Multimedia Tools and Applications (2023) 82:31591–31606
https://doi.org/10.1007/s11042-023-14787-2

Accuracy comparisons of fingerprint based song recognition approaches using very high granularity

Salvatore Serrano[1] · **Marco Scarpa**[1]

Received: 30 June 2022 / Revised: 30 August 2022 / Accepted: 5 February 2023 /
Published online: 21 March 2023

Abstract
Music and song recognition is an activity of wide interest for researchers and companies due to the intrinsic challenges and the possible economical profits it can give. Despite basic algorithms about song recognition are simple in principle, it is quite difficult to obtain an efficient and robust approach able to generate an effective algorithm for identifying short piece of audio on the fly. In this paper, we compare the results obtained using a new algorithm we recently proposed against several baseline approaches in terms of accuracy when very short pieces of audio are processed. Experimental results, performed using both a subset of the MTG-Jamendo dataset and a proprietary audio corpus containing 7000 songs, show our approach outperform the others in particular for excerpts of audio shorter than $3s$.

Keywords Song recognition · Audio fingerprint · Power spectral density · Hamming distance · Binary fingerprints

1 Introduction

In the last six consecutive years, the global recorded music market has seen an increasing of revenues. Specifically, this market grew by 7.4% in 2020. This growth has been obtained thanks to a continued rise in paid subscription streaming revenues which offset a decline in rights revenues due for physical performance probably caused by the COVID-19 pandemic [13]. In this scenario several companies get profit by applications able to automatically extract metadata from broadcast media (in several way, i.e., radio broadcasting, internet streaming, live concerts, music played in public places), specifically, by applications able to recognize copyrighted material in real time analyzing short excerpts of audio signals.

Salvatore Serrano and Marco Scarpa contributed equally to this work.

✉ Salvatore Serrano
sserrano@unime.it

Marco Scarpa
mscarpa@unime.it

[1] Department of Engineering, University of Messina,
C.da Di Dio (Villaggio S. Agata), Messina, 98166, ME, Italy

Several application scenarios permit revenues: monitoring at distributor side, transmission channel or consumer end; added-value services; integrity verification systems and so on [7].

The research whose results are partially reported here takes the cue from a collaboration with an "European - based" company that provides a music recognition service to different kinds of customers such as radio stations and advertisers. Its core business is indeed to measure airtime songs duration with the purpose of generating music charts and song popularity. The current framework of the commercial use consists in fingerprint database, recognition instance and FM transceiver. The primary goal of the collaboration was to improve the recognition algorithm they use in terms of accuracy, efficiency and quantity of manageable data, even if their approach is kept secret. Moreover, the proposed solution have to satisfy the subsequent mandatory constraints considered fundamental by the company for its business:

- ability to fast add new songs inside the dataset of songs to be recognized;
- ability to recognize short excerpts of audio in real time;
- ability to locate the time position of the short excerpt inside the recognized song in real time.

As specified in subsequent Section 2, several researches dealt with song recognition issues, thus some open-source implementations of specific algorithms exist. At the best of our knowledge, none of these works matches the above listed constraints and, at the same time, lets get good performance in terms of accuracy when very short excerpts of audio have to be recognized. Accordingly, the main contributions of this work are:

- analyzing performance of available open-source songs recognition tools in terms of accuracy against very short excerpts of audio to be recognized;
- comparing performance specified in the previous point against performance obtained by our approach and the algorithm currently used by the company;
- performing the previous comparisons using a public available dataset of songs.

As part of the collaboration, the company provided us a testing database containing thousands of songs they use in daily industrial practice; we used it in our experimentation in addition to the public MTG-Jamendo dataset [6].

The common approach for performing recognition of chunks of song is based on on-the-fly extraction of fingerprints characterizing short piece of songs that are then searched into a reference data-base storing fingerprints of original songs [7]. According to [7], requirements of each audio fingerprinting application includes: accuracy, reliability, robustness, granularity, security, versatility, scalability, complexity and fragility.

A song recognition algorithm able to obtain high accuracy with high granularity can also correctly identify very short excerpts of audio. The analysis of this feature and related performance, joined with the ability to add new published songs into the dataset on the fly, is the main motivation of this work. Short excerpts of copyrighted audio is often played inside advertisements and radio-jingles. One of the challenges of the companies working in the song recognition field is the ability to recognize each small excerpt of copyrighted audio on airplay. Indeed, in the music industry, the Performing Rights Organizations (PRO) do the administrative work of collecting performance royalties and distribute them to proper artists or their representatives. Public broadcasting royalties payouts system works as follows: the broadcaster purchases a blanket license from the local PRO; the licence will allow the broadcaster to play all music represented by the PRO; the broadcaster reports the songs it has broadcasted back to the PRO; the PRO uses those data for allocating and distributing the royalties due to right artists and/or their representatives. Often the royalties are related

not only to the air play-time and number of times a specific song is played, but also to the specific hour of the day and the specific day which can affect the audiance of the broadcasting. Broadcaster programmers are obligated to provide a record of every song they have put on the air to the PRO. This record transcription is called "broadcaster log". Given the scale of the operation, these logs are often riddled with missing details and errors (misspelled artist names, straight-up missing track data, something like "Track #" instead of the song's title). Moreover the transmission of different advertisements or jingles are often identified with a single voice like "adv". Corrupted broadcast logs mean that the PROs can't identify the right artist and so the royalties collected will never paid out. Incomplete broadcast logs mean that artists around the globe miss out on millions in potential revenue. The ability to identify short excerpts of copyrighted materials, played also inside jingles and advertisements, permits to perform a more accurate monitoring to the companies which perform airplay monitoring service like the company we collaborate with. Moreover, it will not be necessary to perform a prior association between advertisements/jingles and copyright materials, because the known copyrighted audio will be automatically identified regardless of the context in which it is played.

Specifically, in this paper, we compare accuracy in recognizing very short excerpts of audio materials (lasting from 1 to 5 seconds) obtained using an extension and refinement of our proposed approach introduced in [8, 19] and fully described in [20] against five baseline algorithms: the first one is a Shazam-like approach founded on landmark-based fingerprint method appeared in [29] and implemented as open source version with the name *Audfprint*[1] [11, 30]. The second one is *Dejavu*,[2] another open-source implementation of the Shazam-like approach [29] that uses the constellation algorithm [10]. The third and fourth ones are respectively a new implementation of the classic Shazam algorithm [29], named *Olaf*, and an updated version of the algorithm described in [24], named *Panako*, which uses the Gabor transform to move from time domain to spectral domain; *Panako* and *Olaf* are distributed as an open-source software.[3] The last one is the algorithm, based on a Philips-like approach, used by the company we are collaborating with for its business.

The paper is organized as follows: Section 2 introduces the most relevant works related to this research. In Section 3, we summarize the components and methodology of our proposed approach. The evaluation of our algorithm in comparisons with baselines and company algorithms are presented in Section 4. Finally, Section 5 concludes the overall proposed architecture and introduces future works.

2 Related work

Although song recognition issues can be considered an audio classification or annotation task [12, 16] and, accordingly, it appears could be solved using a "representation learning approach" [4, 18], it has very specific peculiarities:

1. audio/music classification algorithms try to detect, for instance, the music genre or mood of the track [14, 26] or, if the excerpt of audio under test contains the sound of a specific musical instrument, a specific set of musical instruments or/and the voice of a singer [21, 22]; instead, song recognition algorithms try to detect if played excerpts are

[1] https://github.com/dpwe/audfprint
[2] https://github.com/worldveil/dejavu
[3] https://github.com/JorenSix/Panako

extracted from a copyrighted audio track, an identifier of the copyrighted audio track (like song title and the artist name) and, usually, also the time position of the excerpt inside the original audio track;
2. classification algorithms deal with few number of classes while song recognition algorithms deal with hundreds of thousands or millions of tracks to detect;
3. classification algorithms deal with a constant number of classes that do not vary over the time, while song recognition algorithms deal with a number of tracks usually growing over time (due to the publications of new titles);

Application of audio classification includes intelligent recommender systems as a promising technology for music search; they aim to assist users in exploring large-scale music collections by identifying suitable songs based on their preferences [9], while, as mentioned in the previous section, application of song classification is usually airplay monitoring service.

In literature, song recognition issue is usually solved by means of a fingerprint approach, then, in this section, we will focus only on relevant works covering song recognition using this approach.

Fingerprint generation approaches in song recognition systems can be divided into three different types [33]: the first one describes the energy differences between adjacent frequency bands [15]; the second one locates spectral peaks, using either the relationship with other peaks [23, 24, 27, 29] or the energy information around the peaks to form a fingerprint [1]; the last one uses image retrieval techniques [3, 32].

Recently, some works, as papers of Yao et al. [33, 34], use a fingerprint extraction approach based on the technique proposed by Philips in [15] whereas Sonnleitner and Widmer [27] introduce a compact "four - dimensional", continuous hash representation of quadruples of points called *quads*. Although this latter approach can efficiently identify audio in large song collections, and it is robust to noise and audio quality degradation, as well as to severe distortions of speed, tempo and frequency, the generation of each "quad" appears rather complex as reported in [33] making it difficult to be used in real time applications. Moreover, memory requirements to store the data structures used to recognize the songs are very large. In [25], the fundamental frequency components extracted from the audio were matched with the frame-fundamental frequency domain and used to compose what the authors call *fundamental frequency map* (FFMAP). Authors employed also a new hashing method named *spatial adaptive hashing* (SAH) in the similarity calculation process, to compare the audio contents. Even though the approach appears less complex then the quad-based one, it works with an entire song differently from the approach we proposed [5, 19] that is capable of locating the time position of very short snippets inside songs. Authors of [17] present an audio fingerprinting method based on locally linear embedding (LLE). In their approach, the bands around each peak in the frequency domain is divided into four groups of sub-regions and the energy of every sub-region is computed. The LLE is performed in each group and the audio fingerprint is encoded by comparing adjacent energies. Moreover, a matching strategy based on dynamic time warping (DTW) is adopted to solve the distortion due to linear speed changes. The authors of [2] introduced an unsupervised deep learning framework for generating audio fingerprints based on a Sequence-to-Sequence Autoencoder (SA) model composed of two linked Recurrent Neural Networks (RNN). This latter work is, to the best of our knowledge, the only approach who shows results for queries with length shorter than 3s: although experimental results in [2] appear very good (100% of accuracy using excerpt of $1s$), in our opinion the presented research reveals two drawbacks: 1) the dataset used to perform the experimentation (`VoxCeleb1`) is a speech based corpus, accordingly it is not clear how performance are

affected in the context of music recognition; 2) the complexity appears high when compared to a simple fingerprint based approach: it is not said if the SA has to be retrained when the size of the dataset grows with new songs insertion; in this latter case, the computation time could be too high in order to perform real time recognition of several audio tracks.

3 The proposed approach

3.1 Mel-PSD audio fingerprinting

Our proposed fingerprints are based on the estimation of the short time power spectral density (STPSD) of the audio signal obtained on a Mel frequency-scale [28]. Starting from an incoming audio stream of sufficient time-length, a fingerprint F can be built on-the-fly by extracting N_F adjoining linekeys:

$$F = \left[l_0, l_1, \ldots_{N_F-1}\right]. \tag{1}$$

A linekey is a string of B bits built by exploiting both a Welch like approach and an adaptive frequency variant threshold to represent the content of a fixed short piece of audio. We generate a linekey from the samples extracted from a window $W_\sigma(t_n)$ starting at time t_n whose length in time domain is L_σ; in this way, we generate a linekey characterizing an L_σ long piece of music at the t_n position inside the song. Following the Welch's approach, we use a shifting subwindow W_s, whose length is L_s, inside $W_\sigma(t_n)$ for computing K modified periodograms $\mathscr{I}_k(t_n)$, with $k = 0, \ldots, K-1$. We evaluate the periodogram $\mathscr{I}_k(t_n)$ of the k-th subwindow by applying an Hamming windowing to each subwindow and computing the squared magnitude of the FFT over $M > 16 \cdot B$ points, according to the following (2)

$$\mathscr{I}_k(t_n)[m] = \left| \sum_{j=0}^{N-1} x_j \cdot w_j \cdot e^{-\frac{i2\pi mj}{N}} \right|^2, \quad \forall m = 0, \ldots, M-1 \tag{2}$$

where N is the number of samples in each subwindow W_s, x_j are the samples in each subwindow and w_j are the samples of the Hamming window. A Mel frequency-scale bank of B filters is then applied to $\mathscr{I}_k(t_n)$ in order to obtain the energy contained in each of the B sub-bands $\mathscr{E}_k(t_n)$

$$\mathscr{E}_k(t_n)[i] = \sum_{m=M_{i-1}}^{M_i-1} \mathscr{I}_k(t_n)[m], \quad \forall i = 1, \ldots, B \tag{3}$$

The last step is to sum the $\mathscr{E}_k(t_n)$ for all the K periodograms and, to convert all the values to deciBel, obtaining an estimation of the power spectral density $\mathscr{P}(t_n)$ over B frequency sub-bands

$$\mathscr{P}(t_n)[i] = 10 \log_{10} \sum_{k=1}^{K} \mathscr{E}_k(t_n)[i], \quad \forall i = 1, \ldots, B \tag{4}$$

Starting from $\mathscr{P}(t_n)$, we need a threshold based on which we set the binary value at each frequency of the spectrum. To this purpose, we decided to adopt an adaptive frequency variant threshold. Specifically, we use an exponential approximation of the $\mathscr{P}(t_n)$ trend to build the frequency variant threshold using the Least Squares Fitting-Exponential approach [31]. According to this theory, the points of the fitting curve $y[i](t_n)$ are obtained by means of the following (5)

$$y[i](t_n) = a(t_n) \cdot e^{b(t_n) \cdot i}, \ \forall i = 1, \ldots, B \tag{5}$$

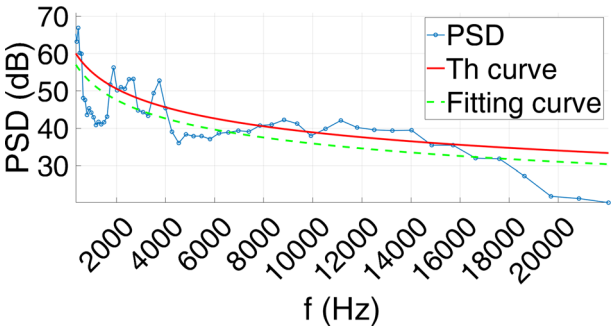

Fig. 1 Binary Linekey extraction

where $a(t_n)$ and $b(t_n)$ are adaptive parameters function of the current $\mathscr{P}(t_n)$. In order to avoid generation of different linekeys due to small oscillations around the fitting curve, we add a constant margin value m to the fitting curve $y[i]$ for deriving the threshold values $T(t_n)[i]$:

$$T(t_n)[i] = y[i] + m, \quad \forall i = 1, \ldots, B. \tag{6}$$

Exploiting both the $\mathscr{P}(t_n)[i]$ values and the frequency variant threshold $T(t_n)[i]$, the binary sequence of B bits is evaluated as

$$l(t_n)[i] = \begin{cases} 1, & \text{if} \quad \mathscr{P}(t_n)[i] > T(t_n)[i] \\ 0, & \text{otherwise} \end{cases}, \quad \forall i = 1, \ldots, B. \tag{7}$$

Figure 1 shows an example of bit sequence generation from $\mathscr{P}(t_n)$ estimation and the related threshold.

3.2 Song recognition by binary hamming distance measure

Exploiting the methodology described in Section 3.1, we represent songs as an ordered set of linekeys, by extracting them from each time interval τ from the beginning of a song. A song S_a, whose time length is L_a, is thus represented as

$$S_a = \{l_a(0), l_a(\tau), \ldots, l_a(n_a \cdot \tau)\},$$

with $n_a = \lceil \frac{L_a}{\tau} \rceil$. We denote the number of songs in the collection with N_s, and we refer to them with the corresponding indices.

Our song recognition process is based on another data structure $\mathcal{L}$ where we store all the linekeys generated by the overall songs collection and associate to each linekey $\mathcal{L}_i$ a list $\mathcal{C}_i$ of couples (id, p), with $i = 0, \cdots, N_l - 1$, where N_l is the overall number of linekeys; a couple represents the index id of a song where the linekey $\mathcal{L}_i$ is located and its position p inside the song.

When a fingerprint $F = (l_0, l_1, \ldots, l_{N_F-1})$ is generated for recognition purposes, we compare each of its linekeys with the linekeys in $\mathcal{L}$ searching for the nearest ones in terms of Hamming distance $d(\mathcal{L}_j, l_i)$. We estimate the song whose F belongs to with two different approaches.

Approach 1. The first approach simply counts the times a song is referenced to: i.e., we increase the counters associated to the songs whose id is in the list $\mathcal{C}_j$ such that $d(\mathcal{L}_j, l_i)$ is minimum; the song with the highest counter is marked as recognized. The song with the smallest index will be picked up, if more than one obtains the same score.

Approach 2. The second approach introduces a penalty for the linekeys far (in terms of hamming distance) from the searched one.

A set of N_s counters is used: one counter for each song in the corpus. For each $l_i \in F$, the counters are increased with the distance $d(\mathcal{L}_i, l_j)$, when the song identifier id is found in the list $\mathcal{C}_j$ such that $d(\mathcal{L}_j, l_i)$ is minimum, otherwise it is increased with B (i.e. the maximum possible distance). The song whose counter has the lowest value is selected as recognized. The song with the smallest index will be picked up, if more than one obtains the same score.

3.3 Discussion about the parameters

The behaviour of the algorithm and, accordingly, the performance in terms of accuracy is affected by the value assigned to each parameter. User can tune the values in order to meet the requirements of the needed task. Specifically, the characteristic parameters in the algorithm are:

- L_σ - The time length of the audio chunk used to evaluate a single linekey; the value should be short enough in order to include only a stationary audio signal (tipically not greater then 150 ms): small values permits to increase granularity but, on the other hand, it will increase the size of the dataset;
- M - The resolution of the FFT used to evaluate each periodogram; this value should be tuned taking into account both the sampling rate of the audio signal f_s and the number of bits B which will form the linekey; anyway, it has to be greater then $16 \cdot B$ in order to have a enough values for obtaining at least a bin inside each filter of the Mel filter bank; accordingly to the FFT algorithm used, it should be a power of 2; it should be chosen in order to be greater than or equal to the number of samples inside L_s audio time: i.e., greater or equal to $N = f_s \cdot L_s$.
- B - The number of bits making up a linekey; this value should be multiple of 8 (one byte) and it corresponds to the number of Mel-filters used in the Mel-filter bank; it affects the frequency resolution of the linekey: greater values permit to obtain higher resolution and, accordingly, more spreading of the space of linekeys; a wider linekey space permits to increase the recognition accuracy using the same number of linekeys in fingerprints but it generate a larger dataset and increases the execution time (both in terms of linekey generation time and linekey searching time).
- K - The number of periodograms used in the Welch's approach in order to perform the power spectral density estimation of the audio signal; accordingly to the theory, its value should be chosen in order to obtain a right compromise between the variance of the estimation and the frequency resolution.
- τ - The time step used to extract each linekey from the audio signal; its value heavily affects the number of linekeys extracted from audio signal; it can be smaller, equal or greater than L_σ: in the first case, overlapped linekeys are generated, i.e. greater resolution and granularity but larger dataset size and longer search time; in the second case, adjoining linekeys are generated, i.e., we use the minimum value in order to not lose any portion of audio signal; in the latter case, linekeys are generated with time jumps in between, i.e. with not covered pieces of audio; the case should be avoided because

both the granularity and the accuracy are negatively affected, even if both dataset size and search time are reduced.

- m (margin value) - is used for tuning the effect of the magnitude of each frequency component during the linekey generation: when this value increases, the gap between the magnitude of the frequency component and the frequency variant fitting curve must be higher in order to change the corresponding bit of the linekey from the value 0 to the value 1; in this the robustness of the linekeys improves but the number of different generated linekey is reduced and, accordingly, the recognition accuracy using few linekeys will result lower;
- N_F - the size of fingerprints in terms of linekeys: greater values mean more audio material is used to build a fingerprint and, accordingly, the accuracy will be positively affected; on the other hand, the time necessary to be able to carry out the recognition will proportionally increase and will not be possible to recognize audio excerpts of shorter duration than the fingerprint.

The reader who wants to deepen the proposed methodology can refer to [20].

4 Results evaluation

4.1 Songs corpus and queries material

In order to evaluate the performance of the different approaches, we used both a subset of the MTG-Jamendo dataset [6] and a specific corpus consisting of $N_s = 7000$ commercial songs. The first one is an open dataset for music auto-tagging. It is built using music available at Jamendo under Creative Commons licenses grouped in 100 folders labeled from 00 to 99 (we used only the folder 00 which contains 586 songs because of the very long time spent by the training phase of the *Audfprint*, *Dejavu*, *Olaf* and *Panako* implementation as provided by the authors). The second one is a subset of the larger one used for its business by the company involved in the project. It was carefully built in order to contain a broad range of different musical genres which belong to contemporary pop music currently played and broadcasted in Italy.[4] To evaluate the impact of corpus size in terms of number of songs, we split this corpus in 7 nested subsets from 1000 to 7000 with a step of 1000. We used each subset of the corpus to perform separated recognition experiments as explained later.

The tests were performed extracting four different excerpts of audio from each song of the datasets. The starting point of each excerpt was randomly selected in the range $[0\ s, t_{max} - 4\ s]$ where t_{max} is the time length of the song. We extracted excerpts of 1, 2, 3 and 4 seconds starting from the same initial point.

4.2 Performance evaluation

The following terms are used in defining our performance measures: t_p (true positives) is the number of cases in which the correct reference is identified from the query. Recognition Accuracy is the proportion of queries whose reference is correctly identified and it is defiend as

$$Accuracy = \frac{t_p}{N}$$

[4]The list of titles of songs in the corpus is available at http://perf.unime.it/?page_id=156

Table 1 Modified parameters in baseline algorithms

Algorithms	Parameter	Value
A1/A2	min-count	0
A2	density	10
O/P	MIN_HITS_UNFILTERED	1
O/P	MIN_HITS_FILTERED	1
O/P	MIN_SEC_WITH_MATCH	0
O/P	MIN_MATCH_DURATION	0

where N is the total number of queries. When true positive queries are performed, we evaluate the error e of the estimated position of the excerpts inside the song as

$$e = p_r - p_e$$

where p_r is the right position and p_e is the estimated one. Therefore, we evaluate the root-mean-square error (RMSE) of the time-positioning as

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N_{t_p}} e_n^2}{N_{t_p}}}$$

where e_n is the error on the n^{th} query with true positive result and N_{t_p} is the total number of queries with true positive result.

For comparison purposes, on the MTG-Jamendo subset 00, we used as baselines: 1) *Audfprint* [11] (using two different parameter settings, denoted as *A1* and *A2* from here on, where in this latter we modified the "density" parameter in order to obtain the same number of linekeys per second as in our approach); 2) *Dejavu* [10] (labelled as *D*); 3) the algorithms implemented in the last released version of Panako [24] (i.e., *Olaf* labelled as *O* and *Panako*, labelled as *P*). We kept all the parameters of baselines algorithms at the implementation defaults except those specified in Table 1 in such a way the reject option is turned off. Table 2 summarizes the values used for the parameters of our approach, denoted as *ME1* and *ME2*.

Table 3 shows results in terms of accuracy obtained on the MTG-Jamendo subset. As expected, performances increase with the size of the excerpts for each analysed approach. Both *Panako* and *Olaf* were unable to identify queries with 1-second-long samples, and, both approaches obtain very poor results also for excerpts with higher time-lengths. *Audfprint* obtains higher accuracy compared to previous approaches and it is affected by the "density" parameter in particular for short excerpts. Accuracy obtained with *Dejavu* are higher then 90% in all excerpts size condition. Anyway, both variants of our proposed approach outperforms all the others. In particular, for queries with 1-second-long samples, the accuracy of our proposed approaches is almost 8% higher than that obtained with *Dejavu*, which is the best of all the baselines.

Table 2 Set of parameters used in the experimentation

$f_s(Hz)$	$L_\sigma(ms)$	K	$\tau(ms)$
44100	100	16	100
M	B	m	N_F
8192	64	$3\,dB$	50

Table 3 Comparisons of accuracy performance (%) obtained on the MTJ-Jamendo-00 dataset for different excerpts lengths

	1s	2s	3s	4s
A1	36.1775	75.4693	87.9266	92.7048
A2	25.3413	61.8601	79.4369	87.9266
D	91.6667	97.7778	98.6325	99.1026
O	0.0000	67.4915	75.0853	77.6877
P	0.0000	0.0000	5.0768	68.3447
ME1	99.2747	99.3174	99.4454	99.4454
ME2	**99.6160**	**99.6587**	**99.7014**	**99.7440**

(best performances in bold)

Results in terms of time-positioning are showed in Table 4. The proposed approach permits to obtain best result for excerpts of 1s while *Olaf* and *Panako* give better result for longer excerpts. Anyway, the poor performance in terms of accuracy of *Panako* and *Olaf* algorithms should be considered to better compare these results. The proposed approach shows a remarkable improvements compared to *Dejavu* and *Audfprint* algorithms which have similar results in terms of accuracy.

Figure 2 shows the experimental Cumulative Density Functions of the time-positioning errors obtained with the considered algorithms for all the excerpts lengths. Each curve in the graphs represents the probability that the error in time-positioning is lower than the time value at the corresponding abscissa. Accordingly, at the same abscissa, an higher value corresponds to better performance. Specifically, Fig. 2-(a) shows the CDFs of time-positioning errors for excerpts lasting 1s: it contains only the curves related to *Audfprint*, *Dejavu* and the proposed approach because *Panako* and *Olaf* are not able to recognize so small excerpts; Fig. 2-(b) shows the CDFs of time-positioning errors for excerpts lasting 2s: in this case, only the curve related to *Panako* is missing for the same reason as in the previous subfigure; Fig. 2-(c) shows the CDFs of time-positioning errors for excerpts lasting 3s and Fig. 2-(d) shows the CDFs of time-positioning errors for excerpts lasting 4s. The trend of these curves permits to detect important information about the time-positioning errors. Of course, these results should be analyzed taking into account the performance in terms of accuracy of each algorithm. Analyzing the curves of the proposed approach is evident that the time-positioning error is lower than 0.1 seconds in 90% of the queries with true positives results. Results of *Olaf* and *Panako* approaches appear always slightly better than the proposed one

Table 4 Comparisons of time-positioning RMSE (s) obtained on the MTJ-Jamendo-00 dataset for different excerpts lengths

	1s	2s	3s	4s
A1/A2	84.7467	72.1993	74.6221	74.5507
D	49.1302	41.589	39.7872	39.7555
O	n.a.	**11.5338**	11.6709	**9.14351**
P	n.a.	n.a.	**7.96764**	14.2642
ME1/ME2	**24.229**	18.7178	18.0896	16.3677

(best performances in bold)

but the performances in terms of accuracy of these approaches are significantly worse. The proposed approach behaves always better than *Audfrpint* and *Dejavu* which have almost the same performance in terms of accuracy.

We then run an experiment on the larger commercial song corpus using as baselines *A1*, *A2* and the algorithm currently used by the company for its business. We didn't consider both *P* and *O* due to their very poor accuracy obtained in the previous experiment, and *D* for the very long time spent in training by its open source implementation. The company algorithm has been provided as a "black box": the number of linekeys generated per second is equal to 100 and the size of each linekey is equal to 64 $bits$. Two different parameter settings were used labelled as *C1* and *C2*.

Results about this second dataset are shown in Fig. 3, where the recognition ratio is depicted by varying N_s, given different excerpt sizes. They show that the analyzed approaches are not particularly affected by the corpus size making them robust with respect to the number of songs in the database. Instead, our approach outperforms both *Audfprint* and the company one for excerpts of length less than or equal to 2 seconds. Figure 3(b) shows that *A2* obtains an accuracy not greater than 90% while both our approaches, *ME1* and *ME2*, and the second company approach (*C2*) obtains an accuracy slightly lower than 100%. The difference in performance is even more pronounced using excerpts of 1s (Fig. 3(a)): in this condition, accuracy of both *Audfprint* approaches dramatically falls down under 50%,

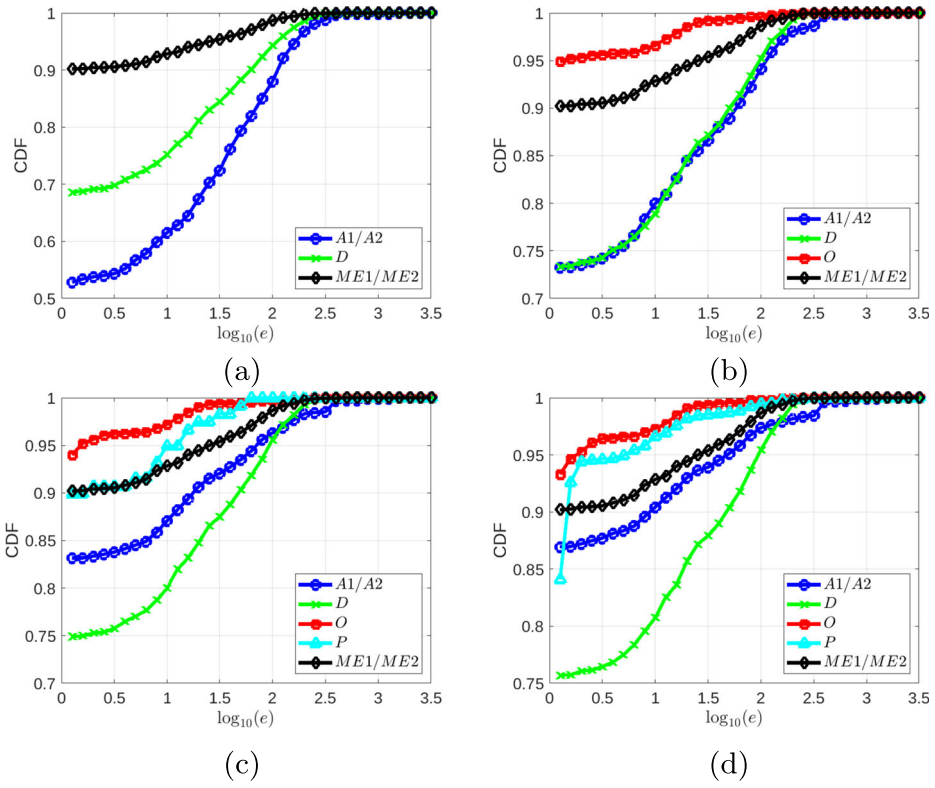

Fig. 2 Cumulative Density Functions (CDFs) of time-positioning errors for excerpts of (a) 1s, (b) 2s, (c) 3s and (d) 4s

both company approaches obtain an accuracy slightly greater than 60%, while both *ME1* and *ME2* maintain the recognition ratio almost near to 100%. In order to better show the results of our approaches, we depicted the details of accuracy results in Fig. 4. We obtain a recognition ratio higher than 98.8% into all the experiments. Specifically, *ME2* outperforms *ME1*, and it permits to obtain accuracy higher than 99.2% for all the corpus sizes also with a granularity of 1s. Using shorter excerpts, the accuracy performance degrades using both our approaches. Anyway, this degradation is always lower than 0.3%.

To justify the good performance of our approach in terms of accuracy, we investigated the characteristics of the generated linekeys. We evaluated the number of unique linekeys obtained increasing the size of the corpus. The results are shown in Fig. 5(a); as can be seen, the number of unique linekeys (N_l) grows linearly with the corpus size (N_s). This means that a linekey appears peculiarly associated with one song, or few at most. Accordingly, only few linekeys are able to correctly classify a song. For comparison purposes, we show in Fig. 5(b) the number of unique hashes obtained by *Audfprint A1* approach on the same corpora: it is considerably less and, primarily, it saturates to about 10^6 when the number of songs increases.

Finally, Fig. 6 shows results in noise condition. We added white noise to the audio queries in order to obtain SNRs equal to 20, 15, 10, and 5dB. Figure 6(a) shows results using excerpts of 4s: our approaches and the company ones outperform Landmark-based approach

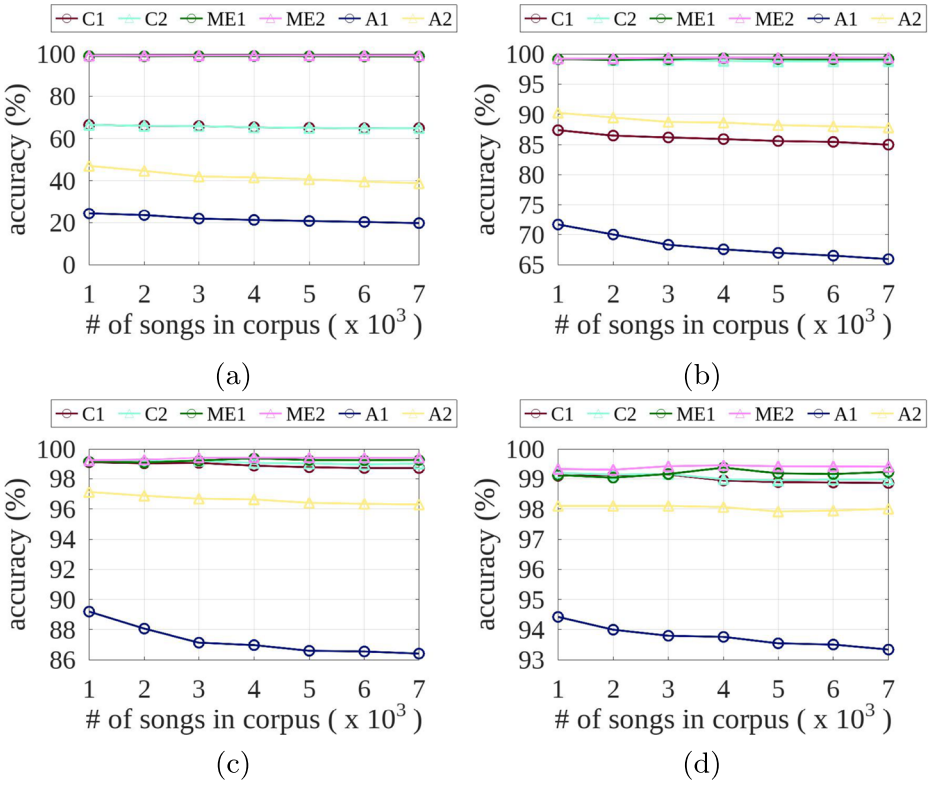

Fig. 3 Accuracy varying number of songs using the company corpus and excerpts of (a) 1s, (b) 2s, (c) 3s and (d) 4s

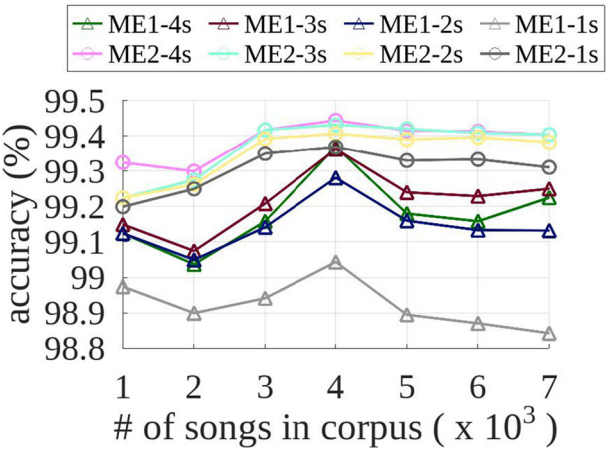

Fig. 4 Accuracy level of ME approaches vs number of songs in the company corpus with different excerpt sizes

for SNR higher than 5dB; a slight performance degradation of *C1* and *ME1* approaches is observable at 5dB, anyway *ME2* and *C2* approaches permit to obtain the best results also at 5dB. Figure 6(b) shows result of accuracy in the same previous noise conditions using the proposed approaches and excerpts of 1, 2, 3 and 4 seconds. For SNRs higher than 10dB, the excerpts size only slightly affects the accuracy (higher than 92.5%). Instead, as expected, we obtained higher degradation for smaller excerpts (Fig. 2).

5 Conclusions and future work

In this paper, we compared the behaviour of a new approach for generating song fingerprints with other well known algorithms when short excerpts of audio are considered.

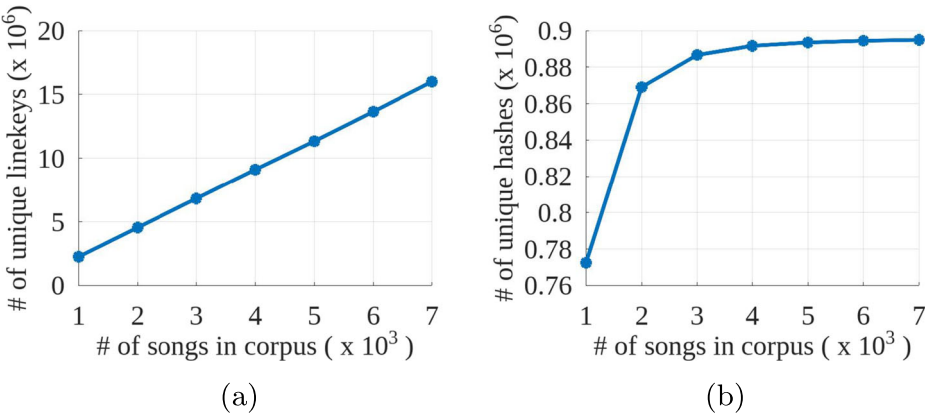

Fig. 5 (a) Number of unique linekeys obtained with proposed approach and (b) number of unique hashes obtained with *Audfprint A1* varying the size of the company corpus

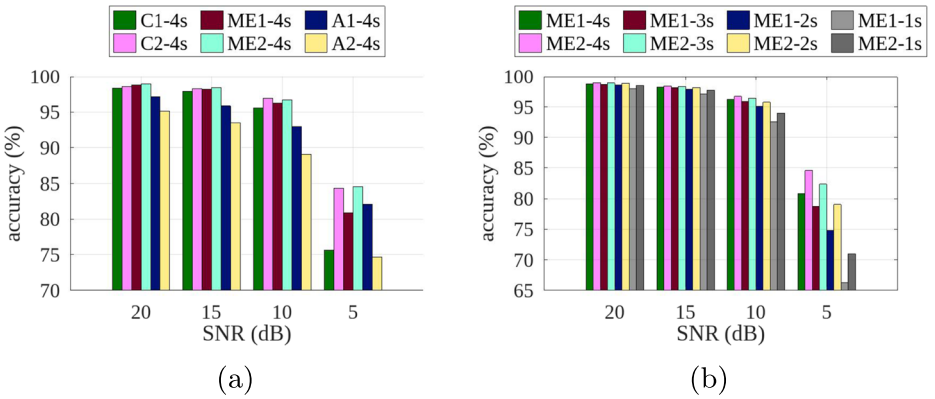

Fig. 6 Accuracy performance comparisons adding white noise to the queries at different SNRs: (a) the proposed approach vs *Audfprint* and *company* approach with excerpts of 4s; (b) the proposed approach with excerpts shorter than 4s

Comparisons with the considered baseline algorithms performed on the MTG-Jamendo subset showed our approach outperforms others and it maintains an accuracy almost equal to 100% varying excerpts lengths (1s, 2s, 3s and 4s). We also investigated the error in time-positioning of the small excerpts inside original track showing our approach exhibits an error lower than 0.1s in 90% of right classifications. We also extended our experiments using a large dataset of 7000 commercial songs highlighting that our proposed approach is not affected by the size of the dataset. Moreover, we added white noise to the audio queries at several SNRs and we showed that accuracy is higher than 98% in clean conditions and remains higher than 90% when SNRs $\geq$ 10dB for all excerpt sizes.

Our future research will focus on the complexity of the classification algorithm and its improvement by means of fast search in Hamming space. Moreover, we also want to investigate the performance of the proposed approach taking into account larger datasets of songs (with more than one hundred thousands songs) and pitch/tempo variations of the excerpts to be recognized.

Data Availability Two datasets have been used for deriving the results presented in Section 4: The Jamendo dataset and data provided by the company collaborating with the authors on the research here presented. The Jamendo data are available in the "MTG-Jamendo Dataset" with the identifier https://doi.org/10.5281/zenodo.3826813 [6]. The second dataset analyzed during the current study is available from an European based company but restrictions apply to the availability of these data, which were used under license from the company, and so are not publicly available. Data are however available from the authors upon reasonable request and permission of the company. The list of sources where the dataset has been created from is available at http://perf.unime.it/?page_id=156. Due to confidentiality agreement, the company likes to be kept anonymous in this publication.

Funding The authors did not receive support from any organization for the submitted work.

References

1. Anguera X, Garzon A, Adamek T (2012) Mask: robust local features for audio fingerprinting. In: 2012 IEEE international conference on multimedia and expo, pp 455–460. IEEE
2. Báez-Suárez A, Shah N, Nolazco-Flores JA, Huang S-HS, Gnawali O, Shi W (2020) SAMAF: sequence-to-sequence autoencoder model for audio fingerprinting. ACM Transactions on Multimedia Computing Communications, and Applications (TOMM) 16(2):1–23

3. Baluja S, Covell M (2007) Audio fingerprinting: Combining computer vision & data stream processing. In: 2007 IEEE international conference on acoustics, speech and signal processing-ICASSP'07, vol 2, p 213. IEEE
4. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
5. Bin Sahbudin MA, Scarpa M, Serrano S (2019) MongoDB clustering using K-means for real-time song recognition. In: 2019 Workshop on computing, networking and communications (CNC), pp 350–354. IEEE
6. Bogdanov D, Won M, Tovstogan P, Porter A, Serra X (2019) The MTG-Jamendo Dataset for Automatic Music Tagging. In: Machine learning for music discovery workshop, international conference on machine learning (ICML 2019), http://hdl.handle.net/10230/42015
7. Cano P, Batlle E, Kalker T, Haitsma J (2005) A review of audio fingerprinting. Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology 41(3):271–284
8. Chaouch C, Sahbudin MAB, Scarpa M, Serrano S (2020) Audio fingerprint database structure using k-modes clustering. Journal of Advanced Research in Dynamical and Control Systems 12(4 Special Issue):1545–1554
9. Cheng Z, Shen J (2016) On effective location-aware music recommendation. ACM Transactions on Information Systems (TOIS) 34(2):1–32
10. Drevo W (2013) Audio fingerprinting with python and numpy. https://willdrevo.com/fingerprinting-and-audio-recognition-with-python/. Accessed 20 Apr 2022
11. Ellis D (2014) The 2014 LabROSA audio fingerprint system. In: ISMIR
12. Fu Z, Lu G, Ting KM, Zhang D (2010) A survey of audio-based music classification and annotation. IEEE Trans Multimedia 13(2):303–319
13. Global Music Report 2021 (2021) https://gmr.ifpi.org/. Accessed 18 Feb 2022
14. Grosse R, Raina R, Kwong H, Ng AY (2012) Shift-invariance sparse coding for audio classification arXiv:1206.5241
15. Haitsma J, Kalker T (2002) A highly robust audio fingerprinting system. In: ISMIR, vol 2002, pp 107–115
16. Hamel P, Lemieux S, Bengio Y, Eck D (2011) Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In: ISMIR, pp 729–734
17. Jia M, Li T, Wang J (2020) Audio fingerprint extraction based on locally linear embedding for audio retrieval system. Electronics 9(9):1483
18. Peng Y, Qi J (2019) Cm-gans: cross-modal generative adversarial networks for common representation learning. ACM Transactions on Multimedia Computing Communications, and Applications (TOMM) 15(1):1–24
19. Sahbudin MAB, Chaouch C, Scarpa M, Serrano S (2020) Audio fingerprint based on power spectral density and hamming distance measure. Journal of Advanced Research in Dynamical and Control Systems 12(4 Special Issue):1533–1544
20. Serrano S, Sahbudin MAB, Chaouch C, Scarpa M (2022) A new fingerprint definition for effective song recognition. Pattern Recognition Letters. https://doi.org/10.1016/j.patrec.2022.06.009
21. Shen J, Shepherd J, Cui B, Tan K-L (2009) A novel framework for efficient automated singer identification in large music databases. ACM Transactions on Information Systems (TOIS) 27(3):1–31
22. Shen J, Shepherd J, Ngu AH (2006) Towards effective content-based music retrieval with multiple acoustic feature combination. IEEE Trans Multimedia 8(6):1179–1189
23. Shibuya T, Abe M, Nishiguchi M (2013) Audio fingerprinting robust against reverberation and noise based on quantification of sinusoidality. In: 2013 IEEE international conference on multimedia and expo (ICME), pp 1–6. IEEE
24. Six J, Leman M (2014) Panako: a scalable acoustic fingerprinting system handling time-scale and pitch modification. In: 15th International society for music information retrieval conference (ISMIR-2014)
25. Son H-S, Byun S-W, Lee S-P (2020) A robust audio fingerprinting using a new hashing method. IEEE Access 8:172343–172351
26. Song Y, Zhang C (2007) Content-based information fusion for semi-supervised music genre classification. IEEE Trans Multimedia 10(1):145–152
27. Sonnleitner R, Widmer G (2015) Robust quad-based audio fingerprinting. IEEE/ACM Transactions on Audio Speech, and Language Processing 24(3):409–421
28. Stevens SS, Volkmann J, Newman EB (1937) A scale for the measurement of the psychological magnitude pitch. J Acoust Soc Am 8(3):185–190
29. Wang AL-C (2003) An industrial strength audio search algorithm. In: ISMIR, vol 2003, pp 7–13. Washington, DC

30. Wang AL-C, Smith III JO (2018) Systems and methods for recognizing sound and music signals in high noise and distortion. Google Patents. USPatent 9,899,030
31. Weisstein EW (2022) Least Squares Fitting–Exponential From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/LeastSquaresFittingExponential.html. Accessed 20 Apr 2022
32. Williams D, Pooransingh A, Saitoo J (2017) Efficient music identification using orb descriptors of the spectrogram image. EURASIP Journal on Audio Speech, and Music Processing 2017(1):1–17
33. Yao S, Niu B, Liu J (2017) Audio identification by sampling sub-fingerprints and counting matches. IEEE Trans Multimedia 19(9):1984–1995
34. Yao S, Wang Y, Niu B (2015) An efficient cascaded filtering retrieval method for big audio data. IEEE Trans Multimedia 17(9):1450–1459