Automatic Selection of Acoustic Features using a Lazy Spitting Method

Simon Bourguigne, Pablo Daniel Agüero, Juan Carlos Tulli, Esteban Lucio Gonzalez, and Alejandro Jose Uriz

Facultad de Ingeniería, Universidad Nacional de Mar del Plata,
Juan B. Justo 4302, 7600 Mar del Plata, Argentina
{sbourguigne,pdaguero}@fi.mdp.edu.ar
http://200.0.183.36/pegasus

Abstract. The increasing amount of music data approaching the scale of ten million of tracks poses the challenge of organizing such huge information. Audio Tag Classification is a sub-area in the Music Information Retrieval. Its objective is predicting human motivated tags given the acoustic data. One major problem in this procedure is the training of the classifier. An important step in the training is the selection of the appropiate acoustical features. This paper explores two selection approaches: greedy and spitting. Experimental results indicate that the proposed spitting algorithm has a superior performance both in classification (F-measure score) and speed (lower computational requirements).

Key words: music information retrieval, audio tag classification, greedy algorithm, spitting algorithm

1 Introduction

Music is one of the most popular types of online information and there are now hundreds of music streaming and download services operating on the World-Wide Web. Some of the music collections available are approaching the scale of ten million tracks and this has posed a major challenge for searching, retrieving, organizing music content, and developing methods for managing collections of musical material for preservation, access, research, and other uses [2][5]. Motivated by this challenges, an interdisciplinary area known as Music Information Retrieval (MIR) has emerged, encompassing areas such as computer science and information retrieval, musicology and music theory, audio engineering and digital signal processing, cognitive science, library science, publishing, and law[5].

The idea of applying automatic information retrieval (IR) techniques to music actually dates back to the $1960'$s[6]. But in particular, MIR has been growing during the past decade out of an explosion of interest in networked collections of musical material in digital form[5]. Consider, for example, the task of organizing a large music repository, this is a tedious and time-intensive job, especially when the traditional solution of manually annotating semantic data to the audio is chosen[7]. These semantic data are commonly refered to as tags. Many published results show that this problem can be tackled using machine learning techniques but it seems, however, that no one has yet found an appropriate algorithm to solve this challenge[1]. The problem of predicting these tags is called automatic tagging. Different groups have been trying to tackle the problem, 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)[1]

When trying to address this problem in terms of machine learning, it is first necessary to determine what set of words people would be likely to use to describe a song and then train a system that can automatically predict what subset of those words better describes a given song. An attempt to solve the first problem has been made by Mandel and Ellis[9] by creating an online game to harvest this descriptions. Analyzing the results they built a dataset known as MajorMiner.

This paper focuses on the problem of tag prediction or automatic tagging using MajorMiner to train a classifier. This classifier is going to be fed by a set of features such as: Mel Frecuency Cepstral Coeficients (MFCC), Spectral Roll-Off, Zero-Crossings Rate (ZCR), etc. These features are computed for each audio frame of a given song which are considered in a collection that ignores their order. Later on they are aggregated by computing there mean and standard deviation values. The goal in this paper is to select the optimal combination of acoustical features for each tag.

This paper is organized as follows. Section 2 describes the task of automatic audio tagging, explaining briefly each subtask. At the end of this section, the proposed feature selection algorithm is shown. Section 3 shows the experimental results with the greedy and spitting feature selection approaches. Finally, conclusions and future work are drawn in Section 5.

2 Automatic Audio Tagging

The task of automatic audio tagging consists of labelling a set of songs with a predefined group of tags. A tag is a user generated keyword associated with some resource, in this case audio. In general audio tracks (or segments of a track) are tagged, but it is also possible to talk about tagging albums or artists by aggregating predictions made over tracks.

This task can be divided in several subtasks: creating a corpus of labelled data used as example, extract useful acoustic features, training a classifier using machine learning techniques, and evaluate the performance of the resulting classifier using cross-validation techniques and classification measures.

2.1 Corpus of labelled data

A proper dataset of labeled [audio,tag] pairs is necessary to let machine learning techniques to find relationships between acoustic features and tags. The machine learning assumption is that if enough examples are shown to an algorithm, the correlation between acoustic features and tags will become clear. However, the following tradeoff remains: gathering more examples help, but as a consequence it is necessary 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.

There have been many attempts to build datasets. In the procedure for the generation of the dataset MajorMiner [9], 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 produced by other resources, such as Last.fm.

2.2 Useful audio features

There are many audio features proposed in the literature that range from time domain based features to spectral based features. This paper uses an in house feature extractor named Ursula which generates the following features:

- Linear Predictive Coding Coefficients: it is a tool used mostly in audio signal processing and speech processing for representing the spectral envelope of a digital signal of speech in a compact form, using the information of a linear predictive model.
- Line Spectral Pairs: they are used to represent linear prediction coefficients (LPC) for transmission over a channel. LSPs have several properties (e.g. smaller sensitivity to quantization noise) that make them superior to direct quantization of LPCs.
- Mel-frequency cepstrum coefficients: it is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.
- Spectral centroid: it is a measure used in digital signal processing that indicates where the “center of mass” of the spectrum is located.
- Spectral flux: it is a measure of how quickly the power spectrum of a signal is changing, calculated by comparing the power spectrum for one frame against the power spectrum from the previous frame.
- Spectral flatness: it is a measure used in digital signal processing to characterize an audio spectrum and quantify how tone-like a sound is, as opposed to being noise-like.
- Spectral crest factor: it indicates how flat or “peaky” the power spectral density is in a given subband.

These features are aggregated into texture windows with a length of M frames. The aggregation consists in calculating the mean and standard deviation for each acoustic feature for each texture window. Later on, the sequence of texture vectors is collapsed into a single feature vector representing the entire audio clip description by taking again the mean and standard deviation of all the texture windows. This process produces mean-mean, mean-std, std-mean and std-std values for each acoustic feature of the clip. This approach is the same one used by the software Marsyas which is one of the most widely used tools for MIR[3].

2.3 Training a classifier using machine learning

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 tags and audio features can be seen as a machine learning algorithm and be applied to automatic tagging.

Support vector machines (SVMs) are one of the most widely used machine learning algorithms, and have been applied in many papers to automatic tagging[1][8][10]. A support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data points of any class (so-called functional margin). In general, the larger the margin the lower the generalization error of the classifier.

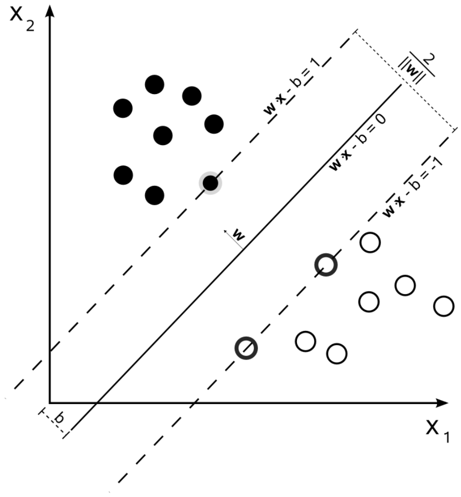

x_2
$\mathbf{w}\cdot\mathbf{x}-b=1$
$\frac{2}{\|\mathbf{w}\|}$
$\mathbf{w}\cdot\mathbf{x}-b=0$
$\mathbf{w}\cdot\mathbf{x}-b=-1$
$\mathbf{w}$
b
x_1

Fig. 1. Maximum-margin hyperplane and margins for an SVM trained with samples from two classes.

In many supervised learning problems, feature selection is important for a variety of reasons: generalization performance, running time requirements, and constraints and interpretational issues imposed by the problem itself. Support Vector Machines are not an exception. It is important to select a subset of features while preserving or improving the discriminative ability of a classifier.

As a brute force search of all possible features is a combinatorial problem, it is necessary to take into account both the quality of solution and the computational expense of any given algorithm.

Greedy methods are a simple heuristic solution to such problem. The number of features included in the feature vector grows step by step, each stage taking the results of the previous stage into account. The greedy algorithm begins with an empty initial feature vector, and in each stage appends an additional feature that contributes to a better global performance of the classifier.

A different approach is taken by Francois[4]; instead of eating features, they train with all of them and spit the most useless one, they re-train with the new set of features and keep on spitting until they stop according to some predefined criteria. They called this the spitting method.

The algorithm proposed in this paper shares the spitting behaviour, but as features are being spat, the SVM is not re-optimized. This algorithm, named lazy spitting method, begins with a full feature vector, and in just one stage deletes all the features that once removed do not impact in the global result of the classifier.

A more detailed description of the steps of the lazy spitting training algorithm are:

- **Initialization** All features are included in the initial vector, and optimal parameters C and W of the SVM are estimated using a grid search algorithm. $C > 0$ is the penalty parameter of the error term of the classifier, and W is a penalty of the wrong classification for positive (+1) and negative (-1) examples.
- **Feature evaluation** Each feature is individually deleted to evaluate the impact in the global performance of the classifier. If such performance is better, the feature is marked for future deletion.
- **Feature deletion** All feature marked for deletion are removed from the feature vector.
- **Final parameter tuning** Optimal parameters C and W of the SVM are estimated using a grid search algorithm with the remaining features in the input vectors.

The reasoning behind the proposed spitting training method is the stability of the optimal parameters C and W after the deletion of one feature. If such parameters are still optimal after the removal, the analysis of the importance of such feature will not be misleaded. However, the multiple remotion in the third step may lead to a suboptimal classifier if C and W are kept the same. Hence, it is necessary to perform a new parameter tuning to obtain a final optimal classifier.

3 Experiments

The experiments performed in this paper are focused in the evaluation of the improvement in the classification scores of SVMs by the selection of the appropiate features for each tag using greedy and spitting algorithms.

3.1 Classification Model

The classification model is a SVM with a variable size input feature vector and one output that can get the values +1 when the tag is set in the clip, and -1 if it is not set. One SVM is individually trained for each tag, using the feature selection and parameter tuning algorithm shown in Section 2.

The SVM software used in the experiments is LIBSVM. LIBSVM is a library for Support Vector Machines (SVMs) that has gained wide popularity in machine learning and many other areas. The parameters tuned in the linear kernel used in the experiments were C and W.

3.2 Dataset

MajorMiner tags dataset was used in the experiments. The tags included in this corpus belong to the following categories:

- Genre (e.g: rock, pop, electronic, hip hop).
- Style (e.g: drum-and-bass).
- Instruments (e.g: piano, drum-machine, strings).
- Tempo (e.g: fast, slow).
- Dynamics (e.g: loud, soft).
- Vocal style (e.g: vocal, vocals).

The MajorMiner game has collected a total of about 73000 taggings, 12000 of which have been verified by at least two users. In these verified taggings, there are 43 tags that have been verified at least 35 times, for a total of about 9000 verified uses. The music of this corpus consists of 2300 clips selected at random from 3900 tracks.

3.3 Evaluation Metrics

In the context of classification tasks, the terms true positives (tp), true negatives (tn), false positives (fp) and false negatives (fn) are used to compare the classification of an item (the tag assigned to the item by a classifier) with the desired correct classification (the tag the item actually belongs to).

Precision and recall are then defined as:

$$\text{Precision} = \frac{tp}{tp + fp} \tag{1}$$

$$\text{Recall} = \frac{tp}{tp + fn} \tag{2}$$

A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

F-measure is the performance metric used in this paper.

4 Results

The experimental results with twenty fold cross-validation is shown in Figure 2. The first columns are the tags under evaluation, second and third columns are the mean and standard deviation of the F-measure score of the folds when spitting algorithm is used for feature selection, and fourth and fifth columns are the mean and standard deviation of the F-measure score of the folds when greedy algorithm is used. The tags are ordered according to their frequency. Drums is the more frequent tag, and r&b (rhythm and blues) is the less frequent.

The sixth column is the difference between the mean F-measure of the second column (spitting algorithm) and the fourth column (greedy algorithm). The backgroud color light gray indicates that the mean F-measure of the spitting algorithm is better than the mean F-measure of the greedy algorithm. Dark gray points out the opposite.

The results indicate a higher number of positive differences in the mean F-measure in favor of the spitting algorithm. Therefore, after these experiments, it is possible to conclude that the spitting algorithm has a superior performance compared with the greedy approach for these experimental conditions.

The global results in terms of F-measure follows the procedure of MIREX 2010, which means averaging the F-measure of all clips to obtain a global score. The global F-measure reveals that the spitting algorithm has a small positive difference with respect to the greedy approach. However, such small difference is not statistically significant and more experiments are necessary to do better significance tests.

Although spitting and greedy algorithms have alike global performances, the former has an important advantage in terms of training speed. The training time for spitting algorithm is ten times faster than the greedy approach because it removes the unimportant features in just one stage.

5 Conclusions

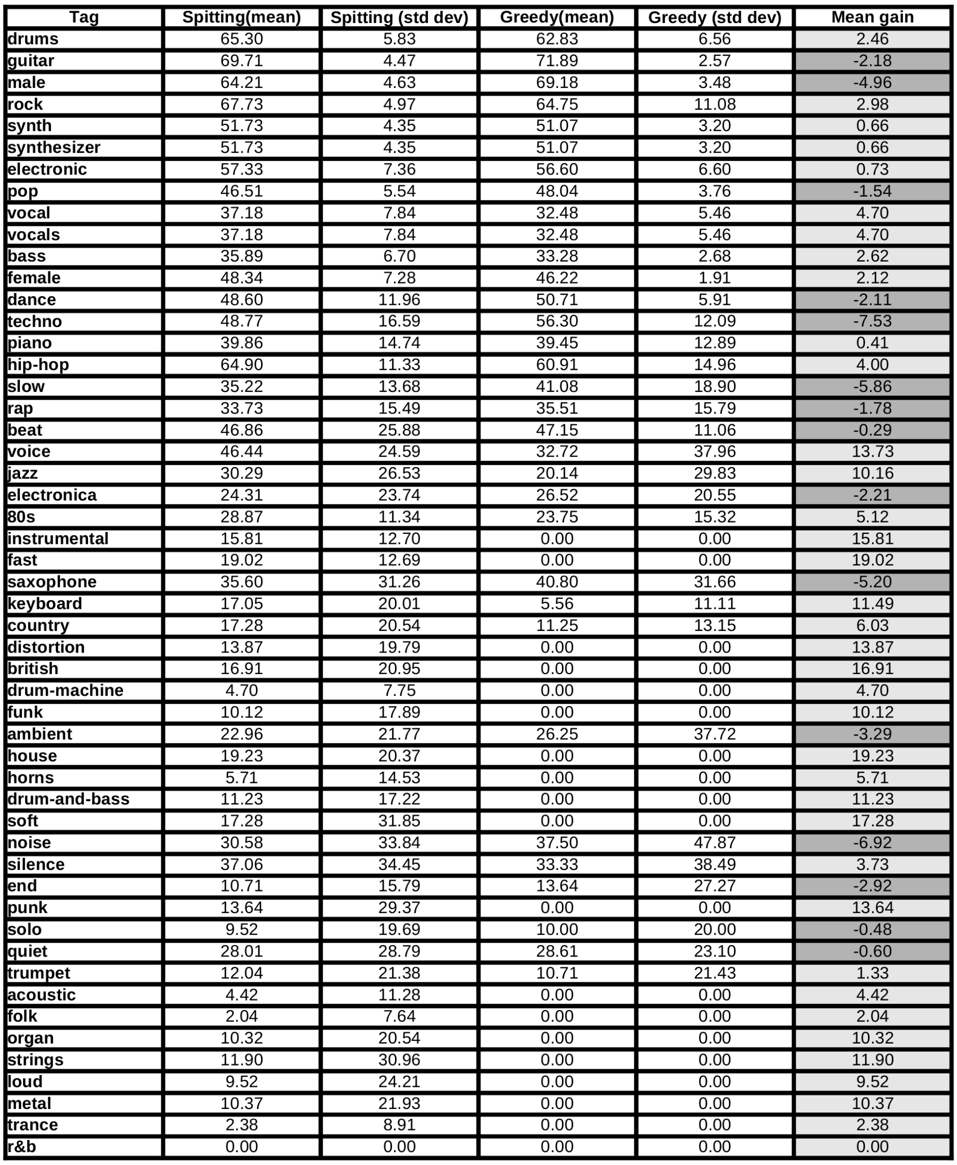
In this paper was presented an introduction of the work in the area of Music Information Retrieval at the Engineering Faculty. The main goal was to obtain a feature selection algorithm to improve the results of support vector machine classifiers in the task of automatic audio tagging.

Experimental results show that the proposed spitting algorithm has better F-measure scores that the baseline greedy algorithm. Another important result is the reduction of the training time by a factor of ten, which is crucial to participate in MIREX (24hs limitation to train a fold).

Future work will focus in the evaluation of additional acoustic features and aggregation techniques to improve F-measure scores.

References

1. Bertin-Mahieux, T., Eck, D., Mandel, M.: Automatic tagging of audio: The state-of-the-art. In: Wang, W. (ed.) Machine Audition: Principles, Algorithms and Systems. IGI Publishing (2010), in press
2. Casey, M.A., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., Slaney, M.: Content-based music information retrieval: Current directions and future challenges. In: Proceedings of the IEEE. vol. 96, pp. 668–696 (2008)
3. Downie, J., Byrd, D., Crawford, T.: Ten years of ismir: Reflections on challenges and opportunities. In: Proceedings of the Tenth International Conference on Music Information Retrieval. pp. 34–41 (2009)
4. Francois, H., Boeffard, O.: The greedy algorithm and its application to the construction of a continuous speech database. In: Proceedings of LREC-2002. vol. 5, pp. 1420–1426 (2002)
5. Futrelle, J., Downie, J.S.: Interdisciplinary research issues in music information retrieval: Ismir 2000-2002. In: Journal of New Music Research. vol. 32, pp. 121–131 (2003)
6. Kassler, M.: Toward musical information retrieval. In: Perspectives of New Music. vol. 2, pp. 59–67 (1966)
7. Lidy, T., Rauber, A.: Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In: Proceedings of the Sixth International Conference on Music Information Retrieval. pp. 34–41 (2005)
8. Mandel, M., Ellis, D.: Song-level features and support vector machines for music classifcation. In: Proceedings of the Sixth International Conference on Music Information Retrieval. pp. 594–599 (2005)
9. Mandel, M., Ellis, D.: A web-based game for collecting music meta-data. In: Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR). pp. 369–374 (2007)
10. Xu, C., Maddage, N., Shao, X., Cao, F., Tian, Q.: Musical genre classifcation using support vector machines. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. pp. 429–432 (2003)



Tag	Spitting(mean)	Spitting (std dev)	Greedy(mean)	Greedy (std dev)	Mean gain
drums	65.30	5.83	62.83	6.56	2.46
guitar	69.71	4.47	71.89	2.57	-2.18
male	64.21	4.63	69.18	3.48	-4.96
rock	67.73	4.97	64.75	11.08	2.98
synth	51.73	4.35	51.07	3.20	0.66
synthesizer	51.73	4.35	51.07	3.20	0.66
electronic	57.33	7.36	56.60	6.60	0.73
pop	46.51	5.54	48.04	3.76	-1.54
vocal	37.18	7.84	32.48	5.46	4.70
vocals	37.18	7.84	32.48	5.46	4.70
bass	35.89	6.70	33.28	2.68	2.62
female	48.34	7.28	46.22	1.91	2.12
dance	48.60	11.96	50.71	5.91	-2.11
techno	48.77	16.59	56.30	12.09	-7.53
piano	39.86	14.74	39.45	12.89	0.41
hip-hop	64.90	11.33	60.91	14.96	4.00
slow	35.22	13.68	41.08	18.90	-5.86
rap	33.73	15.49	35.51	15.79	-1.78
beat	46.86	25.88	47.15	11.06	-0.29
voice	46.44	24.59	32.72	37.96	13.73
jazz	30.29	26.53	20.14	29.83	10.16
electronica	24.31	23.74	26.52	20.55	-2.21
80s	28.87	11.34	23.75	15.32	5.12
instrumental	15.81	12.70	0.00	0.00	15.81
fast	19.02	12.69	0.00	0.00	19.02
saxophone	35.60	31.26	40.80	31.66	-5.20
keyboard	17.05	20.01	5.56	11.11	11.49
country	17.28	20.54	11.25	13.15	6.03
distortion	13.87	19.79	0.00	0.00	13.87
british	16.91	20.95	0.00	0.00	16.91
drum-machine	4.70	7.75	0.00	0.00	4.70
funk	10.12	17.89	0.00	0.00	10.12
ambient	22.96	21.77	26.25	37.72	-3.29
house	19.23	20.37	0.00	0.00	19.23
horns	5.71	14.53	0.00	0.00	5.71
drum-and-bass	11.23	17.22	0.00	0.00	11.23
soft	17.28	31.85	0.00	0.00	17.28
noise	30.58	33.84	37.50	47.87	-6.92
silence	37.06	34.45	33.33	38.49	3.73
end	10.71	15.79	13.64	27.27	-2.92
punk	13.64	29.37	0.00	0.00	13.64
solo	9.52	19.69	10.00	20.00	-0.48
quiet	28.01	28.79	28.61	23.10	-0.60
trumpet	12.04	21.38	10.71	21.43	1.33
acoustic	4.42	11.28	0.00	0.00	4.42
folk	2.04	7.64	0.00	0.00	2.04
organ	10.32	20.54	0.00	0.00	10.32
strings	11.90	30.96	0.00	0.00	11.90
loud	9.52	24.21	0.00	0.00	9.52
metal	10.37	21.93	0.00	0.00	10.37
trance	2.38	8.91	0.00	0.00	2.38
r&b	0.00	0.00	0.00	0.00	0.00

Fig. 2. Mean and standard deviation of F-Measure for spitting and greedy selection algorithm for each tag