Special Paper
방송공학회논문지 제26권 제7호, 2021년 12월 (JBE Vol. 26, No. 7, December 2021)
https://doi.org/10.5909/JBE.2020.26.7.838
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

# Score Image Retrieval to Inaccurate OMR performance

Haekwang Kim[a)‡]

## Abstract

This paper presents an algorithm for effective retrieval of score information to an input score image. The originality of the proposed algorithm is that it is designed to be robust to recognition errors by an OMR (Optical Music Recognition), while existing methods such as pitch histogram requires error induced OMR result be corrected before retrieval process. This approach helps people to retrieve score without training on music score for error correction. OMR takes a score image as input, recognizes musical symbols, and produces structural symbolic notation of the score as output, for example, in MusicXML format. Among the musical symbols on a score, it is observed that filled noteheads are rarely detected with errors with its simple black filled round shape for OMR processing. Barlines that separate measures also strong to OMR errors with its long uniform length vertical line characteristic. The proposed algorithm consists of a descriptor for a score and a similarity measure between a query score and a reference score. The descriptor is based on note-count, the number of filled noteheads in a measure. Each part of a score is represented by a sequence of note-count numbers. The descriptor is an n-gram sequence of the note-count sequence. Simulation results show that the proposed algorithm works successfully to a certain degree in score image-based retrieval for an erroneous OMR output.

Keywords: image retrieval, musical score, symbolic representation, optical music recognition, descriptor

## I. Introduction

Score is used as a representation of a musical work in western culture. Fig.1 shows an example of score. It has two parts. Each part consists of measures that is separated by barlines. Each measure has notes and rests. A note has duration and pitch as its properties. Its pitch is represented by the vertical position on stafflines. Nowadays, a musical work is represented in a digital form such as in mp3 audio format for audio music or in MIDI file format for symbolic music. Nonetheless, score written on paper is still one of the mostly favored presentation tool for musical communication among composers, performers, and conductors.

Music retrieval has a long history as a research topic, for easy access to target music in enormous database or on Internet. ‘Search by Humming’ is a retrieval tool that finds similar music to a melody of sound generated by human humming voice. A melody is matched as pattern with a sequence of notes on temporal axis[1]. ‘Search by example

a) Department of Computer Engineering, Sejong University
‡ Corresponding Author : Haekwang Kim
E-mail: hkkim@sejong.ac.kr
Tel: +82-2-3408-3757
ORCID: https://orcid.org/0000-0002-0302-0627
※This work is from the sabbatical leave grant by Sejong University for the second semester 2017.
· Manuscript received August 30, 2021; Revised September 23, 2021; Accepted September 28, 2021.

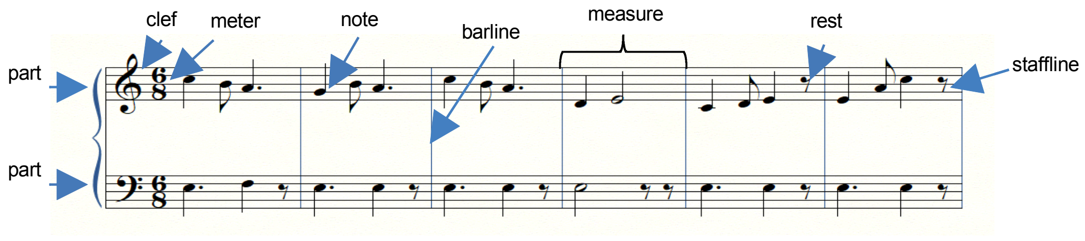

Fig. 1. An example of score

sound' retrieves the same music by audio signatures extracted by signal processing technologies from audio sound. In this paper, a score retrieval method with a score image as input is presented. The proposed algorithm addresses the retrieval problems due to failures by an OMR (Optical Music Recognition). An overview of the related work is presented in section II. The proposed algorithm is explained in section III. Experiment results are analyzed in section IV and This paper concludes in section V.

## II. Existing work

Image retrieval in general is based on extracting features from or tagging keywords to an image, forming descriptor(s). A similarity is calculated for a query image and a reference image using the descriptor(s) and images at the top ranks in terms of similarity are returned from a database of images as search result for further browsing. The MPEG-7 international standard defines descriptors of color, texture, and shape features for content-based image retrieval[2]. As this handmade style of descriptors did not show sufficient performance in the field, deep learning-based descriptors has been the focus of research for image retrieval with its better performance[3].

Score image retrieval is to identify a query score image for retrieving related information. A score is a written document conveying musical meaning in a musical language with musical symbols. Conventional image descriptors are not adequate for score retrieval. The score image retrieval uses a symbolic transcription of a scanned score image obtained with OMR (Optical Music Recognition) technology. Jsymbolic implements 246 descriptors for symbolic music retrieval for melody, pitch, rhythm features with all of them without consideration of errors induced from OMR[4]. An OMR based retrieval suggests pitch information be extracted with OMR and corrected for retrieval[5]. Score reading and correction requires people have

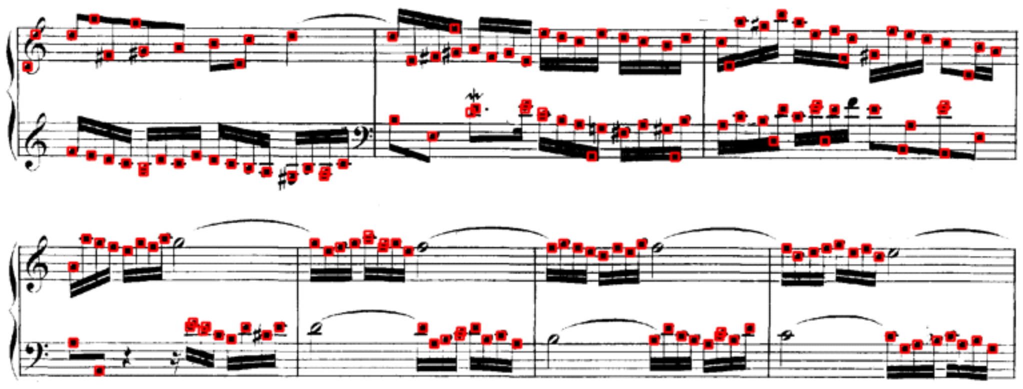
Fig. 2. Example of musical symbol recognition of a scanned score image by Moonlight OMR

hard training for long time with a lot of exercises and even for professional musicians, the task is cumbersome and tedious. One of important features for commercial OMR is easy UI for OMR error correction. Moonlight OMR is a tensorflow based opensource project for OMR by Google, claiming not officially supported[6]. Fig.2 shows an example of OMR result for a part of a scanned score image by Moonlight OMR. The whole page consists of 44 measures 451 filled noteheads. A small square indicates the location of recognized filled noteheads. Most of filled noteheads are correctly detected except only a few errors like at a treble clef. There are 34 detection errors (7.5%) for 451 filled noteheads. For total of 44 measures, 4 measures are with 0 detection error, 13 measures with 1 error, 3 measures for 2 errors and 2 measures for more than 3 errors. Approximately 84% of measures are detected at most 1 error per measure. With the state of the art OMR, the accuracy will be much higher than this experimental OMR.

Among the musical symbols on a score, it is observed that filled noteheads are rarely detected with errors with its simple black filled round shape characteristic for OMR processing. OMR takes a score image as an input; 1) pre-processes the input image improving image quality such as noise removal, contrast enhancing and binarization; 2) musical symbols are recognized by computer vision technology, notably using deep learning; 3) the detected symbols are combined to reconstruct a score structure with MIDI or MusicXML model and saved into a file [7]. Although the performance of the state-of-the art OMR is quite remarkable, there are still errors due to the complexity of musical notation system, much more complicated than the usual alphabet-based text content.

## III. Proposed algorithm

Proposed descriptor: n-gram vector sequence for a part

The input is a score image. A score consists of one or more parts: score = <part1, part2, ⋯, parth>. Each part corresponds to an instrument such as piano, violin or a voice such as soprano, tenor. A part consists of one or more measures: part = <measure1, measure2, ⋯, measurek>.

The proposed descriptor is constructed by the following process:

***Step1)*** a MusicXML score is generated from an input score image by OMR.

***Step2)*** the number of filled noteheads is counted for each measure and a note-count sequence is created for each part. For example, if a note-count sequence for a part is <3, 4, 2, 7, 3>, then the part has 5 measures, the first measure has 3 filled noteheads, the second measure has 4 filled noteheads, and so on.

***Step3)*** From each note-count sequence, an n-gram vector sequence is generated with zero padding to the front and the rear of the note-count sequence. The padding size depends on the n as $\left\lfloor \frac{n}{2} \right\rfloor$. If n is 3, the note-count sequence is padded with 1 zero at the front and the rear. The example note sequence in Step2 is padded as <0, 3, 4, 2, 7, 3, 0>. From this padded sequence, the 3-gram vector sequence is generated with window size of 3 at each position in the original note-count sequence as [<0,3,4>, <3,4,2>, <4,2,7>, <2,7,3>, <7,3,0>]. The proposed descriptor for score is the set of n-gram vector sequences obtained from parts of the score.

Similarity measure of a reference score for a query score

For a query score, similarity is calculated for a reference score in a database by the 4 steps. In step1, a distance matrix is calculated for a n-gram reference vector sequence with an n-gram query vector sequence using Euclidean distance. In step2, the best alignment of a query sequence along a reference sequence is obtained from the distance matrix from step1 with matching percentage as the length of the best alignment over the query length. In step3, each query part is checked the match with a reference part one-to-one from all possible pairs of query and reference.

In step4, similarity of query score and reference score is computed as the average of matching percentage over all the query parts from step3. Detailed description of each process is explained.

***Step1)*** A distance matrix is formed for each pair of parts, one from query and one from reference. The value at (i, j) position of the distance matrix is Euclidean distance of the ith n-gram vector for the query part and the jth n-gram vector for the reference part. For example, a query vector sequence is [<0, 2, 1>, <2, 1, 2>, <1, 2, 2>, <2 ,2, 0>] for 4 measures and a reference vector sequence is [<0, 3, 2>, <3, 2, 2>, <2, 2, 3>, <2, 3, 0>] for 4 measures, then its distance matrix size is 4x4. The value at (0, 2) position of the distance matrix is Euclidean distance of <0, 2, 1> at position 0 of the query and <2, 2, 3> at position 2 of the reference. The value is $\sqrt{(0-2)^2+(2-2)^2+(1-3)^2} = \sqrt{8}$.

***Step2)*** A binarized distance matrix is formed by thresholding: if a distance in the matrix is smaller than a threshold, then the value is set to ‘1’, otherwise set to ‘0’. The value of ‘1’ indicates that two vectors are matched. A high threshold makes the algorithm robust to detection of filled note errors of OMR, losing distinguishability for different scores. The maximum matching length of a query sequence alignment along a reference sequence is calculated from the binarized distance matrix by dynamic programming: equation (1) indicates that if the value of binarized distance matrix is 1 at (i, j), the matching length at (i, j) position is increased by 1 to the previous matching length at the position (i-1, j-1). If the value of binarized distance matrix is 0 at (i, j), then matching length at (i, j) is reset to 0. Equation (2) defines maximum matching length of a query sequence alignment along a reference sequence as the biggest matching length from equation (1) over all (i, j) positions.

The similarity of a reference part for a query part is defined in equation (3) as the ratio of maximum matching length over the size (the number of measures) of the query part.

***Step3)*** A matching part matrix for a query score and a reference score is obtained. The value at (i, j) position of this matrix is matching_percentage of the ith part of the query score and the jth part of the reference score. From this matching part matrix, one-to-one matching is found for each query part. The biggest value of the matching part matrix is chosen at position (i, j), then query part i is mapped to reference part j with its matching_percentage as best_pair (i, j, matching_percentage) and the row i is removed from the matrix. Repeat this process until the matrix empty. The result is the set of best_pair (i, j, matching_percentage) for all parts of the query score.

***Step4)*** The similarity of query score and a reference score is the average of the matching_percentage of the best_pair (i, j, matching_percentage) over all the query parts as in equation (4).

$$matching\_length(i,j) = \begin{cases} matching\_length(i-1,j-1)+1, & if\ matching\ at\ (i,j)=1 \\ 0, & ,otherwise \end{cases} \tag{1}$$

$$max\_matching\_length(part\_q, part\_r) = max_{i,j}(matching\_length(i,j)) \tag{2}$$

$$matching\_percentage(part\_q, part\_r) = \frac{max\_matching\_length(part\_q, part\_r)}{the\ \nu mber\ of\ measures\ in\ part\_q} \tag{3}$$

$$similarity(que, ref) = \frac{\sum_i matching\_percentage(i)}{the\ number\ of\ query\ parts},\ where\ i\ is\ \mathrm{for}\ the\ i^{th}\ part\ of\ the\ query \tag{4}$$

## Ⅳ. Experimental results

The algorithm is implemented in Python language with Music21 package developed by MIT. The Music21 provides an integrated programming environment for music analysis including descriptors such as pitch histograms, interval histograms, etc [8]. These descriptors may be extracted from arbitrary segment of a symbolic score such as in MusicXML file format. These descriptors assume that there are no errors on the input score. For example, a pitch histogram descriptor is completely broken if its related key signature has error from an OMR recognition process. For the simulation experiment of the proposed algorithm, 608 music scores in MusicXML file formats in the Music21 package are used for the data set. The scores in the package are European classics such as Bach, Beethoven, etc. For n-gram vector of the descriptor, n is set to 3. The threshold for the n-gram vector matching is set to 2. To simulate OMR errors, noises with 3 levels of error strength are added to note-count sequencies extracted from the scores. For each note-count for a measure, noise is uniform randomly added by equation (5). Sign is equally randomly produced from -1 and 1. U is standard uniform generated, U ~ Uniform(0,1).

$$\text{Noise} = \text{round}(\text{Sign} * U * \text{error\_strength}),\quad \text{where error strength} = 1, 2, 3. \tag{5}$$

Error_strength set to the value of d simulates d note count error per measure from OMR for scanned score image.

For the experiment, all 608 music scores in the dataset are used as query. For each query score, similarity in equation (4) is calculated for all the reference scores in the data set. the reference scores are sorted in the increasing order of similarity. The rank of the ground truth (query score) in the sorted list is obtained. Top10 rank is true if the ground truth is within top 10 positions in the sorted list. The performance of the proposed algorithm is measured by the average rank and the number of top10 ranks. The range of the rank is 1 to 608. Table 1 shows the performance results of the proposed algorithm from the simulation test. The case with no error shows that the implementation is correct with this case: average rank and top10 rank count are perfect with 1.0 and 608, respectively. With error level 1, the case of one filled note error per measure, the algorithm is still robust with average rank 2.2 and the top10 rank count 597. Most of queries are in top 10 ranks. However, with more severe errors with level 2 and level 3 cases, the algorithm did not retrieve the ground truth in sufficiently top ranks. Only 123 queries are in the top 10 rank for the error level 2 case and miserable 21queries for error level 3 case for total of 608 queries.

Table 1. Test results for 4 datasets with different noise level

| | No error | Error level 1 | Error level 2 | Eror level 3 |
|---|---|---|---|---|
| Average rank | 1.0 | 2.2 | 75.3 | 183.6 |
| Number of top10 ranks (percentage) | 608 (100%) | 597 (98.2%) | 123 (20.2%) | 21 (3.4%) |

Even for non-commercial experimental OMR such as Moonlight, detection errors for filled notehead per measure is much less than error level 1 condition, so the proposed method's performance (98.2% for top10 rank) for error level 1 shows promising for usage in the field with existing OMR technology.

Conventional descriptors using pitch or interval is catastrophic for score retrieval with OMR errors. A note on the centerline of a staff, its pitch value depends on clef for the note: its pitch is 'B' for treble clef and 'D' for a base clef. And all the musical symbols such as accidentals, keys, stems, beams and notes affect severe influences on music retrieval.

## Ⅴ. Conclusion

An algorithm for score image-based retrieval for musical score is presented with its simulation results. The number of notes is counted per measure and used for the proposed descriptor to be strong to errors for an OMR for a scanned score image as input. The experimental results show the possibility of the application in the field for the proposed algorithm with successful performance with average rank of 2.2 and 597 top 10 ranks over 608 queries for the case with one note count error per measure. This algorithm needs to be further improved with more experiments in realistic test conditions integrated with a state-of-the-art OMR software in the market.

## References

[1] D. Garfinkle, C. Arthur, P. Shubert, J. Cumming and I. Fujinaga, "PatternFinder: Content-based Music Retrieval with music21," Proceedings of International Workshop on Digital Libraries for Musicology, Shanghai, China, pp. 5-8, October 2017.

[2] T. Sikora, "The MPEG-7 visual standard for content description-an overview," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 696-702, June 2001.

[3] H. Wang, Y. Cai, Y. Zhang, H. Pan, W. Lv and H. Han, "Deep Learning for Image Retrieval: What Works and What Doesn't," Proceedings of IEEE International Conference on Data Mining Workshop, Atlantic City, USA, pp. 1576-1583, November 2015.

[4] C. McKay, J. E. Cumming and I. Fujinaga, "Jsymbolic2.2: extracting features from symbolic music for use in musicological and MIR research," Proceedings of 19th International Society for Music Retrieval Conference, Paris, France, pp. 348-354, September 2018.

[5] A. Hankinson, J. A. Burgoyne, G. Vigliensoni, A. Porter, J. Thomson, W. Liu, R. Chiu and I. Fujinaga, "Digital document image retrieval using optical music recognition," Proceedings of 13th International Society for Music Retrieval Conference, Porto, Portugal, pp. 577-582, October 2012.

[6] D. Ringwalt, L. Li, N. Jesus, Y. D. Yang, and C. Clauss "GitHub - tensorflow/moonlight: Optical music recognition in TensorFlow", Version v2020.08.14, August 2020

[7] E. Shatri and G. Fazekas, "Optical music recognition: State of the art and major challenges", Proceedings of the International Conference on Technologies for Music Notation and Representation, Hamburg, Germany, pp. 175--184, May 2021.

[8] M. S. Cuthbert, and C. Ariza. "music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data.", Proceedings of 11th International Society for Music Information Retrieval Conference, Utrecht, Netherlands, pp. 637-42, August 2010.

## Introduction Authors

Haekwang Kim

- 1986 : B.E. degree in Electronics Engineering, Hanyang University, Seoul, Korea
- 1994 : D.E.A in Informatics, Paul-Sabatier University, Toulouse, France
- 1997 : Docteur in Informatics, Paul-Sabatier University, Toulouse, France
- 1986 ~ 1991 : Researcher, Samsung Electronics, Korea
- 1992 ~ 1997 : Student researcher, IRIT, Toulouse, France
- 1997 ~ 2000 : Prinipal researcher, Hyundai Electronics, Korea
- 2000 ~ present : Professor, Sejong University, Korea
- ORCID : http://orcid.org/0000-0002-0302-0627
- Research of Interest : video compression, music processing