

# Music tagging with *What's that Tune*

## End of Semester Report

Emily Fortuna

May 4, 2008

## 1 Rationale for the Project

*What's that Tune* seeks to harness the power of human computation in order to create "audio tags" for familiar popular music. It then uses this accumulated data to train a neural network on the same task, so that this process can be automated and applied to sounds and music that are less familiar.

The results of this project have multiple applications. Determining optimal preview clips can help potential customers select music before purchasing it and also allow users to quickly scan through the pieces of a playlist to ensure continuity. Tagging music with its representative clip allows computers to present music search results to users in a more native, informative format than just words. Furthermore, search speed of music can become more competitive with text search if a computer only needs to analyze a representative clip of the music.

## 2 Background and Literature Review

Current trends in computer usage have shifted toward using it as a tool to access the wealth of information on the internet, increasingly audio and video files. This trend makes the ability to search through audio files and select good previews of clips of a file a high priority.

Approaches to making the audio search space usable include text-mining of music articles for semantic tags [10], hand-tagging [3], query-by-humming [2], and lower level signal analysis and genre classification schema [5, 7, 13]. More recently, researchers have turned to using *human computation* as a method to retrieve information from audio files.

## 2.1 Genre Identification Research

The genre identification research generally implements a two stage process: feature extraction from the audio signal and then classification. Researchers have extracted a variety of different features from audio signals using zero-crossing rate, signal bandwidth, spectral centroid, signal energy and Mel-frequency cepstral coefficients (MFCCs), inherited from speech recognition work. The majority of recent research focuses on incorporating MFCCs into their systems. Numerous classification schemes have been applied to music, including Gaussian mixture models, self-organizing maps, neural networks, K-nearest neighbor clustering, and hidden Markov models.

### 2.1.1 Low Level Feature Extraction Work

McKinney and Breebaart suggest the particular classification model does not result in a strong difference in accuracy of the classifications. Instead, they posit that finding new features to extract from audio signals will improve classification performance. They extracted four different features from audio signals: low-level signal properties, MFCC, psychoacoustic features including roughness, loudness, and sharpness, and an auditory model representation of temporal envelope fluctuations (AFTE) [7]. To these features McKinney and Breebaart then applied a Gaussian-based quadratic discriminant analysis (QDA) as a classification model. Overall, they found the AFTE feature set to be the most powerful for a variety of audio classes. However, the standard low-level features more successfully classified crowd noise and classical music while psychoacoustic features were more successful at speech classification. Therefore, no single feature set can be used at the exclusion of others for classification at this time.

### 2.1.2 Classification Studies

Researchers Logan and Salomon developed a method for measuring the similarity of songs using K-means clustering of spectral features [5]. For this process, Logan divided the audio signal into frames, which then used MFCCs, “the discrete cosine transform of the log amplitude Mel-frequency spectrum that can be compared using the Euclidean distance measure”. These frames for a given song are then clustered together into groups based on similarity. Each member of the set of clusters has a weight, mean, and covariance that together form the signature of that particular song. With these signatures, Logan then calculated the Earth Mover Distance (EMD) between songs (so named because the measure is analogous to the cost of moving probability mass, or “piles of earth” from one cluster to another). The EMD between a song and itself is 0. Logan and Salomon found this technique fairly successful in measuring the similarity between to songs. Songs on the same album, and then in the same genre were most frequently judged most similar. They also found success in measuring the high similarity of a corrupted song (a song with a 30 second clip removed from the file) and the original song.

## 2.2 Human Computation and Music

However, given the overall limited success of current approaches (and time constraints in the case of hand-tagging), Jean-Julien Aucouturier and Francois Pachet suggest that tasks such as genre classification have reached an unsurpassable “glass ceiling” unless we find a way to incorporate higher level auditory processing in the classification task [1]. In this vein, more recently Louis von Ahn’s idea of human computation and creating games with a purpose, manifested in the image-labeling ESP Game [12], has been applied to the music search task in the projects TagATune [4], MajorMiner [6], and the Listen Game [11]. All three of these game-based approaches acquire semantic tag information about the quality of the music through human inputs during the game.

TagATune pairs players up to list words that their partner would use to describe sound clips from the FreeSound database. Unlike the other two projects, this game works on tagging all forms of sound, not just music. During game play, participants list words that describe the given sound based on a

listed category from *object, place, action, color, mood, movie genre, opposite, and freebie* (an open description). A word becomes an official tag when a statistically significant number of people use a description for a particular audio clip. Because sound quality is significantly more subjective than image labeling and the categories did not always apply to all of the sounds, application of the *games with a purpose* idea for in this domain posed additional challenges.

In the Listen Game, many online players work together to provide qualitative annotations for well known music clips. Players listen to a fifteen second clip of a song from a set of 250 popular western songs and must select the word from a list of six words that best describes the clip and worst describes the clip given the category *instrumentation, usage, or genre*. These six words come from a predefined 174-word vocabulary used in the CAL500 survey. A player’s score is determined by the amount of agreement between her word selection and all the other player’s selections. The data from the Listen Game was then applied to create a Supervised Multiclass Labeling (SML) model for automatically labelling other songs. Researchers used a Gaussian mixture model (GMM) over an audio feature space combined with the semantic descriptions using the mixture hierarchies expectation-maximization algorithm (MH-EM). To build this SML, Turnbull extended the MH-EM algorithm in order to work with non-binary data.

In MajorMiner, in which players play in groups to create both original and common descriptions of ten second music clips. Players receive two points for the first time an answer has been “verified” (ie, the player is the first one to guess a particular word that has been already submitted) and no points if the word has been verified more than once or if the word is completely original. Mandel found that players most commonly described genre, instrumentation, and gender of the singer, if present. Emotional, descriptive, and rhythm words were less frequently used.

## 3 Project Overview

This project seeks to adapt the idea of human computation in order to acquire information in a manner more closely related to music itself than words. In the “What’s That Tune” game, two players online are paired together. First, each player is simultaneously presented with a different well known musical

piece (currently the database is focusing on Classic Rock, but it shall branch out). He or she must choose some portion of the piece that he or she believes is representative of that song. When both players have selected their portion, each player listen to the other player's selected window from the piece. Each guesser tries to identify the song by name in a multiple choice question. If the person guesses incorrectly, then the guessor has two more chances to guess the correct answer.

Players receive more points for guessing correctly on the first try, and fewer for the second try, and so forth. Additionally, players receive more points for selecting the song window and guessing the correct answer in a shorter amount of time. The partner also receives points when his or her partner correctly guesses the song title, as an incentive for the partner to make a good song choice. The guessor may only listen to a song selection three times, and additional listens decreases the number of points that can be earned from correctly identifying the piece. Players try to identify as many songs as possible within a given time limit.

This game acquires test data to help computers determine the parts of a song that humans believe make it unique. In the background our computer would be learning via a neural network how to select which portions of the song are most "representative" of the piece, useful for more obscure pieces.

Once our computer learner has reached a specific level of competence, we could then pair up the computer with a human player. The human player would try to guess the song from the window chosen by his or her partner, the computer. The computer refines its neural network based on whether the user could correctly identify the tune if it had selected a good window from the audio clip. Results of this game are helpful for refining a computer's capabilities in an audio search and helping a computer learn to extract the human significant pieces from an audio signal.

## 4 Project Process

For the first month and a half I researched current work that has been done in analyzing the music search space, and I refined the project proposal. My research log chronicling the initial research I read can be found here: [http://docs.google.com/Doc?docid=ddnt5fww\\_8dxbszhm6&hl=en](http://docs.google.com/Doc?docid=ddnt5fww_8dxbszhm6&hl=en).

Then I built the "model" framework for the game,

and set up a SQLite database. Players first create an account, and their username is stored in the database, so that they can log in again with their current level and points saved. The model queries the database to generate questions and which contain references to pieces and incorrect answers. By answering questions correctly, the user gains points, and advances to subsequent levels, in which the player must identify a piece from increasingly shorter time windows.

Next I constructed the control, which consists of the hooks to the user interface of the game. Lastly, I designed the user interface with Yan, and then implemented the interface. I ran into a large number of obstacles in this stage. I initially developed the project as a stand-alone application that ran on my local computer. I had to modify an existing library to play mp3 files and then worked on getting them to stream in from the internet. I also had to make the application multithreaded so that the user could listen to the audio and interact with the UI simultaneously. I added a timer to advance the game forward for players at a reasonable pace. I rewrote the code to interface with a MySQL server instead of SQLite, to prepare my program for porting to a typical server setup.

The troubles arose when I finally got access to a server and began porting my project over to the new server in the form of an applet. First I structured the framework so that all of the views were part of one single, larger applet. Next, I wrote a custom UI slider component that not only updated with the progress of a song played on my local computer, but also responded to user clicks by creating a "song window" that the progress slider would snap to when stopped. This was a considerable challenge due the fact that upon initial examination, I could only get information about the song in three inconvertible forms: current frame position, current number of bytes read, and total milliseconds. Because mp3s have a variable bitrate, frames to bytes was not a feasible conversion, and milliseconds to frames also did not prove feasible. When testing with music that resided on my local machine, I initially created a solution involving the BitStreamReader size, but unfortunately this technique did not transfer when I tried to stream media in from the internet. I ended up writing my own mp3 player library that determined the current frame position, and ultimately used metadata from the http request to calculate what fraction of the song had been played, in frames.

More obstacles surfaced as I learned more about applet security. I needed my applet, code that is executed and run on the user's local computer, to be able to make requests to a database running on the server. Not until placing this applet on the server did I discover that there is a security restriction placed on applets so that they cannot execute or connect to code on a remote computer (in this case, the server). I investigated Remote Method Invocation (RMI), and restructured my program to instead use a client-server structure to incorporate RMI. I set this code up on the server I had been working on. For some reason the server, running Ubuntu, was not configured correctly to let the server Java process necessary for RMI run. I set up a server on my own machine, also running Ubuntu, and encountered the same problem. I worked to reconfigure the server, but all of these efforts ultimately proved fruitless. Ultimately I moved all of my code to a FreeBSD server, on which the RMI Server code functioned correctly. I set up Java and a new database on this server. I also had to add sound on the server, to prevent sound card errors (even though the server itself wasn't playing the sounds, it was generating errors!). Next I made the objects passed via RMI serializable.

Once these wrinkles had been ironed out, I ran into additional security issues with the applet being restricted in connecting to other websites. I created a self-signed jar and applet to allow access to these websites, but ultimately this approach was abandoned because of persisting permissions issues. I have decided to instead create a downloadable jar file that can be executed on the user's local computer, and connects to the server database via RMI. I am currently still encountering problems with getting my local jar to connect to the server through RMI.

The code for this project is available here: <https://svn.rice.edu/r/fortuna/devikaResearch>.

## 5 Future Work

Once the last of the RMI communication issues are resolved and the database is populated with songs, we can begin collecting user data in earnest. The RMI issue is expected to be resolved by May 5, and, barring future obstacles, the game should be available on the web by May 9.

I will then write the framework for the neural net. Using this framework, Yan and I will each create our

own complimentary neural learners that learn based on different features of the music. These learners will use collected data from the game to refine their strategies. My principle concern for this stage is getting enough data (ie, players to play) through our game in order to have enough data for our neural net to train from.

## References

- [1] J.-J. Aucouturier and F. Pachet, "Improving timbre similarity: How high is the sky?" *Journal of Negative Results in Speech and Audio Science*, vol. 1, no. 1, pp. 1–13, 2004.
- [2] R. B. Dannenberg and N. Hu, "Understanding search performance in query-by-humming systems," in *International Conference on Music Information Retrieval*, 2004.
- [3] M. Goto, "AIST annotation for RWC music database," in *ISMIR*, 2006.
- [4] E. L. M. Law, L. von Ahn, R. B. Dannenberg, and M. Crawford, "Tagatune: A game for music and sound annotation," in *8th International Conference on Music Information Retrieval*, 2007.
- [5] B. Logan and A. Salomon, "A music similarity function based on signal analysis," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, August 2001.
- [6] M. I. Mandel and D. P. W. Ellis, "A web-based game for collecting music metadata," in *8th International Conference on Music Information Retrieval*, 2007.
- [7] M. F. McKinney and J. Breebaart, "Features for audio and music classification," in *Proceedings of the 4th International Conference on Music Information Retrieval*. Baltimore, MD: Johns Hopkins University, October 2003.
- [8] A. Raskin. Songza. Humanized, Inc. [Online]. Available: <http://songza.com>
- [9] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *Proceedings of the IEEE 22nd International Conference on Acoustics, Speech, and Signal Processing*, 1997, pp. 1331–1334.

- [10] D. Turnbull, L. Barrington, and G. Lanckriet, "Modeling music and words using a naïve bayes approach," in *International Conference on Music Information Retrieval*, 2006.
- [11] D. Turnbull, R. Liu, L. Barrington, and G. Lanckriet, "A game-based approach for collecting semantic annotations of music," in *8th International Conference on Music Information Retrieval*, 2007.
- [12] L. von Ahn and L. Dabbish, "Labeling images in a computer game," in *ACM Conference on Human Factors in Computing Systems (CHI)*, 2004, pp. 319–326.
- [13] K. West and P. Lamere, "A model-based approach to perceptual music similarity," *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 149–149, 2007.