EFFICIENT DATABASE PRUNING FOR LARGE-SCALE COVER SONG RECOGNITION

J. Osmalskyj, S. Piérard, M. Van Droogenbroeck, J.J. Embrechts

INTELSIG Laboratory, Departement EECS
University of Liège, Belgium
{josmalsky, jjembrechts}@ulg.ac.be

ABSTRACT

This paper focuses on cover song recognition over a large dataset, potentially containing millions of songs. At this time, the problem of cover song recognition is still challenging and only few methods have been proposed on large scale databases. We present an efficient method for quickly extracting a small subset from a large database in which a correspondence to an audio query should be found. We make use of fast rejectors based on independent audio features. Our method mixes independent rejectors together to build composite ones. We evaluate our system with the Million Song Dataset and we present composite rejectors offering a good trade-off between the percentage of pruning and the percentage of loss.

Index Terms— Million Song Dataset, Music Information Retrieval, Chromas, Rejectors, Cover Songs

1. INTRODUCTION

Recent years have seen an increasing availability of large music databases and services. Companies such as Spotify and Shazam make extensive use of such databases which usually contain millions of songs. Music Information Retrieval (MIR) allows the development of new techniques for browsing such collections. One typical MIR task is cover song recognition, whose goal is to identify different versions of the same underlying musical piece. Such a version can be very different from the original track in terms of instrumentation, pitch, tempo, etc.

Cover song recognition has been widely studied in the past years. Most of the existing methods make use of direct comparisons of chroma features (see Section 3.2) between pairs of songs using dynamic programing techniques [1]. Although these methods produce interesting results on small and medium size datasets, they require a huge amount of computation, making them unsuitable with large datasets. However, some work was done to handle larger sets: Casey and Slaney [2, 3] use Locally-Sensitive Hashing (LSH) to compare chroma patches. Yu *et al.* [4] also use LSH to compare song statistics. An overview of methods for cover song recognition can be found in [5].

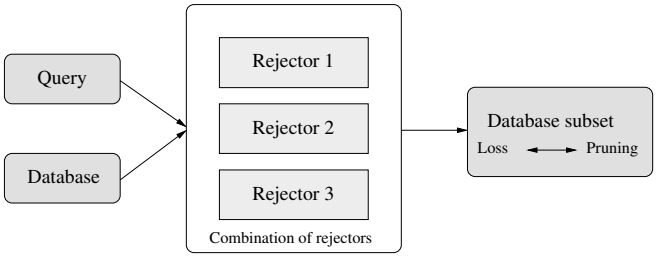

Fig. 1. For a given query song, our method prunes a database with a combination of rejectors that returns a small subset of the original dataset. There is a trade-off between the percentage of pruning and the loss of the rejectors. These two characteristics are defined in Section 2.

For the development of these techniques, researchers had to face the lack of a large public dataset, forcing them to evaluate their systems on individual datasets containing at most a few hundred songs. In 2011, the Million Song Dataset (MSD) [6] was released to solve this dataset issue. It contains audio features for one million tracks, including chroma vectors. It also features a list of 12,960 cover songs, the SecondHandSongs dataset (SHSD), making it suitable for our task.

Bertin-Mahieux *et al.* [7] were the first to propose a scalable method, usable on the MSD dataset, using hash codes inspired by [8]. More recently, they proposed a new approach [9] by projecting an entire song into a small dimension space and using nearest neighbors as candidate covers.

In this paper, we propose a new approach to the problem of cover song recognition in large-scale databases, and we evaluate it on the MSD. We propose a fast method to prune the search database by reducing the size of the search set (10^6 in the case of the MSD) to a smaller subset of songs (see Figure 1). We make use of the features available in the MSD to create rejectors, whose role is to reject a subset of the database and to keep a smaller amount of songs in which further processing could be applied to find the best cover match. We first create two simple rejectors based on the tempo and the duration of the query song. We also create a more powerful rejector based on bag-of-words of chroma features (see Section 3.2). Our chroma rejector is particularly efficient because

it uses the Euclidean distance (which is fast to compute) to compare pairs of songs. We finally show that mixing these rejectors produces better results than using them individually.

The remaining of this paper is organized as follows. The next section introduces the notion of rejector and explains our assessment method. Section 3 presents the three elementary rejectors used in our method. Then, in Section 4, we develop the combination of rejectors to obtain better results. Finally, we conclude the paper in Section 5.

2. REJECTORS AND EVALUATION METHOD

Cover song recognition deals with databases containing millions of songs. In this paper, we focus on retrieving at least one version corresponding to a query. Therefore, our goal is to dramatically decrease the size of the search set very fast, while ensuring the presence of one corresponding version in the remaining subset. To achieve such a reduction, we prune the dataset based on audio criteria using rejectors. A rejector takes a query song q as an input, and returns a subset $\mathcal{S}$ of the search database $\mathcal{D}$ containing only songs related to that query according to a criterion κ:

$$\mathcal{S}(\mathcal{D}, q) = \text{Rejector}_{\kappa}(\mathcal{D}, q) \text{ with } \mathcal{S}(\mathcal{D}, q) \subseteq \mathcal{D}.$$

The criteria are based on the audio features precomputed in $\mathcal{D}$. A criterion can depend on a parameter Δ. Such a parameter allows one to tune the tolerance when two tracks are compared with respect to κ in order to determine if they represent two versions of the same song. This paper focuses on criteria consuming low temporal resources when applied on large databases.

To evaluate the performances of a rejector, one has to consider both the pruning rate and the risk to reject all the corresponding versions from $\mathcal{D}$. Ideally, the pruning should be maximized while the loss is minimized, but in practice there is a trade-off between these two aims. In the remaining of this paper, we present our results in the form of plots displaying all the reachable $(\text{pruning}(\Delta), \text{loss}(\Delta))$ pairs. We call these plots Prune-Loss Curves (PLC). When $\Delta \in \mathbb{R}$, these points define a curve and when $\Delta \in \mathbb{R}^n$ with $n > 1$, they define a surface. Only its lower boundary is of practical interest. A rejector obtained with a parameter Δ is of practical interest only if there exists no Δ' such that $\text{pruning}(\Delta') > \text{pruning}(\Delta)$ and $\text{loss}(\Delta') \leq \text{loss}(\Delta)$, or such that $\text{pruning}(\Delta') \geq \text{pruning}(\Delta)$ and $\text{loss}(\Delta') < \text{loss}(\Delta)$.

Let us denote the set of queries by $\mathcal{Q}$. The loss and pruning rates are computed as follows:

$$\text{loss} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \left(1 - \mathcal{L}\left(\mathcal{S}\left(\mathcal{D}, q\right), q\right)\right)$$

$$\text{pruning} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{|\mathcal{D}| - |\mathcal{S}\left(\mathcal{D}, q\right)|}{|\mathcal{D}|},$$

where

$$\mathcal{L}(\mathcal{X}, x) = \begin{cases} 1 & \text{if } \exists x' \in \mathcal{X} : \mathcal{T}_{\kappa}\left(x, x'\right) = 1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\mathcal{T}_{\kappa}\left(x, x'\right) = \begin{cases} 1 & \text{if } x \text{ and } x' \text{ are two different} \\ & \text{versions of the same song} \\ 0 & \text{otherwise.} \end{cases}$$

The presented results are obtained with the MSD ($\mathcal{D}$) and the SHSD ($\mathcal{Q}$). We have $|\mathcal{D}| = 10^6$ and $|\mathcal{Q}| = 12,960$. The performances of the rejectors studied in this paper are compared to the performances of a naive one on the graphs. By definition, a naive rejector takes its decision without considering the information provided in the query. Let Δ_{ξ} be the probability to drop a track , and let us denote, by p_j, the proportion of queries $q \in \mathcal{Q}$ that have exactly j versions $x \in \mathcal{D}$ such that $\mathcal{T}_{\kappa}(x, q) = 1$. We have $(\text{pruning}(\Delta_{\xi}), \text{loss}(\Delta_{\xi})) = \left(\Delta_{\xi}, \sum_{j=1}^{\infty} p_j \Delta_{\xi}^{j}\right)$.

3. ELEMENTARY REJECTORS

Cover songs are often very different from the original underlying piece. To compare versions, we need a criterion which is common to each version, but insensitive to the differences between them. In this section, we present three elementary rejectors and evaluate them individually. In Section 4, we will merge them into a more powerful composite rejector. We first consider two simple rejectors in Section 3.1, and then a rejector based on chroma features in Section 3.2.

3.1. Duration and tempo rejectors

The duration rejector works as follows. For a query q whose duration is $t(q)$, it selects the songs in $\mathcal{D}$ with a duration comprised in $t(q) \pm \Delta_t\%$. Δ_t is the parameter of this rejector. The tempo rejector works in a similar way. Let us assume that the tempo of the query q is $b(q)$, in beats per minute. The rejector keeps the songs in $\mathcal{D}$ whose tempo is in the range $b(q) \pm \Delta_b$. Δ_b is expressed in beats per minute.

Figure 2 shows the results of these two simple elementary rejectors. In the following, we introduce a more powerful rejector which lowers the prune-loss curve towards the ideal point $(\text{pruning}, \text{loss}) = (100\,\%, 0\,\%)$.

3.2. Chroma rejector

To achieve better results, we consider a harmonic related criterion. Harmonic information is insensitive to the version of a song [10]. Descriptors related to harmonic information exist in the literature and are referred to as *chroma* features. They are derived from the spectrogram and describe the harmonic content of a song.

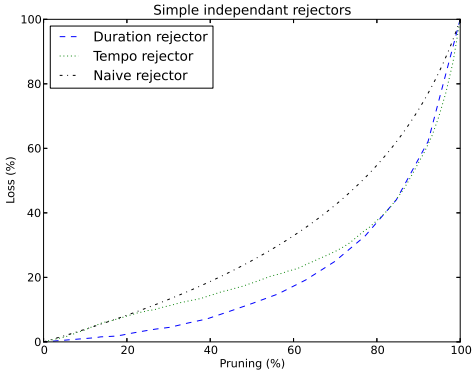

Fig. 2. Performances of the duration and tempo rejectors on the MSD. The parameter Δ_t of the former varies in the $[0, \infty]$ % range. The parameter Δ_b of the latter varies in the $[0, 100]$ bpm range.

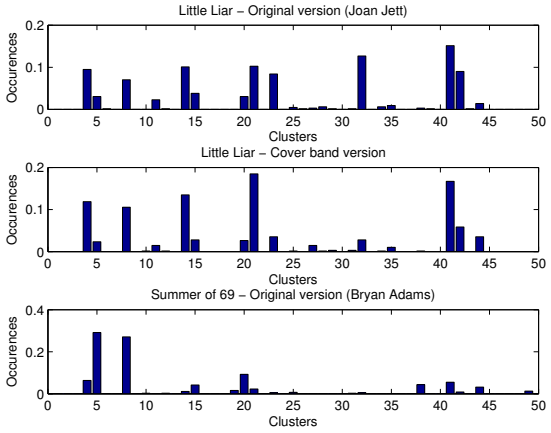

Fig. 3. Bag-of-words representations of 3 songs. The upper ones correspond to two versions of "Little Liar" by Joan Jett, and the third one to a different song ("Summer of 69" by Bryan Adams). We note similarities between the versions of the same piece, and differences with an unrelated song.

Chroma features were first proposed by [11] as pitch class profiles (PCP). Many improvements over the original PCP were introduced later, including HPCP [12] and CENS [13]. An overview of chroma features can be found in [14]. According to [9], none of these features were fully satisfying in their raw format for cover songs recognition. Indeed, the processing of such a huge amount of information requires much computation, making it unsuitable for large datasets such as the MSD. A more compact representation still based on chroma features is presented in [15], where the description of the songs is based on a clustering algorithm over chroma patches. We created an alternative representation which requires less processing.

Our method is based on the assumption that many chroma vectors could be grouped to form a set of similar vectors. In order to obtain a set of clusters of similar chromas, we applied a K-Means algorithm [16] on a random set of 2×10^6 chroma vectors extracted from the MSD. The number of clusters was arbitrarily set to 50 clusters.

Using the clustering model, we computed a bag-of-words representation as follows. The clusters can be seen as a base vocabulary for our description. To compute a bag-of-words, we simply count the number of occurrences of each cluster in a song by classifying each chroma vector in one of the fifty clusters. Therefore, we obtain a compact 50-dimensional representation which still describes well the harmony of a song. Moreover, we make this description invariant to pitch changes by applying the Optimal Transposition Index (OTI) [17] method when comparing two songs. This method computes the best shift to transpose a song A to the pitch of a song B. Clustering the resulting transposition gives comparable pitch-invariant bag-of-words. Figure 3 illustrates that cover songs have similar bag-of-words, while they allow to differentiate from a completely different song.

Using our bag-of-words representation, we created a rejector which returns the $N = \Delta_c |\mathcal{D}|$ nearest neighbors of the query song in terms of harmonic content. Δ_c is the parameter of the rejector, which takes its value in the $[0\,\%, 100\,\%]$ range. From a query song, the chroma features are extracted either from the MSD in the case of the SHSD evaluation, or from the EchoNest [18] using their *Analyze* engine. This engine provides exactly the same features as those available in the MSD, allowing us to use our system with any audio input. Once the chromas are retrieved, each of them is classified in a cluster using our K-Means model, thus providing a bag-of-words describing the song. The resulting histogram is then normalized so that the sum of all bins equals 1.

The bag-of-words is then compared to each song of the database. Since each song is represented by a low-dimensional histogram, the computation of distance measures (such as the Euclidean distance or the Bhattacharyya distance [19]) is very efficient. Before computing the distance between pairs of songs, we compute the OTI to shift the query so that it matches the pitch of the database song. Therefore, the distance between two versions does not depend on the pitch. The resulting distances are sorted and the N first matches are returned by the rejector.

Figure 4 shows the results obtained with the bag-of-words rejector. The comparison of Figures 2 and 4 establishes that the bag-of-words rejector improves the performance considerably. In the next section, we show that we can combine our three elementary rejectors to obtain a composite rejector producing even better results.

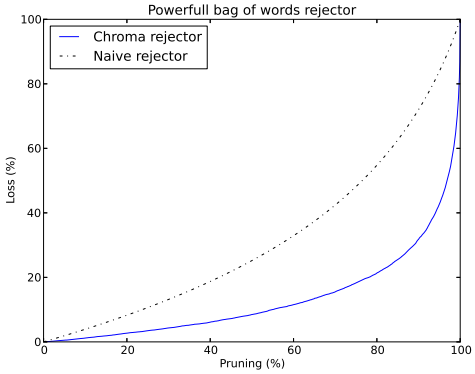

Fig. 4. Performance of our bag-of-words rejector. The parameter Δ_c varies between $0\,\%$ and $100\,\%$. This rejector outperforms the ones presented in Figure 2.

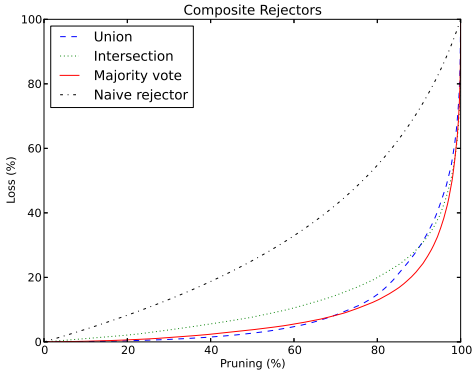

Fig. 5. Performances of the three composite rejectors evaluated on the MSD. Only the rejectors of interest, as defined in Section 2, are shown for each composite rejector.

4. COMPOSITE REJECTORS

In this section, we combine our three elementary rejectors. To achieve better results, we define the following three combinations of the rejectors:

$$S_k^*(\mathcal{D},\, q) = \left\{ d \in \mathcal{D} \,\middle|\, \sum_{i=1}^{3} \mathcal{L}(\mathcal{S}_i(\mathcal{D}, q),\, d) \geqslant k \right\}$$

where $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ denote the subsets provided by the elementary rejectors. The first composite rejector, $\mathcal{S}_1^*$ is named *union* since one can demonstrate that $\mathcal{S}_1^* = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3$. The second one is obtained following a *majority vote* strategy. And the third one is named *intersection* since $\mathcal{S}_3^* = \mathcal{S}_1 \cap \mathcal{S}_2 \cap \mathcal{S}_3$. From a computational point of view, the last one can been seen as a cascade of rejectors [20, 21].

The results of our composite rejectors are depicted in Figure 5. If the goal is to minimize the loss while keeping the pruning above a significant level, then the best results are obtained with the majority vote rejector. However, if the goal is to maximize the pruning while maintaining the loss below a low threshold, then the union rejector is preferable.

Inevitably, composite rejectors are built on top of elementary rejectors. We have proposed three of them, nevertheless others could be considered in conjunction or in replacement. It should however be stressed that the elementary rejectors need to be selected carefully. To minimize the computation time, we aim at extracting and treating the information only once. Therefore, we have selected mutually independent criteria. For example, it is preferable to consider the tempo over the total number of beats, since the number of beats depends on the song duration, while the tempo is not.

In this paper, we have expressed the intrinsic trade-off of rejectors based on a loss *vs* pruning analysis. Another way of expressing the same compromise would be based on a loss *vs* expected computation time analysis. Such an analysis is left for future work since the computation time of a composite rejector does not depend only on the pruning rate. It depends on (i) the complexity of the elementary rejectors, (ii) their parameters, (iii) their pruning, (iv) the order in which they are queried, and (v) the way their answers are combined (*i.e.* union, intersection, majority vote, *etc*). Moreover, our three composite rejectors are implemented with short-circuit operations, which complicates the analysis.

5. CONCLUSIONS

This paper focuses on cover song recognition in large-scale databases. In order to speed up queries, we have developed a fast database pruning method. Our goal is not to find a unique match to an audio query, but rather to reduce the search set as fast as possible. Further analysis can be subsequently conducted on the remaining subset to identify a precise match.

Our database pruning method is built on top of rejectors. We have provided the performance achieved by three elementary rejectors related to independent audio features, namely the duration, the tempo, and the harmonic content of the songs. We have also presented the results obtained with three composite rejectors, obtained by combining the elementary ones. We have established that such a combination improves the overall performance. Our method applied on the Million Song Dataset is able to reduce the search size with a very low risk of dropping the target song. Furthermore, our algorithm has the big advantage of being very fast.

To our knowledge, only few methods are usable for large datasets such as the Million Song Dataset and this paper is a step forward towards cover song recognition for very large datasets. Moreover, our method can be seen as a universal strategy to develop database pruning methods, and could be applied to other computer science fields such as computer vision or data mining.

6. REFERENCES

[1] J. Serra, *Identification of versions of the same musical composition by processing audio descriptions*, Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, 2011.

[2] M. Casey and M. Slaney, "Fast recognition of remixed audio," in *Proceedings of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2007.

[3] M. Casey, C. Rhodes, and M. Slaney, "Analysis of minimum distances in high-dimensional musical spaces," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 16, no. 5, pp. 1015–1028, July 2008.

[4] Y. Yu, M. Crucianu, V. Oria, and L. Chen, "Local summarization and multi-level lsh for retrieving multivariant audio tracks," in *Proceedings of the seventeen ACM international conference on Multimedia*, 2009, pp. 341–350.

[5] J. Serra, E. Gomez, and P. Herrera, "Audio cover song identification and similarity: Background, approaches, evaluation, and beyond," in *Advances in Music Information Retrieval*, pp. 307–332. Springer Berlin / Heidelberg, 2010.

[6] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, 2011.

[7] T. Bertin-Mahieux and D. Ellis, "Large-scale cover song recognition using hashed chroma landmarks," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2011.

[8] A. Wang, "An industrial-strength audio search algorithm," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, 2003, pp. 7–13.

[9] T. Bertin-Mahieux and D. Ellis, "Large-scale cover song recognition using the 2D Fourier transform magnitude," in *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR)*, 2012.

[10] J. Osmalskyj, J.-J. Embrechts, M. Van Droogenbroeck, and S. Piérard, "Neural networks for musical chords recognition," in *Journées d'informatique musicale*, Mons, Belgium, 2012.

[11] T. Fujishima, "Realtime chord recognition of musical sound: a system using Common Lisp music," in *Proceedings og the International Computer Music Conference (ICMC)*, 1999, pp. 464–467.

[12] E. Gomez and P. Herrera, "Automatic extraction of tonal metadata from polyphonic audio recordings," in *Proceedings of the 25th International Audio Engineering Society Conference (AES)*, 2004.

[13] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, 2005, pp. 288–295.

[14] N. Jiang, P. Grosche, V. Konz, and M. Müller, "Analyzing chroma feature types for automated chord recognition," in *Proceedings of the 42nd AES Conference*, 2011.

[15] T. Bertin-Mahieux, R. Weiss, and D. Ellis, "Clustering beat-chroma patterns in a large music database," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, 2010.

[16] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *5th Berkeley Symposium on Mathematical Statistics and Probability*. 1967, vol. 1, pp. 281–297, University of California.

[17] J. Serra, E. Gomez, P. Herrera, and X. Serra, "Chroma binary similarity and local alignment applied to cover song identification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 6, pp. 1138–1152, 2008.

[18] The Echo Nest team, "The echo nest," `http://the.echonest.com`.

[19] M. Deza and E. Deza, *Encyclopedia of Distances*, Springer, 2009.

[20] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Kauai, USA, December 2001, vol. 1, pp. 511–518.

[21] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, New York, USA, June 2006, vol. 2, pp. 1491–1498.