MARSYAS SUBMISSIONS TO MIREX 2007

George Tzanetakis
University of Victoria
Computer Science
gtzan@cs.uvic.ca

ABSTRACT

Marsyas is an open source software framework for audio analysis, synthesis and retrieval with specific emphasis on Music Information Retrieval. It is developed by an international team of programmers and researchers led by George Tzanetakis. In MIREX 2007 we participated in the following tasks: Audio Artist Identification, Audio Classical Composer Identification, Audio Genre Classification, Audio Music Mood Classification, and Audio Music Similarity and Retrieval Results. In this abstract we describe the specific algorithmic details of our submission and provide information about how researchers can use our system using the MIREX input/output conventions on their own datasets. Also some comments on the results are provided especially highlighting the excellent running time performance of our system (an order of magnitude faster than any other submission while remaining competitive in task performance).

1 INTRODUCTION

Marsyas is an open source software framework for audio processing with specific emphasis on Music Information Retrieval (MIR). It has been around since 1999 and in 2002-2003 underwent a major restructure/rewrite (version 0.2) [3]. This version has now matured and has been progressing nicely in 2006-2007 with the addition of several new developers and finally some decent documentation. Therefore we were excited to try out the software for the Music Information Retrieval Evaluation Exchange (MIREX) in 2007.

There are two main advantages of *Marsyas* compared to other solutions for building MIR systems:

- **Integration:**

  Marsyas strives to support all the necessary algorithmic and software building blocks required to build full MIR systems. Frequently MIR researchers use a variety of different software systems to achieve their goal. For example MATLAB might be used for feature extraction and WEKA might be used for machine learning/classification. There are two main problems with such non-integrated approaches. The first is reduced performance due to communication bottlenecks between each part of the process. The second which is more deep but not really utilized in our submission this year is the ability of integrated systems to combine signal processing and machine learning on several different abstraction layers and with both bottom-up and top-down processing. In constrast typically the use of non-integrated approaches follows the classic bottom-up sequential approach of feature extraction followed by classification.

- **Runtime performance:**

  As most practitioners of MIR for audio signals know, it takes a lot of computation time. One of the major goals of Marsyas is to reduce this computation time as much as possible. Unlike many other computer applications, computation time differences in audio MIR can play an important role in the ability to conduct experiments especially over large audio collections. An experiment that completes in 30 minutes is much easier to handle compared to one that completes in 8 hours and can be repeated with different parameters. Being able to process a million sound clips can result in better statistics for feature extraction than processing 100 sound clips and so on. Marsyas achieves fast run-time performance using a variety of different means which include: 1) a dataflow architecture that minimizes the need for memory allocation and can process audio files using large networks of computation blocks with a fixed memory footprint 2) fast, optimized C++ code for all operations 3) the ability to process large collections of audio files in one run with fixed memory footprint. Frequently other approaches to MIR operate on one file at a time adding significant computation time to start/stop a process, allocate memory etc every time a file is processed.

The main goal of our MIREX submission was to highlight these characteristics of *Marsyas* and hopefully motivate more researchers to explore the framework and contribute to it. Anyone can download the software framework, look at the corresponding code and run experiments on their own datasets. Therefore the selected set of features and classification approach we choose to utilize was straight-forward, well-known and most important fast to compute.

Moreover, we have significant experience using these features over a large number of various audio datasets so we felt more confident about their robustness dealing with unknown audio collections. More complicated feature extractors for example based on rhythmic, pitch, and stereo information are supported at various levels of completeness in *Marsyas* but unfortunately will have to wait for next MIREX.

2 SYSTEM DESCRIPTION

For all the tasks we participated we decided to represent each audio clip as a single feature vector. Even though much more elaborate audio clip representations have been proposed in the literature we like the simplicity of machine learning and similarity calculation using single feature vectors per audio clip. Coupled with a decent classifier this approach worked reasonably well compared to other much more complex ones.

The features used are Spectral Centroid, Rolloff, Flux and Mel-Frequency Cepstral Coefficients (MFCC). To capture the feature we compute a running mean and standard deviation over the past M frames:

$$m\Phi(t) = mean[\Phi(t-M+1),..,\Phi(t)] \quad (1)$$
$$s\Phi(t) = std[\Phi(t-M+1),..,\Phi(t)] \quad (2)$$

where $\Phi(t)$ is the original feature vector. Notice that the dynamics features are computed at the same rate as the original feature vector but depend on the past M frames (40 in our case corresponding to approximately a so called "texture window" of 1 second). This results in a feature vector of 32 dimensions at the same rate as the original 16-dimensional one. This process is illustrated in Figure 1. The sequence of feature vectors is collapsed into a single feature vector representing the entire audio clip by taking again the mean and standard deviation across the 30 seconds (the sequence of dynamics features) resulting in the final 64-dimensional feature vector per audio clip. A more detailed description of the features can be found in Tzanetakis and Cook [2].

For the audio similarity and retrieval task once all the feature vectors (one per audio clip) have been computed the features are normalized so that the minimum of each feature is 0 and the maximum in 1 (Max/Min Normalization) and Euclidean distance over the normalized features is used for the distance matrix.

For all the classification tasks (audio artist identification, audio classical composer identification, audio genre classification, audio music mood classification) a linear support vector machine classifier was used.

3 IMPLEMENTATION

In this section we provide information about how to download *Marsyas* and find information for installing and using the framework as well as specific information for running the tasks we participated using the MIREX 2007 input/output conventions. We hope that providing this information will help other researchers and practitioners run our system on their own datasets and motivate them to participate in the *Marsyas* developer and user communities. *Marsyas* can be compiled under Linux, OS X (Intel and PPC), and Windows (Visual Studio, Visual Studio Express, Cygwin and MinGW).

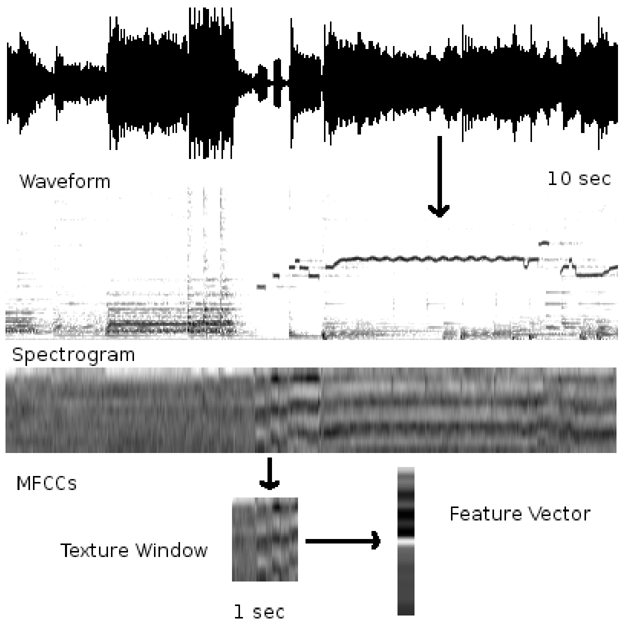

Figure 1. *Feature extraction and texture window*

To download *Marsyas* use the following url:
`http://www.sourceforge.net/projects/marsyas`
For information and documentation use the following url:
`http://marsyas.sourceforge.net`

System specific installation instructions are provided in the documentation. Once compiled it is straightforward to run the MIREX 2007 tasks we participated. For the audio music similarity/retrieval task the following command-line commands are required:

```
> ./bextract -e SVSTFTMFCC
   -c /pathto/collection.txt
   -wd /Users/gtzan/mirex/
   -w marsyas_mirex2007.arff
> ./kea -wd /Users/gtzan/mirex
   -m distance_matrix
   -dm /pathto/marsyas_mirex2007_dm.txt
   -w marsyas_mirex2007.arff
```

The first command extracts features using the SVSTFTMFCC extractor to the Weka .arff file *marsyas_mirex2007.arff* using as working directory */Users/gtzan/mirex/*. The second command calculates the corresponding distance matrix in the specified MIREX format. The matrix is stored in the file *marsyas_mirex2007_dm.txt*. The command-line arguments to the two executables have been broken into separate lines for presentation purposes but should be typed in a single line.

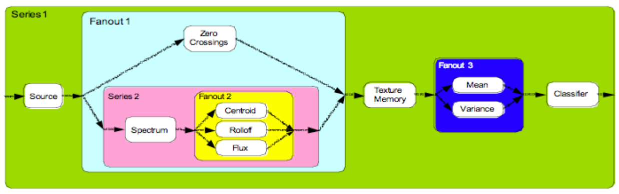

Figure 2. *Feature extraction data flow network in Marsyas 0.2*

For all the audio classification tasks a single executable for feature extraction/training/classification is used. In MIREX this program was executed once for each fold. Due to time constraints we decided to avoid dealing with associating feature vectors with audio clips and therefore audio features for the same audio clip might be recomputed across different folds. Even more computational savings could have been achieved if we had implemented that feature. The command for audio classification is:

```
./bextract -e SVSTFTMFCC
  /pathto/traincollection.txt
  -tc /pathto/testcollection.txt
  -pr /pathto/predictedcollection.txt
```

For classification the *Marsyas* MIREX submissions utilized a linear support vector machine trained using libsvm [1] which is directly integrated into the source code. Figure 2 shows the *Marsyas* dataflow diagram for the feature extraction that is common among all tasks.

4 DISCUSSION OF RESULTS

Overall we were very pleased with the performance of the *Marsyas* submissions to MIREX 2007. In all tasks the Marsyas submissions performed comparably to the other submissions. The detailed results are available from the MIREX 2007 webpage [1] so in this section we only briefly highlight some of the evaluation results that are specific to Marsyas.

In our opinion the most important numbers are the significantly faster run-time performance of the Marsyas submissions to the audio classification tasks despite the fact that no cashing of feature extraction was performed. Table 4 is based on the MIREX 2007 results. The following changes were made: the running times are provided in minutes which are easier to grasp, the Marsyas entry has been moved to the top and the total run time (feature extraction, train, classification) is provided. Other notable distinctions was that the Marsyas submission ranked first in the Audio Mood Classification task and ranked second (with statistically insignificant differences between the top 4 entries) in the Audio Music Similarity and Retrieval task.

[1] `http://www.music-ir.org/mirex2007/index.php/Main_Page`

	AAId	ACCl	AGCl	AMCl
GT(Marsyas)	**10**	**9**	**19**	**2**
ME	57	64	139	10
MEspec	57	47	137	10
KL	107	49	-	9
IM-KNN	50	41	135	10
IM-SVM	49	44	115	9
TL	473	348	905	26
GH	-	-	382	-
CL	-	-	-	184
KL2	-	-	-	8

Table 1. Total run times (feature extraction, training and classification) in minutes for the MIREX 2007 audio classification tasks (AAId is Audio Artist Identification, ACCId is Audio Classical Composer Identification, AGCl is Audio Genre Classification and AMCl is Audio Mood Classification)

5 FUTURE WORK

There is plenty of interesting future work to be explored. Now that we have the MIREX Input/Output conventions fully supported we are very excited about participating in MIREX in the future. Our submissions this year can be considered a baseline and we can only improve in the future. In no particular order here are some of the directions we would like to explore for the tasks we participated this year: more complex audio clip representations and similarities than the single vector approach, additional features (rhythm-based, pitch/chroma based, stereo panning), and better utilization of domain knowledge such as hierarchies. In addition we hope to participate in more tasks in the following years.

Finally we would like to encourage other practitioners to explore and hopefully contribute to *Marsyas*. We are also happy to offer assistance to anyone interested in porting their existing systems into *Marsyas*.

6 REFERENCES

[1] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

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

[3] George Tzanetakis. Marsyas-0.2: a case study in implementing music information retrieval systems. In *Intelligent Music Information Systems*. IGI Global, 2007. to appear.