Automatic Tagging of Audio: The State-of-the-Art

Thierry Bertin-Mahieux
Columbia University, USA
tb2332@columbia.edu

Douglas Eck
University of Montreal, Canada
douglas.eck@umontreal.ca

Michael Mandel
University of Montreal, Canada
and *Columbia University, USA*
mim@ee.columbia.edu

ABSTRACT

Recently there has been a great deal of attention paid to the automatic prediction of tags for music and audio in general. Social tags are user-generated keywords associated with some resource on the Web. In the case of music, social tags have become an important component of ``Web 2.0" recommender systems. There have been many attempts at automatically applying tags to audio for different purposes: database management, music recommendation, improved human-computer interfaces, estimating similarity among songs, and so on.

Many published results show that this problem can be tackled using machine learning techniques, however, no method so far has been proven to be particularly suited to the task. First, it seems that no one has yet found an appropriate algorithm to solve this challenge. But second, the task definition itself is problematic. In an effort to better understand the task and also to help new researchers bring their insights to bear on this problem, this chapter provides a review of the state-of-the-art methods for addressing automatic tagging of audio. It is divided in the following sections: goal, framework, audio representation, labeled data, classification, evaluation, and future directions. Such a division helps understand the commonalities and strengths of the different methods that have been proposed.

INTRODUCTION

Many tasks require machines to *hear* in order to accomplish them. In the case of music, we would like computers to helps us discover, manage, and describe the many new songs that become available every day. The goal is not necessarily to replace humans: the best description of an album is probably the one of a music expert. That being said, it is now impossible for any group of experts to listen to every piece on the Internet and summarize it in order for others to discover it. For example, let's consider an online radio like Pandora (www.pandora.com) or Last.fm (www.last.fm). As a service, they provide listeners with a personal continuous stream of music based on their musical tastes. In order to make recommendations, they rely heavily on collaborative filtering: if a listener liked songs A and B and you liked A, you might like B. Such algorithms have proven to be extremely efficient. However, it leaves two major problems. First, what is often called the cold-start problem, the fact that new items entering the

system are not popular, hence never recommended, hence never popular. Secondly, some songs that start as popular end up even more popular because they do get recommended. A machine that could understand music as experts do would improve both situations.

Understanding music is a very general statement. In this work, we are interested in describing music with appropriate tags (or labels, keywords), addressing the previous situation in a natural way. Knowing a set of tags that applies to a song is in itself a description. One can also manage music by grouping pieces that are described in a similar way. Recommendations can also be made from the space of tags, e.g. one can identify that a listener like "rock" music with "female voices". Note that on Pandora and Last.fm, users can already browse music according to tags that were applied by other users. Once again, we do not claim that machines would create better tags than human listeners. Automatically created tags, or *autotags*, are useful for new or unpopular music, as well as for completing the set of tags applied by humans. Describing music with a "bag of words" is the same approach taken by text search engines that describe web pages using the words they contain. With the ability to create an appropriate set of tags that applies to any given audio, we could develop as many tools as Google and Yahoo did for web documents and change the way we handle music.

Many different groups have been trying to predict tags from audio, yet there have been few attempts at uniting the community behind a clear shared task definition. This was partially addressed at MIREX 2008. MIREX stands for Music Information Retrieval Evaluation eXchange, a set of contests held each year at the International Conference on Music Information Retrieval (ISMIR). Throughout this paper when we refer to a MIREX contest we mean the 2009 audio tag classification task, unless otherwise mentioned. More information can be found at http://www.music-ir.org/mirex/2009/index.php/Main_Page. Though we applaud the hard work of the contest organizers, there is room for improvement. For example, the contest could have been more clearly defined in terms of evaluation and we discuss it later on in this chapter. One goal of our review is to bring together the many proposed evaluation methods and work towards a common framework for evaluation. In addition we hope that this effort will help bring new researchers into this area by offering them a clear set of goals.

Note that the lack of a common goals does not mean that learning tags have been useless so far. For instance, Eck et al. (2008) and Barrington et al. (2009) both showed that automatically generated tags can improve music recommendation. Also, Turnbull et al. (Turnbull, Barrington, Torres & Lanckriet, 2008) explain how to manage a sound effect database using automatic tagging.

This chapter focuses on automatic tagging of music. However, regarding the vast and diverse set of tags that have been used and the absence of prior knowledge assumed on the audio, this work addresses tagging of audio in general. For instance, someone working on a speech versus music classifier should also find the following methods and algorithms interesting.

Human tags

We presented the goal as finding a set of descriptive tags for given audio frames, but it is important to first understand how humans tag audio. We often call them **social tags** as they are applied by humans on some collaborative platform, as opposed to tags produced by an automatic tagging algorithm. The different tags obtained are better understood when we know why people tag in the first place. Lamere lists six reasons (Lamere, 2008) that we summarize here:

- Memory and context – items are tagged to assist personal retrieval, e.g. having access to all "pop" songs in a database.
- Task organization – items are tagged to assist in the organization of music discovery tasks, e.g. tag a new song "check out" or create a future playlist of "emo" artists.
- Social signaling – items are tagged as a way for an individual to express their tastes to others, e.g. the use of the tag "seen live".

- Social contribution – items are tagged to add to the group knowledge, e.g. a metal fan tagging a song “not metal” if it is not considered truly metal.
- Play and competition – items are tagged as part of a game, more details in the Sources of Data section.
- Opinion expression – items are tagged to convey an individual's opinion, e.g. the use of the tag “awesome”.

The types of tag themselves can be associated with a few categories, see Table 1. Finally, here is specific example of tagging of the band *The Shins* in Table 2.

Tag Type	***Frequency***	***Examples***
Genre	68%	Heavy metal, punk
Locale	12%	French, Seattle, NYC
Mood	5%	Chill, party
Opinion	4%	Love, favorite
Instrumentation	4%	Piano, female vocal
Style	3%	Political, humor
Misc	2%	Coldplay, composers
Personal	1%	Seen live, I own it
Organizational	1%	Check out

Table 1: Distribution of tag types on Last.fm (in 2007)

Tag	***Freq***	***Tag***	***Freq***	***Tag***	***Freq***
Indie	2375	The Shins	190	Punk	49
Indie rock	1183	Favorites	138	Chill	45
Indie pop	841	Emo	113	Singer-songwriter	41
Alternative	653	Mellow	85	Garden State	39
Rock	512	Folk	85	Favorite	37
Seen live	298	Alternative rock	83	Electronic	36
Pop	231	Acoustic	54	Love	35

Table 2: Top 21 tags applied to The Shins on Last.fm (in 2007)

In this paper, we present how to learn to apply similar tags to audio based on human tagging examples as the ones in Tables 1 and 2. We can already highlight a limitation of automatic tagging systems based on audio, unlike humans they could not learn to apply tags like "seen live" and "love" since there is no correlation with the audio itself. However, tags being categorized as *genre*, *mood*, and *instrumentation* are within an algorithm's reach.

Definitions

A **tag** is a user-generated keyword associated with some resource, in our case audio. In general we tag audio tracks (or segments of a track) but we will also talk about tagging albums or artists by aggregating predictions made over tracks. **Tag** and **label** are used interchangeably. **Autotag** is a short name for a label applied to some audio by an automatic tagging algorithm. We may refer to an automatic tagging algorithm as an **autotagger**, but we must be careful as that term as been used for a specific algorithm in Bertin-Mahieux et al. (2008).

FRAMEWORK

One goal of this paper is to provide a unified view of all automatic tagging systems for audio that are currently being investigated. These systems can be decomposed into four parts: audio representation, tagging data, machine learning algorithm and evaluation (see Figure 1). Evaluation is included as a full part of the system as it has impact on the way the machine learning algorithm is trained. For instance one might use a different algorithm to train a model evaluated for artist similarity prediction than for precision/recall of tag vocabularies. Regarding machine learning algorithms, the term must be understood here in its most general definition, as a function that takes an input and produces an output depending on the input. In the case of automatic tagging, an algorithm can be viewed as a "machine" that takes a piece of audio, and a set of tags, and decides which tags apply and eventually to what degree. Classification can be seen as the case where the degree must be either 0 or 1.

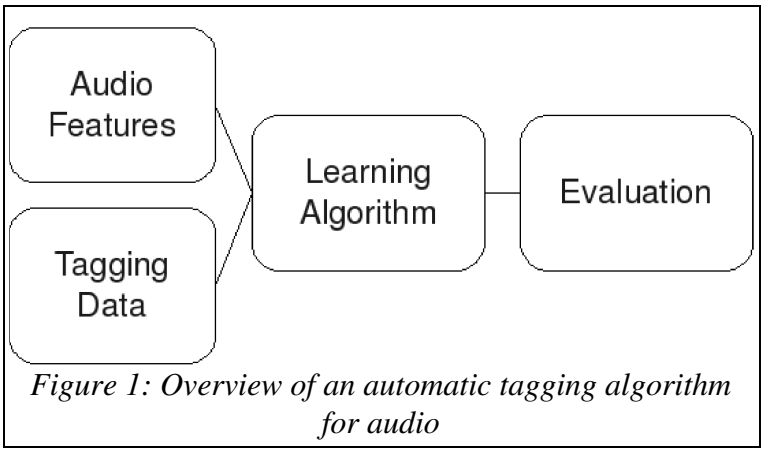

Figure 1: Overview of an automatic tagging algorithm for audio

The next sections follow the four parts of the framework. An algorithm can be described by how it deals with each of these parts: what audio features and tagging data it uses, what learning algorithm is used, and how performance is evaluated. Choices for each part can usually be made independently of the choices in the other parts.

AUDIO REPRESENTATION

We briefly discuss the vast area of representing audio, as it is always an issue when applying machine learning techniques to sounds. Let's start by stating the obvious: there is no one single best representation that will work in all cases. Furthermore, it is difficult to know in advance which representation will work with which technique. In this section we present the most typical approach, which is to mix together many representations. We present some specific choices that were made in recent papers on automatic tagging of audio. For an introduction to audio features or specific details on the implementations, we refer the reader to the future reading section.

Bag of frames

The "bag of frames" representation is an analogy to the term "bag of words" in the text search community: to describe a text document, one keeps a vector of word counts from that document, disregarding the order in which the words appeared in that document. In the case of audio, for each "frame", one computes one or many features, which are considered in a collection that ignores their order.

An example of a bag of words setting would be the following approach: for each 100 ms of audio, compute the following features: fourier transform, mel-frequency cepstral coefficients (MFCC), chromagrams, autocorrelation coefficients, delta MFCC, zero-crossing rate. Each of these features can be seen as a vector of size at least 1, and all are simply concatenated. Another specific example from (Turnbull, Barrington, Torres & Lanckriet, 2008), authors use MFCC, delta-MFCC and delta-delta-MFCC on 10ms-length frames.

In the bag of frames representation, these features are then aggregated over longer durations. One simple aggregation technique is taking the mean and possibly the variance of each dimension of the previously created vectors. This has merit when dealing with frames of length 1 to 10 seconds. The intuition is that, for automatic labeling, the occurrence of an event is of interest but not necessarily its time stamp. A time division up to 100 or 10 ms can therefore be too much precision, but this property always has to be verified empirically. Aggregate features were successful for music genre classification in Bergstra et al. (2006).

Future reading

A complete description of typical audio features for Music Information Retrieval applications can be found in Aucouturier & Pachet (2003) or Gold & Morgan (1999). A more recent approach is the one of sparse coding, i.e. use representations that better compress the information into a few values. For work on sparse coding in the frequency domain, see Plumbley et al. (2006) and Gardner & Magnasco (2006). For work in the time domain, see (Grosse et al. 2007; Manzagol et al., 2007; Smith & Lewicki, 2006). For commonly used features, one can use the two following toolboxes that are freely available online: MIR toolbox (http://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox) and jAudio (http://jmir.sourceforge.net/jAudio.html). Some companies even offer to analyze audio online and provide you with features, e.g. the Echo Nest (http://www.echonest.com).

OBTAINING LABELED DATA

More specific to our task than representing audio is finding a proper dataset of labeled <audio,tag> pairs. As we explained in the introduction, the tags applied by humans are of great quality. The machine learning assumption is that if we show an algorithm enough examples, the correlation will become clear. However, the following trade-off remains: gathering more examples help, but we usually need to explore less reliable sources to do so. For instance, tags applied by music companies are usually of little value since they are chosen according to commercial interests instead of the music itself. The

ultimate goal would be to have a large set of examples that is "clean" at the same time, i.e. that could have been created by a human expert. There have been many attempts to build datasets, and (Turnbull, Barrington & Lanckriet, 2008) presents them as five possible sources. In the following section, we will follow Turnbull et al. and present the first four sources (the fifth one refers to automatic tagging, the purpose of this paper). Then, we discuss the size of the available datasets and their implications. Finally, we briefly present different ways to "clean up" the tagging data and arrange it in either a classification or regression setting for the machine learning algorithm. The number of tags to be learned simultaneously is also problematic.

Sources of Data

We follow (Turnbull, Barrington & Lanckriet, 2008) and present four sources of data.

Survey. Most straightforward and costly method, one can ask people to listen to songs and tag them. One problem usually arises, either the participants are paid, or there are not enough of them. The CAL lab at the University of California in San Diego (UCSD) paid undergraduates to fill out a survey, the resulting dataset CAL500 is available online (Turnbull, Barrington, Torres & Lanckriet, 2007). It consists of tags applied to 500 songs by an average of approximately 3 listeners. The vocabulary (set of possible tags) was fixed by the survey. Later in this section we talk about obtaining tag data from games. This overcomes many of the survey problems, participants fill the survey because of a reward (winning), but the reward is non monetary, hence acquiring data is not as costly.

Social tags. Tags applied by human users. Last.fm is an online radio that enables users to tag an artist, an album, or a song through a webbrowser or a donwloaded client. By the beginning of 2007, the database contained a vocabulary of 960,000 free-text tags and millions of songs were annotated. Last.fm data is available through their Audioscrobbler service page (Audioscrobbler). Last.fm, provides the largest freely available collection of tagging data, but other data available from the web exist, including MusicBrainz (http://musicbrainz.org). Last.fm data have been used or described in (Eck et al., 2008; Lamere, 2008; Mandel & Ellis, 2008b) .

Game. Different tagging games have been developed by research teams to gather clean data (Kim et al, 2008; Law et al., 2007, Mandel & Ellis, 2007; Turnbull, Liu, Barrington & Lanckriet, 2007). The idea is to give users an incentive to apply appropriate tags to songs or song snippets. For instance, in MajorMiner (Mandel & Ellis, 2007), users get points if they are the first or second person to use a tag on a particular excerpt. This avoids usage of random, unrelated, or mischievous tags. Cheating is always possible, but there are ways to counter it, usually by tracking a user behavior over some time. Data acquired this way are usually very clean, but still many orders of magnitude smaller in size than social tags. The largest game data available has been released by the Magnatagatune team (http://tagatune.org/Datasets.html) and contains tags applied to about 20,000 songs.

Web documents. A fourth idea is to use documents available on the Internet to describe audio. Celma et al. (2006) crawls MP3 blogs to obtain tags. For instance, one could search for words that are more often associated with a particular artist than with an "average artist", and use it as a tag. One can easily gather millions of tags, but the main drawback of this method is the noise in the data. Other approaches include querying search engines (Knees et al., 2008) and mining music websites (Whitman & Ellis, 2004).

Size of Datasets

Following what we saw as the different sources of tagging data, we emphasize the question of the size of the dataset. These days, the choice can be summarized into *small and clean* versus *large and noisy*. A typical small and clean data would come from a game, and a typical large and noisy dataset would be Last.fm data. To help clarify the choice and its implications, let's focus on the two following questions:

- **What is the application?** If the goal is to build a new online music recommender based on tags, the choice is obvious, the data has to be large. No performance obtained on small datasets can be guaranteed in a real life setting of millions of songs. However, if the goal is to classify a set of songs into 10 or 20 categories, it makes sense to learn these categories on very clean, unambiguous data, even if we sacrifice the size.
- **What is the algorithm?** We will discuss in the next section the different algorithms used for automatic tagging, but it is clear that some are more suited than others for a particular size of dataset. Algorithms like the support vector machine (SVM) classifier can be trained well on small datasets, but their training time is on the order of the square of the number of examples. On the other hand, algorithms such as neural networks with stochastic gradient descent can be trained on very large datasets, but can be defeated by too few examples. If a specific algorithm must be used (for research purpose for instance), choosing a proper dataset size is essential.

The authors believe that most of the future applications will rely on large datasets, a trend already seen in image information retrieval, e.g. in (Li et al., 2006). The incredible amount of data should overcome the inherent noise, except for some specific task where the classification is expected to be clear and very accurate. This is rarely the case in real life, a sound can be tagged in various ways, and the tagging can change with context and over time. We encourage new researchers to mine web documents and make extensive use of available datasets like Last.fm. We also hope that in the end, online games will produce larger and more reliable datasets. Magnatagatune (tagatune.org/Magnatagatune.html), a recently published tagging dataset of 20.000 audio songs, is a step in the right direction.

Cleanup of data

Here we mention different heuristics one might want to use to improve tagging data. No dataset coming from the four sources we described can be efficiently used “out of the box”. Furthermore, the way the data is prepared has to be consistent with the task at hand, the machine learning algorithm, and the evaluation method. For instance, if a band like *Radiohead* has been tagged ten thousand times more often than others, should the algorithm see ten thousands time more often audio coming from *Radiohead*? Another example, some tags are synonyms, or the negation of one another. Should they be combine into one label, and do we penalize the algorithm during learning if it predicts a synonym instead of the tag it has seen?

The first thing is to select the vocabulary. As we mentioned, Last.fm data has approximately one million different tags. Since it would be unlikely for anyone to try to learn them all, a subset has to be chosen. Eck et al. (2008) simply take the most popular tags (the one applied the most) on Last.fm. It is straightforward, but sometimes misleading. For instance, many listeners tag songs as “favorite” so they can make playlists out of their favorite pieces. Unfortunately, no algorithm should be able to learn such a tag unrelated to the music itself. A better list of tags to be used would be the tags people use to search for music. A user can search for “R&B” music, but he has no reason to search for “favorite” music unless it is

his own. The list of tags used for search is therefore cleaner and probably more useful to learn. Another method is also described in the Future reading section.

Another thing to be careful about is popularity bias. Some artist or songs are more popular than others, hence more frequently tagged. Additionally, some tags are more popular than other, hence more frequently applied. This bias appears in data coming from social tags or web documents, and to some extent in games data (tag popularity). Easy examples would be *Radiohead* versus most of the existing artists, and "rock" versus "japanese ska". One can decide to either leave the data as it is, or to try to reduce the bias, usually with the hope of helping the learning algorithm later on. See future readings for pointers.

Organization of data

We discuss the last two essential choices that must be made: should the data be presented as a classification or regression? Then, should the tags be learned one or many at a time?

A classification implies a set of binary questions: does this tag apply to this audio, yes or no. With a regression, the question becomes: to what degree can this audio be described by this tag? In the real world, there are few cases where a belonging can either be 100% or zero. It is especially true for genre tags for music, where songs never belong to only one. Thus, a regression setting is the natural choice for an automatic tagging algorithm. There are many ways to transform a tagging dataset into a regression dataset. If it comes from a survey, one can take the percentage of people that tagged a song with a particular tag.

There are two major reasons to transform a tagging dataset into a classification dataset: 1) it is easier for a learning algorithm to find a proper class than a proper belonging value, and 2) values one gets for pairs <audio,tag> can be extremely noisy, transforming them into a classification can mitigate the noise. An example where the value <audio,tag> can be too noisy to be used is the CAL500 dataset (Turnbull, Barrington, Torres & Lanckriet, 2007). Most of the songs are listened to by 3 people. If we would take the proportion of listeners that tagged a song "rock" for instance, the numbers could only be 0, 0.33, 0.66 and 1. Obviously, the granularity does not account for real life and the number of listeners is not large enough to be confident in the value. To transform a regression dataset into a classification dataset, we usually use a threshold. In the case of CAL500, the authors suggest that we consider that a tag was applied to a song only if 80% of the listeners agreed. If the amount of data is large enough, e.g. coming from Last.fm or web documents, you can use rankings to determine if a tag applies to some audio or not. Eck et al. (2008) consider that a tag was applied only to the top 10 artists the most related to the tag (according to TFxIDF, described in the next section). The other applications of the tag were ignored or discarded.

Regarding the number of tags to be learned simultaneously, we can safely say that in theory, the more the better. Tags often share mutual information, e.g. the use of the tag "r&b" indicates that the tag "hip hop" will probably be applied as well. Learning a set of tags instead of one enables us to use those correlations in addition to those coming from audio features. That being said, the vocabulary can be a very large set, and can even grow with time. Not every machine learning algorithm can handle multiple outputs, and the computational complexity often grows accordingly. Another practical restriction is the use of cloud computing: if we learn one tag at a time, one can launch N learning tasks on N processors. Thus, the learning of the whole vocabulary scales inversely with the number of CPUs.

In the literature, Turnbull et al. (2008), Eck et al. (2008), and Mandel & Ellis (2008a) learn tags independently, and Trohidis et al. (2008) learn them simultaneously.

Future reading

Another method for selecting tags to be learned is described in Torres et al. (2007) and involves Canonical Correlation Analysis (CCA). CCA is especially designed to find dependencies between two

datasets living in different but related spaces. The two spaces here are the audio features and the tagging space. The idea is that a tag is meaningful if it is linked to some audio event (seen in the features). From a practical point of view, it amounts to choosing a vocabulary that can probably be learned.

Regarding popularity bias, few papers face this problem directly, mostly because published results often come from small datasets. We mention Eck et al. (2008), which uses term-frequency, inverse document-frequency (TFxIDF) to approach the issue. TFxIDF come from the text search community that deal with documents of different lengths and words used with varying frequency in a particular language. Seeing artists (or songs) as documents and tags applied to them as words of these documents, TFxIDF produces a value of how much a tag is particular to a given artist (or song). This problem could also be seen as bd-matching where we try to associate users to products. Each user has to be associated with b products, and each products associated with d users. The framework fits the popularity bias we face, but its application to tagging data is ongoing research. Huang & Jebara (2007) present a bd-matching implementation example.

Note that the TFxIDF method can be used to get a score in order to transform the data into a regression setting, the value to predict is TFxIDF's ouput.

MACHINE LEARNING METHODS

The central part of any automatic tagging algorithm is the model that links tags to audio features. Being as general as possible, any method that finds (possibly highly complex) correlations between the two can be seen as a machine learning algorithm and be applied to automatic tagging. Such a large research field is impossible to cover or summarize. We will focus on three algorithms recently used for this task. For a general introduction to the field, or for pointers to other interesting algorithms, see the future reading section.

We split machine learning methods into two classes: classification methods, and regressions and probabilistic methods. This refers to the kind of output the methods learn to produce, and it is a typical division between supervised learning methods. It is appropriate here since, as we saw before, our tagging data can exist as one or the other.

Classification Methods

Support vector machines (SVMs) are one of the most widely used machine learning algorithms, and have been applied in M. Mandel's papers to automatic tagging. Their performance as a classifier has been demonstrated and they can also handle regression, but a concern with it is the training speed, which is of order N^2, N being the number of examples (audio frames here). Mandel & Ellis (2008a) use SVMs in a multiple-instance learning framework (Dietterich et al., 1997) In this framework, supervision for training is provided not at the level of clips, but of at the level of collections of clips (e.g. tracks, albums, or artists), and the classifier's job is to predict whether a classification is relevant to each of the clips in that collection and to clips in a separate test set. For music, if an artist is tagged with “saxophone”, some of their clips should contain saxophone, but not necessarily all of them.

Boosting is a meta-algorithm in the sense that it works on top of other learning algorithms. For now on, we discuss boosting single-stumps, i.e. a decision tree with two leaves. Boosting has been used by the Gamme lab at the University of Montreal, first for genre classification (Bergstra et al., 2006), then for automatic tagging (Bertin-Mahieux et al., 2008; Eck et al., 2008; Kegl et al., 2008). In an iterative way, boosting finds the best possible classifier (in the set of all possible single-stumps) and adds it to a “strong classifier”. It then reweights the examples so that classifying the hard ones is more rewarding. As

single-stumps see only one dimension at a time, the information contained in those dimensions must be highly relevant. Hence, the authors used boosting in conjunction with aggregate features. Boosting training time scales linearly in the number of examples.

Regression and Probabilistic Methods

We already mentioned SVMs and neural networks that have extensions to handle the regression case. It is also the case for k-Nearest Neighbors (kNN), but it has been rarely used in the literature. We now focus our attention on the method used by the CAL group at UCSD (Turnbull, Barrington, Torres & Lanckriet, 2008).

Gaussian mixture is a learning algorithm that models a distribution using gaussians. In our case, audio frames are a point in the features space and we model their position. The assumption is that, if a new point (audio frame) is close in the feature space, it should be tagged the same way. The advantage of the Gaussian mixture model is that it gives a probability, i.e. the probability that a tag X was applied to an audio frame F. Gaussian mixtures are trained using the well-known EM algorithm (Bishop, 2006). Gaussian mixtures are more powerful (in terms of representation capacity) than an algorithm that only classifies because it estimates the likelihood of a data point. The drawback of such methods is that, if the goal is to be discriminative, resources are spent modeling the data. Specifically, if two classes are 99% the same, the generative approach tries to model each class and probably ignores the 1% difference between them, while the discriminative approach focuses on that 1% that is different. Another disadvantage is that training Gaussian mixtures are extremely demanding in computational resources as soon as the number of audio frames get large.

To overcome the computational complexity, Turnbull et al. use the **Hierarchical Gaussian mixture model** (HGMM), a method introduced for vision in (Vasconcelos, 2001). The idea is to train Gaussian mixtures on a subset of audio frames, for instance all frames from one song. This gives a single Gaussian mixture per song, meaning 500 Gaussian mixtures for the CAL500 dataset. Vasconcelos showed how to approximate all these mixture by a new Gaussian mixture. This divide-and-conquer approach is very efficient.

According to MIREX 2008 results and some comparisons done in (Bertin-Mahieux et al., 2008), HGMM seems the better performing algorithm for automatic tagging of music until MIREX 2009. However, Bertin-Mahieux et al. raise concern about the small size of the datasets that were used for the comparisons as it could favor HGMM.

Future reading

For a complete introduction to the machine learning field, we refer the reader to (Bishop, 2006) or (Duda et al., 2000).

K-nearest neighbors (KNN) is one of the most simple, and surprisingly effective, machine learning techniques. Due to its simplicity, KNN should be a first comparison again any new tagging algorithm. KNN has been used in (Sordo et al., 2007) to explicitly propagate labels among neighbors, neighbors being derived from audio features. See also (Cano & Koppenberger, 2004).

Trohidis et al. (2008) discuss four algorithms to classify audio into emotions. Three algorithms: **binary relevance** (BR), **label powerset** (LP), and **random *k*-labelsets** (RAKEL) are taken from their previous work (Tsoumakas et al., 2007) and **ML*k*NN** is described in (Zhang & Zhou, 2007). These are interesting methods because they aim at directly predicting a subset of classes. In our audio case, it could mean they build a predictor from a set of audio features to the subset of tags {“rock”, “metal”, “hardcore”}. These algorithms are clearly of interest, but since their relatively poor results in the MIREX 2008 contest, their application must be refined for audio and automatic tagging in particular.

Neural networks have not been been used in recent papers on automatic tagging of audio, but there is no reason why it would not work efficiently. In particular, it handles multi-label classification and regression cases, and new developments in the field (Bengio et al., 2007) allow the use of very large networks, hence possibly capturing highly complex relations between audio and tags. Neural networks have been used successfully for music genre classification in (Bergstra, 2006)..

Finally, Bergstra (2006) does a review of algorithms that have been used for genre classification and all of them have potential for automatic tagging.

As the size of datasets increases, algorithms need to scale efficiently. Online algorithms are trained by seeing only a few examples at a time, thus avoiding the need of an increasingly large memory. For online learning versions of some of the algorithms we presented, see (Bordes et al., 2005) (SVM), (Bradley & Schapire, 2008) (boosting) and (Declercq & Piater, 2008; Song & Wang, 2005) (gaussian mixture model).

EVALUATION

Evaluation may be the most problematic part of the automatic tagging of audio task. We can be more general and say that the task definition itself is problematic, and that is why the community has not agreed on a set of measure functions. We gave the goal of having a machine that can describe music the same way a human expert would. Thus, the perfect evaluation would be to have human experts grade the quality of the autotags produced by different algorithms. Unfortunately, such an evaluation is often impossible in practice, and other evaluation measures need to be used as an approximation for the judgment of an expert.

We split evaluation measures into three categories : direct measures, indirect measures, and human evaluation. Direct measures are tightly related to the machine learning setting. For instance, classification accuracy for a classification task or mean squared error for a regression task. Indirect measures imply using autotags in a model for another task, and measuring how well the model performs. For instance, what is the quality of recommendations based on autotags? Finally, human evaluations involve humans asserting the quality of the autotags.

Although the separation between the three types of evaluations is not always crisp, it helps questioning the results reported so far in the literature.

Direct Measures

A first measure, especially easy to compute if the problem is posed as a classification, is classification accuracy. It is the percentage of songs that were correctly identified as being tagged or not tagged with a particular label. We often see the two numbers presented separately, as the number of songs tagged and not tagged are usually very unbalanced. Also, for some applications, not predicting a label is not as penalizing as predicting a wrong label. Therefore, classification accuracy for positive examples (song being tagged) and negative examples (song not being tagged) should be analyzed independently.

This leads us to precision and recall (Figure 2), a pair of measures that tries to analyze similar aspects but from a retrieval point of view. More specifically, it asks the two questions: when I ask for a song with tag X, how often do I get one correctly? Then, can I get all songs tagged with X? One aspect can be more important than the other depending on the goal. Precision measures how often we tag a song with the correct label. Recall measures the fraction of relevant songs that are correctly identified.

$$precision = \frac{|relevant\ documents| \cap |documents\ retrieved|}{|document\ retrieved|}$$

$$recall = \frac{|relevant\ documents| \cap |documents\ retrieved|}{|relevant\ documents|}$$

Figure 2: formulas for precision and recall

Good recall avoids having an algorithm that never tags anything in order to avoid a mistake. However, if a user is looking for a "rock" song on the internet, he does not care to get billions of them, but the ones he receives should really be "rock", thus precision is predominant.

One usually wants a good combination of both precision and recall. We might also want to compare algorithms according to both, F-measure is the average of precision and recall. Two other methods are presented in the future reading section.

The problem is that all these measures represent slightly different aspects of a good tagging algorithm. We can either select one of them as a community standard and ignore the rest, use a large number of them and get overwhelmed with data (often contradictory), or devise new and more rounded methods of measuring a model's performance. We discuss the latter in the next subsections.

Indirect Measures

We have seen measures that are closely linked to what the algorithm is trained to do, for instance classification accuracy. These are not the only way to state whether our autotagging algorithm produces pertinent labels or not. We can use tags, and autotags, in derivate tasks and see if both perform as well. Eck et al. (2008) use tags to compute a similarity measure between artists and compare it against a ground truth similarity. Note that it is far from obvious how to build a ground truth similarity metric and they actually rely on one that is "good enough", details can be found in their paper. At the end, Eck et al. can claim that autotags do not perform as well as real tags, but clearly better than random. Thus, autotags "have merit". Other derivative tasks can easily be imagined. If one has access to an online radio station like Last.fm, one could look at user satisfaction with their recommendations when they search for a song by tags. Derivative tasks also enable us to add a human in the evaluation loop, and that is what we discuss in the next subsection.

Human Evaluation

The most direct human evaluation is to set up a panel of experts, or simply a large survey of individuals, and ask them to rate how appropriate a tag is for a particular song. If one started from clean data (from a survey for instance), one can hide some entries during the training of our algorithm and then test how likely the test entries are according to our model. However, there is one major limitation with this method. It is quite easy to find positive examples, i.e. a tag that was applied to a song. It is harder to determine that a tag should not have been applied to the song. The test set from a survey data is therefore half complete, and we still need a human to evaluate the output of an automatic tagger.

Here is a specific example of what we just said. Consider the band *U2*. It has obviously been tagged with "rock". Thus, if an algorithm produces that tag for a *U2* song, one can mark that as a success. Next, assume that the algorithm produced the tag "country". This does not fit perfectly the style of *U2*, but it is closer than "rap" for instance. If none of these tags were applied by listeners, both would be penalized the same way. However, a human judge would prefer an algorithm that decides that *U2* is more "country" than "rap". This illustrates the need for a human evaluation.

Setting up a panel to judge algorithms is difficult, as they are required to tag thousands or even millions of songs. A very promising approach, already considered in the section about obtaining labeled

data, is to use a game to encourage users to becoming judges. E. Law suggested we modify the Tagatune game for that purpose (Law, 2008). In Tagatune, two players listen to a song simultaneously and the goal is to determine if they are listening to the same one. Players are only allowed to communicate by typing labels which are visible to both of them and must make their decision based on these tags. If an automatic tagging system outputs appropriate tags, it should enable a human partner to decide if he is listening to the same song or not. Hence, the algorithm that yields the best average score for its human partners is considered the most accurate of all models. As we are writing, this evaluation experiment is underway, with many submissions from MIREX 2008 automatic tagging contest (Law et al., 2009).

Concerns about Current Evaluation Methods

Many researchers in the field feel we did not find the proper evaluation methods for automatic tagging, most of these concerns are expressed in (Law, 2008). We try to summarize the main arguments here. Classification accuracy is somewhat irrelevant if the testing data are noisy. One can always argue that, as long as the training and testing data come from the same distribution, the noise becomes part of the task and the evaluation is valid. That being said, when we consider that non only are the data noisy, but they are overwhelmed with negative examples (tag X was not applied to audio frame F), the confidence in the test set gets very low. The *U2* example applies here, the fact that *U2* is tagged neither "country" nor "rap" does not mean that both predictions are equally wrong.

Precision and recall are more relevant to actual applications of an automatic tagger. In particular, for an online radio like Last.fm, it is important to provide a "pop" song or "ska" song to a user that asked for that genre. Precision measures exactly that, and we argue that it is more important than recall in most real-life applications.

All tags are not born equals, and some are more popular than others. Turnbull et al. find that the most useful tags are the one that are neither the most popular nor the most obscure. Thus, we should consider evaluation methods that take the popularity of tags into account, but this is extremely difficult to do. For MIREX 2008, the beta-binomial model (Gelman et al., 2003) was used and one of its properties is to put more weight on tags that are better represented. According to Turnbull et al.'s finding, this can be problematic.

Finally, as we mention before, the size of the datasets is problematic. If the goal is to create a program that can be implemented on a very large scale like the Internet, one needs to measure the performance of algorithms on very large datasets. This has not been the case so far and it is likely to remain a problem for the years to come. Fortunately, indirect and human evaluations like we presented have the potential to circumvent the matter.

Future reading

Another direct evaluation measure is Area under the Receiver Operating Characteristic curve (AROC). AROC assumes that, for a tag X, you can order songs by how much they should be tagged with X. AROC computation can be found in numerous papers, for instance (Turnbull, Barrington, Torres & Lanckriet, 2008). Many of the concerns expressed above have been explored by the search engine community. We recommend the work of Herlocker et al. for a complete overview (Herlocker et al., 2000; Herlocker et al., 2004).

THREE PUBLISHED IMPLEMENTATIONS

In this section, we present three approaches that have been successfully implemented and published recently. According to the results of the 2008 MIREX contest on automatic tagging, they also

appear as the three best performing algorithms. We can therefore consider them as the state-of-the-art and a reference point for future research. Then, we briefly presents the MIREX 2008 results according to one measure, F-measure. The full results are available online and are too detailed to be presented here.

UCSD - CAL

A first successful implementation was done at the University of California in San Diego in the Computer Audition Laboratory (CAL) headed by Professor G. Lanckriet.

The feature they use are MFCC, delta-MFCC and delta-delta-MFCC over frames of approximately 23ms. The tagging data comes from CAL500 dataset. The machine learning algorithm is hierarchical gaussian mixture. Evaluation is done with precision / recall and AROC.

Publications about this work are: (Turnbull, Barrington, Torres & Lanckriet, 2007; Turnbull, Barrington, Torres & Lanckriet, 2008).

Columbia - LabROSA

A second method was developed by M. Mandel at Columbia in the Laboratory for the Recognition and Organization of Sound and Audio (LabROSA) headed by Professor D. Ellis.

The features used are the mean and unwrapped covariance of MFCCs and rhythmic features that measure modulations in a few gross frequency bands. The training data comes from the MajorMiner game. The machine learning algorithm is SVMs. Evaluation is done by measuring the classification accuracy on a test set with the same number of positive and negative examples, ensuring a constant baseline of 50%.

Publications about this work are: (Mandel & Ellis, 2008a; Mandel & Ellis, 2008b).

University of Montreal - GAMME

A third method was investigated at the GAMME laboratory headed by Professor D. Eck and in collaboration with Sun Microsystems (Search Inside the Music project, main researcher was P. Lamere).

The features used are aggregate spectrograms, MFCC and autocorrelation coefficients. The training data comes from Last.fm. The machine learning algorithm is multi-class boosting of single stumps. The evaluations include classification accuracy and artist similarity.

Publications about this work are: (Eck et al., 2007; Eck et al., 2008; Bertin-Mahieux et al., 2008; Kegl et al, 2008).

Results of MIREX 2008 Audio Tag Classification Task

In Table 3 we present the results of MIREX 2008 according to the F-measure. The complete and detailed results are available online. The first three teams are the implementations we presented above. Note that the "Smurf" team is a sort of control submission. The algorithm always tagged any audio with the most popular tags. The number of tags output was previously optimized to maximize the F-measure on a training set similar in size to the one used at MIREX. Also, the entry from G. Peeters was a generic algorithm which was submitted to all MIREX tasks, thus it was not optimized for automatic tagging.

Team	***Algorithm***
UCSD	HGMM
Columbia	SVM

University of Montreal	Boosting
Smurf	Tag popularity
Geoffrey Peters	ircamclassification (generic)
Trohidis et al.	RAKEL / ML*k*NN

Table 3: Submissions to MIREX 2008 audio tag classification task, teams ordered according to F-measure. When a team has multiple entries, only the best entry is reported. For more details, see: http://www.music-ir.org/mirex/2008/

Note on MIREX 2009 Audio Tag Classification Task

The results for MIREX 2009 were released just before this work went to press. Unfortunately, most of the teams from 2008 did not resubmit, but we can look at the results on the MajorMiner dataset, the same one used in 2008. Summarized results are shown in Table 4.

Team	***F-measure***
Lo et al., Taiwan (2009)	0.31
Tzanetakis, CAN (2009)	0.29
UCSD (2008)	0.28
Columbia (2008)	0.26
University of Montreal (2008)	0.19

Table 4: Winning entries from MIREX 2008 and MIREX 2009 audio tag classification task, teams ordered according to F-measure on the "major minor" dataset. When a team has multiple entries, only the best entry is reported. For more details, see: http://www.music-ir.org/mirex/2009/

As we can see from the F-measure, the accuracy of the systems are improving. Lo et al. (2009) used an ensemble classifier, something similar to boosting. Tzanetakis (2009) used two layers of SVM, one to learn each tag independently, and a second to take advantage of the correlations between tags. The detailed results from 2008 and 2009 are available on the MIREX website.

FUTURE RESEARCH DIRECTIONS

Automatic tagging of audio, and automatic tagging of multimedia in general, are active research fields and research directions are numerous. We want to highlight two improvements that we hope to see

in the next five years, namely the use of large and scalable algorithms, and the proper handling of sparse coding.

The future of audio and machine learning belongs to a fast online learning algorithm. The audio data is accessible, and so is metadata (tags for instance). The possibilities offered by such an amount of information are endless. From a machine learning point of view, we can describe any manifold that holds meaningful information, e.g. the subspace of scary audio sounds. From an industry point of view, we are able to spot any new trends or peculiar taste that might develop among listeners. The only problem is that we can not currently synthesize so much data. The perfect algorithm should be online because no matter how cheap memory gets, we will never be able to store everything in main memory at once. The perfect algorithm should be fast for obvious reasons, and being online helps. Let us add something there, the perfect algorithm will probably be simple, and its low complexity will be balanced by the quantity of data seen. Low memory usage, fast, and simple, it should be pure gain for developers, and it is made possible by the enormous amounts of data online.

On a more signal processing aspect, we are looking forward to an algorithm that will handle correctly sparse encodings, especially encodings performed in the time domain. As it is convincingly exposed in (Smith & Lewicki, 2005; Smith & Lewicki, 2006), frame-based audio features have inherent limitations, namely the frequency accuracy / time accuracy trade-off. Note that developing "sparse algorithms" is a general trend in machine learning research in the recent years. The recent work done by Weinstein and Moreno (2007) on audio fingerprinting is a good example of a successful application of sparse coding.

CONCLUSION

We described automatic tagging algorithms for audio as a set of four components, and we presented the state-of-the-art in each of these components. We also presented three successful implementation that have been published and that performed well at MIREX. Finally, we emphasize the current trend toward large-scale data, and explained why algorithms that handle such data should be investigated. In accordance with the goal of this paper, we presented an overview of the existing models and gave the appropriate pointers and references needed by a researcher trying to develop his own approach.

Automatic tagging is a step towards a better understanding of music by machines. One road map to achieve this goal consists of two steps: describing audio with meaningful labels, and using these labels to manage and discover new products. The second step has been intensively developed by web search engines, and it is the greatest incentive to create a good automatic tagging algorithm.

References

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2003) Support vector machines for multiple-instance learning, in *Advances in Neural Information Processing Systems 15, MIT Press.*

Aucouturier, J.J. & Pachet, F. (2003) Representing Musical Genre: A State of the Art, in *Journal of New Music Research*, 32, 83-93

Audioscrobbler, Web Services described at http://www.audioscrobbler.net/~data/webservices/

Barrington, L., Turnbull, D., Yazdani, M., & Lanckriet, G. (2009) Combining audio content and social context for semantic music discovery, in *Proceedings of the 32th annual international ACM SIGIR conference on Research and development in information retrieval, ACM.*

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007) Greedy layer-wise training of deep networks, in *Advances in Neural Information Processing Systems 19, MIT Press,* 153-160

Bergstra, J. (2006) Algorithms for classifying recorded music by genre, University of Montreal

Bergstra, J., Casagrande, N., Erhan, D., Eck, D. & Kégl, B. (2006) Aggregate Features and AdaBoost for Music Classification, in *Machine Learning*, 65, 473-484

Bertin-Mahieux, T., Eck, D., Maillet, F. & Lamere, P. (2008) Autotagger: a model for Predicting social tags from acoustic features on large music databases, in Journal of New Music Research, special issue: "From genres to tags: Music Information Retrieval in the era of folksonomies.", 37

Bishop, C. (2006) Pattern Recognition and Machine Learning, Springer Verlag.

Bordes, A., Ertekin, S., Weston, J. & Bottou, L. (2005) Fast Kernel Classifiers with Online and Active Learning, in *Journal of Machine Learning Research,* 6, 1579-1619

Bradley, J.K., & Schapire, R. (2008) FilterBoost: Regression and Classification on Large Datasets, in *Advances in Neural Information Processing Systems 20, MIT Press.*

Cano, P. & Koppenberger, M. (2004) Automatic sound annotation, in *lEEE workshop on Machine Learning for Signal Processing*, 391-400

Celma, O., Cano, P. & Herrera, P. (2006) Search Sounds: An audio crawler focused on weblogs, in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR 2006)*

Declercq, A. & Piater, J.H. (2008) Online Learning of Gaussian Mixture Models - a Two-Level Approach, in *3rd International Conference on Computer Vision Theory and Applications (VISAPP)*, 605-611

Dietterich, T., Lathrop, R. & Lozano-Pérez, T., Solving the multiple instance problem with axis-parallel rectangles, in *Artificial Intelligence, Elsevier Science Publishers Ltd.,* 1997, 89, 31-71

Duda, R.O., Hart, P.E. & Stork, D.G. (2000) *Pattern Classification, Wiley-Interscience Publication.*

Eck, D., Bertin-Mahieux, T. & Lamere, P. (2007) Autotagging music using supervised machine learning, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007)*, 2007

Eck, D., Lamere, P., Bertin-Mahieux, T. & Green, S. (2008) Automatic generation of social tags for music recommendation, in *Advances in Neural Information Processing Systems 20*, *MIT Press.*

Garnder, T.J. & Magnasco, M.O. (2006) Sparse time-frequency representations, in *Proceedings of the National Academy of Science*, 103, 6094-6099

Gelman, A., Carlin, J., Stern, H. & Rubin, D. (2003) *Bayesian Data Analysis, Second Edition, Chapman & Hall/CRC*

Gold, B. & Morgan, N. (1999) Speech and Audio Signal Processing: Processing and *Perception of Speech and Music, John Wiley & Sons, Inc.*

Grosse, R., Raina, R., Kwong, H. & Ng, A.Y. (2007) Shift-Invariant Sparse Coding for Audio Classification, in *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence*, 2007

Herlocker, J.L., Konstan, J.A. & Riedl, J.T. (2000) Explaining collaborative filtering recommendations, in *Computer Supported Cooperative Work*, 241-250

Herlocker, J.L., Konstan, J.A., Terveen, L.G. & Riedl, J.T. (2004) Evaluating collaborative filtering recommender systems, in *ACM Trans. Inf. Syst., ACM Press*, 22, 5-53

Huang, B. and Jebara, T. (2007) Loopy Belief Propagation for Bipartite Maximum Weight b-Matching, in *Artificial Intelligence and Statistics.*

Kégl, B., Bertin-Mahieux, T. & Eck, D. (2008) Metropolis-Hastings sampling in a FilterBoost music classifier, in *ICML Workshop on Music and Machine Learning.*

Kim, Y., Schmidt, E. & Emelle, L. (2008) Moodswings: a collaborative game for music mood label collection, in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008).*

Knees, P., Pohle, T., Schedl, M., Schnitzer, D. & Seyerlehner, K. (2008) A document-centered approach to a natural language music search engine, in *European Conference on Information Retrieval (ECIR).*

Lamere, P. (2008) Semantic tagging and music information retrieval, in *Journal of New Music Research, special issue: "From genres to tags: Music Information Retrieval in the era of folksonomies."*

Law, E., v. Ahn, L., Dannenberg, R. & Crawford, M. (2007) TagATune: a game for music and sound annotation, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*

Law, E. (2008) The problem of accuracy as an evaluation criterion, in *ICML Workshop on Evaluation Methods in Machine Learning.*

Law, E., West, K., Mandel, M., Bay, M. & Downie, S. (2009) Evaluation of algorithms using games: the case of music tagging, in *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR 2009).*

Li, X., Chen, L., Zhang, L., Lin, F. & Ma, W.-Y. (2006) Image annotation by large-scale content-based image retrieval, in *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia.*

Lo, H.-Y., Wang, J.-C. & Wang, H.M. (2009) An ensemble method for MIREX audio tag classificationMusic Information Retrieval Evaluation Exchange (MIREX), Kobe, 2009, available at http://www.music-ir.org/mirex/2009/

Mandel, M. & Ellis, D. (2007) A web-based game for collecting music metadata, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*

Mandel, M. & Ellis, D. (2008a) Multiple-instance learning for music information retrieval, in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008).*

Mandel, M. & Ellis, D. (2008b) A web-based game for collecting music metadata, in *Journal of New Music Research, special issue: "From genres to tags: Music Information Retrieval in the era of folksonomies.".*

Manzagol, P.-A., Bertin-Mahieux, T. & Eck, D. (2008) On the use of sparse time relative auditory codes for music, *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008).*

Plumbley, M., Abdallah, S., Blumensath, T. & Davies, M. (2006) Sparse representations of polyphonic music, in *Signal Processing*, 86, 417-431

Smith E. & Lewicki, M.S. (2005) Efficient coding of time-relative structure using spikes, in *Neural Computation, MIT Press*, 17, 19-45

Smith, E. & Lewicki, M.S (2006) Efficient auditory coding, in *Nature*, 439, 978-982

Song, M. & Wang, H. (2005) Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series.*
Sordo, M., Laurier, C. & Celma, O. (2007) Annotating music collections: how content-based similarity helps to propagate labels, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*
Torres, D., Turnbull, D., Barrington, L. & Lanckriet, G. (2007) Identifying words that are musically meaningful, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*
Trohidis, K., Tsoumakas, G., Kalliris, G. & Vlahavas, I. (2008) Multi-label classification of music into emotions, in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008).*
Tsoumakas, G. & Vlahavas, I. (2007) Random k-Labelsets: an ensemble method for multilabel classification, in *ECML '07: Proceedings of the 18th European conference on Machine Learning.*
Turnbull, D., Liu, R., Barrington, L. & Lanckriet G. (2007), A game-based approach for collecting semantic annotations of music, in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*
Turnbull, D., Barrington, L., Torres, D. & Lanckriet, G. (2007) Towards musical query-by-semantic-description using the CAL500 data set, in *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval.*
Turnbull, D., Barrington, L. & Lanckriet, G. (2008) Five Approaches to Collecting Tags for Music, in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008).*
Turnbull, D., Barrington, L., Torres, D. & Lanckriet, G. (2008) Semantic annotation and retrieval of music and sound effects, in *IEEE Transactions on Audio, Speech & Language Processing*, 16
Tzanetakis, G. (2009) Marsyas submissions to MIREX 2009, in Music Information Retrieval Evaluation Exchange (MIREX), Kobe, available at http://www.music-ir.org/mirex/2009/
Vasconcelos, N. (2001) Image indexing with mixture hierarchies, in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001).*
Weinstein, E. & Moreno, P. (2007) Music Identification with Weighted Finite-State Transducers, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP).*
Whitman, B. & Ellis, D. (2004) Automatic record reviews, in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004).*
Zhang, M. & Zhou, Z. (2007) ML-KNN: A lazy learning approach to multi-label learning *Pattern Recognition, Elsevier Science Inc., 40*, 2038-2048

INDEX TERMS

machine learning
music
social tags
automatic tagging
evaluation
game
features
recommendation
labeled data
Music Information Retrieval Evaluation eXchange (MIREX)

KEY TERMS & DEFINITIONS (SUBHEAD 1 STYLE)

(Please refer to author checklist to see if this applies to your submission)

Keyword: Definition of Keyword.

You are requested to provide 7-10 key terms related to the topic of your chapter and provide clear and concise definitions (in your own words) for each term. Place your terms and definitions after the references section of your chapter.

Machine learning: linked to artificial intelligence, machine learning is a science discipline that is concerned with functions that can algorithmically predict an output based on some data.

Social tags: a user-generated keyword associated with some resource, in our case audio.

Automatic tagging: the use of a machine learning algorithm to apply a tag to some resource.

Feature: a descriptor of some resource, e.g. beats per minute for a music performance.

Music Information Retrieval Evaluation eXchange (MIREX): a set of contests held each year at the ISMIR conference

Evaluation: in our case, some measure of how well a particular tag applies to some audio.

Recommendation: the process of suggesting a new and relevant resource to some user.