classifier proved to be the most effective method for genre classification at the recent MIREX 2005 international contests in music information extraction, and the second-best method for recognizing artists. This paper describes our method in detail, from feature extraction to song classification, and presents an evaluation of our method on three genre databases and two artist-recognition databases. Furthermore, we present evidence collected from a variety of popular features and classifiers that the technique of classifying features aggregated over segments of audio is better than classifying either entire songs or individual short-timescale features.

Keywords Genre classification · Artist recognition · Audio feature aggregation · Multiclass AdaBoost · MIREX

Editor: Gerhard Widmer

J. Bergstra (✉) · N. Casagrande · D. Erhan · D. Eck · B. Kégl
Department of Computer Science, University of Montreal, Montreal Quebec H3C 3J7, Canada
e-mail: bergstrj@iro.umontreal.ca

N. Casagrande
e-mail: casagran@iro.umontreal.ca

D. Erhan
e-mail: erhandum@iro.umontreal.ca

D. Eck
e-mail: eckdoug@iro.umontreal.ca

B. Kégl
e-mail: kegl@iro.umontreal.ca

1. Introduction

Personal computers and portable digital music players are increasingly popular platforms for storing and playing music. (Apple alone has sold more than 42 million iPod players and more than a billion songs from their iTunes Store.) As personal music collections continue to grow, the task of organizing and selecting music becomes more challenging. This motivates research in automatic methods for search, classification, and recommendation.

We deal here with one aspect of this challenge, the prediction of the genre or artist of a song based on features extracted from its audio waveform. This task is challenging for at least two reasons. First, the raw musical signal is very high dimensional. In CD quality audio, a single monophonic waveform contains 44100 values per second. Thus a stereo 3 minute song contains nearly 16 million samples. More compact acoustic features can be extracted from the waveform. However, many of these are computed over short frames of audio, thus reducing the degree of compression achieved. Second, music databases can be very large. The website `www.itunesregistry.com` recently reported an average iTunes collection size of 3,542 songs. Commercial databases for online music services can contain 500,000 or more (personal communication, Benjamin Masse, president `radiolibre.ca`). To be useful for real world applications, a classifier must scale to large datasets.

In this paper we present a genre and artist classifier that addresses these issues. We use the ensemble learner ADABOOST, which performs large-margin classification by iteratively combining the weighted votes of a collection of weak learners. This approach has two advantages in the context of the challenges we face. First, our model uses simple decision stumps, each of which operates on a single feature dimension. Thus our model performs feature selection in parallel with classification. Although a feature set with redundant or non-informative features will slow down computation, those features will be left behind by ADABOOST as it iteratively selects informative features. Second, ADABOOST scales linearly with the number of training points provided. If we limit the number of learning iterations, our model has the potential to deal with very large datasets. This compares favorably with other state-of-the-art large margin classifiers such as SVMs, which have an overall computational complexity that is quadratic in the number of data points.

This paper makes two contributions. First we provide experimental evidence that our algorithm is competitive with other state-of-the-art classifiers for genre and artist recognition. Second we explore the issue of feature extraction, focusing on the challenge of aggregating frame-level acoustic features into a form suitable for classification. We analyze experiments on several classifiers using different aggregation segment sizes. The results support the general claim that aggregation is a useful strategy, and suggest reasonable limits on segment size.

The paper is structured as follows: in Section 2 we discuss previous attempts at music genre and artist recognition, focusing separately on feature extraction, feature aggregation and classification. In Section 3 we describe our ADABOOST model and aggregation technique. In Section 4 we provide experimental results for our model. This section includes a discussion of the results of the `MIREX` (Music Information Retrieval Evaluation eXchange) 2005 genre prediction contest and artist recognition contests where our algorithms won first and second prizes, respectively. In Section 5 we present the experimental results from our investigation of segment length for feature aggregation. Finally in Section 6 we offer concluding remarks and discuss future work.

2. Genre and artist classification

In the following section we treat genre and artist classification as a three-step process. While we do not provide a survey, we note that much previous work in music classification fits in this framework and describe it below. The first step is the extraction of acoustic features from short frames of audio data that capture information such as timbre and rhythm. The second step is the aggregation of frame-level features into more compressed segment-level features. The third step is the prediction of genre or artist for a song using these compressed features as input to a classifier.

2.1. Feature extraction and aggregation

To our knowledge there is no accepted theory of which features are best for music classification tasks such as genre and artist recognition. Many different methods have been tried, including Fourier analysis and related measures such as cepstral analysis (yielding MFCCs). In addition to the family of Fourier methods, results have been reported for wavelet analysis (Lambrou et al., 1998), autoregression (Ahrendt and Meng, 2005) and the collection of statistics such as zero-crossing rate, spectral centroid, spectral roll-off and spectral spread. A survey by Aucouturier and Pachet (2003) describes a number of popular features for music similarity and classification, and research continues (e.g. Bello et al. (2005), Pampalk et al. (2005)). In Section 3 we describe the specific features we use in our model.

In order to capture relatively fine-timescale structure such as the timbre of specific instruments, features are often extracted from short ($\sim$ 50ms) frames of data. Our goal, however, is to predict something about an entire song which encompasses many frames ($\sim$ 20 per second in our implementation). This raises the question of how to use this frame level information.

One option is to compress these frame-level features into a *single* set of song-level features. Tzanetakis and Cook (2002) and Li et al. (2003) fit individual Gaussians to each feature (diagonal covariance among Gaussians). More recently, (Mandel and Ellis, 2005a) generated song-level features using Gaussian densities with full-covariance. Gaussian mixtures have also been used to generate song-level features by, e.g., Aucouturier and Pachet (2002) and Pampalk et al. (2005).

On the other end of the spectrum it is also possible to first classify directly the frame-level features themselves, and then to combine the individual classifiers into song labels by a majority vote. Xu et al. (2003), for example, reports good genre classification results on a small dataset using this approach. West and Cox (2004), however, note a significant improvement in performance when they use a “memory” feature, which is the mean and variance of frame-level features over the previous second.

A third option is to aggregate frame-level features over an audio segment that is longer than a single frame but shorter than an entire song. As in the second approach, individual segment classifications are combined to make a decision about an entire song. This technique was described in Tzanetakis and Cook (2002), who summarized a one-second ‘texture window’ with feature means and variances before performing classification. In a similar work, Ahrendt and Meng (2005) used an auto-regressive model in place of Gaussians. West and Cox (2005) used an onset detection algorithm to partition the song into segments that correspond to single notes.

While promising results have been reported for a wide range of segment sizes (all the way from the frame to the song), no one to our knowledge has undertaken a systematic exploration of the effects of segment size on classification error. In Section 5 we investigate this question by training several algorithms using the same dataset. For all experiments we use the popular technique of fitting independent Gaussians (diagonal covariance) to each feature in a segment. We then vary segmentation size and analyze results for evidence that certain segment lengths work better than others.

2.2. Classification

A wide range of algorithms have been applied to music classification tasks. These include minimum distance and K-nearest neighbor in Lambrou et al. (1998), and Logan and Salomon (2001). Tzanetakis and Cook (2002) used Gaussian mixtures. West and Cox (2004) classify individual frames by Gaussian mixtures, Linear Discriminant Analysis (LDA), and regression trees. Ahrendt and Meng (2005) classify 1.2s segments using multiclass logistic regression.

Several classifiers have been built around Support Vector Machines (SVMs). Li et al. (2003) reported improved performance on the same dataset as Tzanetakis and Cook (2002) using both SVM and LDA. Mandel and Ellis (2005a) used an SVM with a kernel based on the symmetric KL divergence between songs. Their model performed particularly well at MIREX 2005, winning the Artist Recognition contest and performing well in the Genre Recognition contest as well. While SVMs are known to perform very well on small data sets, the quadratic training time makes it difficult to apply them on large music databases. This motivates research on applying equally well-performing but more time efficient algorithms to music classification.

3. Algorithm

In this section we describe a genre and artist classifier based on multiclass AdaBoost. In keeping with the organization of Section 2, we organize this section around feature extraction, feature aggregation, and classification.

3.1. Acoustic feature extraction and aggregation

For all experiments, we break an audio waveform into short frames (46.44ms, or 1024 samples of audio at 22050Hz). The feature sets for experiments presented in this work are drawn from the following list.[1] Those unfamiliar with standard methods of audio signal processing may refer to Kunt (1986) for background.

- *Fast Fourier transform coefficients (FFTCs).* Fourier analysis is used to analyze the spectral characteristics of each frame of data. In general we computed a 512-point Fourier transform $\mathcal{F}(\mathbf{s})$ of each 1024-point frame $\mathbf{s}$. The 32 lowest frequency points were retained for our experiments.
- *Real cepstral coefficients (RCEPS).* Cepstral analysis is commonly used in speech recognition to separate vocal excitation from the effects of the vocal tract. See Gold and Morgan (2000) for an overview. RCEPS is defined as $\text{real}(\mathcal{F}'(\log(|\mathcal{F}(\mathbf{s})|)))$ where $\mathcal{F}$ is the Fourier transform and $\mathcal{F}'$ is the inverse Fourier transform.
- *Mel-frequency cepstral coefficients (MFCC).* These features are similar to RCEPS, except that the input x is first projected according to the Mel-scale Junqua and Haton (1996), a psycho-acoustic frequency scale on which a change of 1 unit carries the same perceptual significance, regardless of the position on the scale.

[1] Our feature extractor is available as a **C** program from the first author's website: `http://www-etud.iro.umontreal.ca/~bergstrj`.

- *Zero-crossing rate (ZCR).* The zero-crossing rate (ZCR) is the rate of sign-changes along the signal. In a signal with a single pitched instrument, the ZCR is correlated with the dominant frequency (Kedem, 1986).
- *Spectral spread, spectral centroid, spectral rolloff* Spectral spread and spectral centroid are measures of how power is distributed over frequency. Spectral spread is the variance of this distribution. The spectral centroid is the center of mass of this distribution and is thus positively correlated with brightness. Spectral rolloff feature is the a-quantile of the total energy in $|\mathcal{F}_{\mathbf{s}}|$.
- *Autoregression coefficients (LPC).* The k linear predictive coefficients (LPC) of a signal $\mathbf{s}$ are the product of an autoregressive compression of the spectral envelope of a signal. These coefficients can be computed efficiently from the signal's autocorrelation by the Levinson-Durbin recursion (Makhoul, 1975).

After computing these frame-level features, we group non-overlapping blocks of m consecutive frames into segments. We summarize each segment by fitting independent Gaussians to the features (ignoring covariance between different features). The resulting means and variances are the input to weak learners in ADABOOST.

3.2. Classification with ADABOOST

After extracting the segment-level features, we classified each segment independently using ADABOOST. The training set was created by labeling each segment according to the song it came from. ADABOOST Freund and Schapire (1997) is an *ensemble* (or *meta-learning*) method that constructs a classifier in an iterative fashion. It was originally designed for binary classification, and it was later extended to multi-class classification using several different strategies. In this application we decided to use ADABOOST.MH Schapire and Singer (1998) due to its simplicity and flexibility.

In each iteration t, the algorithm calls a simple learning algorithm (the *weak learner*) that returns a classifier $\mathbf{h}^{(t)}$ and computes its coefficient $\alpha^{(t)}$. The input of the weak classifier is a d-dimensional observation vector $\mathbf{x} \in \mathbb{R}^d$ containing the features described in Section 2.1, and the output of $\mathbf{h}^{(t)}$ is binary vector $\mathbf{y} \in \{-1, 1\}^k$ over the k classes. If $h_\ell^{(t)} = 1$ the weak classifier "votes for" class ℓ whereas $h_\ell^{(t)} = -1$ means that it "votes against" class ℓ. After T iterations, the algorithm outputs a vector-valued discriminant function

$$\mathbf{g}(\mathbf{x}) = \sum_{t=1}^{T} \alpha^{(t)} \mathbf{h}^{(t)}(\mathbf{x}).$$

To obtain a single label, we take the class that receives the "most vote", that is, $f(\mathbf{x}) = \arg\max_\ell g_\ell(\mathbf{x})$.

When no a-priori knowledge is available for the problem domain, small decision trees or, in the extreme case, *decision stumps* (decision trees with two leaves) are often used as weak classifiers. Assuming that the feature vector values are ordered beforehand, the cost of the weak learning is $O(nkd)$, so the whole algorithm runs in $O(nd(kT + \log n))$ time. In the rest of the paper, we will refer to this version of ADABOOST.MH as AB.STUMP. We also experimented with small decision trees as weak learners, and we refer to this version of ADABOOST.MH as AB.TREE.[2]

[2] Our implementation of ADABOOST.MH is available as a C++ program from `http://sourceforge.net/projects/multiboost`.

Table 1 Summary of databases used in our experiments

Database	Magnatune		USPOP		Tzanetakis
Type	Genre	Artist	Genre	Artist	Genre
Number of training files	1005	1158	940	1158	800
Number of test files	510	642	474	653	200
Number of classes	10	77	6	77	10
Average song length		$\sim$ 200s			30s

Note that if each weak classifier depends on only one feature (as a decision stump does) and the number of iterations T is much less than the number of features d, then AdaBoost.MH acts as an implicit feature extractor that selects the T most relevant features to the classification problem. Even if $T > d$, one can use the coefficients $\alpha^{(t)}$ to order the features by their relevance.

4. Experiment A—MIREX 2005

In this section we present results from our entry in the 2005 MIREX (Music Information Retrieval Evaluation eXchange) contest. MIREX is an annual series of contests whose main goal is to present and compare state-of-the-art algorithms from the music information retrieval community. It is organized in parallel with the ISMIR conference. We participated in two contests: Audio Genre Classification Bergstra et al. (2005a), and Audio Artist Identification Bergstra et al. (2005b). Both contests were straightforward classification tasks. Two datasets were used to evaluate the submissions: Magnatune[3], and USPOP[4]. Both databases contain mp3 files of commercially produced full-length songs. The Magnatune database has a hierarchical genre taxonomy with 10 classes at the most detailed level (ambient, blues, classical, electronic, ethnic, folk, jazz, new age, punk, rock), whereas the USPOP database has 6 genres (country, electronic and dance, new age, rap and hip hop, reggae, rock). Each of the datasets has a total of 77 artists. Table 1 summarizes the parameters of the experimental setup.

Competition rules demanded that every algorithm must train a model and make a class prediction for the elements of the test set within 24 hours. Due to the large number of contest submissions, contest organizers could not perform a K-fold cross validation of results. Only a single train/test run was performed for each <entry, database> pair.

4.1. Parameter tuning

The 24 hour limit on the training time required us to take extra care in setting the hyper-parameters of the algorithm: the number of boosting iterations T and the number of frames per segment m. Since the contest databases Magnatune and USPOP were kept secret, we tuned our algorithm using a database of song excerpts, furnished to us by George Tzanetakis. This is the same dataset used in Tzanetakis et al. (2002), Tzanetakis and Cook (2002), and Li and Tzanetakis (2003). This database has a total of 1000 30-second song openings. Each excerpt is labeled as one of ten genres (blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock). To our ears, the examples are well-labeled, and exhibit a wide range of styles and instrumentation within each genre. Although the artist names are not associated

[3] www.magnatune.com

[4] www.ee.columbia.edu/~dpwe/research/musicsim/uspop2002.html

with the songs, our impression from listening to the music is that no artist appears twice. Thus, the so-called *producer effect*, observed in Pampalk et al. (2005) is not a concern for these parameter tuning experiments.[5]

To tune the hyper-parameters we extracted frame-level features (256 RCEPS, 64 MFCC, 32 LPC, the lowest 32 of 512 Fourier coefficients, 16 spectral rolloff, 1 LPC error, and the zero-crossing rate) from the Tzanetakis database, and estimated parameter performance by 5-fold cross-validation. This gave us 800 training points, slightly lower than in the MIREX contests. More importantly, the length of a song excerpt in the Tzanetakis database is 30s, much shorter than the average length of a commercially produced song which we estimated to be $\sim$ 200s. This means that using the same segment length m and number of iterations T, 24 hours of training on the Magnatune and USPOP databases would be roughly equivalent to 3 hours of training on the Tzanetakis database. Thus, we set the time limit in our tuning experiments to 2 hours.

Although ADABOOST is relatively resistant to overfitting, the test error may increase after a large number of iterations. To avoid overfitting, we can validate T using data held-out from the training set. In our trials, we did not observe any overfitting so we decided not to validate T. Instead we let the algorithm run the longest possible within the time limit. In fact, given the time limit, underfitting was a more important concern. Since the number of training examples n (and so the training time) is inversely proportional to the segment length m, selecting a small m might prevent our algorithm from converging within 24 hours. In Section 5 we present an experimental study in which we determined that the optimal segment length is between 50 and 100 frames, if ADABOOST is permitted to converge. Based on our performance on the Tzanetakis database, however, we estimated that setting $m = 100$ could result in our algorithm being terminated before converging. Thus, to avoid underfitting, in the contest submission we set $m = 300$ which corresponds to segments of length 13.9s. Since the Tzanetakis database did not contain different song excerpts from the same authors, we did not tune the algorithm separately for the artist recognition contest: our submissions in the two contests were identical.

4.2. Results

The overall performance of each entry was calculated by averaging the raw classification accuracy on USPOP with a hierarchical classification accuracy on Magnatune.[6] Table 2 and Table 3 summarize the contest results.[7]

Our submissions based on AB.STUMP and AB.TREE ranked first and second on both genre tasks. On the artist databases, our algorithms placed first and third, and second and third.

Note that all the tests were done on single folds (not cross-validation) and there were only around 500 test files in each database, so the significance of the differences is difficult to assess. However we can assert that in trials before the contest, our model obtained a classification rate of 83% on the Tzanetakis database. This compares favorably with the best published classification rate on the Tzanetakis database, of 71% obtained by Li et al. (2003).

[5] The producer effect comes from the fact that songs from the same album tend to look the same through the lens of popular feature-extractors. If an album is split between training and testing sets, then we can expect artificially high test-set performance.

[6] More details regarding the evaluation procedure can be found at `http://www.music-ir.org/mirex2005/index.php/{Audio_Genre_Classification,Audio_Artist_Identification}`.

[7] More detailed results, including class confusion matrices and brief descriptions of each algorithm, can be found at `http://www.music-ir.org/evaluation/mirex-results/audio-{genre,artist}/index.html`.

Table 2 Summarized results for the genre recognition contest at MIREX 2005

Rank	Participant	Overall	Magnatune	USPOP
1	AB.TREE	**82.34**%	**75.10**%	**86.92**%
2	AB.STUMP	81.77%	74.71%	86.29%
3	Mandel & Ellis	78.81%	67.65%	85.65%
4	West, K.	75.29%	68.43%	78.90%
5	Lidy & Rauber [1]	75.27%	67.65%	79.75%
6	Pampalk, E.	75.14%	66.47%	80.38%
7	Lidy & Rauber [2]	74.78%	67.65%	78.48%
8	Lidy & Rauber [3]	74.58%	67.25%	78.27%
9	Scaringella, N.	73.11%	66.14%	75.74%
10	Ahrendt, P.	71.55%	60.98%	78.48%
11	Burred, J.	62.63%	54.12%	66.03%
12	Soares, V.	60.98%	49.41%	66.67%
13	Tzanetakis, G.	60.72%	55.49%	63.29%

Table 3 Summarized results for the artist identification contest at MIREX 2005

Rank	Participant	Mean performance	Magnatune	USPOP
1	Mandel & Ellis	**72.45**%	76.60%	**68.30**%
2	AB.STUMP	68.57%	**77.26**%	59.88%
3	AB.TREE	66.71%	74.45%	58.96%
4	Pampalk, E.	61.28%	66.36%	56.20%
5	West & Lamere	47.24%	53.43%	41.04%
6	Tzanetakis, G.	42.05%	55.45%	28.64%
7	Logan, B.	25.95%	37.07%	14.83%

5. Experiment B—Choosing segment length

In this section we investigate the effect of segment length on classification error. Selecting the correct segment length m is crucial both for achieving the highest possible classification accuracy and for calibrating training time. In the experiments described below we examine how the segment length affects classification accuracy given unlimited training time. Based on our experiments we conclude that for several popular genre prediction algorithms, the optimal segment length is around 3.5 seconds, with good values ranging from approximately 2 to 5 seconds.

To pursue this question, we tested four classifiers on the Tzanetakis database: AB.STUMP, AB.TREE, sigmoidal neural network (ANN), and SVM with Gaussian kernel. The ANN Bishop (1995) was a feed-forward network of 64 hidden nodes, fit by gradient descent. An implicit $L2$-norm weight-decay was introduced by initializing the network parameters to small values, and performing early-stopping using a validation set. No explicit regularization was applied during optimization. Classification was performed by a bagged ensemble of 10 networks, as described in Breiman (1996). To train our SVM Cortes and Vapnik (1995), we optimized the width of the kernel first—using a relatively low value of the soft-margin parameter—and then optimized the soft-margin parameter C, while fixing the optimal width. Both of the optimizations were done on a validation set and the optimal set of hyper-parameters was selected by 5-fold cross-validation. Both versions of ADABOOST.MH were trained for 2500 iterations, which sufficed to reach a plateau in performance.

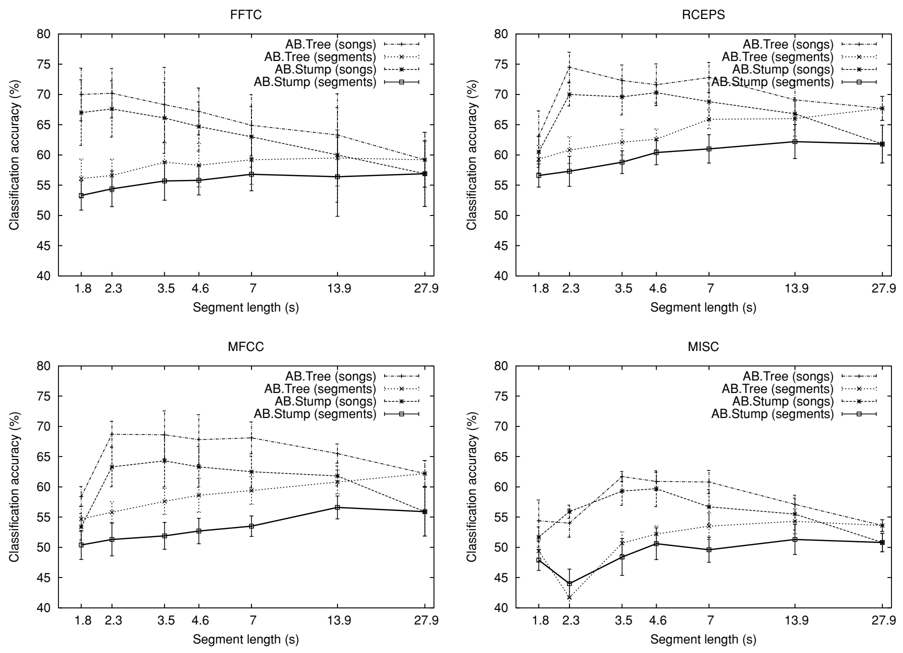

Fig. 1 The classification accuracy on four different feature sets trained with AdaBoost.MH using segment lengths $m \in \{40, 50, 75, 100, 150, 300, 600\}$

To evaluate the robustness of the segmentation procedure across different feature types, we ran separate experiments with four different feature sets described in Section 2: 64 MFCC, 128 RCEPS, 128 FFTC, and 19 MISC (including 1 ZCR, 1 Spectral Centroid, 1 Spectral Spread, and 16 Rolloff). Each of the 16 <feature set,classifier> pairs was evaluated using $m \in \{40, 50, 75, 100, 150, 300, 600\}$ frames/segment (corresponding to 1.8s, 2.3s, 3.5s, 4.6s, 7.0s, 13.9s, and 27.9s, respectively). All tests were done using 5-fold cross-validation. For all tests, we split the dataset at the song level to ensure that the segments from a single song were put into either the training or test set.

Figures 1 and 2 show the results. In all experiments, the general trend was that the segment classification rate rose monotonically with the segment length, and the whole-song classification rose and then fell. Although the fold variance makes it difficult to identify a clear peak, it appears that whole-song performance is optimal at around 3.5s segments across feature types and classification algorithms, except for the SVM that seems to prefer slightly longer segments. In the case of the neural network, we do not see the same degradation in segment-level performance as with AdaBoost, but we see the same rise and fall of the whole-song performance. The performance of the SVM is comparable to that of AdaBoost and the neural net when we use MFCC and RCEPS features, but it is much worse on the FFTC and MISC features. Note that it was impractical to obtain results using the 2.3- and 1.8-second segments with the SVM due to the training time which is quadratic in the number of training points. Figure 3 summarizes the results for every feature and algorithm on an entire song and on a segment size of 3.5 seconds.

The general rise and fall of the song-level classification can be explained in terms of two conflicting effects. On one hand, by partitioning songs in to more pieces, we enlarge our training set. This is universally good for statistical learning, and indeed, we see the benefits in

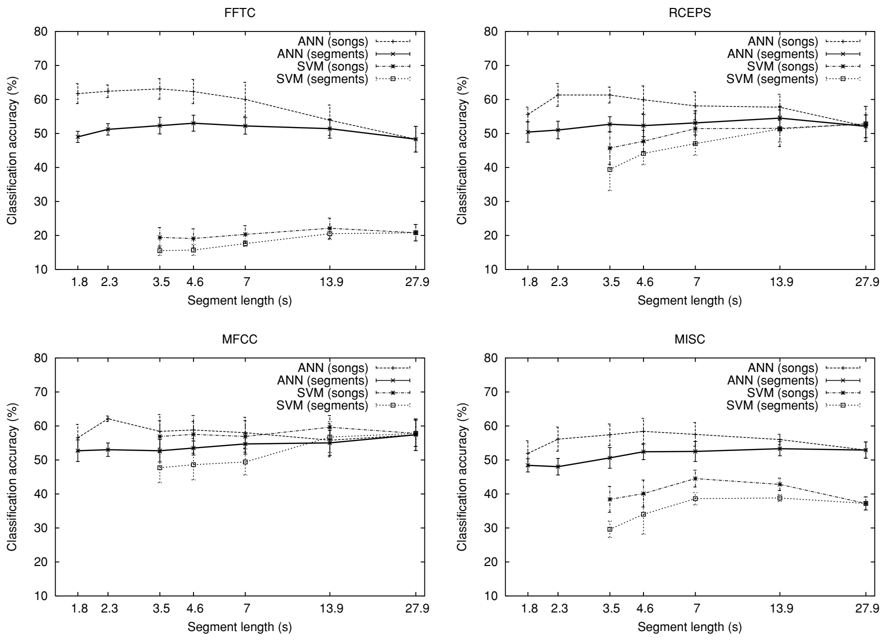

Fig. 2 The classification accuracy on four different feature sets trained with ANN and SVM using segment lengths $m \in \{40, 50, 75, 100, 150, 300, 600\}$

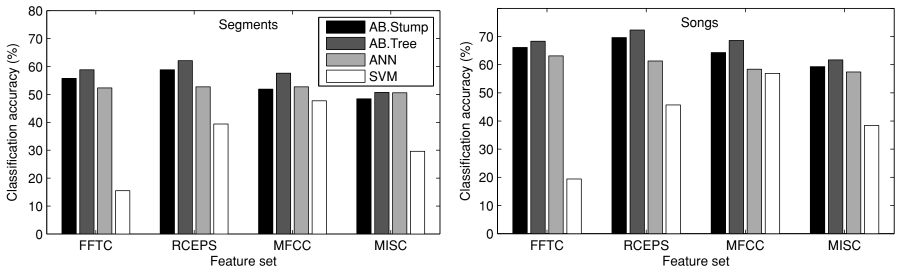

Fig. 3 The classification accuracy on four feature types and four classification algorithms using 3.5s segments

each classifier. On the other hand, smaller segments mean that our aggregate features (frame-level sample-means and sample-variances) have higher variance. For small-enough segments, the value of aggregating frame-level features is null, and previous work, particularly West and Cox (2004), supports the general claim that some amount of aggregation is necessary for good performance.

6. Conclusions and future work

In this work, we have presented a supervised learning algorithm for classifying recorded music. We have demonstrated with results on three databases that this algorithm is effective in genre prediction; to our knowledge, at the time of writing these results are the highest genre prediction rates published for each of Tzanetakis' database, Magnatune, and for the

USPOP database. We attribute this effectiveness to our particular strategy for aggregating frame-level, and to the use of ADABOOST to produce a classifier.

Recent work by Lippens et al. (2004) and an earlier work by Soltau (1997) (described in Aucouturier and Pachet (2003)) suggest standards to which we may hold these results. Lippens et al. (2004) conducted a study in which people were asked to do genre-recognition on a small dataset involving six quite different genres. It was found that on average, people were correct only around 90% of the time. In a Turing-test conducted by Soltau (1997), he finds that his automatic music classifier is similar in performance to 37 subjects who he trained to distinguish rock from pop using the same training and test sets. Given the steady and significant improvement in classification performance since 1997, we wonder if automatic methods are not already more efficient at learning genres than some people.

Future work would explore the value of segmentation when additional frame-level features act on longer frames to compute rhythmic fingerprints and short-term timbre dynamics. This includes evaluating features extracted from the beat histogram Tzanetakis and Cook (2002) and the autocorrelation phase matrix Eck and Casagrande (2005). We could also enrich the label set. Although in our experiments we considered a single label for each observation (each song belongs to one and only one category), it is more realistic to use hierarchical, or generally overlapping class labels for labeling music by its style. ADABOOST.MH extends naturally to *multi-label* classification by allowing the label vectors to contain more than one positive entry.

Music classification algorithms are approaching a level of maturity where they can aid in the hand labeling of data, and can verify large labeled collections automatically. However, to be useful in a commercial setting, these algorithms must run quickly and be able to learn from training sets that are one, two, even three orders of magnitude larger than the ones dealt with in our experiments. To this end, it would be useful to establish large databases of music that can be legally shared among researchers.

Acknowledgements Thanks to Alexandre Lacoste for many interesting ideas in conversation. Thanks also to George Tzanetakis for providing the genre classification dataset. This research is supported by grants from the Quebec Fund for Research in Nature and Technology (FQRNT) as and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Ahrendt, P., & Meng, A. (2005). Music Genre Classification using the multivariate AR feature integration model. Extended Abstract. MIREX genre classification contest (www.music-ir.org/evaluation/mirex-results).

Aucouturier, J., & Pachet, F. (2002). Music Similarity Measures: Whats the Use?. In: Fingerhut, M. (ed.): *Proceedings of the Third International Conference on Music Information Retrieval (ISMIR 2000).*

Aucouturier, J., & Pachet, F. (2003). Representing musical genre: A state of the art. *Journal of New Music Research 32*(1), 1–12.

Bello, J., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. (2005). A Tutorial on Onset Detection in Music Signals. *IEEE Transactions on Speech and Audio Processing*.

Bergstra, J., Casagrande, N., & Eck, D. (2005a). Artist Recognition: A Timbre- and Rhythm-Based Multiresolution Approach. MIREX artist recognition contest.

Bergstra, J., Casagrande, N., & Eck, D. (2005b). Genre Classification: A Timbre- and Rhythm-Based Multiresolution Approach. MIREX genre classification contest.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Breiman, L. (1996). Bagging Predictors. *Machine Learning 24*(2), 123–140.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning 20*(3), 273–297.

Crawford, T., & Sandler, M. (eds.) (2005). Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005).

Eck, D., & Casagrande, N. (2005). Finding Meter in Music Using an Autocorrelation Phase Matrix and Shannon Entropy. In: *Proc. 6th International Conference on Music Information Retrieval (ISMIR 2005).*

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences 55*(1), 119–139.

Gold, B., & Morgan, N. (2000). *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. Wiley.

Junqua, J., & Haton, J. (1996). *Robustness in Automatic Speech Recognition*. Boston: Kluwer Academic.

Kedem, B. (1986). Spectral analysis and discrimination by zero-crossings. *Proc. IEEE 74*(11), 1477–1493.

Kunt, M. (1986). *Digital Signal Processing*. Artech House.

Lambrou, T., Kudumakis, P., Speller, R., Sandler, M., & Linney, A. (1998). Classification of audio signals using statistical features on time and wavelet tranform domains. In: *Proc. Int. Conf. Acoustic, Speech, and Signal Processing (ICASSP-98)*, *6*, 3621–3624.

Lippens, S., Martens, J., & De Mulder, T. (2004). A comparison of human and automatic musical genre classification. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*, *4*, 233–236.

Li, T., Ogihara, M., & Li, Q. (2003). A comparative study on content-based music genre classification. In: *SIGIR 03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. New York, NY, USA, (pp. 282–289) ACM Press.

Li, T., & Tzanetakis, G. (2003). Factors in automatic musical genre classification. In: *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*.

Logan, B., & Salomon, A. (2001). A music similarity function based on signal analysis. In: *2001 IEEE International Conference on Multimedia and Expo (ICME'01).* (p. 190).

Makhoul, J. (1975). Linear Prediction: A Tutorial Review. In: *Proceedings of the IEEE*, *63*, 561–580.

Mandel, M. I., & Ellis, D. P. (2005a), Song-level features and support vector machines for music classification. In Crawford and Sandler (2005).

Mandel, M., & Ellis, D. (2005b). Song-level features and SVMs for music classification. Extended Abstract. MIREX 2005 genre classification contest (www.music-ir.org/evaluation/mirex-results).

Pampalk, E., Flexer, A., & Widmer, G. (2005). Improvements Of Audio-Based Music Similarity And Genre Classification. In (Crawford and Sandler, 2005).

Schapire, R. E., & Singer, Y. (1998). Improved boosting algorithms using confidence-rated predictions. In: *COLT 98: Proceedings of the eleventh annual conference on Computational learning theory*. New York, NY, USA, (pp. 80–91) ACM Press.

Soltau, H. (1997). Erkennung von Musikstilen. Masters thesis, Universitat Karlsruhe.

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing 10*(5), 293–302.

Tzanetakis, G., Ermolinskyi, A., & Cook, P. (2002). Pitch histograms in audio and symbolic music information retrieval. In: Fingerhut, M. (ed.): *Proceedings of the Third International Conference on Music Information Retrieval: ISMIR 2002*, 31–38.

West, K., & Cox, S. (2004). Features and classifiers for the automatic classification of musical audio signals. In: *Proc. 5th International Conference on Music Information Retrieval (ISMIR 2004).*

West, K., & Cox, S. (2005). Finding an Optimal Segmentation for Audio Genre Classification. In (Crawford and Sandler, 2005).

Xu, C., Maddage, N.C., Shao, X., & Tian, Q. (2003). Musical Genre Classification Using Support Vector Machines. In: *In International Conference of Acoustics, Speech & Signal Processing (ICASSP03).*