XIV Reunión de Trabajo Procesamiento de la Información y Control -- RPIC 2011, pp. 459-464, 2011.

Assessment on Automatic Acoustic Features Selection for Automatic Tag Classification

Simon Bourguigne† Pablo Daniel Agüero† Juan Carlos Tulli†
Esteban Lucio Gonzalez† Alejandro Jose Uriz‡

†Facultad de Ingeniería
Universidad Nacional de Mar del Plata, Argentina
sbourguigne@fi.mdp.edu.ar
‡CONICET, Facultad de Ingeniería
Universidad Nacional de Mar del Plata, Argentina

Abstract— **The increasing amount of musical data approaching the scale of tens of millions of tracks poses the challenge of organizing such a huge amount of information. Audio Tag Classification is a sub-area in Music Information Retrieval. It's objective is being able to predict human motivated tags given the acoustic data. One major problem in this attempt is training a classifier. An important step in training a model is the selection of the appropriate acoustical features. This paper explores two selection approaches: spitting and lazy spitting. Experimental results indicate that the proposed lazy spitting algorithm has a superior performance both in classification (F-measure score) and speed (lower computational requirements).**

Keywords— **music information retrieval, audio tag classification, spitting algorithm, lazy 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 [3][8]. 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 [8].

The idea of applying automatic information retrieval (IR) techniques to music actually dates back to the 1960′s [9]. 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 [8]. 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 [10]. These semantic data are commonly referred 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 [2]. 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) [2].

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 [12] 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 Frequency Cepstral Coefficients (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 (bag-of-frames approach). Later on they are aggregated by computing their mean and standard deviation values. The goal in this paper is to select the optimal combination of acoustical features for each tag, and two approaches are explored: spitting and lazy spitting.

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 spitting and lazy spitting feature selection approaches. Finally, conclusions and future work are drawn in Section 4.

2 AUTOMATIC AUDIO TAGGING

The task of automatic audio tagging consists of labeling 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 labeled data used as example, extracting useful acoustic features, training a classifier using machine learning techniques, and evaluating the performance of the resulting classifier using cross-validation techniques and classification measures.

2. 1 Corpus of labeled data

A proper dataset of labeled [audio,tag] pairs is necessary to let machine learning techniques 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 trade-off 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 MajorMiner [12] dataset, 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 deliberately erroneous 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: they represent the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear frequency mel scale.
- 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 sub-band.

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 within each texture window, resulting in two sequences (mean and std) for each feature. Later on, each sequence is collapsed into a single feature vector representing the entire audio clip by taking again the mean and standard deviation of each sequence. 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 [5].

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 they have shown to be a useful technique in the task of music classification [2][11][13]. 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.

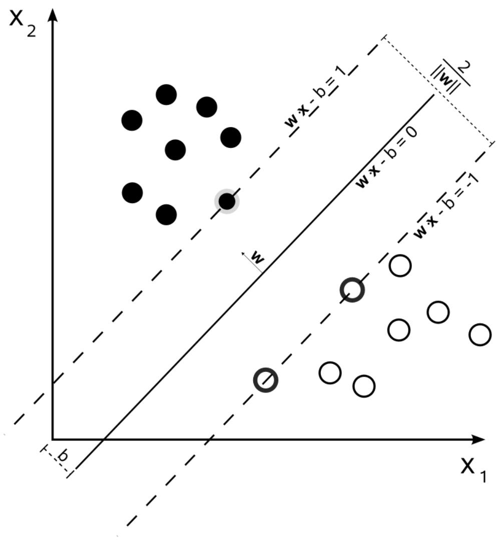

Figure 1: Maximum-margin hyperplane and margins for an SVM trained with samples from two classes.

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 François [7]; instead of "eating" features, they train with all of them and in each iteration they "spit" the most useless one, after that 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 behavior, but features are only marked to be spat. Later on the SVM is re-optimized when all marked features are finally deleted (or spat). 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 momentarily 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 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. The algorithm, doesn't impose a stopping criteria. As it is discussed in Section 3. 4, a single-pass implementation was chosen.

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 appropriate features for each tag using spitting and lazy 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 [4]. 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. A more detailed insight on the F-measure can be found at Downie et al. [6].

3. 4 Performed experiments

In this paper we performed five experiments which are detailed below:

- **Best model (BEST)**. This experiment evaluates how good the lazy spitting method can perform when it knows the best C and W parameters, and acoustic features. These parameters and features set are obtained using the proposed method but the optimization is performed by looking at the test set and obtaining the F-measure from it. Doing this modification the results are biased towards the testing data. It is an oracle behavior in the training to discover “How high is the sky?” [1] for this particular problem.
- **No feature selection (NS)**. No selection of acoustic features for deletion is also explored to uncover the usefulness of spitting and lazy spitting algorithms. In these experiments only C and W were optimized.
- **Spitting algorithm (SP)**. Approach proposed by François [7]; 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. In this case, the used criteria is to iterate until the F-measure decreases, in which case we keep the previous feature set.
- **Lazy spitting algorithm (LSP)**. Approach proposed in this paper to reduce the training speed time. In order to meet the time requirements imposed by MIREX a single-pass implementation was adopted.

3. 5 Results

The experimental results with twenty fold cross-validation is shown in Table 1. The first column shows the tags under evaluation. Second, third, fourth and fifth columns show the mean of the F-measure score of the folds for BEST, NS, SP and LSP experiments. The tags are ordered according to their relative frequency. Drums is the more frequent tag, and r&b (rhythm and blues) is the less frequent.

The sixth column (LSP vs NS) is the difference between the mean F-measure of the third column (No Selection) and the fifth column (Lazy Spitting). The red color indicates that the mean F-measure of the spitting algorithm is worse than the mean F-measure of the NS experimental condition.

The results indicate a higher number of positive differences in the mean F-measure in favor of the lazy spitting algorithm. Therefore, after these experiments, it is possible to conclude that the lazy spitting algorithm has a superior performance compared to the no selection approach for these experimental conditions.

The seventh column (LSP vs SP) shows the difference between the mean F-measure of the fourth column (Spitting) and the fifth column (Lazy Spitting). These results also indicate a higher number of positive differences in the mean F-measure in favor of the lazy spitting algorithm. Therefore, the lazy spitting algorithm has a superior performance compared with the spitting approach for these experimental conditions.

The global results in terms of F-measure reveal that the lazy spitting algorithm has a positive difference with respect to no selection and spitting approaches. Such difference is significant and encourages the use of lazy spitting algorithm in future MIREX meetings.

Lazy spitting algorithm also has an important advantage in terms of training speed. The experiments were run on a AMD^{TM}Athlon II X4 640 Processor (3GHz) with 8GB RAM. Table 2 shows the minimum, average and maximum times to train all the classifiers for each fold. The methods shown are just the two spitting algorithms under evaluation. The training time for lazy spitting algorithm is 33% faster than the spitting approach because it removes the unimportant features in just one stage. Although the gain in time might seem small, when the experiments take days to be executed, such gain becomes important.

Table 2: Spitting methods speed performance (in seconds)

Method	**Min**	**Average**	**Max**
Spitting method	844	892	920
Lazy spitting method	645	665	700

4 CONCLUSIONS

Experimental results show that the proposed lazy spitting algorithm has better F-measure scores that the baseline spitting algorithm. Another important result is the 33% gain in the training time, which is crucial to participate in MIREX (24hs limitation to train a fold). Although a significance test is necessary to confirm such superiority in terms of F-measure, given that it systematically yields better results we speculate it performs at least as well as the original spitting method. Therefore, whether it gets a better F-measure or not, it is an advantageous approach taking into account its speed and the time limitations imposed in the MIREX competitions.

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

REFERENCES

[1] J.J. Aucouturier and F. Pachet. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1, 2004.

[2] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010. In press.

[3] M. A. Casey, Re. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. In *Proceedings of the IEEE*, volume 96, pages 668–696, 2008.

[4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[5] J. Downie, D. Byrd, and T. Crawford. Ten years of ismir: Reflections on challenges and opportunities. In *Proceedings of the Tenth International Conference on Music Information Retrieval*, pages 34–41, 2009.

[6] J.S. Downie, A.F. Ehmann, M. Bay, and M.C. Jones. The music information retrieval evaluation exchange: Some observations and insights. *Advances in Music Information Retrieval. Berlin: Springer.*, pages 93–115, 2010.

[7] H. Francois and O. Boeffard. The greedy algorithm and its application to the construction of a continuous speech database. In *Proceedings of LREC-2002*, volume 5, pages 1420–1426, 2002.

[8] J. Futrelle and J. S. Downie. Interdisciplinary research issues in music information retrieval: Ismir 2000-2002. In *Journal of New Music Research*, volume 32, pages 121–131, 2003.

[9] M. Kassler. Toward musical information retrieval. In *Perspectives of New Music*, volume 2, pages 59–67, 1966.

[10] T. Lidy and A. Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proceedings of the Sixth International Conference on Music Information Retrieval*, pages 34–41, 2005.

[11] M. Mandel and D. Ellis. Song-level features and support vector machines for music classifcation. In *Proceedings of the Sixth International Conference on Music Information Retrieval*, pages 594–599, 2005.

[12] M. Mandel and D. Ellis. A web-based game for collecting music meta-data. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 369–374, 2007.

[13] C. Xu, N. Maddage, X. Shao, F. Cao, and Q. Tian. Musical genre classifcation using support vector machines. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 429–432, 2003.

Table 1: Mean F-Measure for BEST, NS, SP and LSP experimental conditions for each tag

TAG	BEST	NS	SP	LSP	LSP vs NS	LSP vs SP
drums	65.17	62.28	62.11	62.18	-0.10	0.07
guitar	70.26	66.97	67.39	69.61	2.64	2.22
male	63.23	54.93	55.15	59.63	4.70	4.48
rock	68.58	65.34	63.64	68.38	3.03	4.74
synth	51.20	42.43	41.83	44.27	1.83	2.44
synthesizer	51.20	42.43	41.83	44.27	1.83	2.44
electronic	58.48	48.60	49.16	55.26	6.66	6.10
pop	48.28	39.26	36.76	41.52	2.26	4.76
vocal	38.77	31.75	30.77	32.78	1.03	2.00
vocals	38.77	31.75	30.77	32.78	1.03	2.00
bass	37.36	31.18	27.79	33.24	2.06	5.45
female	40.03	30.19	28.76	44.46	14.27	15.71
dance	59.34	49.33	46.92	48.26	-1.07	1.34
techno	64.47	52.22	57.19	48.12	-4.10	-9.07
piano	52.46	39.34	42.87	41.96	2.62	-0.91
hip-hop	62.43	47.65	51.21	63.20	15.55	11.99
slow	46.74	27.55	28.66	34.87	7.31	6.21
rap	45.84	29.78	31.50	35.39	5.62	3.90
beat	51.74	41.75	37.38	49.08	7.33	11.69
voice	49.78	37.03	39.02	43.43	6.40	4.41
jazz	41.38	29.92	29.46	38.73	8.81	9.27
electronica	43.97	25.47	32.55	36.04	10.57	3.49
80s	35.08	18.62	12.58	26.13	7.51	13.55
instrumental	33.36	11.72	10.75	13.68	1.96	2.93
fast	32.74	21.41	20.89	15.45	-5.96	-5.44
saxophone	44.14	36.34	37.47	40.86	4.52	3.39
keyboard	19.44	15.44	13.8	26.37	10.93	12.57
country	21.08	8.72	8.48	18.46	9.74	9.98
distortion	24.61	12.16	8.05	15.32	3.16	7.27
british	21.67	7.93	8.48	7.47	-0.46	-1.02
drum-machine	15.38	7.44	7.22	10.49	3.05	3.28
funk	4.17	4.26	0	12.54	8.28	12.54
ambient	40.72	21.93	25.81	18	-3.93	-7.81
house	56.04	23.97	20.36	14.01	-9.96	-6.35
horns	11.25	5.15	5	7.06	1.91	2.06
drum-and-bass	0	4.41	2.08	11.85	7.44	9.77
soft	12.50	8.99	15.63	22.87	13.89	7.25
noise	32.74	24.50	12.5	35.30	10.80	22.80
silence	42.50	38.73	39.88	33.60	-5.12	-6.27
end	28.91	17.29	18.61	17.04	-0.25	-1.57
punk	24.44	12.48	22.77	12.90	0.42	-9.87
solo	29.40	15.26	14.56	10.74	-4.52	-3.81
quiet	34.31	15.89	19.86	28.70	12.81	8.84
trumpet	25	9.76	12.05	15.19	5.43	3.15
acoustic	29.17	11.27	19.79	9.69	-1.58	-10.10
folk	16.67	12.82	8.33	8.87	-3.95	0.54
organ	12.50	1.90	3.13	3.86	1.95	0.73
strings	18.75	2.22	3.57	9.12	6.90	5.55
loud	24.58	13.11	17.5	9.94	-3.17	-7.56
metal	18.75	10	12.5	15.58	5.58	3.08
trance	36.25	8.67	18.33	3.17	-5.49	-15.16
r&b	0	0	0	0	0.00	0.00
Mean F-Measure	50.28	41.65	41.41	44.95		