Chorus Detection Using Music Structure Analysis

Zhengyu Cao[1], Yongwei Gao[1], and Wei Li[1,2(✉)]

[1] School of Computer Science, Fudan University, Shanghai 201203, China
{zycao18,ywgao16,weili-fudan}@fudan.edu.cn
[2] Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 201203, China

Abstract. This paper describes a novel chorus detection method based on extracting the functional structure of music from its self-similarity matrix. An existing similarity measure was enhanced firstly by using a key-shift invariant distance and by introducing a chroma-like pitch feature that exploits melody extraction results of the music. The repeated sections in the audio were extracted using a graph-based algorithm and clustering-merging method assuming transitivity of similarity then. Finally, a classifier to detect the chorus from the repeated sections was trained. The evaluation results show that our method is comparable with the state-of-the-art algorithms on both multiple and single chorus section detection tasks.

Keywords: Chorus detection · Music structure · Graph algorithm

1 Introduction

Music tend to be structured audio as described in [11], composed of repeating patterns/segments in hierarchies, from repeating phrases to sections. Among them, the longest repeating segments which correspond to sections or functional parts in the song are especially useful. For popular music, the basic song structure consists of an intro, verse, bridge, chorus and outro section. Chorus sections as the most representative parts of pop music are of special interest in many music-related applications, like auto music clipping, music thumbnailing, preview, retrieval and recommendation. For example, with the rapid growth of short-video services, the catchiest part of the music was preferred for making the videos. The service provider usually have large repositories of digital music clips which means clipping and choosing the chorus section manually is difficult, auto-clipping solves the problem.

2 Previous Work

To catch chorus sections, approaches based on music structure analysis were pervasively adopted, though there exist methods like [9] which directly estimate

X. Shao et al. (Eds.): CSMT 2020, LNEE 761, pp. 3–17, 2021.
https://doi.org/10.1007/978-981-16-1649-5_1

chorus sections from music audio. Self-similarity matrix (SSM) is the key component in many structure analysis algorithms [4,6,7,10,21,24].

One challenge faced in SSM based music structure analysis methods is how to extract meaningful segments from a raw SSM, which involves SSM enhancement and analysis. Various methods have been proposed to enhance the SSM, like matrix fusion technique from [2] used by [24], non-negative matrix factorisation (NMF) based methods [4,12] and methods [10] using augmentation of transposition and tempo invariance. As for extracting segments from the SSM, in [24], a spectral clustering method based on eigenvector decomposition of Laplacian matrix of the SSM was used to group the frames; in [4], a checkerboard kernel was applied to the SSM to generate a novelty curve, then peaks in the novelty curve were detected as segment boundaries; in [6] and [7], lines from the SSM diagonals were extracted first and merged using various hand-craft rules to form segments.

In [4], transitivity (which means if A and B are similar, and A and C are similar, then A and C should be similar) was enforced to the output music structure on the last step, the proposed method pushed it further: transitivity of similarity was considered at the first place, and this constraint was kept throughout the following steps. We proposed a novel graph-based algorithm to extract repeating segments from the SSM using a clustering-merging method. The clustering step can be seen as a repetition based method, comparing to the stripe detection approach used in previous repetition based methods like [7,14,18], the proposed method focus on detecting repeating patterns of smaller size but more repetitions, thus involve less effort integrating the repeating patterns and better reflects the repetition in music.

Melody extraction results were introduced to enhance the SSM in the proposed method, the reason is twofold. On the one hand, the feature-level similarity fusion in [24] draws good results in SSM enhancement while it supports arbitrarily many features as input so that new features could be added. On the other hand, the results of melody extraction algorithms has been greatly improved from salience-based approaches [22] to data-driven approaches like [1,3,13].

Heuristic methods were heavily used for chorus detection in previous works [6,7,17], focusing on distinguish chorus sections by repetition counts, durations and other features. Since the number of features could be large, and annotated datasets were available, supervised methods were preferred, in [25], a random forest classifier were used to detect chorus segments. We adopted the data-driven method, and combined melody extraction results into the features for chorus classifying.

Experiment on RWC Pop Database [8] shows our method is better comparing to the music thumbnailing algorithm in [9] on single section chorus detection, and has comparable performance with the best of 5 structure analysis algorithms mentioned in [19]. For reproducibility, the proposed algorithm and evaluation code is available on https://github.com/beantowel/chorus-from-music-structure.

3 Method

Figure 1 demonstrates the process of the proposed method. Firstly, acoustic features as pitch chroma, MFCC, chroma and tempogram were calculated from the input music recording. Then self-similarity matrices were generated on these features and fused into one. Low-level patterns were extracted by graph algorithms assuming transitivity of similarity and merged to form top-level structures. In the end, a classifier learns from the training data to detect chorus sections and makes predictions on structural information and melody features of the input sections.

Figure 2 shows the results from the pipeline. The fused SSM were plotted in the upper-left subfigure, the ground truth chorus sections and detected chorus sections were represented by the green stripes in the upper half and lower half of the box. The upper-right subfigure shows the ground truth structure annotations of the music, the green squares were verse sections and the blue squares were the chorus sections. The lower-right and lower-left subfigure shows the extracted low level and top-level structure of the music, different colors were used only to identify different repeating patterns.

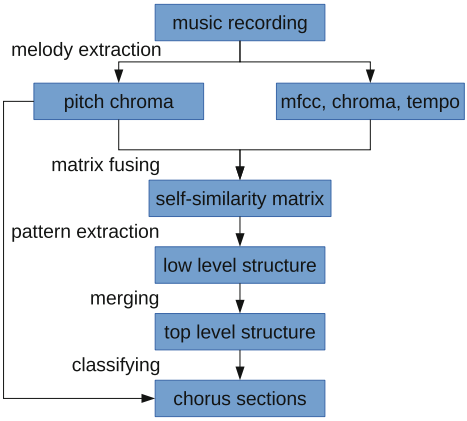

Fig. 1. Overview of the proposed method

3.1 Pitch Chroma Feature

The proposed method adopts the melody extraction result from [13] which is a melody line: a sequence of estimated fundamental frequency $\{f_0, f_1, f_2, \dots\}$ corresponding to each timestamp $\{t_0, t_1, t_2, \dots\}$. Though the algorithm has state-of-the-art results, wrong estimations in the output make the raw sequence not suitable for measuring similarity directly. Inspired by chroma feature, a pitch chroma feature vector is derived from the fundamental frequency sequence which is robust to the errors.

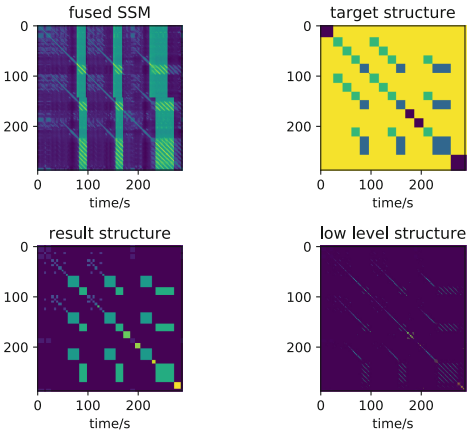

Fig. 2. Results for song 'Dream magic' from RWC Pop database

For a given window of frequencies $\{f_i, \ldots, f_j\}$ and a given number of pitch classes N_{class}, frequency values belonging to each pitch class were counted as pc_i, comprising a feature vector reflecting the occurrence of the pitches $\left[pc_0 \ldots pc_{N_{class}-1}\right]$. The frequency value f is mapped to its pitch class in a similar way to that in the chroma feature:

$$pitchClass(f) = N_{class} \log_2(f) \bmod N_{class} \tag{1}$$

the occurrences were counted as:

$$pc_k = \sum_{l=i}^{j} [pitchClass(f_l) = k] \tag{2}$$

In the proposed method, the number of pitch classes is set to $N_{class} = 24$ which gives the vector a finer resolution. The window size is $0.1 * 10\,\mathrm{s}$ long, while the SSM used in the proposed method has a frame size of $0.23\,\mathrm{s}$.

3.2 Key-Shift Invariant Distance

Modulation is the change of tonality, modulated sections are considered as the same pattern in the proposed structure analysis method. To deal with the key change in modulated sections, a key-shift invariant distance is used as the similarity measure for chroma and pitch chroma feature vectors.

For two feature vectors denoted as:

$$\mathbf{x} = \left[x_0 \ldots x_{n-1}\right], \mathbf{y} = \left[y_0 \ldots y_{n-1}\right] \tag{3}$$

n cosine similarity values is calculated by rolling the element in vector $\mathbf{y}$ by an offset of $i = 0, \ldots, n-1$ and evaluating the similarity between $\mathbf{x}$ and

$[y_{0+i} \cdots y_{n-1+i \bmod n}]$. The maximum similarity value, or the minimum distance among them is presented as the key-shift invariant distance of the two vectors. For example, the chroma feature has 12 pitch classes, then the distance is invariant to key changes in semitones.

3.3 Repeating Pattern Extraction

Low Level Pattern Extraction. Using the modified version of the algorithm from [24] with key-shift invariant distance, a self-similarity matrix is calculated by fusing SSMs of Mel-frequency cepstral coefficients (MFCC), chroma, pitch chroma and tempogram feature vectors. The fused SSM is 'cleaner' where stripe patterns corresponding to repeating segments were highlighted.

The fused SSM was binarized according to a threshold of $\exp(-5)$, values lower than the threshold were set to 0 while the rest were set to 1. The proposed method takes the binarized matrix as the adjacency matrix of the self-similarity graph (SSG) g_{ss} where vertices represents audio frames and edges represents the similarity relation between the frames.

Low-level patterns, or short repeating segments, were captured first. Similar frames forms a fully connected subgraph, or a clique in the SSG. By extracting cliques from the SSG, redundant or wrong edges representing a similarity relation were removed. However, cliques may overlap in the SSG, as there are noise/errors in the generated graph breaking the transitivity of similarity. To deal with the noise and find the repeating segments, the clique with maximum size was iteratively extracted from the graph as described in Algorithm 1, once a clique was extracted, the vertices in that clique were removed from the graph. This step requires to find all possible cliques in the SSG which is time-consuming when the graph is big, so literally only the first 10000 cliques found were used for selecting the largest clique, with less cliques to select, results will be slightly worse while decreasing processing time.

Figure 3 shows 2 example cases of extracting cliques from an undirected graph, cliques were enclosed in red-dotted circles, in the right subfigure, there are possible cliques that overlap with each other.

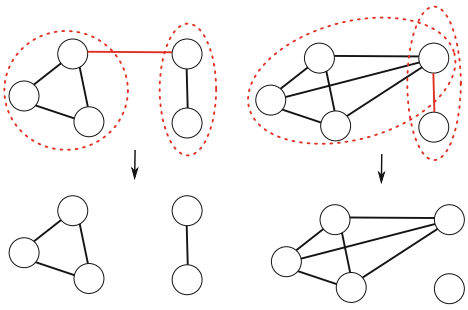

Fig. 3. Extracting cliques from undirected graph

Algorithm 1. Extract cliques from graph g_{ss}

Require: g_{ss} is undirected graph
Ensure: $C = \{c, \dots\}$ are non-overlap cliques in g_{ss}
  $C \leftarrow \{\}$
  while g_{ss} is not empty **do**
    $x \leftarrow findCliques(g_{ss})$
    $c \leftarrow maxSizeClique(x)$
    insert c into C
    remove c from g_{ss}
  end while
  return C

The extracted cliques were treated as low level patterns, each represents a group of repeating segments. For clarity, unique numbers can be assigned to the cliques, then the music structure will be represented by a sequence of label numbers for audio frames. For visualization, a labeled SSM can be constructed using the label sequence. The three representations, cliques, label sequence and labeled SSM, are equivalent data structures since they transform to each other freely. For example, cliques $C = \{(0, 1), (2, 3)\}$, label sequence $S = \{1, 1, 2, 2\}$ and labeled SSM $M_{label} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \end{bmatrix}$ all refer to the same structure.

Cliques Merging. To get functional level structure of music, the method merged the original extracted cliques to form larger repeating segments. Consider a music piece with structure ABA for example, the low level structure would look like $a_0a_0a_1a_2b_0b_1a_0a_0a_1a_2$ and the cliques extracted would be $\{(0, 1, 6, 7), (2, 8), (3, 9), (4), (5)\}$. The target structure ABA, however, yields a target list of clique as $C = \{(0, 1, 2, 3, 6, 7, 8, 9), (4, 5)\}$. The principle is, if two cliques are sequentially adjacent, like a_0 and a_1 whose clique representation is $(0, 1, 6, 7)$ and $(2, 8)$, they can be merged into the same larger clique, as depicted in Fig. 4, where sequentially adjacent frames were connected by red lines. The transitivity of similarity relation were kept between merged repeating segments.

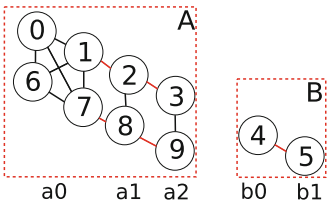

Fig. 4. Merging cliques into larger repeating segments

To decide whether two cliques were sequentially adjacent, we introduce the notion of 'ends' of the clique: it's composed of heads and tails (endpoints excluded) of the consecutive segments in the clique. For example, $(0, 6)$ and $(1+1, 7+1)$ are the heads and tails of clique a_0. Given two clique c_i, c_j where the minimum frame number satisfies $min(c_i) < min(c_j)$, tails of the former clique $tails(c_i)$ and heads of the latter clique $heads(c_j)$ were compared to tell if they met the adjacent condition.

Ideally, if the two cliques were adjacent, the heads and the tails should match as $tails(c_i) = heads(c_j)$, but to tolerate deviations the method uses a predicate which is a conjunction of:

- repeating counts restriction: difference between the numbers of consecutive segments in c_i and c_j is within D_{block}, which means for the lengths of the heads and tails, condition

$$|len(heads(c_i)) - len(tails(c_j))| < D_{block} \tag{4}$$

  is satisfied.
- distance restriction: distance between $tails(c_i), heads(c_j)$ is within D_{adj}, which means for most (except for at most D_{block} items) of the items $x \in tails(c_i)$ or $y \in heads(c_j)$, condition

$$\min_{\forall y \in heads(c_j)} |x - y| < D_{adj} \tag{5}$$

  or

$$\min_{\forall x \in tails(c_i)} |y - x| < D_{adj} \tag{6}$$

  is satisfied.

To merge original cliques into larger cliques. An adjacency matrix of the cliques M_{clique} whose items are $m_{i,j}$ is constructed by evaluating the predicate mentioned above for cliques c_i and c_j where $i < j$. According to the principle 'sequentially adjacent cliques be merged into the same larger clique', let g_{clique} be a graph with adjacency matrix M_{clique}, cliques in the same connected components were to be merged into one as described in Algorithm 2.

Smoothing and Adaptive Merging. To reduce noise/errors in the final output of structure analysis, a median filter with window size K_{window} is applied to the label sequence representation of the structure which smooth the output and keeps large repeating segments in one piece.

The parameter D_{block} is crucial for controlling the hierarchy of the output structure, with bigger D_{block} the criteria in Sect. 3.3 is more tolerant and results in larger repeating segments, with smaller D_{block}, the criteria is more strict and results in lower-level patterns. There is no optimal value of D_{block} for every song and its SSM, thus an adaptive merging method was adopted. From multiple merging outputs with $D_{block} \in \{0, \ldots, 2\}$ and $K_{window} \in \{23, 37, 47\}$, result whose count of cliques is greater than 3 and of minimum 'error' is selected.

Algorithm 2. Merge cliques

Require: $C = \{c, \dots\}$ are cliques
Ensure: $C_{merge} = \{c', \dots\}$ are largest possible merged cliques
  $M_{clique} = [m_{i,j}]$
  for $c_i, c_j \in C$ **do**
    $m_{i,j} \leftarrow isAdjacent(c_i, c_j)$
    $m_{j,i} \leftarrow m_{i,j}$
  end for
  $C_{merge} \leftarrow \{\}$
  for $x \in components(M_{clique})$ **do**
    $c' \leftarrow ()$
    for $c \in x$ **do**
      add c into c'
    end for
    insert c' into C_{merge}
  end for
  return C_{merge}

The error of clique merging process is modeled by comparing the labeled SSMs of the original clique and that of the merged clique. Empirically, a good merging result is close to the original cliques, thus we use an error function composed of two terms: false negative rate and false positive rate. Given the original cliques C and merged cliques C_{merge}, their labeled self-similarity matrix representations $m_c = M_{label}(C)$ and $m_{c'} = M_{label}(C_{merge})$ were compared. The error function is:

$$Error(C, C_{merge}) = \alpha E_{Neg} + max(\beta E_{Pos} - 0.1, 0) \tag{7}$$

where $E_{Neg} = sum(m_{c'} = 0 \wedge m_c \neq 0)$ is the number of false negatives and $E_{Pos} = sum(m_{c'} \neq 0 \wedge m_c = 0)$ is the number of false positives, coefficients $\alpha = \frac{1}{sum(m_c \neq 0)}$ and $\beta = \frac{1}{sum(m_c \neq 0)}$ were used to normalize the importance of the terms as both type of errors are considered equally important. Good merged cliques should cover the original cliques, thus the false positives were inevitable and always greater than 0, so the minus-then-max function clips the false positive rate lower then 0.1.

3.4 Chorus Detection

Based on the music structure analysis results, the chorus detection task is just of choosing the right repeating segments as the chorus. The proposed method uses a random forest classifier to learn which cliques are the chorus sections. The chorus sections tend to have different acoustic features to other sections and specific positions in the song, thus acoustic and structural features of the cliques $C = \{c, \dots\}$ were provided to train the classifier, the features used were listed below:

- *clique duration*: duration occupied by the clique normalized by duration of the song.
- *voicing rate*: the ratio of the number of voicing frames to that of all frames in the clique by counting non-zero frequencies in the melody line within the clique.
- *melody median*, *minimum* and *maximum*: median, minimum and maximum value of the frequencies in the melody line within the clique.
- *clique head* and *last clique head*: the smallest and the biggest frame number in the heads of the clique $heads(c)$.
- *segments count*: the number of consecutive segments in the clique by measuring the size of $heads(c)$.

Apart from the structural features like *clique head*, we added more features to expose the positional/structural information of the cliques. Based on the 8 features mentioned above, in 3 ways additional features were generated:

- *ranking*: features of the cliques in a music recording were ranked by sorting their values, the ranking numbers were used as additional features.
- *normalizing*: features were normalized by the maximum value from the cliques in a music recording to generate additional features.
- *stacking*: cliques were sorted by occurrence (their minimum frame number), then features of a clique's predecessor and successor were copied and added as additional features.

In the training phase, the proposed structure analysis algorithm was first applied to the music recordings in the training set to get repeating segments, then each clique was compared with the ground truth annotation to decide whether to label it as a chorus section or not. The comparison is done by measuring the overlap ratio between the clique and the ground truth chorus sections, given the length of the clique l_{clique}, the length of the chorus section and of overlap part $l_{chorus}, l_{overlap}$, two metrics can be calculated as:

- precision: $p = \frac{l_{overlap}}{l_{clique}}$
- recall: $r = \frac{l_{overlap}}{l_{chorus}}$

Cliques with precision $p > 50\%$ and recall $r > 10\%$ were labeled as chorus sections, the others were labeled as non-chorus sections. For each clique, an 8 dimensional feature vector and an $8 * 4 = 32$ dimensional additional feature vector were calculated, the data were used to train a random forest classifier with 1000 decision trees.

In the prediction phase, features of the cliques extracted by the structure analysis algorithm were fed to the classifier, the output were directly used as the result of chorus detection.

4 Evaluation

4.1 Database and Metrics

The RWC Pop Database [8] used for evaluation contains 100 popular songs, each with one functional structure annotation file. To train the classifier for chorus

detection, the dataset was randomly split into a training set and a validation set which constitutes of 70 songs and 30 songs respectively. The evaluation do not distinguish between different chorus sections like 'chorus A' and 'chorus B' which were used in the annotation. Chorus detection performance was measured by overlap-based metrics like in Sect. 3.4 and [7]:

- precision: $P = \frac{\text{total length of correctly detected chorus sections}}{\text{total length of detected chorus sections}}$
- recall: $R = \frac{\text{total length of correctly detected chorus sections}}{\text{total length of correct chorus sections}}$
- f-measure: $F = \frac{2RP}{R+P}$

For comparison with algorithm from [9] which outputs single chorus section, a modified version of the above metrics was used. Only nearest ground truth chorus section was considered when measuring the output chorus section. If there were multiple output chorus section, the length of distinct nearest correct chorus sections were summed up as $L_{nearest\ chorus}$. Given the total length of correctly detected nearest chorus sections $L_{nearest\ overlap}$, the modified metrics were denoted as:

- precision-single: $P_{single} = \frac{L_{nearest\ overlap}}{\text{total length of detected chorus sections}}$
- recall-single: $R_{single} = \frac{L_{nearest\ overlap}}{L_{nearest\ chorus}}$
- f-measure-single: $F_{single} = \frac{2R_{single}P_{single}}{R_{single}+P_{single}}$

The modified metrics are suitable for algorithms detecting a single chorus section and are compatible with algorithms detecting multiple chorus sections, for the latter case, the precision and recall can be viewed as an averaged score for the multiple detected chorus sections.

4.2 Reference Methods

The proposed method was denoted as 'seqRecur', for better comparison with [9], a modified version of the method denoted as 'seqRecurS' which has the same single section output format were also evaluated. The fixed-length single chorus section which covers most of the chorus sections predicted by the proposed method was selected as the output of 'seqRecurS'.

Apart from the proposed method, we evaluated 6 reproducible algorithms, denoted as 'highlighter', 'scluster', 'sf', 'olda', 'cnmf' and 'foote' [5,9,15,16,20,23]. MSAF [19] implementation of the latter 5 structure analysis algorithms from https://github.com/urinieto/msaf were used. To evaluate the performance of structure analysis algorithms on chorus detection task, extra steps were taken.

For label algorithms from MSAF (scluster, cnmf) which outputs the music structure via labeled sections, the chorus detection method as in Sect. 3.4 can be applied. For boundary algorithms from MSAF (sf, olda, foote) which split a music recording into sections, since the output has no recurrent music structure but only boundaries, the output sections was labeled by maximizing similarity on the SSM used in Sect. 3.3, then the same chorus section classifier can be applied on these sections.

To assess the effect of the chorus detection classifier independently, ground truth music structure were provided in the training and prediction phase of the chorus classifier, the result was denoted as 'gt'. To assess the effect of the structure analysis algorithm independently, the output sections can be assigned labels to achieve the highest possible precision, i.e. the section was labeled as chorus if more than 50% of its length overlap with ground truth chorus sections, its results were denoted by plus sign suffix like 'scluster+' as it represents the upper bound chorus detection precision of a structure analysis method.

4.3 Results

The average precision, recall and f-measure for songs in the validation set were listed in Table 2. The violin plot which shows the minimum, maximum and average value of F, F_{single} for songs in the validation set were shown in Fig. 5. The proposed method 'seqRecur' was the best on R, F among other structure analysis algorithms, though its upper bound performance 'seqRecur+' was worse than that of 'olda+'. The modified proposed method 'seqRecurS' were comparable on F_{single} than 'highlighter', which was designed for detecting a single chorus section.

The high scores of 'gt' shows that the chorus detection classifier was capable of learning from the human-labeled ground truth structure annotations. The performance decrease from results of highest possible precision 'X+' to its correspondence 'X' using the classifier to detect chorus sections shows that the output of existing structure analysis algorithms didn't fit chorus detection task well, one possible reason is that the output structure lacks consistency, making it difficult to learn to distinguish the chorus from other functional sections.

Table 1. Reference method categories

	Ground truth structure	Calculated structure
Ground truth chorus	–	X+
Calculated chorus	gt	X

4.4 Ablation Study

To verify the effect of enhancing SSM by introducing the pitch chroma feature, we conduct an ablation study by removing the pitch chroma feature used in the SSM fusion step. With the same parameter settings, the evaluation results were listed in Table 3. By utilizing the melody extraction results with the pitch chroma feature, the performance of the chorus detection system increased by 2% in f-measure for method 'seqRecur'.

Table 2. Average results on validation set. Depending on whether the two stages of the algorithms: structure analysis and chorus detection have used ground truth results, the reference methods can be divided into 3 categories as described in Table 1.

algo	P	R	F	P_{single}	R_{single}	F_{single}
seqRecur	0.8050	**0.7688**	**0.7726**	0.7957	0.7771	0.7701
seqRecurS	0.8533	0.3317	0.4680	0.8198	0.7212	0.7508
highlighter	**0.8820**	0.3354	0.4762	**0.8571**	0.7784	0.7910
scluster	0.6772	0.6528	0.6038	0.6436	0.6912	0.6324
sf	0.7889	0.7068	0.6906	0.7446	0.8303	0.7423
olda	0.7793	0.7615	0.7313	0.7304	0.8675	0.7571
foote	0.8368	0.6812	0.7030	0.7886	**0.8738**	**0.8068**
seqRecur+	0.8573	0.8031	0.8169	**0.8436**	0.8155	0.8168
scluster+	0.8545	0.8972	0.8672	0.8107	0.9227	0.8470
sf+	0.8108	0.8841	0.8323	0.7624	0.9221	0.8141
olda+	0.8683	**0.9344**	**0.8940**	0.8221	**0.9548**	**0.8762**
cnmf	0.5510	0.6482	0.5714	0.5262	0.6966	0.5805
cnmf+	0.7929	0.8594	0.8078	0.7457	0.9063	0.8065
foote+	**0.8717**	0.8684	0.8621	0.8270	0.8898	0.8452
gt	0.9395	0.9460	0.9423	0.9253	0.9463	0.9328

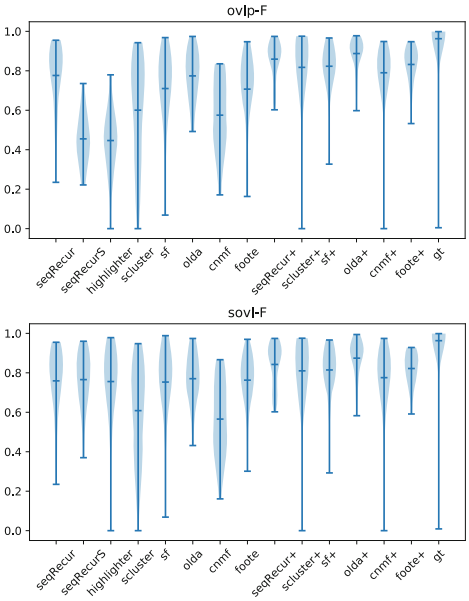

Fig. 5. Distribution of F (ovlp-F) and F_{single} (sovl-F) on validation set

Table 3. Performance increase when pitch chroma feature was used to enhance SSM

algo	ovlp-P	ovlp-R	ovlp-F	sovl-P	sovl-R	sovl-F
seqRecur	−0.98%	5.23%	2.31%	−1.48%	3.85%	1.02%
seqRecurS	−6.52%	−1.69%	−2.76%	−6.82%	−3.58%	−5.26%
seqRecur+	2.21%	−5.31%	−2.00%	1.74%	−4.13%	−1.44%

5 Conclusion

This paper proposed a chorus detection method based on music structure analysis results. To better compute the music similarity, we enhanced an existing similarity fusing method by introducing a new feature which exploits melody extraction algorithms and a key-shift invariant distance to deal with the key changes. A novel structure analysis method using graph algorithms and a chorus detection method using supervised learning was proposed.

The chorus detection method were applied to the output of the proposed structure analysis algorithm and the other 5 state-of-the-art algorithms. Evaluation results shows the method was comparable with the state-of-the-arts algorithms on both multiple and single chorus section detection tasks. The adapted structure analysis methods using part of the proposed method to detect chorus sections also reach high performance.

Results shows utilizing music structure analysis and melody extraction algorithms for chorus detection was viable and competitive. However, the performance was still not satisfactory comparing to the upper bound. Two reasons lie behind this: the structure analysis algorithms were not good enough, and the ambiguity of what 'chorus' means since the arrangement of songs varies and the functional sections were annotated by humans.

Ackowlegement. This work was supported in part by National Key R&D Program of China (2019YFC1711800), NSFC (61671156).

References

1. Bittner, R.M., McFee, B., Salamon, J., Li, P., Bello, J.P.: Deep salience representations for F0 estimation in polyphonic music. In: The 18th International Society for Music Information Retrieval Conference, Suzhou, China, pp. 63–70 (2017)
2. Wang, B., Jiang, J., Wang, W., Zhou, Z.-H., Tu, Z.: Unsupervised metric fusion by cross diffusion. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, Providence, RI, pp. 2997–3004 (2012). https://doi.org/10.1109/CVPR.2012.6248029, http://ieeexplore.ieee.org/document/6248029/
3. Chen, M.T., Li, B.J., Chi, T.S.: CNN based two-stage multi-resolution end-to-end model for singing melody extraction. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, pp. 1005–1009. IEEE (2019). https://doi.org/10.1109/ICASSP.2019.8683630, https://ieeexplore.ieee.org/document/8683630/

4. Cheng, T., Smith, J.B.L., Goto, M.: Music structure boundary detection and labelling by a deconvolution of path-enhanced self-similarity matrix. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, pp. 106–110. IEEE (2018). https://doi.org/10.1109/ICASSP.2018.8461319, https://ieeexplore.ieee.org/document/8461319/
5. Foote, J.: Automatic audio segmentation using a measure of audio novelty. In: 2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532), New York, NY, USA, vol. 1, pp. 452–455. IEEE (2000). https://doi.org/10.1109/ICME.2000.869637, http://ieeexplore.ieee.org/document/869637/
6. Gao, S., Li, H.: Popular song summarization using chorus section detection from audio signal. In: IEEE 17th International Workshop on Multimedia Signal Processing (MMSP), Xiamen, China, pp. 1–6. IEEE (2015). https://doi.org/10.1109/MMSP.2015.7340798, http://ieeexplore.ieee.org/document/7340798/
7. Goto, M.: A chorus section detection method for musical audio signals and its application to a music listening station. IEEE Trans. Audio Speech Lang. Process. **14**(5), 1783–1794 (2006). https://doi.org/10.1109/TSA.2005.863204, http://ieeexplore.ieee.org/document/1677997/
8. Goto, M., Hashiguchi, H., Nishimura, T., Oka, R.: RWC music database: popular, classical and jazz music databases. In: ISMIR, Paris, France, vol. 2, pp. 287–288 (2002)
9. Huang, Y.S., Chou, S.Y., Yang, Y.H.: Pop music highlighter: marking the emotion keypoints. Trans. Int. Soc. Music Inf. Retrieval **1**(1), 68–78 (2018). https://doi.org/10.5334/tismir.14, http://transactions.ismir.net/articles/10.5334/tismir.14/
10. Jiang, N., Muller, M.: Estimating double thumbnails for music recordings. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Queensland, Australia, pp. 146–150. IEEE (2015). https://doi.org/10.1109/ICASSP.2015.7177949, http://ieeexplore.ieee.org/document/7177949/
11. Jun, S., Rho, S., Hwang, E.: Music structure analysis using self-similarity matrix and two-stage categorization. Multimedia Tools Appl. **74**(1), 287–302 (2015). https://doi.org/10.1007/s11042-013-1761-9
12. Kaiser, F., Sikora, T.: Music structure discovery in popular music using non-negative matrix factorization. In: The 11th International Society for Music Information Retrieval Conference, Utrecht, Netherlands, p. 6 (2010)
13. Kum, S., Nam, J.: Joint detection and classification of singing voice melody using convolutional recurrent neural networks. Appl. Sci. **9**(7), 1324 (2019). https://doi.org/10.3390/app9071324, https://www.mdpi.com/2076-3417/9/7/1324
14. Lu, L., Wang, M., Zhang, H.J.: Repeating pattern discovery and structure analysis from acoustic music data. In: Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval - MIR 2004, New York, NY, USA, p. 275. ACM Press (2004). https://doi.org/10.1145/1026711.1026756, http://portal.acm.org/citation.cfm?doid=1026711.1026756
15. McFee, B., Ellis, D.P.: Learning to segment songs with ordinal linear discriminant analysis. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, pp. 5197–5201. IEEE (2014). https://doi.org/10.1109/ICASSP.2014.6854594, http://ieeexplore.ieee.org/document/6854594/
16. McFee, B., Ellis, D.P.W.: Analyzing song structure with spectral clustering. In: The 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan, p. 6 (2014)

17. Mildner, V., Klenner, P., Kammeyer, K.D.: Chorus detection in songs of pop music. In: Proceedings of ESSV, Universität Karlsruhe, Karlsruhe, p. 8 (2003)
18. Müller, M., Kurth, F.: Towards structural analysis of audio recordings in the presence of musical variations. EURASIP J. Adv. Sig. Process. **2007**(1) (2006). https://doi.org/10.1155/2007/89686, https://asp-eurasipjournals.springeropen.com/articles/10.1155/2007/89686
19. Nieto, O., Bello, J.P.: Systematic exploration of computational music structure research. In: The 17th International Society for Music Information Retrieval Conference, New York City, USA, p. 7 (2016)
20. Nieto, O., Jehan, T.: Convex non-negative matrix factorization for automatic music structure identification. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, pp. 236–240. IEEE (2013). https://doi.org/10.1109/ICASSP.2013.6637644, http://ieeexplore.ieee.org/document/6637644/
21. Paulus, J., Klapuri, A.: Audio-based music structure analysis. In: The 11th International Society for Music Information Retrieval Conference, Utrecht, Netherlands, p. 12 (2010). citation Key Alias: paulusAUDIOBASEDMUSICSTRUCTURE2010a
22. Salamon, J., Gómez, E.: Melody extraction from polyphonic music signals using pitch contour characteristics. IEEE Trans. Audio Speech Lang. Process. **20**(6), 1759–1770 (2012)
23. Serra, J., Muller, M., Grosche, P., Arcos, J.L.: Unsupervised detection of music boundaries by time series structure features. In: Twenty-Sixth Conference on Artificial Intelligence, Toronto, Ontario, Canada, p. 7 (2012)
24. Tralie, C.J., McFee, B.: Enhanced hierarchical music structure annotations via feature level similarity fusion. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, pp. 201–205. IEEE (2019). https://doi.org/10.1109/ICASSP.2019.8683492, https://ieeexplore.ieee.org/document/8683492/
25. Wu, F., Sun, S., Xue, W.: Automatic extraction of popular music ringtones based on music structure analysis. In: 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, Japan, pp. 1–5. IEEE (2016). https://doi.org/10.1109/ICIS.2016.7550919, http://ieeexplore.ieee.org/document/7550919/