# ENHANCING COVER SONG IDENTIFICATION WITH HIERARCHICAL RANK AGGREGATION

**Julien Osmalskyj, Marc Van Droogenbroeck, Jean-Jaques Embrechts**
INTELSIG Laboratory - University of Liège - Belgium
josmalsky@ulg.ac.be, jjembrechts@ulg.ac.be

## ABSTRACT

Cover song identification involves calculating pairwise similarities between a query audio track and a database of reference tracks. While most authors make exclusively use of chroma features, recent work tends to demonstrate that combining similarity estimators based on multiple audio features increases the performance. We improve this approach by using a hierarchical rank aggregation method for combining estimators based on different features. More precisely, we first aggregate estimators based on global features such as the tempo, the duration, the overall loudness, the number of beats, and the average chroma vector. Then, we aggregate the resulting composite estimator with four popular state-of-the-art methods based on chromas as well as timbre sequences. We further introduce a refinement step for the rank aggregation called "local Kemenization" and quantify its benefit for cover song identification. The performance of our method is evaluated on the Second Hand Song dataset. Our experiments show a significant improvement of the performance, up to an increase of more than $200\,\%$ of the number of queries identified in the Top-1, compared to previous results.

## 1. INTRODUCTION

Given an audio query track, the goal of a cover song identification system is to retrieve at least one different version of the query in a reference database, in order to identify it. In that context, a version can be described as a new performance or recording of a previously recorded track [22]. Retrieving covers is a challenging task, as the different renditions of a song can differ from the original track in terms of tempo, pitch, structure, instrumentation, etc. The usual way of retrieving cover songs in a database involves extracting meaningful features from an audio query first in order to compare them to the corresponding features computed for the other tracks of the database using a pairwise similarity function. The function returns a score or a probability of similarity. Many researchers have been using exclusively chroma features [10, 13, 14, 22] to characterize the songs in the database. Chroma vectors describe the harmony of the songs and are robust to changes in instrumentation and timbre, which makes them quite popular for the task. While chromas are the most used features in the literature, other works investigate the use of different features, such as timbral features [23] or cognition based features [4].

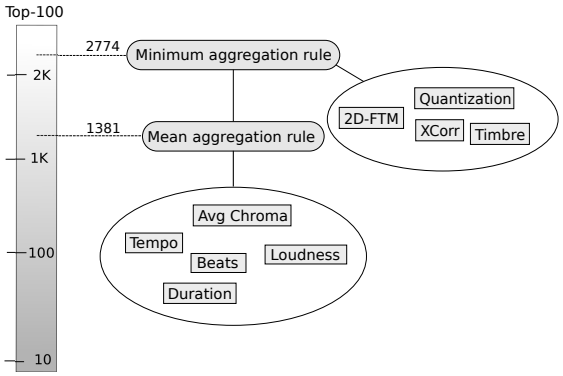


**Figure 1**. Hierarchical rank aggregation of estimators based on audio features. Global features are first aggregated using the *mean rule*, identifying 1,381 tracks in the top-100, out of 5,464 tracks sampled from the Second Hand Song dataset. The resulting composite estimator is then aggregated with four remaining features using the *minimum rule*, identifying 2,774 tracks in the top-100.

In recent work [16], we established that combining multiple audio features improves the performance of cover song identification systems: designing several classifiers based on different features and combining them through probabilistic rules or rank aggregation techniques improves the performance. In light of this, it seems important to study how state-of-the-art features perform when they are combined for cover song identification. In this paper, we improve upon previous work by considering a total of nine features, including four state-of-the-art ones. These features cover a wide range of audio characteristics, from low-dimensional ones such as the tempo or the duration of the songs, to higher level characteristics such as chromas and timbral sequences. We build similarity estimators for each feature, using supervised machine learning for some of them, and combine them in a hierarchical way to design a new combination method. In this method, we first aggregate five estimators that are based on global features:

tempo, duration, loudness, beats, and averaged chroma. These global features are computed for the entire song, rather than for individual chunks of audio. We show that combining such estimators using rank aggregation methods improves the performance, compared to probabilistic fusion [16]. We then take the resulting aggregated estimator and combine it with four state-of-the-art methods using a different aggregation rule, as shown in Figure 1. We further achieve a higher performance by applying a refinement step called *"local Kemenization"* [7, 8]. We found that this refinement step significantly increases the number of queries identified immediately (Top-1).

## 2. METHOD OVERVIEW

Identifying cover songs with respect to an audio query involves comparing the query to a set of tracks in a reference database using a similarity function. Considering two input tracks, the function should return a score indicating whether the tracks are considered as being similar or not. Our approach follows the combining method that we proposed in [16]. As there exist several effective features in the literature, the idea is to take advantage of all of them by combining them. We therefore design several pairwise comparison functions, called similarity estimators, based on different audio features. We first consider the same set of estimators as the one used in [16]. We make use of global low-dimensional features such as the tempo, the duration, the number of beats, the overall loudness, and the average chroma vector of a song, learning a probabilistic model to predict the similarity. We also include three estimators based on chromas features. The first and second ones were used in previous works and are respectively based on the quantization of chroma features [11, 16] and the cross-correlation of entire chroma sequences [9]. We add a third chroma estimator based on an efficient large-scale method proposed by Bertin-Mahieux *et al.* [2]. Finally, to take into account timbral information, we include an estimator based on MFCC features. This method, introduced by Tralie *et al.* [23], showed that some covers could be identified based on timbre only.

### 2.1 Weak estimators

In previous work [16], we demonstrated that global low-dimensional features (tempo, duration, etc.) bring information that helps in the identification process. However, using only such features for identifying cover songs is not enough to achieve good performance. Indeed, such features are considered as *weak* because they only slightly improve a classifier with respect to a purely random classifier. While we combined these features using probabilistic combination rules, we innovate by combining them using rank aggregation techniques. For each feature, we build a probabilistic estimator, using supervised machine learning. To determine the similarity of candidates with respect to the query, we perform pairwise comparisons using the learned probabilistic models to predict probabilities of similarities. Each query is compared to the database using each estimator. We then aggregate the rankings produced by each estimator to build an improved list of results.

### 2.2 Chroma estimators

#### *2.2.1 Cross-correlation*

We design the same cross-correlation estimator as the one used in [16]. This estimator is useful to take into account temporal information. It computes two dimensional cross-correlations between high-pass filtered chroma sequences of the tracks to be compared. The similarity score between two songs is computed as the reciprocal of the peak value of the cross-correlated signal. We refer the reader to the original work [9] for details.

#### *2.2.2 Quantization*

To take into account the harmonic distribution of the songs, we make use of an estimator based on the quantization of chroma features [11, 17]. For each track, chroma vectors are mapped to specific codewords. Codewords are determined using a K-Means clustering of 200,000 chromas vectors. We retain 100 clusters for the feature, resulting in a 100-dimensional feature vector. The similarity score is computed as the cosine similarity between two 100-dimensional vectors. To account for key transposition, we make use of the optimal transposition index [21] (OTI) technique, as it has been used in other works [1, 20].

#### *2.2.3 2D Fourier transform magnitude coefficients*

This method was first introduced by Bertin-Mahieux *et al.* [2] and was designed as a fast and accurate feature for cover song identification. The idea is to encode harmonic information in a compact representation, to make it invariant to local tempo changes and pitch shifts. First we extract patches of 75 consecutive chromas with an overlap of 1. We then compute the 2D FFT magnitude coefficients for each patch. Next, we aggregate all the patches pointwise using a median rule. Finally, we project the resulting 900-dimensional representation on a 50 dimensional PCA sub-space. Each track is therefore represented by a 50-dimensional vector. The final score between two tracks is computed as the cosine similarity between two projections.

### 2.3 Timbre estimator

In our base set of estimators, we also include a method proposed by Tralie *et al.* [23], that takes into account the relative evolution of timbre over time. Using careful centering and normalization, the authors were able to design features that are approximately invariant to cover. The features are based on self-similarity matrices of MFCC coefficients and can be used to identify cover songs. Being based on timbre rather than harmony, this feature demonstrates that if the pitch is blurred and obscured, cover song identification should still be possible (see the original paper [23] for a detailed explanation). We designed an estimator based on features that were kindly computed for us by the authors of the method. The similarity score is computed using the Smith-Waterman alignment algorithm.

## 3. HIERARCHICAL RANK AGGREGATION

### 3.1 Rank aggregation techniques

To take advantage of all the features that we use, we need a way to combine them. One way of doing that is through probabilistic combination rules. Under the hypothesis that all estimators return probabilities, we can experiment several rules such as the probabilistic product, sum or median rules [5, 6]. The problem is that not all of our estimators return a probability. Some estimators return probabilities, while others return different kinds of scores, for example a cosine similarity or a cross-correlation peak value. One solution for using such rules would be to map scores to probabilities, but there is no straightforward way of doing that. Furthermore, an independent dataset is often mandatory for such a mapping.

Another solution is to combine estimators through rank aggregation techniques, as we proposed in [16]. As a single query $q$ is compared to the entire database using $N$ estimators, we obtain $N$ different orderings of the database. Each track of the database can be found at different positions in the resulting orderings. Based on the positions of the tracks, rank aggregation techniques compute a new position by applying simple rules such as computing the new rank as the mean of the ranks of each track in the initial orderings. Other rules include the minimum, maximum or median rules. Rank aggregation techniques are popular in the web literature [7]. Such techniques are interesting compared to score-based combination because they are intrinsically calibrated and scale-insensitive [19].

In this paper, we aggregate features at different levels. We first aggregate weak features, experimenting with multiple rules. We next use the resulting ranking as a new input for another aggregation rule, by considering our four remaining estimators. We therefore build a hierarchy of two aggregated classifiers (Figure 1) and achieve improved performance compared to previous results [16].

### 3.2 Optimizing rank aggregation

After several input rankings $r_1$, $r_2$, ..., $r_k$ have been aggregated into one final ranking $\mu$ using one of the rules proposed before, we can apply a refinement step called *local Kemenization* [8] to further improve the ranking $\mu$. An aggregated ranking is *locally Kemeny optimal* if there are no pairwise swaps of items in the list that will reduce the sum of Kendall $\tau$ [8] measures between each input ranking $r_i$ and $\mu$, where $i = 1, \ldots, k$. The sum of the Kendall $\tau$ measures with respect to each initial ranking is called "*aggregated Kendall measure*". The Kendall $\tau$ measure determines the correlation between two rankings of equal size. It measures the degree to which one list agrees with another [15]. In practice, one way of computing it is to count the number of swaps needed by a bubble sort algorithm to permute one list to the other. Formally, the Kendall $\tau$ distance is defined by

$$\tau = \frac{n_c - n_d}{n(n-1)/2}, \qquad (1)$$

where $n_c$ is the number of concordant pairs and $n_d$ is the number of discordant pairs. The denominator corresponds to the total number of pairs of $n$ items in the lists. A pair of tracks $(i, j)$ is *concordant* if $i$ is ranked above $j$ in both lists, and *discordant* otherwise.

Based on this distance measure between rankings, the local Kemenization procedure considers each pair of adjacent tracks in $\mu$ and verifies whether a swap will improve the aggregated Kendall measure. In practice, for two adjacent tracks $(i, j)$ in $\mu$, with $i$ ranked above $j$, the procedure checks whether track $j$ is ranked above $i$ in the majority of the input rankings. If yes, it swaps the two items as it refines the aggregated list with a reduced Kendall distance. The procedure starts from the beginning of the list, and is repeated iteratively for all pairs of tracks, requiring $n - 1$ checks for an aggregated list of length $n$. Note that the consecutive swaps of the Kemenization process take into account the inclusion of earlier swaps. For implementation details, we refer the reader to our own implementation of several rank aggregation rules with local Kemenization in a C++ library [1]. We used that code to produce results for this paper.

For our task of cover song identification, we apply the local Kemenization step to our final aggregations to improve the overall performance. Detailed results are given in Section 4.

## 4. EXPERIMENTS AND RESULTS

### 4.1 Experimental setup

#### *4.1.1 Evaluation database*

We evaluate our method on the Second Hand Song dataset [2] (SHS), a subset of the Million Song Dataset (MSD) [3]. The SHS is structured in 5,854 *cliques*, which are groups of 3 cover songs on average, for a total of 18,196 tracks. The dataset does not provide any audio data. Rather than that, it proposes a set of pre-computed features. Since we need independent learning and test sets for learning probabilistic models, we split all tracks in a learning set (LS) containing 70 % of the tracks, and a test set (TS) containing the 30 % remaining tracks. We learn our models on the LS, and evaluate our final system on the TS, containing 5,464 tracks. Following the procedure explained in [16], we get rid of duplicate tracks in the SHS, thus reducing the number of cliques to 5,828.

#### *4.1.2 Estimators settings*

For each estimator, we use the pre-computed features in the SHS. As the chroma features provided in the dataset are aligned on onsets rather than the beats, we re-align them on the beats to account for tempo variations within covers, as done in other works [2, 13].

For our weak estimators, we learn probabilistic models using the ExtraTrees algorithm [12], to estimate probabilities of similarity. The models are learned with 1,000 trees

[1] https://github.com/osmju/librag

[2] http://labrosa.ee.columbia.edu/millionsong/secondhand

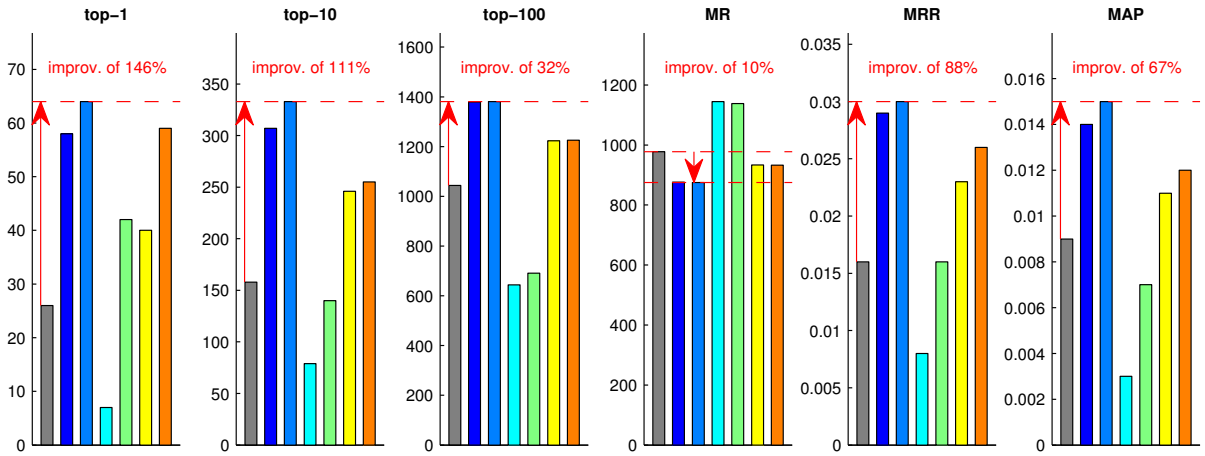


**Figure 2**. Comparison of the performance of rank aggregation methods for combining estimators based on weak features. The baseline is the probabilistic fusion of weak features proposed in [16]. The mean rank aggregation rule significantly outperforms the baseline, especially in the top-1 with an improvement of 146 %. The red arrows quantify the improvement compared to the baseline. gray baseline, blue mean rank rule, dark blue mean rank rule with local Kemenization, cyan minimum rank rule, green minimim rank rule with local Kemenization, yellow median rank rule, orange median rank rule with local Kemenization.

to reduce the variance, and a maximum depth of 20. The trees are not completely developped to avoid over-fitting. The implementation we use is the Python Scikit-Learn library [18].

To account for key transpositions in our estimator based on the average chroma and in the quantization estimator, we use the optimal transposition index (OTI) technique [21].

For the 2D-FTM estimator, we closely follow the original implementation. We wrote our own C++ implementation, based on the Python code [3] provided by Humphrey *et al.* [13]. We use the FFTW library for computing the 2D-FFT of chroma patches.

Finally, our timbre estimator is close to the original implementation [23], as the features were computed by the author itself for us. The only difference in the implementation comes from the fact that the MFCC sequence of a song in the SHS does not have the same resolution than in the original implementation. We implemented our own version of the sequence alignment Smith-Waterman to compute the final score.

### 4.2 Aggregation of weak features

Our first experiment consists in aggregating weak features, experimenting with several fusion rules. We take estimators based on the tempo, the duration, the number of beats, the average chroma vectors, the loudness, and aggregate them. We compare the performance to the baseline results obtained in our previous work [16]. We evaluate the performance of the system using standard information retrieval statistics, such as the Mean Rank of the first identified track (MR), the Mean Reciprocal Rank (MRR), and the Mean Average Precision (MAP). Note that the lower the MR is, the better it is, while the goal is to maximize

[3] https://github.com/urinieto/LargeScaleCoverSongId

| | Baseline | Mean | Kemeny | Increase |
|---|---|---|---|---|
| Top-1 | 26 | 58 | **64** | **+ 146 %** |
| Top-10 | 158 | 307 | **333** | **+ 111 %** |
| Top-100 | 1,044 | 1,379 | **1,381** | **+ 32 %** |
| Top-1000 | 3,729 | 3,911 | **3,911** | **+ 5 %** |
| MR | 977.6 | 876.2 | **875** | **+10 %** |
| MRR | 0.016 | 0.029 | **0.03** | **+ 88 %** |
| MAP | 0.009 | 0.014 | **0.015** | **+ 67 %** |

**Table 1**. Performance achieved with weak estimators when applying the mean rank aggregation rule ("mean" column), and applying the refinement step ("Kemeny" column) to improve the performance.

the MAP and the MRR. We also evaluate the number of queries for which a match is identified in the Top-$K$, $K$ being a parameter. We present results for the number of tracks identified in the top-1, 10 and 100.

Figure 2 displays the performance of three aggregation rules for the weak estimators with respect to all the metrics. We can notice immediately that the mean aggregation rule outperforms all other combinations. Without local Kemenization, the mean rule provides an improvement of 123 % compared to the baseline, with 58 tracks identified in the top-1 (against 26 in the baseline). That result shows that rank aggregation of these features outperforms the probabilistic rules proposed in previous work. Improvements in terms of the other metrics are also significant and demonstrate the strength of the method. Applying the refinement local Kemenization step, we further improve the performance to 64 tracks identified in the top-1, which corresponds to an increase of 146 % compared to the baseline. Note that the refinement provides surprisingly good improvement, especially for the minimum aggregation rule. Without optimization, we identify 7 tracks in the

| | Baseline | Mean | Min | Median |
|---|---|---|---|---|
| Top-1 | 328 | 832 | **1010** | 839 |
| Top-10 | 1015 | 1479 | **1785** | 1499 |
| Top-100 | 2015 | 2669 | **2774** | 2681 |
| Top-1000 | 4158 | **4456** | 4416 | 4385 |
| MR | 726.4 | **563** | 582 | 595 |
| MRR | 0.107 | 0.194 | **0.234** | 0.198 |
| MAP | 0.055 | 0.105 | **0.132** | 0.106 |

**Table 2**. Hierarchical aggregation of all features, with local Kemenization. Best performance is achieved with the minimum rule.

| | Mean | Min | Median |
|---|---|---|---|
| Top-1 | 503 | **784** | 519 |
| Top-10 | 1187 | **1577** | 1106 |
| Top-100 | 2435 | **2535** | 2299 |
| Top-1000 | **4423** | 4290 | 4309 |
| MR | **593** | 651 | 650 |
| MRR | 0.13 | **0.19** | 0.13 |
| MAP | 0.07 | **0.1** | 0.07 |

**Table 3**. Performance of single aggregation rules without hierarchization, with local Kemenization. Performance is not as good as with hierarchization.

top-1. This number jumps to 42 tracks, without changing anything to the base estimators, simply by applying the algorithm presented in Section 3.2. Table 1 quantifies the performance and improvement compared to the baseline for the mean aggregation rule, as it provides the best performance. Note that it is interesting to realize that using such weak features, we still can identify cover songs much better than random guessing.

### 4.3 Hierarchical aggregation

As the mean rule produces the best experimental results with weak estimators, we consider that resulting aggregated ranking as a single estimator by itself, and combine it with the four remaining estimators based on chroma and timbral features. We experiment the hierarchical combination with the mean, minimum and median rules, as for the weak estimators. Figure 3 displays the performance of each rule, with the local Kemenization step applied. It is straightforward to notice that the minimum rule with Kemenization significantly outperforms the other rules, especially for the top-1, with 1,010 tracks identified in the top-1. For the top-1 metric, we achieve the best performance so far on the subset we use for evaluation, with a MAP set to $0.132$ and a MRR set to $0.234$. Table 2 quantifies the metrics for all rules. The minimum rule achieves an impressive performance, especially for the top-1 metric, with an improvement of $208\,\%$, and for the MRR and the MAP, with an improvement of respectively $119\,\%$ and $140\,\%$. Note that as we used a significant subset of the SHS (70 %) as a learning set for our probabilistic models, it is difficult to compare our results with other works. Therefore, the baseline here corresponds to the best combination results proposed in our previous work [16]. The baselines in Figures 2 and 3 are different because they respectively correspond to the combination of weak features, as done in [16], and the combination of weak features and chroma based estimators, also as proposed in [16]. Figure 4 shows the performance curves of the aggregations corresponding to the bars in Figure 3. The horizontal axis corresponds to the top-k cutoff, that is the proportion of tracks that are rejected from the final set (the tracks ranked below k). The vertical axis corresponds to the loss, that is the proportion of queries for which no matches at all have been found in the top-k. If at least one corresponding track matches the query, then the loss is set to zero for that query. The second and third charts correspond to zooms in the lower left corner and in the upper right corner. From the performance curves, we clearly observe the improvement compared to the baseline. We also observe that the final curve (minimum rule, green) fits very closely the upper right part of the chart, corresponding to a very high cutoff value. We can reasonably tell that approximately half of the input queries are identified in the top-1% of the returned ranking. Note however how the mean curve (blue) takes the best position at low cutoff values.

### 4.4 Single aggregation of all features

To quantify the benefit of using *hierarchical* rank aggregation rather than running a single combination, we combined all the features with the three aggregation rules, and applied the refinement step. Aggregating all features in a single run corresponds to setting equal weights to all features. On the other hand, aggregating the results in a hierarchical way corresponds to set different weights to the features. Table 3 gives the performance of all aggregation rules with local Kemenization without any hierarchization. The best performing rule in terms of Top-{1, 10, 100} is again the minimum rule. Similar conclusions yield for the MRR and MAP metrics. For the Top-1000 and the MR, the best rule is the mean aggregation. Overall, the results are worse than using hierarchical aggregation. For the top-1 metric, the number of identified tracks drops by $22\,\%$ for the minimum rule, which is quite significant. For the MRR and the MAP respectively, the performance is decreased by $19\,\%$ and $24\,\%$ for the minimum rule. This demonstrates that attributing different weights to the estimators allows to achieve better performance. Avoiding the hierarchization would lead to a decreased performance, as indicated by the results in Table 3, compared to Table 2.

## 5. CONCLUSION

In this paper, we improve cover song identification by evaluating multiple rank aggregation rules. Based on previous work, we first construct probabilistic estimators based on multiple weak features such as tempo, duration, loudness, number of beats, and average chroma vectors. We use supervised machine learning to learn models predicting probabilities of similarity. Then, rather than combining the estimators through probabilistic rules, we have evaluated

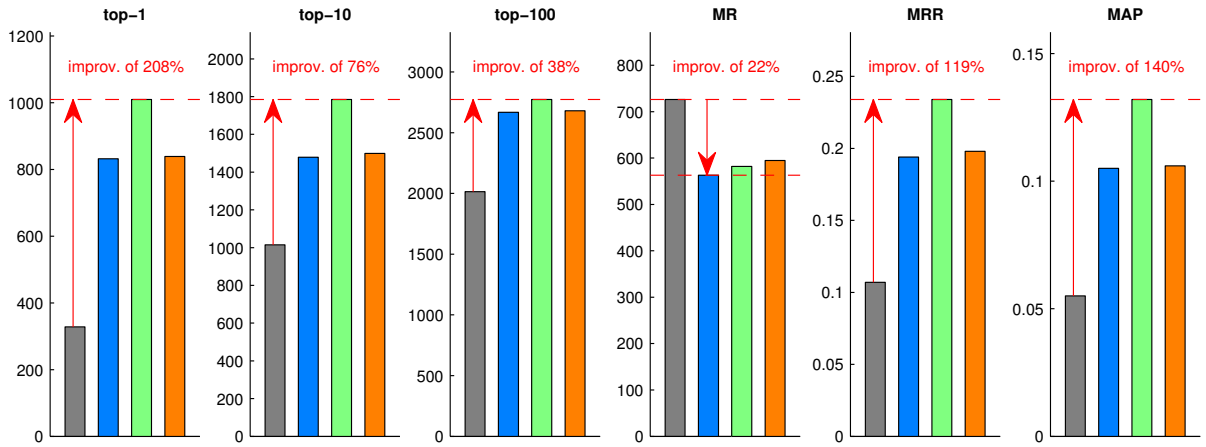


**Figure 3**. Hierarchical aggregation of all estimators with local Kemenization. The weak estimators are first aggregated using the mean rank aggregation rule. The figure shows how the performance vary when considering different top-level aggregation rules. The red arrows quantify the improvement compared to the baseline. ■ baseline [16], ■ mean rank rule with local Kemenization, ■ minimum rank rule with local Kemenization, ■ median rank rule with local Kemenization.

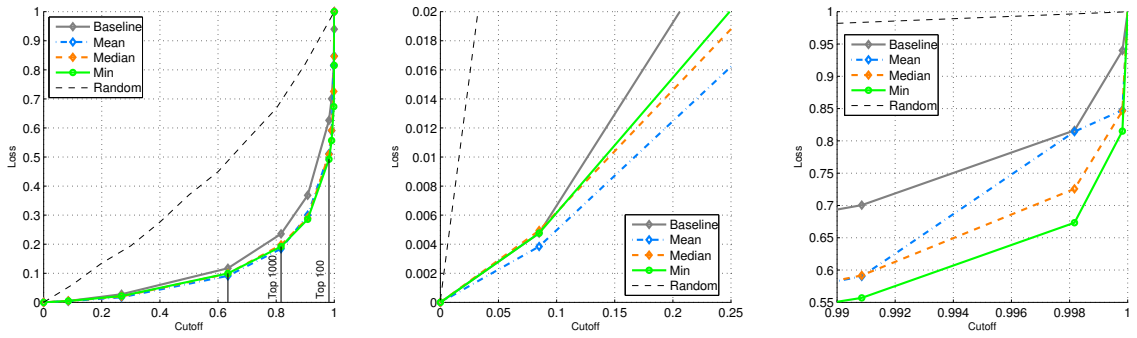


**Figure 4**. Performance curves of the refined hierarchical aggregation using all features. The x-axis corresponds to the proportion of tracks considered dissimilar by the method and the y-axis corresponds to the proportion of lost queries, that is the proportion of queries for which no matches at all have been found. The second and third charts correspond respectively to a zoom in the lower left part of the first chart, and to a zoom in the upper right corner of the first chart.

several rank aggregation rules, and prove that the mean aggregation rule provides improved results, compared to the baseline. Considering the resulting combined estimator, we further aggregate it with four estimators based on four state-of-the-art features. The selected features take into account harmonic information through chroma features, and timbral information through self-similarity matrices of MFCC coefficients. We further introduce an optimization step, called local Kemenization, that builds an improved aggregated ranking by swapping tracks to the top of the list, with respect to the agreement with each base estimator. To combine the estimators, we aggregate them all in a hierarchical way, evaluating several hierarchical rank aggregation rules. To highlight the gain of using hierarchicak rank aggregation, we also aggregate all nine features through a single aggregation rule, thus allocating an identical weight to all features. We show that such a combination degrades the performance. Our method is evaluated on the Second Hand Song dataset, displaying the performance in terms of standard statistics such as the mean rank of the first identified query, the mean reciprocal rank, the mean average precision and the number of tracks identified at the top-k cutoff. Best results are achieved with the minimum aggregation rule with local Kemenization. Indeed, we are able to identify 1,010 tracks at the first position, which corresponds to $18\,\%$ of the database. In the first 10 tracks returned, we identify 1,785 tracks (32 % of the database), which is a significant improvement over previous work. Compared to previous work on combination, we improve the results by $208\,\%$ in terms of the number of tracks identified in the top-1. In terms of mean reciprocal rank and mean average precision, we achieve improved performance with a value of $0.234$ for the MRR and $0.132$ for the MAP. The results show that aggregating multiple features, and therefore taking into account multiple sources of musical information, leads to significant improvements in the field of cover song identification. Our method takes advantage of all the best from the literature in that field, and suggests that following that direction of research might eventually lead to an even better performance.

## 6. REFERENCES


[1] T. Ahonen. Compression-Based Clustering of Chromagram Data : New Method and Representations. In *International Symposium on Computer Music Multidisciplinary Research*, pages 474–481, 2012.

[2] T. Bertin-Mahieux and D. Ellis. Large-scale cover song recognition using the 2d fourier transform magnitude. In *Proceedings of the 13th International Society for Music Information Retrieval (ISMIR)*, pages 241–246, 2012.

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval (ISMIR)*, pages 591–596, 2011.

[4] S. Downie, H. Xiao, Y. Zhu, J. Zhu, and N. Chen. Modified perceptual linear prediction liftered cepstrum (mplplc) model for pop cover song recognition. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 598–604, Malaga, 2015.

[5] R. Duin. The combining classifier: to train or not to train? *16th International Conference on Pattern Recognition*, 2:765–770, 2002.

[6] R. Duin and D. Tax. Experiments with classifier combining rules. *Lecture Notes in Computer Science*, 31(15):16–29, 2000.

[7] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, 2001.

[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank Aggregation Revisited. *Systems Research*, 13(2):86–93, 2001.

[9] D. Ellis, C. Cotton, and M. Mandel. Cross-correlation of beat-synchronous representations for music similarity. In *International conference on acoustics, speech and signal processing (ICASSP)*, pages 57–60, Las Vegas, 2008. IEEE.

[10] D. Ellis and G. Poliner. Identifying Cover Songs withchroma featires and dynamic beat tracking. In IEEE, editor, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–16, New York, 2007. IEEE.

[11] P. Foster, S. Dixon, and A. Klapuri. Identifying Cover Songs Using Information-Theoretic Measures of Similarity. *IEEE Transactions on Audio, Speech and Language Processing*, 23(6):993–1005, 2015.

[12] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[13] E. Humphrey, O. Nieto, and J. Bello. Data Driven and Discriminative Projections for Large-scale Cover Song Identification. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, pages 4–9, 2013.

[14] M. Khadkevich and M. Omologo. Large-Scale Cover Song Identification Using Chord Profiles. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, pages 5–10, 2013.

[15] A. Langville and C. Meyer. *The Science of Rating and Ranking - Who's #1?* Princeton University Press, 2012.

[16] J. Osmalskyj, P. Foster, S. Dixon, and J.J. Embrechts. Combining features for cover song identification. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 462–468, Malaga, 2015.

[17] J. Osmalskyj, S. Pierard, M. Van Droogenbroeck, and J.J. Embrechts. Efficient database pruning for large-scale cover song recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 714–718, Vancouver, BC, 2013.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] R. Prati. Combining feature ranking algorithms through rank aggregation. In *Proceedings of the International Joint Conference on Neural Networks*, pages 10–15, 2012.

[20] J. Serrà. *Identification of Versions of the Same Musical Composition by Processing Audio Descriptions*. PhD thesis, 2011.

[21] J. Serrà, E. Gómez, and P. Herrera. Transposing Chroma Representations to a Common Key. In *IEEE Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pages 45–48, 2008.

[22] J. Serrà, 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*, 16(6):1138–1151, 2008.

[23] C. Tralie and P. Bendich. Cover song identification with timbral shape sequences. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 38–44, Malaga, 2015.