COMMUNITY-BASED COVER SONG DETECTION

Marc Sarfati
Spotify
marcs@spotify.com

Anthony Hu
Spotify
anthonyh@spotify.com

Jonathan Donier
Spotify
jdonier@spotify.com

ABSTRACT

Audio-based cover song detection has received much attention in the MIR community in the recent years. To date, the most popular formulation of the problem has been to compare the audio signals of two tracks and to make a binary decision based on this information only. However, leveraging additional signals might be key if one wants to solve the problem at an industrial scale. In this paper, we introduce an ensemble-based method that approaches the problem from a many-to-many perspective. Instead of considering pairs of tracks in isolation, we consider larger sets of potential versions for a given composition, and create and exploit the graph of relationships between these tracks. We show that this can result in a significant improvement in performance, in particular when the number of existing versions of a given composition is large.

1. INTRODUCTION

With the rise of online streaming services, it is becoming easier for artists to share their music with the rest of the world. With catalogs that can reach up to tens of millions of tracks, one of the rising challenges faced by music streaming companies is to assimilate ever-better knowledge of their content – a key requirement for enhancing user and artist experience. From a musical perspective, one highly interesting aspect is the detection of composition similarities between tracks, often known as the *cover song* detection problem. This is, however, a very challenging problem from a content analysis point of view, as artists can make their own version of a composition by modifying any number of ingredients – instruments, harmonies, melody, rhythm, structure, timbre, vocals, lyrics, among others.

Over the years, it has become customary in the Music Information Retrieval (MIR) literature to address the cover song detection problem in what is arguably the most challenging setting. Indeed, most papers attempt to detect composition relationships between pairs of tracks based on their two audio signals only – in other words, completely out of context and without using any metadata information. While this well-defined task makes sense from an academic perspective, it might not be the optimal approach for solving the problem at an industrial scale [4].

The second starting point of our work is the fact, often mentioned in cognitive science, that commonly observed patterns are represented and stored in a redundant fashion in the human brain, which makes them more likely to be retrieved, recognised and identified than patterns that are observed less frequently [14]. If true, this would apply to our assessment of composition similarities as well. The main idea behind our work is that the corpus of existing versions of a composition can be precisely a substitute for these multiple representations.

Following these guiding intuitions, we turn to a new use case, where we do not just have pairs but a pool of *candidates* that are likely to be instances of some given musical work (according *e.g.* to some first metadata analysis). We then compare these candidates not only to *one* reference version (*e.g.* the original track, if it exists) but also to other candidate versions. We then build a graph of all these versions to identify composition clusters. Sometimes, when hundreds or thousands of versions of a given work exist (which is quite common in the catalogue of a streaming company), this ensemble-based approach can result in substantial improvements on the cover detection task.

In Section 2 we present a review of the literature on cover identification. In Section 3, we present the 1-vs-1 cover identification algorithm that we use throughout the paper, which is heavily based on [23]. The main contribution of this paper lies in Section 4, in which we present the new use case for cover identification described in the previous paragraph.We then showcase our method with examples in Section 5 and discuss some challenges in Section 6.

2. RELATED WORK

A number of possible approaches to cover song identification [12, 15] have been developed in the last decade. The authors of [9] introduced a first solution to this problem which has been used as a starting point for many subsequent studies. The main idea is to extract a list of beat-synchronous [6] chroma features from two input tracks and quantify their similarity by applying dynamic programming algorithms to a cross-similarity matrix derived from these features. This algorithm has been refined in [8] by adding a few modifications such as tempo biasing. Harmonic Pitch Class Profile (HPCP) features (chroma features) have proven very useful in cover identification [9, 18–20] as they capture meaningful musical information for composition. Other features have subsequently been introduced, such as self-similarity matrices of MFCC features [23, 24]. To take advantage

of the complementary properties of different types of features, [23] further introduced a method to combine several audio features by fusing the associated cross-similarity matrices, which resulted in a significant increase in performance compared to single-feature approaches. Having extracted audio features from two tracks to be compared, most methods use dynamic programming (either Dynamic Time Warping or the Smith-Waterman algorithm [22]) to assign a score to the pair [9, 23, 24]. One drawback of these methods is that they are computationally expensive and cannot be run at scale. Scalable solutions have been developed by mapping audio features to smaller latent spaces. For instance, [1, 13] use Principal Component Analysis (PCA) to compute a condensed representation of audio features which they use to perform a large-scale similarity search (*e.g.* a nearest neighbor search). In the same vein, [10, 17] use deep neural networks to learn low-dimensional representations of chroma features.A smaller number of papers take an ensemble-based approach and report increases in performance. In particular, [3, 23] leverage the network of songs using Similarity Network Fusion thereby fusing scores obtained via different methods, while [21] investigate several clustering methods and report in particular that the original song tends to be central within their communities.

3. PAIRWISE MATCHING

As mentioned above, our ensemble-based cover identification method consists of two steps. For a given work, we proceed to: ***(i)*** a pairwise (1-vs-1) comparison of all the tracks in a pool of potential candidates, ***(ii)*** a clustering of these candidates based on the results of step (i). In this section we present the 1-vs-1 cover song identification algorithm (i) which will be used as a starting point for our ensemble-based approach, and evaluate its performance on two distinct cover datasets.

3.1 The algorithm

For the purposes of this work, any 1-vs-1 similarity measure could be used for step (i), as we are mainly interested in quantifying the impact of step (ii) on the overall performance. We have chosen to rely on an implementation of the algorithm introduced in [23], as the algorithm achieves the best results to date on the *Covers80* [7] and *MSD* (*Covers1000*) datasets [2]. For a high-level overview of the pipeline, please refer to Figure 2 in [23]. As with most algorithms presented in Section 2, it can be decomposed into two stages: first, it extracts a list of meaningful audio features from the two tracks to be compared, then it computes a similarity score based on these. The details of this method are not directly relevant to our work, so we will focus here on a quantitative assessment of its performance, to give the reader a quantitative idea of our starting point. More details on the algorithm can be found in [23].

3.2 Quantitative evaluation of the 1-vs-1 method

We evaluate our implementation of [23] on two different datasets, and compare it with the numbers reported in the original paper as well as with a publicly available implementation of [23] by its author. [1] To make the comparison more interpretable, we evaluate two versions of our implementation with two sets of parameters: *Params1* mimics the parameters used in [23], and should therefore produce numbers that very similar to those described in the original paper, while *Params2* uses shorter 8-beats-long blocks.

We first compare the algorithms on the widely used *Covers80* dataset [7] to enable comparison with other published methods. The dataset is composed of 160 tracks that are divided into two sets (A and B) of 80 tracks each, with every track in set A matching one (and only one) track in set B. For each of the 160 tracks, we compute its score with all the other 159 tracks and report the rank of its true match. Table 1 reports the Mean Rank (MR) of the true match (1 is best), the Mean Reciprocal Rank (MRR) [5], as well as the Recall@1 (R@1) and Recall@10 (R@10). We also compute the so-called *Covers80 scores* by querying each track in set A against all the tracks in set B and reporting the number of matches found with rank 1. [2] Overall, our results are close to the ones reported in [23] – even though we could not quite reach the numbers given in their paper.

	Covers80					**Internal Dataset**	
	MR	MRR	R@1	R@10	*Covers80* score	Recall	Recall (no Jazz)
[23] paper	**7.8**	**0.85**	**82.4%**	89.9%	**68/80**	-	-
[23] code	8.6	0.77	71.7%	**91.2%**	62/80	73.2%	81.8%
Params1	10.5	0.81	78.6%	85.5%	64/80	79.2%	87.8%
Params2	13.2	0.75	73.0%	80.5%	60/80	**85.8%**	**95.1%**

Table 1: Comparison of our implementations (*Params1* and *Params2*) against the implementations of [23], on the *Covers80* dataset and on our internal dataset. The recall rates for the internal dataset correspond to a false positive rate of 0.5%. For each column, the best performance is printed in bold.

To complement this baseline, we have created an internal dataset of 452 pairs of covers grouped into several categories, obtained by metadata filtering based on the keywords *Acoustic Cover, Instrumental Cover, Karaoke, Live, Remix, Tribute* as well as some *Classical* and *Jazz* covers. Such granularity allows us to compare the performance of our algorithm across genres and cover types, providing a new perspective on the problem, as shown in Table 2. We have tested the two versions of our algorithm on the 452 positive pairs and 10,000 negative pairs selected uniformly at random. We selected the classification threshold to ensure a very low false positive rate below 0.5%. Results are presented in Table 1. Our algorithm reaches 85.8% recall, versus 73.2% for the publicly available implementation of [23] [3] . Note that jazz is the most challenging genre to

[1] https://github.com/ctralie/GeometricCoverSongs

[2] Each track from set A is now queried against the 80 tracks from set B, instead of all other 159 tracks.

[3] As the computational time is much higher for this algorithm, we only computed the false positive rate using 500 negative pairs.

detect, as jazz covers include a lot of improvisation that can be structurally different from their parent track (see Table 2). If we remove jazz covers from the dataset, the recall increases to 95.1% with the *Params2* implementation.

Type	Acoustic	Instr.	Karaoke	Live	Remix	Tribute	Classical	Jazz
# of pairs	57	63	46	57	31	53	77	68
Recall	94%	84 %	97%	100%	93 %	96 %	100 %	35 %

Table 2: Recall rates for each genre in our internal dataset, with a $< 0.5\%$ false positive rate.

In view of these results, we will use our own implementation with *Params2* throughout the rest of this paper, as it is faster and performs best on our internal dataset, which is larger and more diverse than *Cover80*.

3.3 Distributions of scores

Figure 1 presents the histogram of pairwise scores for all the positive and negative pairs in our internal dataset. The distribution of scores for the negative pairs is short-tailed and tightly concentrated around $s = 2$. This means that above $s \simeq 5$, all the pairs can be matched with high confidence. The distribution of scores for the positive pairs is much wider. As we can see from the histogram, a non-negligible fraction of these pairs lies below the classification threshold (dashed vertical line) and thus cannot be detected with this 1-vs-1 method. The purpose of the next section will be to apply an ensemble method to a pool of candidate versions of a given work, to bring these undetected candidates above the threshold by exploiting the many-to-many relationships between the candidates.

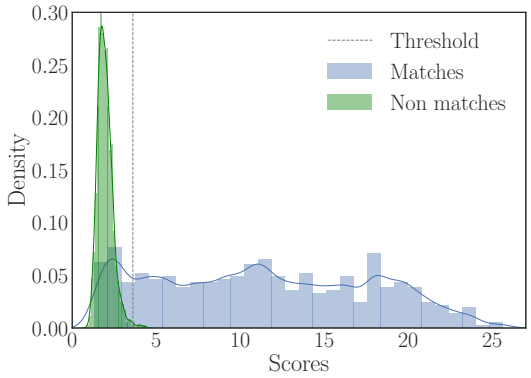
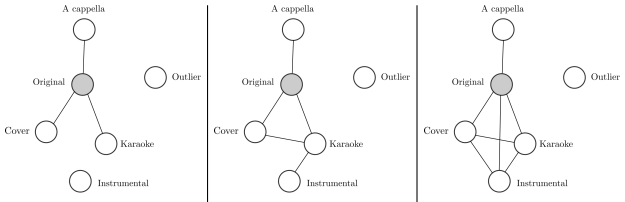


Figure 1: Histogram of the scores for positive *(blue)* and negative *(green)* pairs on our internal dataset. The threshold corresponds to the threshold used for Table 2.

4. ENSEMBLE ANALYSIS

While the 1-vs-1 algorithm we presented in Section 3 gives satisfying results overall, it still struggles on covers that are significantly different from their original track. Here we show how analyzing a large pool of candidate covers for one given reference track can improve the quality of the matching. The intuition behind this idea is that a cover version can match the reference track poorly, but match another intermediate version which is closer to the reference. For instance, an acoustic cover can be difficult to detect on a 1-vs-1 basis, but might match a karaoke version which itself strongly matches the reference track. We therefore turn to a new use case, where we not only compare single pairs (*e.g.* one reference track against one possible cover), but instead start from a pool of candidates that are all likely to be instances of some given composition (or *work*). Usually, this pool corresponds to candidates that have been pre-filtered according to some non-audio related signal, *e.g.* their title, and might comprise up to a few thousands candidates, depending on the popularity of the work and the specificity of the pre-filtering step.



(a) Computing scores versus the reference track

(b) Computing all pairwise scores

(c) Final graph

Figure 2: Direct (a) vs. ensemble-based approach (b)-(c).

4.1 Computing all pairwise scores

Given a set of N candidate versions of a work, we first compare all possible pairs of candidates within the set, resulting in $\frac{N(N-1)}{2}$ distinct scores $\{s_{ij}\}_{1 \leq i < j \leq N}$. As mentioned above, if the candidates have been pre-filtered using some metadata-matching algorithm, N typically varies from a few dozen to a few thousand candidates.

4.2 Scores to distances

Figure 1 shows that almost all negative pairs have scores between 0 and 4 while scores above 8 always correspond to positives. Scores above 8 should thus indicate a high probability of a true match regardless of the score, while a variation in score around 4 should have a significant impact on that probability. To account for this fact, we convert our scores into more meaningful distances using a logistic function: $d_{ij} = \left(1 + e^{-\frac{s_{ij} - m}{\sigma}}\right)^{-1}$, where s_{ij} is the score associated to pair (i, j) and d_{ij} is the resulting distance. We have found that the values $\sigma = 0.5$ and $m = 4.3$ work well with the distance-collapsing algorithm introduced in the next section.

4.3 Collapsing the distances

Let $D = \{d_{ij}\}$ denote the pairwise distance matrix between all pairs of candidates (see Figure 3, top left). The idea behind the ensemble-based approach is to exploit the geometry of the data to enhance the accuracy of the classification – for example, the fact that a track can match the reference track better through intermediate tracks than

directly. We use a loose version of the Floyd-Warshall algorithm [11] to update the distances in D, such that the new distances satisfy the triangular inequality most of the time [4]. The method is presented in Algorithm 1.

Algorithm 1 Loose Floyd-Warshall

procedure COLLAPSEDISTANCES(distance matrix D)
  while D still updates **do**
    for i, j in $1..N$ **do**
      $\widetilde{D}(i,j) \leftarrow \mathbf{min}^{(2)}_{k \neq i,j} D(i,k) + D(k,j) + \eta$
      $D(i,j) \leftarrow \mathbf{min}\Big(D(i,j),\ \widetilde{D}(i,j)\Big)$

Here $\min^{(k)}(x)$ denotes the k^{th} smallest value of a vector x. Using $k > 0$ allows to gain robustness as several short paths are required to merge clusters. We have found that the algorithm is slightly more robust when imposing a penalty $\eta > 0$ for using an intermediate node, which we have set to $\eta = 0.01$ after performing a grid-search optimization.

Figure 3 shows the distance matrix before (top left) and after (top right) updating the distances using Algorithm 1, for a set of candidates versions of *Get Lucky* by Daft Punk. We can see that the updated distance matrix has a more neatly defined division between clusters of tracks. The figure shows one large cluster in which all tracks are extremely close to each other (the white area), a few smaller clusters (white blocks on the first diagonal) and a number of isolated tracks that match only themselves.

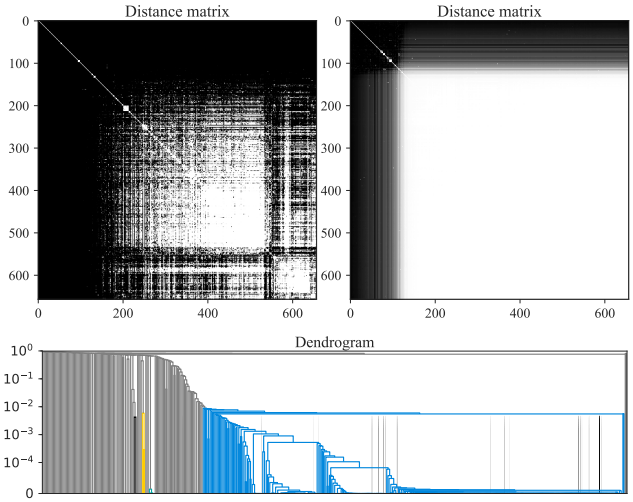


Figure 3: Top: The Floyd-Warshall algorithm applied to the distance matrix of *Get Lucky*, with (left) original distance matrix and (right) the distance matrix after applying the Floyd-Warshall algorithm. For better visual interpretation, tracks are reordered along the axes according to proximity. Darker shades correspond to larger distances. Bottom: dendrogram representation of the hierarchical clustering on the Floyd-Warshall distance matrix.

[4] The distances that would be obtained by applying the original Floyd-Washall algorithm to D would always satisfy the triangular inequality, but the resulting configuration would be very sensitive to outliers. Our method is more robust to outliers, as it requires to find more than one better path to update the distance between two points.

4.4 Hierarchical clustering

We then proceed to a clustering of the tracks using the updated distance matrix defined in 4.3, denoted D'. We use hierarchical clustering with centroid linkage [16] as we have no prior knowledge on the number of clusters in the graph. Figure 3 (bottom) shows a dendrogram representation of the hierarchical clustering applied to D'. In this example, if we apply a relatively selective threshold, we find one major cluster (colored in blue in Figure 3) that contains 97% of the true positives and no false positives. Most other clusters contain a single element, which are all the negative tracks and the remaining 3% of the positives. If we set the clustering threshold lower, then we can get more granular clusters within a same work.

4.5 Final score

In order to assign each track a final score that measures its similarity to the reference track, we use the *cophenetic distance* to the reference track that is produced by the hierarchical clustering (*i.e.* the minimum distance threshold for which they would find themselves in the same cluster as the reference track). Each track is thus assigned a final score in $0 - 100$, simply taken equal to $100 \times$ (1 - cophenetic distance), such that exact matches have a score of 100.

5. ANALYSIS OF REAL WORLD EXAMPLES

5.1 Data

We now apply the above to real world data. Our dataset consists of 10 sets of candidates that correspond to 10 works that we want to find the versions of. These 10 works span multiple genres and musical styles, including Hip Hop, R&B, Rap, Pop and Jazz. For a given work, we create the set of candidates by performing a metadata search of the given work's title on the whole Spotify catalogue, which we then annotate manually. Across the given works that we study, this produces sets of candidates whose sizes vary from a few hundred to a few thousand candidate tracks. Each set includes a reference track, which will be the anchor point for that composition. More details on the dataset can be found in Table 3.

Work	# tracks	% positives	Reference artist
Airplane	811	19%	B.o.B
Believer	2552	5%	Imagine Dragons
Blurred Lines	386	71%	Robin Thicke
Bodak Yellow	110	78%	Cardi B
Brown Sugar	721	5.8%	D'Angelo
Embraceable You	1319	94%	Sarah Vaughan
Get Lucky	657	83%	Daft Punk
Halo	2995	7.9%	Beyoncé
Heartless	1747	5.3%	Kayne West
Imagine	2044	50%	John Lennon

Table 3: The "10 works" dataset. For each work, we have selected a reference track that will be our anchor point for that composition. *Click on a work to play the reference track in the browser.*

5.2 Outline of the analysis

For each of these works, we analyze the set of candidates following the steps outlined in the previous two sections, providing us with two sets of outputs for each work: ***(a)*** the *direct score*, defined as the output of the 1-vs-1 algorithm between each candidate and the reference track, as described in Section 3 (rescaled between 0 and 100); ***(b)*** the *ensemble-based score*, produced by the method described in Section 4 (also between 0 and 100).

In the next section we start by quantitatively evaluating our ensemble-based approach (b) against the direct approach (a), before turning to some qualitative examples.

5.3 Quantitative results

We define two different metrics to evaluate the direct and the ensemble-based methods:

Ranking metric: For each work, we pick the value of the threshold that minimizes the number of classification errors, and report the number of errors. We call this a *ranking metric* as the number of errors is minimized when positives and negatives are perfectly ranked, regardless of their scores. We also report the corresponding recall and false positive rates for this threshold.

Classification metric: We fix a universal classification threshold and compute the corresponding number of classification errors.

Work	Best thr.	Ranking errors - direct					
		False negatives		False positives		Both	
		Abs.	Rel.	Abs.	Rel.	Abs.	Rel.
Airplane	12.1	33	21.9%	1	0.2%	34	4.2%
Believer	18.2	6	5.2%	0	0.0%	6	0.2%
Blurred Lines	10.1	19	7.0%	9	8.3%	28	7.3%
Bodak Yellow	6.1	6	7.0 %	6	33.3%	12	10.9%
Brown Sugar	12.1	2	4.8%	1	0.1%	3	0.4%
Embraceable You	4	0	0%	74	98.7 %	74	5.6 %
Get Lucky	10.1	17	3.1%	3	2.6%	20	3.0%
Halo	11.1	8	3.4%	8	0.3%	16	0.5%
Heartless	12.2	15	16.3%	2	0.1%	17	1.0%
Imagine	15.2	72	7.1%	17	1.7%	89	4.4%

(a) Direct approach.

Work	Best thr.	Ranking errors - ensemble-based					
		False negatives		False positives		Both	
		Abs.	Rel.	Abs.	Rel.	Abs.	Rel.
Airplane	70.7	4	2.6%	3	0.5%	7	0.9%
Believer	85.9	0	0.0%	0	0.0%	0	0.0%
Blurred Lines	52.5	0	0.0%	0	0.0%	0	0.0%
Bodak Yellow	29.3	9	10.5%	0	0.0%	9	8.2%
Brown Sugar	70.7	0	0.0%	1	0.1%	1	0.1%
Embraceable You	40.4	22	1.8 %	19	25.3 %	41	3.1 %
Get Lucky	78.8	5	0.9%	2	1.7%	7	1.1%
Halo	98.0	4	1.7 %	20	0.7%	24	0.8%
Heartless	83.8	8	8.7%	1	0.1%	9	0.5%
Imagine	96.0	1	0.1%	5	0.5%	6	0.3%

(b) Ensemble-based approach.

Table 4: Optimal thresholds and corresponding results for the ranking metric.

Table 4 shows the results for the ranking metric for each work in our dataset. For the optimal thresholds, we report the number of false negatives, false positives and the sum of both (*i.e.* the total number of classification errors). We also compute the corresponding false negative rate, false positive rate and total error rate.

Table 4a shows the ranking results for the direct approach. Interestingly, the number of false negatives tends to be higher than the number of false positives. [5] This is in line with the histogram in Figure 1, which shows a short-tailed distribution for the negatives and a wider distribution for the positives. Overall, the error rate lies between $0-10\%$, corresponding to a recall rate between 80% and 97% and a false positive rate below 10% (except for *Bodak Yellow* and *Embraceable You* which have a very small number of negatives to begin with – the latter case is in fact degenerate as nearly all tracks are classified as matching).

Table 4b shows the ranking results for our ensemble-based approach. The number of ranking errors is substantially lower than for the direct approach, including both the number of false positives and false negatives, as the total error rate goes down below 1% in most cases. Again, the main exception is *Bodak Yellow*, which has the smallest number of candidates. [6] *Embraceable You* is the second most challenging work, but remarkably its threshold is no longer degenerate, meaning that the method has now found a way to separate the candidates. Notably, the number of false negatives no longer outnumbers the number of false positives: the ensemble-based approach has successfully caught most of the difficult tracks that poorly matched the reference track. Among the few tracks that are still missed, several are actually very close to the threshold, and only a handful are still completely undetected (cf Table 7).

Table 5 shows the results for the classification metric. The universal threshold for each approach is defined as the median of the optimal thresholds obtained in the ranking experiment above. Again, we report the number of false negatives, the number of false positives and the sum of both. We also compute the corresponding false negative rate, false positive rate and total error rate. Here again, the results of the ensemble-based approach are overall superior to the direct approach, mostly due to an increase in recall. Although Table 5a is quite similar to Table 4a, which is a sign that the threshold on direct scores can be chosen in a nearly universal way, Table 5b differs considerably from Table 4b for some specific works (namely *Halo*, *Imagine* and *Believer*). This happens as the optimal threshold is significantly higher on these works (often $> 95\%$), letting a large number of false positives above the 78.8% threshold.

5.4 Examples

For each work, we can identify the cases where the ensemble-based approach has allowed us to detect previously undetected tracks, and trace back the optimal path that joined the reference track and the newly found track. Table 6 shows a few examples of such paths for various works. For each example, the reference track is shown at

[5] *Embraceable You* is an exception, as its threshold is degenerate and all tracks are classified as matching.

[6] It was also a genuinely difficult example and we struggled to annotate it.

Work	Thr.	Classification errors - direct					
		False negatives		False positives		Both	
		Abs.	Rel.	Abs.	Rel.	Abs.	Rel.
Airplane	12.1	33	21.9%	1	0.2%	34	4.2%
Believer	12.1	4	3.5%	91	3.7%	95	3.7%
Blurred Lines	12.1	28	10.3%	0	0%	28	7.3%
Bodak Yellow	12.1	49	57 %	0	0%	49	44.5%
Brown Sugar	12.1	2	4.8%	1	0.1%	3	0.4%
Embraceable You	12.1	753	60.5 %	3	4 %	756	57.3 %
Get Lucky	12.1	23	4.2%	1	0.9%	24	3.7%
Halo	12.1	12	5.1%	4	0.1%	16	0.5%
Heartless	12.1	15	16.3%	2	0.1%	17	1.0%
Imagine	12.1	49	4.8%	94	9.3%	143	7%

(a) Direct approach.

Work	Thr.	Classification errors - ensemble-based					
		False negatives		False positives		Both	
		Abs.	Rel.	Abs.	Rel.	Abs.	Rel.
Airplane	78.8	9	6.0%	1	0.2%	10	1.2%
Believer	78.8	0	0.0%	27	1.1%	27	1.1%
Blurred Lines	78.8	2	0.7%	0	0.0%	2	0.5%
Bodak Yellow	78.8	19	22.1%	0	0.0%	19	17.3%
Brown Sugar	78.8	2	4.8%	1	0.1%	3	0.4%
Embraceable You	78.8	70	5.6 %	1	1.3 %	71	5.4 %
Get Lucky	78.8	5	0.9%	2	1.7%	7	1.1%
Halo	78.8	0	0 %	163	5.9%	163	5.4%
Heartless	78.8	7	7.6%	4	0.2%	11	0.6%
Imagine	78.8	0	0%	158	15.6%	158	7.7%

(b) Ensemble-based approach.

Table 5: Universal threshold and corresponding results for the classification metric.

Work	Depth	Main artist (& link)	Direct scores	Ensemble-based scores
Imagine	0	John Lennon	**100**	**100**
	1	Classic Gold Hits	**60.0**	**99.99**
	2	A Perfect Circle	**21.6**	**97.9**
	3	Yoga Pop Ups	8.6	**97.9**
Heartless	0	Kanye West	**100**	**100**
	1	The Fray	**34.1**	**99.85**
	2	William Fitzsimmons	9.5	**90.7**
Get Lucky	0	Daft Punk	**100**	**100**
	1	Samantha Sax	**40.4**	**99.95**
	2	Dallas String Quartet	6.9	**86.6**
Halo	0	Beyonce	**100**	**100**
	1	LP	**27.6**	**99.96**
	2	Dion Lee	7.96	**99.16**
Halo	0	Beyonce	**100**	**100**
	1	Karaoke Universe	**20.5**	**99.96**
Halo Halo	2	Fajters	7.45	**99.35**

Table 6: Some examples of tracks that are undetected by the direct approach and captured by the ensemble-based approach, in the ranking experiment. The scores that are above the detection thresholds for each method are displayed in bold (the corresponding detection thresholds can be found in Table 4). *Click on an artist to play in the browser.*

the top of the cell (depth 0), and the newly found track at the bottom of the cell (depth > 1), with the intermediate tracks that allowed to bridge the gap in between. All the examples are true positives, except for the last example (*Halo Halo* by Fajters), which has been erroneously matched to a karaoke version of the reference track.

What about the tracks that are still undetected? Table 7 shows examples of tracks that are still undetected by our ensemble-based approach for a couple of works. No clear pattern emerges – apart from the fact that they are often in a very different musical style from the original.

Work	Main artist - Title	Direct score	Ensemble -based score
Get Lucky	The Getup - *Get Lucky*	6.4	24.9
Halo	Amanda Sense - *Halo*	12.4	94.3
Imagine	Dena De Rose - *Imagine*	10.4	86.2
Embraceable You	Earl Hines - *Embraceable You*	9.6	36.3
	Samina - *Embraceable You*	5.8	17.0
Heartless	Bright Light - *Heartless*	11.9	50.3
	Rains - *Heartless*	6.4	47.7
Bodak Yellow	Josh Vietti - *Bodak Yellow*	5.8	14.2
	J-Que Beenz - *Bodak Yellow*	5.6	13.0
Airplanes	Em Fresh - *Airplanes*	5.5	66.1
	Lisa Scinta - *Airplanes*	9.6	55.0

Table 7: Some example of tracks that are undetected by the ensemble-based approach in the ranking experiment, with their scores for both methods. *Click on a title to play in the browser.*

6. DISCUSSION

One main challenge associated with our ensemble-based approach is how to correctly handle transitivity. This issue emerges from the fact that compositions are not mutually exclusive. For example, a medley might constitute a bridge between two distinct composition groups, which our algorithm would then merge together (which is undesirable). There are probably at least two ways around this issue: one is metadata-based (*i.e.* identify these potential outliers from the metadata and exclude them from the graph computation), while another is to detect them directly from the graph structure (identify bridges between otherwise unrelated clusters).

7. CONCLUSION

In this paper, we consider the following formulation of the cover song identification problem: among a pool of candidates that are likely to match one given reference track, find the actual positives. We have introduced a two-step approach, with a first step that computes pairwise similarities between every pair of tracks in the pool of candidates (for which any known 1-vs-1 approach can be used), and a second ensemble-based step that exploits the relationships between all the candidates to output final results. We have shown that this second step can significantly improve the performance compared to a pure 1-vs-1 approach, in particular on the ranking task, where the error rate is down from a few percents to less than 1% in general. The classification task is naturally more challenging as the optimal threshold might vary from work to work, suggesting that the method would be best exploited as a complement to human annotations – where the human's task would mainly be to find the optimal threshold for the classification. Automating this last step turned out to be non-trivial and is left for future work.

8. REFERENCES

[1] Thierry Bertin-Mahieux and Daniel PW Ellis. Large-scale cover song recognition using the 2d fourier transform magnitude. In *International Conference on Music Information Retrieval (ISMIR)*, pages 241–246, 2012.

[2] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *International Conference on Music Information Retrieval (ISMIR)*, 2012.

[3] Ning Chen, Wei Li, and Haidong Xiao. Fusing similarity functions for cover song identification. *Multimedia Tools and Applications*, 77(2):2629–2652, 2018.

[4] Albin Andrew Correya, Romain Hennequin, and Mickaël Arcos. Large-scale cover song detection in digital music libraries using metadata, lyrics and audio features. *arXiv preprint arXiv:1808.10351*, 2018.

[5] Nick Craswell. Mean reciprocal rank. In *Encyclopedia of Database Systems*, pages 1703–1703. Springer, 2009.

[6] Simon Durand, Juan Pablo Bello, Bertrand David, Gaël Richard, Simon Durand, Juan Pablo Bello, Bertrand David, Gael Richard, Bertrand David, Gael Richard, et al. Robust downbeat tracking using an ensemble of convolutional networks. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(1):76–89, 2017.

[7] Daniel PW Ellis. The "Covers80" cover song data set. *URL: http://labrosa. ee. columbia. edu/projects/coversongs/covers80*, 2007.

[8] Daniel PW Ellis and C Cotton. The 2007 labrosa cover song detection system. *Music Information Retrieval Evaluation eXchange (MIREX)*, 2007.

[9] Daniel PW Ellis and Graham E Poliner. Identifying cover songs with chroma features and dynamic programming beat tracking. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 4, pages 1429–1432. IEEE, 2007.

[10] Jiunn-Tsair Fang, Chi-Ting Day, and Pao-Chi Chang. Deep feature learning for cover song identification. *Multimedia Tools and Applications*, 76(22):23225–23238, 2017.

[11] Robert W Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

[12] Peter Grosche, Meinard Müller, and Joan Serrà. Audio content-based music retrieval. In *Dagstuhl Follow-Ups*, volume 3. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2012.

[13] Eric J Humphrey, Oriol Nieto, and Juan Pablo Bello. Data driven and discriminative projections for large-scale cover song identification. In *International Society for Music Information Retrieval (ISMIR)*, pages 149–154, 2013.

[14] Ray Kurzweil. *How to create a mind: The secret of human thought revealed*. Penguin, 2013.

[15] Meinard Müller. *Fundamentals of music processing: Audio, analysis, algorithms, applications*. Springer, 2015.

[16] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

[17] Xiaoyu Qi, Deshun Yang, and Xiaoou Chen. Audio feature learning with triplet-based embedding network. In *AAAI*, pages 4979–4980, 2017.

[18] Suman Ravuri and Daniel PW Ellis. Cover song detection: from high scores to general classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 65–68. IEEE, 2010.

[19] Joan Serra, Emilia Gómez, and Perfecto Herrera. Transposing chroma representations to a common key. In *IEEE CS Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pages 45–48, 2008.

[20] Joan Serra, Emilia Gómez, Perfecto Herrera, and Xavier Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 16(6):1138–1151, 2008.

[21] Joan Serrà, Massimiliano Zanin, Perfecto Herrera, and Xavier Serra. Characterization and exploitation of community structure in cover song networks. *Pattern Recognition Letters*, 33(9):1032–1041, 2012.

[22] Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[23] Christopher J Tralie. Early MFCC and HPCP fusion for robust cover song identification. In *International Society for Music Information Retrieval Conference (ISMIR)*, page 294–301, 2017.

[24] Christopher J Tralie and Paul Bendich. Cover song identification with timbral shape sequences. In *International Society for Music Information Retrieval Conference (ISMIR)*, page 38–44, 2015.