Query by Humming for Song Identification Using Voice Isolation

Edwin Alfaro-Paredes, Leonardo Alfaro-Carrasco, and Willy Ugarte(✉)

Universidad Peruana de Ciencias Aplicadas, Lima, Peru
{u201611810,u201212551}@upc.edu.pe, willy.ugarte@upc.pe

Abstract. There are some methods for searching in large music databases, like searching by song name or artist name. However, in some cases these methods are not enough. For instance, a person might not remember the name of a song, but might remember its melody. In Music Information Retrieval, there is a task called Query-By-Humming, which allows retrieving a rank of songs that are similar to an audio humming. In this research, we propose the use of vocal isolation methods to improve query-by-humming systems. To achieve this, different configurations of Query-by-Humming systems were tested to analyze the results and determine in which cases our proposal works better. The results showed that vocal isolation improves the performance of Query-by-Humming systems when the music collection consists of modern songs.

Keywords: Query-by-Humming · Music similarity · Melody extraction · Music information retrieval

1 Introduction

In today's world, there is an ever-growing content of music available to the public and an equally growing access to information. An example of this is the number of songs that are indexed in digital databases like Spotify, where 40,000 new songs are added each day[1]. Although parts of this information are structured and organized, other parts are difficult to find or retrieve. Due to the latter, there is a need for searching methods in large amounts of music data. For example, one way of searching for a song is by its name or artist. However, this isn't helpful if the user doesn't know or doesn't remember the name of a song or any identifying information other than the song's melody. For this reason, the investigation area of Music Information Retrieval has an approach called Query-By-Humming/Singing, where new methods for music recognition, using the user's hum or voice, are studied.

A Query-By-Humming System is composed by two main parts, as described by [20] in their proposed diagram. First, a descriptor extractor, which processes the songs and returns a descriptor of each song. This is an important task,

[1] Music Business Worldwide - https://bit.ly/37TQNE4.

because the database is constructed with the extracted descriptors. Also, the user's hum or voice needs to be processed to extract a descriptor that represents similar information of the hummed song as the one in the system's database. The second part consists of a method to measure the similarity between a song and the user's interpretation (e.g., correlation). The ranking results use these scores to retrieve the songs in similarity order. Therefore, the user can recognize the song only by their interpretation.

Recognizing a song by a user's hum or voice is a difficult task, because the user does not necessarily have good singing/humming abilities, as a research affirms that 62% of non musicians were poor singers [9]. This might lead to significant differences between the user's interpretation and the original song [10]. Furthermore, the process of searching in a large database is likely a time consuming task, thus the solution mitigates all the mentioned problems.

Our main motivation for this research project is the relatively small amount of proposals in the state of art related to the direct analysis of frequencies in an audio file. Starting from this, our contributions are as follows:

- We want to demonstrate that vocal isolation could improve results for Query-By-Humming systems, because the user is more likely to hum the voice part from a given song.
- To obtain the best performance on our Query-By-Humming system, we have made an exhaustive comparison between different configurations, which includes descriptors and alignment methods.
- Two different datasets were used to find in which cases our contribution could improve the performance of Query-By-Humming systems.

This paper is organized as follows. In Sect. 2, we define the main concepts of Query-By-Humming systems. In Sect. 3, we present our contribution. Section 4 contains several works that we have reviewed. Our experiments and results are shown in Sect. 5. Finally, in Sect. 6, we analyze our results and give some recomendations for future works.

2 Background

In this section, we define all those concepts and methods that are related with the Query-By-Humming task. These concepts are important to understand this research and how these systems are constructed using different approaches.

2.1 Cover Song Identification

This task consists of identifying all the possible covers from a database given a song [21]. Similarly to Query-By-Humming, this approach is composed of two main parts, descriptor extraction and similarity computation. Usually, for this task, the descriptor is constructed from melody and harmony, like this research project [20]. There are some clear similarities between Cover Song Identification and Query-By-Humming. For this reason, the latter task can be considered as a general case from the other, where hummings and songs could be related to the concept of cover [20]. The main difference is noted in the descriptor extraction process, where only the melody is used, because the user's hum is a monophonic audio and represents the main melody of a song [20].

2.2 Melody Extraction Process

The purpose of this process is the automatic extraction of the main melody from a polyphonic audio, and it's worth noting its importance in many music information retrieval tasks [19]. In Query-By-Humming systems, the output of this process is used to compute the descriptor of a song or hum.

There are mainly two kinds of melody extraction methods in the literature:

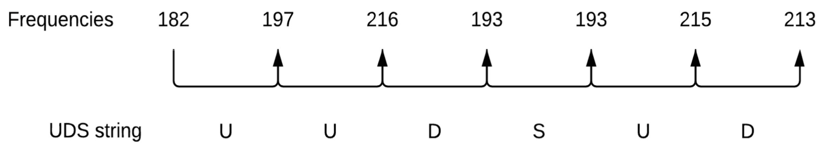
- First, the database is created from midi files, where the melody is extracted based on channel elimination like [15], or based on a main salience function that constructs a route of the melody between all the available channels [8].
- Second, the database is created from wav files, where frequency analysis is applied to recognize the main melody [18,19,27].

2.3 Descriptor Computation Process

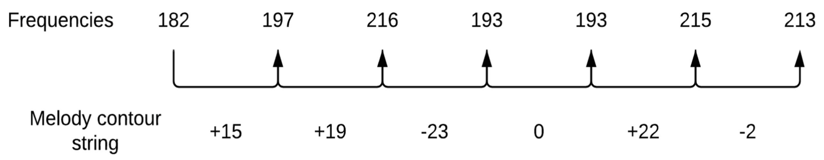
As mentioned earlier, to construct a Query-By-Humming database, it's necessary to compute the descriptors from the songs and hummings. These descriptors should have some important properties: compactness, expressiveness and portability [6]. Thus, the descriptors can be stored efficiently, they can represent the most outstanding melody expressions and the system can be more resistant to different types of inputs (key and tempo variation) [6].

The most basic form to represent the melody is with the UDS descriptor (see Fig. 1a), which uses the letter "U" to represent an increase between two consecutive notes, the letter "D" to represent a decrease between two consecutive notes, and the letter "S" to represent no change between two consecutive notes [4]. Another descriptor is the melody contour string (see Fig. 1b), which uses the pitch difference between two consecutive notes [11]. There are some descriptors that are usually used in Cover Song Identification task, like chromagrams, which represent the frequencies from an audio in twelve different pitch classes. Some of these chromagrams can be used in Query-By-Humming [20].

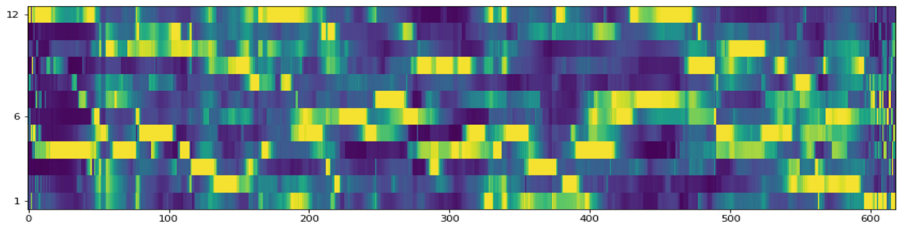
The Harmonic Pitch Class Profile (HPCP) (see Fig. 1c) is a type of chromagram used to represent the chords of an audio. We use HPCP in this research for our experiments and test a method for Cover Song Identification in Query-By-Humming. Another chromagram we used is the proposed by [20], a Semitone-band Based Chromagram (henceforth denoted as SBBC) (see Fig. 1d). To obtain this descriptor, the cents and semitones are computed from the fundamental frequency of the melody. Then, the results are mapped into a single octave. Finally, a pitch class histogram is computed and a normalization is applied.



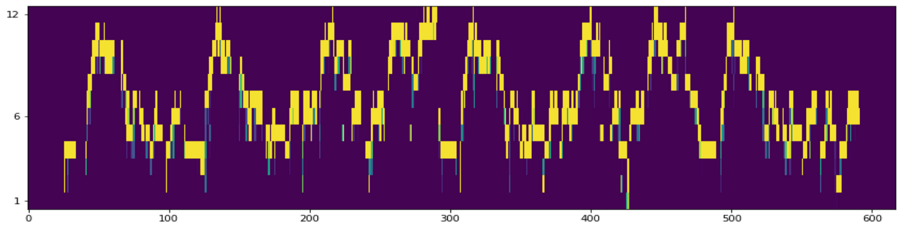
(a) UDS Descriptor



(b) Melody Contour Descriptor



(c) HPCP Descriptor

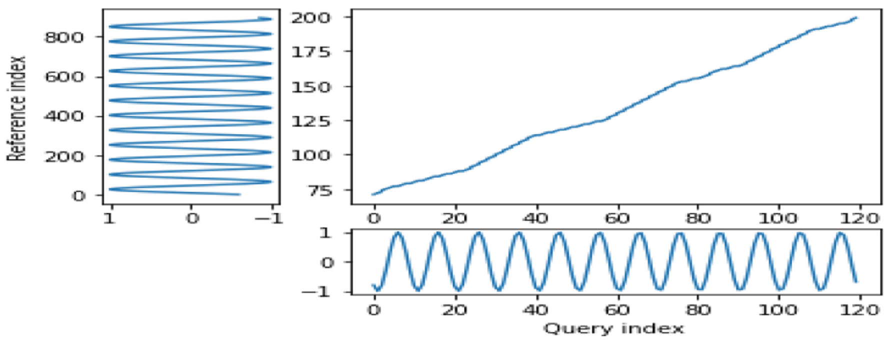


(d) SBBC Descriptor based on [20]

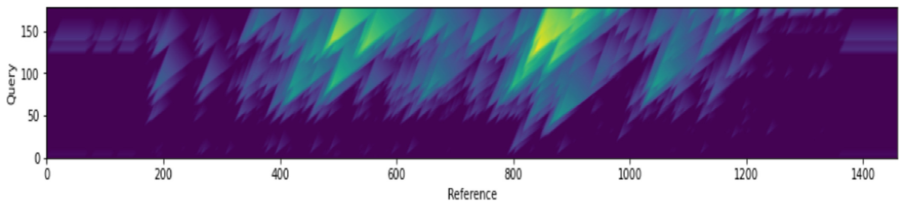
Fig. 1. Melody descriptors

2.4 Alignment Methods

These methods allow us to compute a similarity score between a query (humming) and a reference (songs from database). Thus, it is possible to obtain a similarity ranking, ordered from the most similar song to the least one.



(a) DTW Alignment (dtw-python package)



(b) Qmax Alignment (Essentia Package)

Fig. 2. Alignment methods

Now, we are going to define two methods that have shown a good performance in many music information retrieval tasks:

- First, Dynamic Time Warping (DTW) (see Fig. 2a), defined from [23]: *"this algorithm allows finding the optimal scaling of the time axis of the compared sequences by minimizing the cost of matching one of them to the other"*.
- Second, Qmax algorithm (see Fig. 2b), proposed by [22], which computes a similarity score between two time series, based on local alignment [20].

This method receives a similarity matrix as an input, which is computed from the query and the reference into their form of descriptor (chroma). For this research, we used both DTW and the Qmax algorithm in order to test different configurations and find out which one gives better results.

3 Related Works

To the best of our knowledge, the earliest work on Query-by-Humming systems is [4], where a simple but robust UDS descriptor is used and a fuzzy matching algorithm is applied to account for errors in the user input when comparing it to a database of MIDI song descriptors. Another work is [13] in which a more expressive descriptor is created with strings indicating a higher or lower note than the previous one, a pitch difference and note duration.

A different approach is explored in [3] where n-grams, fixed-length overlapping sub-sequences, are used as descriptors, while in [17] neural networks are trained and used to determine the best starting and ending positions of the stored descriptors to be used as parameters for the dynamic time warping algorithm. In [20], a pitch histogram, inspired on chromagrams, is proposed as a melody descriptor and the Qmax algorithm is used for the matching process.

In [11,25], the authors propose the use of additional information to complement the results obtained by a Query-by-Humming system. In the first one, a preliminary ranking is constructed as a first step of the system by a melody matching algorithm, then it is reordered by giving priority to results that belong to genres that the user has previously searched. In the second one, a Query-by-Humming/Singing system is proposed, where an additional comparison is done between any lyrics obtained from the user (singing) query and the lyrics.

Finally, other works have taken an approach to tackle the high processing time problem that a Query-by-Humming system inevitably faces with a big database of song descriptors. In [5], the song descriptors are repeatedly hashed and assign to "buckets", then the user input is subsequently hashed and only compared with the songs that belong to the same bucket that the query falls into, effectively reducing the number of matching candidates and thus search time. In [26], a Piecewise Aggregate Approximation transformation is applied to create a much shorter and simpler pitch sequence of the user query that is then used to filter out descriptors that are less likely to match.

All these works have focused on common steps of Query-by-Humming system but, to the best of our knowledge, none of them have added any pre-processing steps to the data used to build the descriptors database. We noticed there is room for improvement that may significantly improve the results of Query-by-Humming systems by filtering the raw data that is then processed and used as a database to look for matches. Our proposal applies a voice extraction process to the music tracks which eases up the following descriptor extraction step by reducing the amount of noise that the algorithm has to detect and filter.

4 Main Contribution

Our main interest is to find out if separating the vocals and the accompaniment of each music track may improve the descriptor extraction process that is an inherent part of Query-by-Humming systems. This separation will be made as a preprocessing step, thus, instead of using directly songs to compare, using the separated vocals and the accompaniment of each song as an input.

Isolating the vocals from a music track should reduce the amount of noise a melody extraction algorithm must detect and remove, allowing for better descriptors and an overall higher precision in the final results. This isolation was done by pretrained machine learning models, Spleeter [7] and Demucs [2], to create derived datasets that contain only the vocals of the original songs. They were selected since they have the best results in the literature to our knowledge [2,7], and because are provided by serious companies like Deezer (specialized in music related products) and Facebook Research.

Furthermore, algorithms for both descriptor extraction and melody matching have a huge impact on the precision of the results. Thus, we compare and setup different combinations of these algorithms, which includes HPCP and SBBC for descriptor extraction process, and for the matching similarity task Qmax and Dynamic Time Warping. Additionally, musical collections that differ in their "oldness" (i.e., the average release year of their songs) might influence the behavior of the algorithms and its results. Thus, we compare and setup different experiments on two musical collections that differ on their "oldness".

Our test planification included experiments to verify how the different configurations for Query-by-Humming systems behave in a real situation with popular music. However, we noticed that the MTG-QBH collection[2] is composed by old songs. For this reason we had to create our own collection, which is composed by "modern" popular songs (i.e., on average, from the last 20 years). This collection also includes 24 humming queries of 16 songs.

5 Experiments

5.1 Experimental Protocol

Data:

– *Musical Collections:* To evaluate our method, we used two musical collections.
  - The first one is a subset of the MTG-QBH (See footnote 2) dataset, which consists of 481 canonical songs and 118 humming queries. Because our method works by extracting the vocals from an audio file, 78 tracks that contain no lyrics were excluded from the original collection for a total of 403 songs used for our experiments.
  - The second dataset was constructed for this research, and consists of 50 songs and 24 humming queries.

  Both datasets differ from each other in their songs' release dates: the first one contains music released between 1926 and 2004, with an average of 1972, while the second one consists of music from 1965 to 2018 with an average of 2004.

– *Dataset Partition:* Since the aforementioned datasets do not have the same amount of songs and hummings, and if we want to compare both collections, it is important to create samples of the first one to make it as similar as possible in size and distribution terms to the second one.

  For this reason, during our experiments, we grouped the queries from each dataset by their target songs and then randomly selected hummings from the first collection's groups to mirror the second query distribution. Thus, we made 24 queries (10 queries where each one targets a different song, 8 queries

[2] Music Technology Group - https://bit.ly/2IdElnG.

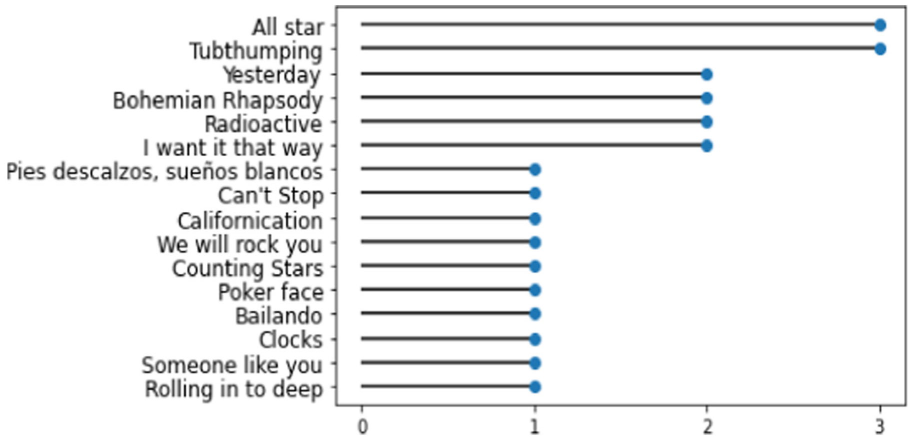


Fig. 3. Distribution of hummings from music collection 2

where each pair targets the same song, and 6 queries where each half has the same target song) for collection 1. Finally, 34 songs were randomly selected (in addition to the 16 songs targeted by the previous queries) (Fig. 3).

Equipment:

– The computer we used for the experiments has the followings characteristics: AMD Ryzen 7 3700X 8-Core Processor 3.60 GHz, 16 GB RAM 3200 MHz, and NVIDIA GeForce RTX 2060 SUPER.
– The programming language used was Python 3.7.
– The code and datasets are publicly available at https://bit.ly/3ngQh7o.

Experiment Configurations: For our experiments:

– We use two types of descriptors, HPCP from the Essentia package [1] and SBBC based on [20].
– For the alignment methods: we use the DTW and Qmax algorithms.
– For comparing full melody from songs against the vocal part from songs for both collections: we use vocal isolation techniques Spleeter [7] and Demucs [2] (two pre-trained models for separating vocals and accompaniments).
– For measuring the performance of these configurations, we use Mean Reciprocal Rank (MRR) (Eq. 1) and Top-X Hit Rate (Eq. 2) from [14].

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{r_i} \tag{1}$$

$$Top\text{-}X = \frac{|\{c \in Q \mid r_c \leq X\}|}{|Q|} \tag{2}$$

where r_i = result position of i-th query and Q = Query set

Each configuration used a unique combination of a dataset, a melody descriptor, and a matching algorithm and all configurations were run against three different versions of the musical collections: the full songs (vocals + accompaniment), vocal isolation processed by Spleeter, and processed by Demucs. The Query-by-Humming framework used by each configuration is as follows: For each song on the database and the query (user's humming), its main melody is extracted with Melodia algorithm, which is based on frequency analysis [19]. Then, a descriptor is calculated from each extracted melody. Finally, the query in its form of descriptor is compared with each song's descriptor using an alignment method, and the ranking is elaborated from the scores of each alignment.

Each experiment with the collection 1 was run five times, generating different samples, and the average was calculated.

5.2 Discussion

Quantitative Results: In Table 1 shows our results of experiments. For instance, 0.14 is the MRR for collection 2 using SBBC as descriptor, which is constructed with the melody of the voice part of a song extracted with Spleeter, and Qmax as the alignment method.

We could bring another example, 7.50% is the Top 5 for collection 1 using HPCP as descriptor, which is constructed with the melody of the full song, and DTW as the alignment method. Also, there are some numbers in bold, which means the best result for that configuration. For instance, 0.26 is the MRR obtained for the best configuration (SBBC as descriptor and Qmax as alignment method) for the first collection using the melody of the full song.

On one hand, in Table 1 that, for both music collections, the SBBC descriptor has generally the best performance for all metrics (most of the bold values are in columns 4, 5, 8, 9, 12, 13, 16 and 17). This descriptor obtains the best result for the combination of melody extracted from the full song (first line for collection 1) and Qmax alignment, obtaining an MRR of 0.26 (numbers in bold for column 5), a Top 1 of 15.83% (numbers in bold for column 9), a Top 5 of 35% (numbers in bold for column 13), and a Top 10 of 50% (numbers in bold for column 17).

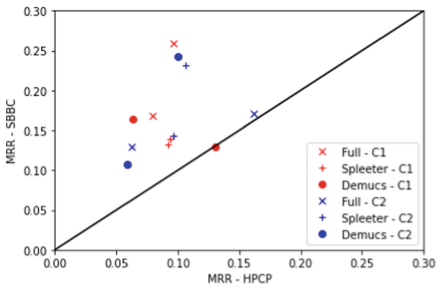
Table 1. Comparison of the descriptor extraction methods, alignment methods, and Collections 1 and 2 with MRR and Top-X hit rate metrics.

	MRR				Top 1				Top 5				Top 10			
	HPCP		SBBC		HPCP		SBBC		HPCP		SBBC		HPCP		SBBC	
	DTW	Qmax	DTW	Qmax	DTW	Qmax	DTW	Qmax	DTW	Qmax	DTW	Qmax	DTW	Qmax	DTW	Qmax
Collection 1																
Full	0.08	0.10	0.17	**0.26**	2.50	2.50	9.17	**15.83**	7.50	10.83	20.83	**35.00**	13.33	25.00	30.83	**50.00**
Spleeter	0.10	0.09	**0.14**	0.13	2.50	4.18	**6.67**	4.17	8.33	5.83	13.33	**17.50**	21.67	15.83	**27.50**	25.83
Demucs	0.13	0.06	0.13	**0.16**	5.00	0.83	5.00	**8.33**	15.83	5.00	14.17	**20.00**	33.33	8.30	24.17	**49.17**
Collection 2																
Full	0.16	0.06	**0.17**	0.13	**8.33**	0.00	**8.33**	4.17	16.67	4.17	**20.83**	12.50	**29.17**	12.50	20.83	25.00
Spleeter	0.11	0.10	**0.23**	0.14	0.00	4.17	**16.67**	8.33	20.83	12.50	**25.00**	16.67	**29.17**	16.67	**29.17**	20.83
Demucs	0.10	0.06	**0.24**	0.11	4.17	0.00	**16.67**	4.17	4.17	4.17	**25.00**	12.50	25.00	20.83	**37.50**	16.67

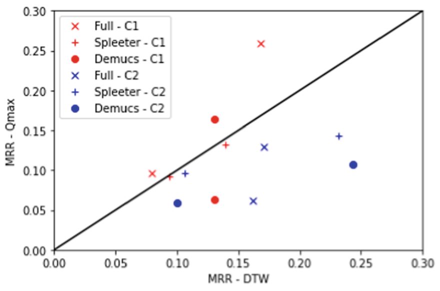
On the other hand, in Table 1, the DTW algorithm combined with the vocal isolation approaches obtain better results for the collection 2, being 0.24 its highest value for MRR (number in bold for Demucs in column 4), 16.67% for Top 1 (numbers in bold for Spleeter/Demucs in column 8), 25% for Top 5 (numbers in bold for Spleeter/Demucs in column 12), and 37.5% for Top 10 (number in bold for Demucs in column 16).

Qualitative Results: First, Fig. 4a shows that SBBC descriptor [20] mostly has a much better performance than the HPCP descriptor (most of the points are located in the upper triangle/SBBC side). HPCP was developed to store harmonic information, even if the melody is used to construct it. Thus, HPCP might not be suitable for the Query-By-Humming task. Contrarily SBBC descriptor is suitable for it because they are based on semitone and octave abstraction, which makes the descriptor more robust to octave changes and local expressions [20].

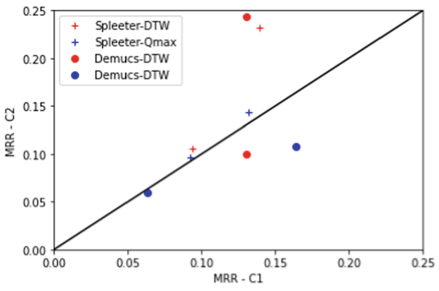
After, Fig. 4b shows that DTW algorithm has a better performance for the Collection 2 (i.e., modern songs). Contrarily, the Qmax algorithm is more suitable for the Collection 1 (i.e., older songs). All of (resp. some of) the blue (resp. red) points are displayed below (resp. above) and relatively far from (resp. close to) the diagonal line for Collection 2 (resp. Collection 1). This is similar to [20], where the authors have obtained good results for the same collection.



(a) SBBC vs. HPCP



(b) Qmax and DTW in both collections



(c) Vocal Isolation in both collections

Fig. 4. Results graphics

Finally, Fig. 4c shows that vocal isolation improves the performance mostly for collection 2 (most of the points are displayed above the diagonal line). This might indicate that there are some important differences between current music and old music that explain this behaviour. For instance, according to [16], there is a decrease in terms of loudness and rythm complexity, and an increase about timbre complexity.

6 Conclusion

In this paper, we have experimented with different configurations for Query-by-Humming systems, which includes two alignment algorithms (DTW and Qmax), two methods for constructing a song descriptor (HPCP and SBBC), and with (or without) an extra pre-processing step for vocal isolation.

Our results show that SBBC descriptor mostly performs much better than the HPCP descriptor, on one hand, the DTW algorithm performs better for modern songs (i.e., Collection 2), on the other hand, the Qmax algorithm is more suitable for older songs (i.e., Collection 1) and the vocal isolation mainly improves the performance only for collection 2.

Our results point to a clear distinction between Collection 1 (older songs) and Collection 2 (modern songs), nevertheless deeper studies need to be performed to confirm this insight. Furthermore, developing a recommender system that automatically adapts the retrieval parameters (e.g., descriptors, alignment metrics, ...) according to the type (or genre) of the hummed song to improve the quality of the results, seems a promising research prospect. These improvements to the Query-by-Humming task may be integrated into a song recognition system that could implement other solutions, e.g. simple tasks like Query-by-Text by asking the user to input genres or artists to reduce the search space, or reading music sheets [12]. Finally, softness may be a key element for fine tunning the parameters [24].

References

1. Bogdanov, D., et al.: Essentia: an audio analysis library for music information retrieval. In: ISMIR (2013)
2. Défossez, A., Usunier, N., Bottou, L., Bach, F.R.: Music source separation in the waveform domain. CoRR abs/1911.13254 (2019)
3. Doraisamy, S., Rüger, S.M.: Robust polyphonic music retrieval with n-grams. J. Intell. Inf. Syst. **21**(1), 53–70 (2003)
4. Ghias, A., Logan, J., Chamberlin, D., Smith, B.C.: Query by humming: musical information retrieval in an audio database. In: ACM Multimedia (1995)
5. Guo, Z., Wang, Q., Liu, G., Guo, J.: A query by humming system based on locality sensitive hashing indexes. Signal Process. **93**(8), 2229–2243 (2013)
6. Gómez, E., Klapuri, A., Meudic, B.: Melody description and extraction in the context of music content processing. J. New Music Res. **32**, 23–40 (2003)
7. Hennequin, R., Khlif, A., Voituret, F., Moussallam, M.: Spleeter: a fast and efficient music source separation tool with pre-trained models. J. Open Source Softw. **5**(50), 2154 (2020)

8. Huang, Y., Lai, S., Sandnes, F.E.: A repeating pattern based query-by-humming fuzzy system for polyphonic melody retrieval. Appl. Soft Comput. **33**, 197–206 (2015)
9. Hutchins, S., Peretz, I.: A frog in your throat or in your ear? searching for the causes of poor singing. J. Expe. Psychol. General **141**, 76 (2011)
10. Khan, N.A., Mushtaq, M.: Open issues on query by humming. In: ICADIWT (2011)
11. Liu, N.: Effective results ranking for mobile query by singing/humming using a hybrid recommendation mechanism. IEEE Trans. Multim. **16**(5), 1407–1420 (2014)
12. Lozano-Mejía, D.J., Vega-Uribe, E.P., Ugarte, W.: Content-based image classification for sheet music books recognition. In: 2020 IEEE Engineering International Research Conference (EIRCON) (2020)
13. Lu, L., You, H., Zhang, H.J.: A new approach to query by humming in music retrieval. In: IEEE ICME (2001)
14. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
15. Ozcan, G., Isikhan, C., Alpkocak, A.: Melody extraction on MIDI music files. In: IEEE ISM (2005)
16. Parmer, T., Ahn, Y.: Evolution of the informational complexity of contemporary western music. In: ISMIR (2019)
17. Pham, T.D., Nam, G.P., Shin, K.Y., Park, K.R.: A novel query-by-singing/humming method by estimating matching positions based on multi-layered perceptron. KSII Trans. Internet Inf. Syst. **7**(7), 1657–1670 (2013)
18. Rocamora, M., Cancela, P., Pardo, A.: Query by humming: automatically building the database from music recordings. Pattern Recognit. Lett. **36**, 272–280 (2014)
19. Salamon, J., Gómez, E.: Melody extraction from polyphonic music signals using pitch contour characteristics. IEEE Trans. Speech Audio Process. **20**(6), 1759–1770 (2012)
20. Salamon, J., Serrà, J., Gómez, E.: Tonal representations for music retrieval: from version identification to query-by-humming. Int. J. Multim. Inf. Retr. **2**(1), 45–58 (2013)
21. Serrà, J., Gómez, E., Herrera, P.: Audio cover song identification and similarity: Background, approaches, evaluation, and beyond. In: Raś, Z.W., Wieczorkowska, A.A. (eds.) Advances in Music Information Retrieval. SCI, vol. 274, pp. 307–332. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11674-2_14
22. Serrà, J., Serra, X., Andrzejak, R.: Cross recurrence quantification for cover song identification. New J. Phys. **11**, 09307 (2009)
23. Stasiak, B., Skiba, M., Niedzielski, A.: FlatDTW - dynamic time warping optimization for piecewise constant templates. Digit. Signal Process. **85**, 86–98 (2019)
24. Ugarte, W., Boizumault, P., Loudni, S., Crémilleux, B., Lepailleur, A.: Soft constraints for pattern mining. J. Intell. Inf. Syst. **44**(2), 193–221 (2013). https://doi.org/10.1007/s10844-013-0281-4
25. Wang, C., Jang, J.R.: Improving query-by-singing/humming by combining melody and lyric information. IEEE ACM Trans. Audio Speech Lang. Process. **23**(4), 798–806 (2015)
26. Wang, L., Huang, S., Hu, S., Liang, J., Xu, B.: Improving searching speed and accuracy of query by humming system based on three methods: feature fusion, candidates set reduction and multiple similarity measurement rescoring. In: INTERSPEECH. ISCA (2008)
27. Zhang, W., Chen, Z., Yin, F.: Main melody extraction from polyphonic music based on modified Euclidean algorithm. Appl. Acoust. **112**, 70–78 (2016)