

## RESEARCH

# Piano Sheet Music Identification Using Dynamic N-gram Fingerprinting

Daniel Yang and T. J. Tsai

This article introduces a method for large-scale retrieval of piano sheet music images. We study this problem in two different scenarios: camera-based sheet music identification and MIDI-sheet image retrieval. Our proposed method combines bootleg score features with a novel hashing scheme called dynamic N-gram fingerprinting. This hashing scheme ensures that every fingerprint is discriminative enough to warrant a table lookup, which improves both retrieval accuracy and runtime. On experiments using all piano sheet music images in the IMSLP database, the proposed method achieves  $>0.8$  mean reciprocal rank with sub-second runtimes. As a practical application, we use our system to find matches between the Lakh MIDI dataset and IMSLP, which augments the IMSLP sheet music data with symbolic music information for a subset of pieces. We release our code and Lakh-IMSLP matches to facilitate future study.

**Keywords:** sheet music; piano; retrieval; search; fingerprinting

## 1 Introduction

This article investigates large-scale retrieval of piano sheet music images through two applications. The first application is camera-based sheet music identification, where a user can take a cell phone picture of a physical page of piano sheet music and identify the matching sheet music in a large database. Using a structured database like the International Music Score Library Project (IMSLP), this sheet-sheet retrieval task could allow a user to conveniently access related resources such as alternate editions of the sheet music or audio recordings of the piece. The second application is MIDI-sheet image retrieval, where a MIDI query is used to find a match in a database of sheet images (or vice versa). In this work, we will use our approach to identify matches between the Lakh MIDI Dataset (LMD) (Raffel, 2016) and IMSLP. This is an extremely large-scale, cross-modal retrieval task. In both applications, our work lays a foundation for searching large databases of sheet music in novel ways.

To understand how our work fits into the broader landscape, we provide an overview of previous work on retrieval of sheet music images. Most previous work on sheet music retrieval comes from the audio-sheet image alignment and retrieval literature. There are three general approaches to this problem. The first approach is to use optical music recognition (OMR) (Calvo-Zaragoza et al., 2020) to convert the sheet music into MIDI, extract chroma features from the MIDI, and then compare

them to chroma features extracted from the audio. This method has been applied to various forms of audio-sheet synchronization (Kurth et al., 2007; Damm et al., 2008; Fremerey et al., 2010; Thomas et al., 2016) and retrieval (Fremerey et al., 2008, 2009). The second approach is to compute a mid-level feature directly from the sheet music images, such as the location of noteheads (Izmirli and Sharma, 2012). This feature can then be related to audio chroma features using musical domain knowledge. The third approach is to use a neural network model to learn a common embedding space for audio and sheet music that captures semantic similarity. This approach has been applied to both audio-sheet alignment (Dorfer et al., 2017, 2016) and various forms of audio-sheet retrieval (Dorfer et al., 2018a, c, 2017). Recent work has also investigated the alignment task as a reinforcement learning problem (Dorfer et al., 2018b; Henkel et al., 2019).

Sheet-MIDI and sheet-sheet alignment & retrieval have also been studied in various forms. Several previous works study the problem of using a sequence of musical symbols as a query to find matches in a set of corresponding sheet music images. This typically involves preprocessing the sheet music with OMR or musical object recognition, and then performing an N-gram lookup (Thompson et al., 2011; Achankunju, 2018) or treating the problem as a string matching (Malik et al., 2013) or keyword spotting (Calvo-Zaragoza et al., 2018) problem. One recent work studies both sheet-MIDI and sheet-sheet retrieval, where OMR is used to convert sheet music into MIDI and then similarity is computed using dynamic time warping over the pitch sequences (Hajič et al., 2018). A few other recent works approach the image-MIDI alignment and retrieval

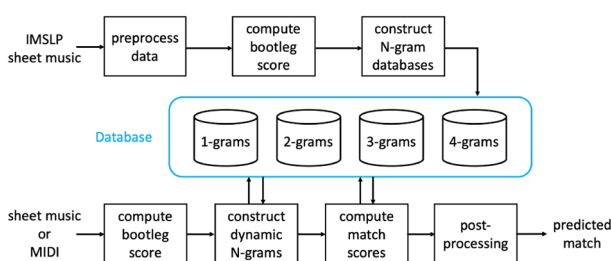
task by computing mid-level features that encode the location of noteheads relative to staff lines in sheet music. This ‘bootleg score’ representation has been applied to MIDI-sheet image synchronization (Tanprasert et al., 2019), large-scale MIDI-sheet retrieval (Tsai, 2020), and MIDI passage retrieval given a cell phone picture of sheet music (Yang et al., 2019; Tsai et al., 2020).

Our approach combines the bootleg score feature representation (Yang et al., 2019) with a novel hashing scheme called dynamic N-gram fingerprinting. This hashing method improves upon recent work (Tsai, 2020) that uses 1-grams as fingerprints. The dynamic N-gram method considers N-grams of different lengths, and it selects the length of each N-gram dynamically at runtime to ensure that every fingerprint is discriminative enough to warrant a table lookup.

This paper has four main contributions.<sup>1</sup> First, we propose a way to perform large-scale sheet music retrieval by combining an existing bootleg score feature representation with a novel hashing scheme called dynamic N-gram fingerprinting. Second, we show that our approach significantly improves upon previously proposed approaches on two different tasks: camera-based sheet music identification and MIDI-sheet image retrieval. In our experiments, we use all solo piano sheet music images in the IMSLP database, which is several orders of magnitude larger than previous studies. Third, we use the proposed system to find matches between the Lakh MIDI dataset and IMSLP, which results in a multimodal dataset containing MIDI files and matching IMSLP sheet music images. Fourth, as a byproduct of our work, we release a precomputed dataset of bootleg score features for all piano scores in IMSLP. Due to the extremely large size of the IMSLP dataset, this precomputed set of features can facilitate other studies on large-scale sheet music retrieval.

## 2 System Description

Our proposed approach for piano sheet music retrieval is shown in **Figure 1**. We will describe the system in three parts. In Section 2.1, we will explain the bootleg score feature representation, which corresponds to the two boxes labeled “compute bootleg score.” In Section 2.2, we explain the offline process of constructing the database, which corresponds to the three blocks in the upper half of **Figure 1**. In Section 2.3, we explain the online process of doing a search, which corresponds to the four blocks in the lower half of **Figure 1**. We will explain all parts of **Figure 1** for completeness, but we point out that our



**Figure 1:** Proposed architecture for large-scale piano sheet music retrieval.

novel contribution is in constructing multiple N-gram databases and using dynamic N-grams at runtime.

### 2.1 Bootleg Score Computation

There are three kinds of bootleg score features: sheet music bootleg scores (for sheet-sheet retrieval), MIDI bootleg scores (for MIDI-MIDI retrieval), and sheet-MIDI bootleg scores (for sheet-MIDI retrieval).

The bootleg score is a symbolic representation that describes the positions of noteheads in sheet music relative to staff lines (Yang et al., 2019; Tsai et al., 2020). The bootleg score itself is a sequence of 62-dimensional binary vectors, which encode the staff line positions of noteheads occurring at the same time. The first 28 elements of the 62-dimensional vector represent noteheads that appear in the left hand staff, corresponding to notes ranging between A0 and G4. The remaining 34 elements represent noteheads that appear in the right hand staff, corresponding to notes ranging between E3 and G8. Note that these two ranges have overlap (E3 to G4), since notes in the middle register can occur in the left or right hand staves. These left and right hand vectors are concatenated to form a  $62 \times N$  binary matrix, where  $N$  indicates the total number of simultaneous notehead events (e.g. a chord containing multiple notes would constitute a single event). Note that this representation does not encode key signature, time signature, accidentals, duration, octave markings, or clef changes. Even though it throws away a lot of information in the sheet music, a sufficiently long sequence of bootleg score events is generally distinctive enough to identify a piece. **Figure 2** shows an example section of sheet music and its corresponding bootleg score. This representation was originally proposed for aligning sheet music and MIDI files (Yang et al., 2019; Tsai et al., 2020), and has been adapted and applied to other tasks (Tsai, 2020; Yang and Tsai, 2020; Shan and Tsai, 2020).

The sheet music bootleg score is effective for sheet-sheet retrieval. Its computation consists of 3 steps. The first step is image preprocessing. The image is converted to grayscale, a blurred version of the image is subtracted to mitigate the effect of variable lighting conditions, and the image is resized to ensure that the separation between adjacent staff lines is within an acceptable range. The second step is object detection. Here, the goal is to detect three different types of musical symbols: filled noteheads, staff lines, and bar lines. Because these objects are all simple geometrical shapes (circles and straight



**Figure 2:** An excerpt of sheet music and its corresponding bootleg score.

lines), they can be detected efficiently and robustly using classical computer vision techniques like morphological transforms and blob detection. The third step is to integrate the detected object locations into a bootleg score. The bar lines are used to group staff lines into left and right hand staves, and the location of filled noteheads relative to the staff lines determine their location in the bootleg score. For a more detailed explanation, the reader is referred to Tsai et al. (2020).

The MIDI bootleg score is effective for MIDI-MIDI retrieval. Its computation consists of three steps. The first step is to extract a list of note onsets and their onset times. The second step is to group note onsets into simultaneous note events, which consist of one or more note onsets that occur within the same discretized time interval. The third step is to project the simultaneous note events onto a bootleg score using the deterministic rules and conventions of Western musical notation (e.g. B4 would appear on the middle staff line in the right hand). This projection ignores note durations and rests, and it only describes the sequence of note events. When there is ambiguity about a notehead location due to enharmonic representations and/or left-right hand attribution, multiple noteheads are placed in the bootleg score at all plausible locations.

The sheet-MIDI bootleg score is a variant that enables cross-modal sheet-MIDI retrieval. There are two differences between the sheet music bootleg score and the MIDI bootleg score that prevent their use in a cross-modal hashing framework. The first difference is with handling left-right hand attribution. While a notehead in the middle register will only appear in either the left *or* right hand staff in the sheet music bootleg score, it appears in both the left *and* right hand staves in the MIDI bootleg score. This difference can be resolved by duplicating noteheads in the middle register so they appear in both the left and right hand staves of the sheet music bootleg score (even though they only occur in one staff in the actual sheet music). The second difference is with handling enharmonic representations. While a notehead will only appear in exactly one staff line position in the sheet music bootleg score, it may appear in two (or more) positions in the MIDI bootleg score if the note is a black key on the piano. For example, the MIDI note number 61 could appear in the sheet music as a C-sharp or a D-flat, and these correspond to two different staff line positions. This difference can be resolved at a system level by generating two completely separate versions of the MIDI bootleg score: one in which all black notes are interpreted as sharps, and one in which all black notes are interpreted as flats. Both versions can be used to query the database, and the one with a higher match score is selected. By modifying the original form of the MIDI and sheet music bootleg scores in these two ways, the resulting representation can be used for large-scale, cross-modal retrieval (Tsai, 2020).

The bootleg score representation has two key benefits that make it an ideal choice for our applications. First, it is very fast to compute. Using only a CPU, it is possible to compute a bootleg score on a high-resolution scan of sheet music in under 1 second. This is much more efficient than

most OMR systems, since OMR is typically construed as an offline task. Second, the bootleg score feature extraction has no trainable weights and is therefore much less prone to overfitting. By simply tuning a few hyperparameters, the bootleg score has been successfully used in very different domains including scanned sheet music, synthetic sheet music, and cell phone pictures of sheet music (Yang et al., 2019; Tsai et al., 2020; Yang and Tsai, 2020; Shan and Tsai, 2020).

## 2.2 Database Construction

There are three steps in constructing the database: data preprocessing, computing bootleg score features, and creating N-gram reverse indices. These three steps are shown in the upper half of **Figure 1** and are explained in the following three paragraphs.

The first step is data preprocessing. To construct a sheet music database, the raw sheet music PDF is converted into a sequence of PNG images at 300 DPI. Because there is an extremely wide range of page sizes and resolutions across IMSLP, we also resize the PNG images to a fixed width of 2550 pixels. Note that this fixed resizing is separate from the page-specific resizing step during the bootleg score feature computation (see Section 2.1). To construct a MIDI database, no preprocessing is necessary.

The second step is to compute bootleg score features for each sheet music or MIDI file. The appropriate bootleg score variant from Section 2.1 should be selected based on the type of retrieval. Since each sheet music file has multiple pages, the bootleg scores from each page are concatenated into a single global bootleg score. In practice, each 62-dimensional (binary) column in the bootleg score is encoded as a single 64-bit integer, so that the bootleg score is simply a list of integers.

The third step is to create N-gram reverse indices. Consider a length  $L$  bootleg score represented by the sequence of integers  $x_1, x_2, \dots, x_L$ . For each offset  $i$  in  $1 \leq i \leq L$ , we construct an N-gram fingerprint  $(x_i, x_{i+1}, \dots, x_{i+N-1})$  for a fixed value of  $N$ . We store two pieces of information about each N-gram fingerprint in the reverse index: the file and the offset  $i$  where the N-gram fingerprint occurs. So, for a given N-gram fingerprint value, the reverse index will contain a list of instances in the database where the N-gram fingerprint occurs. We construct five separate N-gram reverse indices for  $N = 1, 2, 3, 4, 5$ . These reverse indices will enable efficient retrieval during the runtime search.

## 2.3 Search

The search at runtime consists of four steps, as shown in the bottom half of **Figure 1**.





The first step is to compute a bootleg score representation for the query. The same bootleg score variant should be used in the database construction phase and the runtime search. The output of this step is a bootleg score represented by a sequence of integers  $x_1, x_2, \dots, x_L$ .

The second step is to construct a sequence of dynamic N-gram fingerprints from  $x_1, x_2, \dots, x_L$ . For each offset  $i$  in  $1 \leq i \leq L$ , we construct a dynamic N-gram fingerprint  $(x_i, x_{i+1}, \dots, x_{i+N(i)-1})$ , where  $N(i)$  specifies the length of the N-gram

and may vary for different offsets  $i$ . So, for example, if  $N(1) = 1$ ,  $N(2) = 3$ , and  $N(3) = 2$ , the first three dynamic N-grams would be  $(x_1)$ ,  $(x_2, x_3, x_4)$ ,  $(x_3, x_4)$ .  $N(i)$  is computed by determining the smallest integer  $n$  that satisfies  $\text{count}((x_i, x_{i+1}, \dots, x_{i+n-1})) < \gamma$ , where  $\text{count}((x_i, x_{i+1}, \dots, x_{i+n-1}))$  indicates the total number of occurrences of the specific n-gram  $(x_i, x_{i+1}, \dots, x_{i+n-1})$  in the n-gram database and where  $\gamma$  is a hyperparameter specifying the maximum number of database matches we are willing to process for each reverse index lookup. In practice, we also impose a maximum cap  $n_{max}$  on the N-gram size due to memory constraints (i.e. storing all N-gram reverse indices in RAM). In our experiments, we use  $n_{max} = 4$  and  $\gamma = 1000$ , which was tuned to ensure a sub-second average runtime for searching all piano scores in IMSLP.

**Figure 3** shows an example of constructing a single dynamic N-gram at offset  $i$ . If we were to use the 1-gram at offset  $i$ ,  $(x_i)$ , we would need to process a large number of database matches, resulting in slow runtime. This might occur if the 1-gram is not very distinctive (e.g. the bootleg score column only contains a single notehead). On the other hand, if we used the 5-gram  $(x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4})$ , the fingerprint would be overly distinctive and prone to error since a single “bit” error anywhere in the 5-gram would cause the table lookup to fail. Instead, we select  $N = 3$  to ensure a balance between runtime and accuracy. Thus, the dynamic N-gram tries to find a balance between runtime and accuracy at each offset  $i$  by ensuring that every N-gram is discriminative enough to warrant a table lookup.

The third step is to assign each database item a match score using the histogram of offsets method. This method was originally proposed by Wang (2003) as an efficient way to search a database given a sequence of query fingerprints. For every database item, a histogram is constructed based on the relative offset between the query and the database item for matching N-gram fingerprints. For example, if the query N-gram fingerprint  $(x_i, x_{i+1}, \dots, x_{i+N(i)-1})$  matches an N-gram fingerprint  $(y_j, y_{j+1}, \dots, y_{j+N(i)-1})$  in database item  $k$ , a relative offset value of  $j-i$  will be added to the histogram for database item  $k$ . For true matches in the database, a sequence of matching fingerprints will result in a spike in the histogram at the true relative offset (i.e. where the query begins matching the database item’s bootleg score).

Fingerprint	Database	# Matches	$< \gamma$ ?
$(x_i)$		1274353	✗
$(x_i, x_{i+1})$		42353	✗
$(x_i, x_{i+1}, x_{i+2})$		968	✓
$(x_i, x_{i+1}, x_{i+2}, x_{i+3})$		27	

**Figure 3:** An example of constructing a dynamic N-gram. The length of the N-gram is chosen to balance runtime and retrieval accuracy.

Therefore, we can use the maximum histogram bin count as the match score for each database item.

The fourth step is post-processing the results. In our applications, the database items correspond to sheet music PDFs. Since a piece of music may have multiple PDF versions in IMSLP, we calculate the piece match score as the maximum score among its constituent PDFs. Finally, we sort the pieces by their match scores. The final output of the system is a ranked list of IMSLP pieces.

### 3 Case Study: Camera-Based Sheet Music Identification

The first of our two tasks is to identify piano sheet music based on a cell phone picture of a physical page of sheet music. In this section, we study the performance of the proposed system on this task.

#### 3.1 Experimental Setup

The experimental setup will be described in three parts: the cell phone images, the database, and the evaluation metric.

The cell phone query images come from the Sheet MIDI Retrieval dataset (Tsai et al., 2020). It consists of 2000 cell phone images of 200 piano pieces coming from 25 different composers. For each piece, exactly one sheet music PDF from IMSLP was printed onto physical paper. These physical pages were placed in various locations, and 10 cell phone pictures were taken of different sections throughout each piece. The cell phone pictures were taken on four different cell phone models, at varying levels of zoom (capturing between 1 and 5 lines of music), and in various lighting conditions (e.g. with and without flash). In our experiments, we use a train/test split of 400/1600, which is the original split used by Tsai et al. (2020).

The database consists of all piano sheet music scores in IMSLP. First, we scraped the IMSLP website to download all sheet music PDFs and accompanying metadata. This data is about 1.2 TB in size. For each piece, the metadata contains an instrument tag specifying the instrumentation of the piece. Next, we determined a set of instrument tags that correspond to solo piano pieces. We did this by searching for any instrument tags that contain the words “piano”, “clavier”, “harpsichord”, or “keyboard”, and then manually filtering the list to a set of 150 valid tags that correspond to solo piano pieces. Finally, we assemble a list of all non-manuscript PDFs that have a valid instrument tag. The resulting database contains 29,310 PDFs and 374,758 individual pages.

We evaluate performance along two axes: retrieval accuracy and runtime. Because there is only one correct matching piece in the database, we use mean reciprocal rank (MRR) as our metric for retrieval accuracy. MRR ranges between 0 and 1, where 1 indicates perfect performance. To characterize runtime, we report the mean and standard deviation of runtimes on all test queries. The measured runtimes include all data preprocessing steps, such as converting from JPG to PNG and image resizing. All experiments are run on a 2.1 GHz Intel Xeon CPU.



### 3.2 Results

We compare our proposed system to nine baseline systems. The first 4 baselines are image retrieval systems developed in the computer vision community: MAC (Radenovic et al., 2016), SPoC (Babenko and Lempitsky, 2015), GeM (Radenovic et al., 2018), and RMAC (Tolias et al., 2016). All systems use a pretrained CNN as a backbone, but employ different approaches to reduce the activations to a fixed-length feature representation. Because these four retrieval systems were trained on natural images and not sheet music, we expect them to perform poorly. Nonetheless, we include them in our set of baselines to establish that existing out-of-the-box image retrieval systems are inadequate solutions for our task. The last 5 baseline systems are fixed N-gram approaches for  $N = 1, 2, 3, 4, 5$ . These systems use fingerprints of a fixed size and only utilize one N-gram database. The fixed 1-gram system was recently proposed by Tsai (2020) and corresponds to the previous state-of-the-art on this task.

**Table 1** summarizes our experimental results for the sheet music identification task. We report retrieval accuracies for three different variants of the problem. The first variant (and default problem statement) is piece retrieval, where we measure how well the system can identify the correct piece. In this variant, we pool PDF match scores by piece and evaluate the list of ranked pieces. The second variant is version retrieval, where we measure how well the system can identify the PDF that is shown in the image. In this variant, we treat each PDF as a separate item, and only the exact same PDF is considered correct. The third variant is page retrieval, where we measure how well the system can identify the exact page that is shown in the cell phone image. In this variant, we remove alternate PDFs of the same piece from

**Table 1:** Comparing system performance on the camera-based piano sheet music identification task. The first three columns show the mean reciprocal rank on three retrieval tasks at varying levels of specificity. The last two columns show the system runtimes. The 1-gram system was proposed by Tsai (2020) and corresponds to the previous state-of-the-art.

System	Retrieval (MRR)			Runtime (s)	
	Piece	Version	Page	Avg	Std
MAC	.037	.023	.026	1.17	.12
SPoC	.003	.002	.002	1.14	.09
GeM	.025	.017	.017	1.18	.11
R-MAC	.036	.023	.024	0.96	.11
1-gram	.709	.560	.620	21.5	12.5
2-gram	.845	.525	.775	2.76	.36
3-gram	.808	.652	.723	1.99	.21
4-gram	.755	.617	.665	1.23	.13
5-gram	.608	.493	.599	1.07	.08
dynamic	.853	.692	.785	0.98	.12

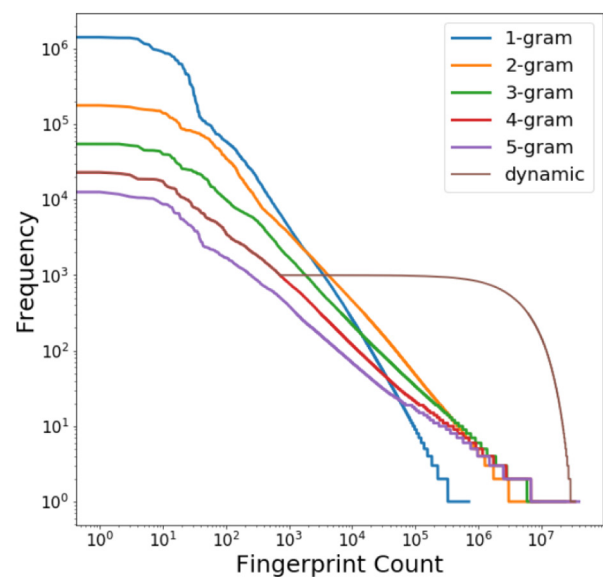
the database, treat each page as a separate database item, and pool page scores within each PDF.

There are 5 things to notice about **Table 1**. First, the image retrieval baselines perform terribly but are better than random guessing (0.001 MRR, standard deviation 0.004). This simply confirms that out-of-the-box image retrieval systems are not viable solutions. Second, the fixed N-gram baselines show a tradeoff between accuracy and runtime for different values of  $N$ . As  $N$  increases, the accuracy generally gets worse and the runtime improves. Third, the dynamic N-gram system has the best retrieval accuracy (0.853 MRR) and best runtime (0.98 seconds). This suggests that our design has successfully avoided a costly tradeoff between accuracy and runtime, and instead captures the best of both worlds. Fourth, the page retrieval results trail the piece retrieval results by around 0.07 MRR. This moderate drop in accuracy can be explained by repeated sections of music. Recapitulations or repeated motifs in the music may lead to correct piece identification but incorrect page identification. Fifth, the version retrieval results trail the piece retrieval results by 0.1-0.3 MRR. The fact that this is a big drop is a good thing: it means that the system confuses different PDFs of the same piece, meaning that it is largely invariant to differences between printed editions.

### 3.3 Analysis

We conduct four different analyses to provide deeper insight into our system’s performance on the sheet music identification task.

The first analysis is to characterize the hashing behavior of the proposed system. **Figure 4** shows the frequency of unique N-gram fingerprints (sorted from most frequent to least frequent) in the fixed N-gram and dynamic N-gram databases. Note that both axes are on a log scale.

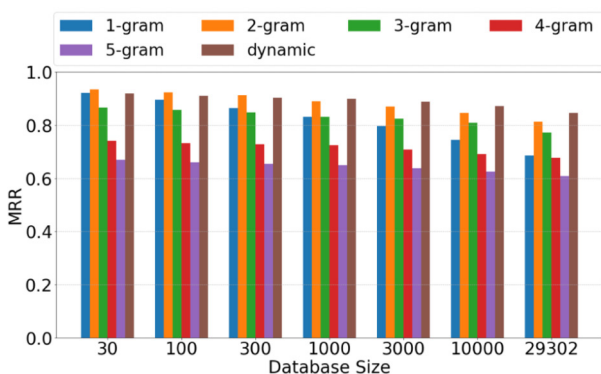


**Figure 4:** Frequency of N-gram fingerprints in the database, where fingerprints have been sorted from most frequent (left) to least frequent (right). Note that both axes are on a log scale.

There are three things we can notice about **Figure 4**. First, there are a small number of very frequently occurring fingerprints. The 1-gram system is the most extreme example of this: there are seven fingerprints that occur more than a million times. Unsurprisingly, the extremely common 1-gram fingerprints tend to be bootleg score columns that contain only a single notehead. Because these fingerprints occur so commonly in music, the system needs to process a very large number of matches in the database, resulting in a slow runtime. Second, there are a very large number of rarely occurring fingerprints. For all systems, the vast majority of fingerprint values occur less than 10 times in the whole database. These fingerprints are overly specific and error-prone, leading to lower retrieval accuracy. They also require a lot of RAM when hosting the database in memory. Third, the dynamic N-gram system has the most desirable frequency distribution of all systems. It has the benefits of a long N-gram – avoiding fingerprints that occur too frequently – but it mitigates the problem of having overly distinctive fingerprints by augmenting the database with 1-grams, 2-grams, and 3-grams that occur less than  $\gamma = 1000$  times.

The second analysis is to characterize the effect of database size. We systematically varied the database size by randomly sampling a desired number of items from the full database. **Figure 5** shows the performance of the fixed N-gram and dynamic N-gram systems for different database sizes. Each histogram bar represents the average performance of 10 different database samplings on the test set.

There are two things to notice about **Figure 5**. First, the dynamic N-gram system scales well. There is an approximately linear decrease in MRR as the database size increases exponentially over several orders of magnitude. For reference, when the database size increases from 10,000 to 30,000, the MRR decreases by about 0.02. Second, the dynamic N-gram system has the highest retrieval accuracy for large databases and is comparable to the best-performing system for small databases. For database sizes smaller than 1000, the fixed 2-gram system has slightly higher retrieval accuracy. Based on these results, the dynamic N-gram system is not worth the effort for small-scale retrieval tasks, but its benefits become clear for large-scale retrieval.



**Figure 5:** Effect of database size on system performance for the sheet music identification task. The largest database size corresponds to the full IMSLP piano database.

The third analysis is to characterize how well the system copes with variations between different printed editions of the same piece. To answer this question, we ran a controlled experiment in which we removed the exact same sheet music version from the database, so that only alternate versions of the sheet music exist in the database. Because some queries only had one sheet music version in IMSLP, this controlled experiment was run on a reduced subset of 930 test queries.

**Table 2** shows the results of this controlled experiment. When the exact matching sheet music version is removed from the database, we see a slight drop in performance of about 0.02-0.06 MRR. These results suggest that the system is relatively robust to variations between different printed editions.

The fourth analysis is to characterize how the value of  $\gamma$  affects system performance. When varying the value of  $\gamma$  from 1k to 5k to 10k to 20k, the MRR slightly decreased from .864 to .865 to .860 to .853, and the average runtime decreased from 1.67s to 1.53s to 1.21s to 0.98s. Therefore, changing the value of  $\gamma$  allows us to trade off a slight decrease in retrieval accuracy for a significant speedup in runtime.

#### 4 Case Study: Large-scale MIDI-Sheet Retrieval

The second of our two tasks is large-scale MIDI-sheet image retrieval. In this section, we describe two sets of controlled experiments (Sections 4.1 and 4.2) as well as an application of our approach to link LMD and IMSLP (Section 4.3).

##### 4.1 Comparison to Previous Approaches

The goal of the first set of controlled experiments is to compare our proposed method to previously proposed approaches on the MIDI-sheet image retrieval task. We used the MIDI-sheet image retrieval benchmark proposed

**Table 2:** Comparing performance on the sheet music identification task under two conditions: when the exact same printed edition exists in the database (left column) and when only an alternate edition of the same piece exists (right column).

System	Piece Retrieval (MRR)	
	Same Version	Alternate Version
MAC	.037	.043
SPoC	.003	.004
GeM	.025	.029
R-MAC	.036	.039
1-gram	.709	.659
2-gram	.845	.784
3-gram	.808	.767
4-gram	.755	.722
5-gram	.688	.668
dynamic	.853	.812

by Tsai et al. (2020), where a cell phone picture of a few lines of sheet music is used to retrieve a specific temporal passage in a MIDI file. We adapted this benchmark to study large-scale retrieval by padding the database with additional MIDI files from LMD.

**Table 3** shows the results of these experiments. The table is divided into four sections according to the size of the database: 1, 200, 2000, and 10000. The top section shows the results on the original benchmark, where the MIDI file is assumed to be known (i.e. the database size is 1). These results are copied directly from Tsai et al (2020), and the reader is referred to the original paper for

**Table 3:** Comparison of system performance on MIDI-sheet image retrieval. Columns indicate the database size (DB), precision (P) and recall (R) and F-measure (F) for passage-level retrieval, mean reciprocal rank (MRR) for file-level retrieval, and average runtime.

System	DB	P	R	F	MRR	$T_{avg}$
Random	1	.16	.16	.16	–	0.0s
SharpEye	1	.43	.08	.13	–	–
PhotoScore	1	.64	.62	.63	–	–
RetinaNet	1	.52	.26	.35	–	11.7s
Dorfer 2018c	1	.69	.28	.40	–	17.5s
Faster R-CNN	1	.84	.87	.85	–	49.9s
DWD	1	.91	.87	.89	–	213s
BS-DTW	1	.89	.89	.89	–	0.90s
BS-DTW	200	.90	.87	.88	.92	10.9s
1-gram	200	.58	.59	.59	.69	1.28s
2-gram	200	.70	.73	.71	.84	1.00s
3-gram	200	.60	.72	.66	.79	1.00s
4-gram	200	.41	.57	.48	.63	1.00s
5-gram	200	.36	.49	.42	.53	1.00s
dynamic	200	.74	.77	.75	.86	0.98s
BS-DTW	2k	.89	.83	.86	.88	167s
1-gram	2k	.48	.51	.49	.59	2.28s
2-gram	2k	.63	.69	.66	.78	1.10s
3-gram	2k	.49	.63	.55	.69	1.09s
4-gram	2k	.39	.55	.46	.60	1.04s
5-gram	2k	.35	.47	.40	.51	1.05s
dynamic	2k	.66	.72	.69	.80	.98s
BS-DTW	10k	.83	.77	.80	.83	458s
1-gram	10k	.28	.34	.31	.43	6.78s
2-gram	10k	.43	.58	.49	.68	1.45s
3-gram	10k	.40	.53	.46	.56	1.10s
4-gram	10k	.34	.48	.40	.50	1.04s
5-gram	10k	.28	.42	.34	.46	1.00s
dynamic	10k	.55	.66	.60	.73	.99s

information about the baseline systems. The remaining three sections compare the best-performing system on the original benchmark (BS-DTW) to the fixed N-gram and dynamic N-gram systems for increasing database sizes. We evaluate both file-level retrieval (MRR) and passage-level retrieval (precision, recall, F-measure).

There are three things to notice about **Table 3**. First, only one of the systems evaluated on the original benchmark (i.e. database size 1) has acceptable retrieval accuracy *and* runtime: the BS-DTW system proposed by Tsai et al. (2020). This system computes bootleg score features on the sheet music and then uses subsequence DTW to find the best matching subsequence in the MIDI bootleg score. Second, the N-gram fingerprinting methods (i.e. the fixed N-gram and dynamic N-gram) are a scalable solution, whereas the BS-DTW approach is not. We can see that the runtime for BS-DTW increases approximately linearly with the database size, resulting in exorbitantly long run times. In contrast, the N-gram fingerprinting methods have worse retrieval accuracy but scale very well with database size. Third, the dynamic N-gram method is the best approach for large-scale retrieval. We can see that it has a lower runtime and higher retrieval accuracy than the fixed N-gram approaches across all database sizes.

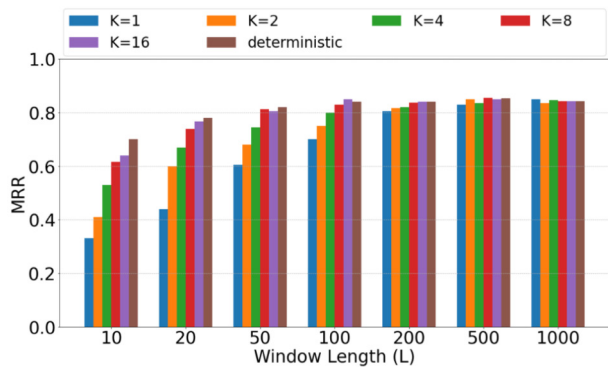
#### 4.2 Handling Jumps & Repeats

The goal of the second set of controlled experiments is to determine an effective strategy for handling discontinuities due to jumps or repeats. Though the pieces in the Sheet MIDI Retrieval dataset do not contain any discontinuities, it is necessary to answer this question in anticipation of applying our system to link LMD and IMSLP, where repeats and jumps may occur in the data. Note that our proposed system relies on finding a *sequence* of matching N-grams, so a single discontinuity could potentially shorten the sequence of matching N-grams in half.

One way to handle repeats and jumps with our proposed system is to break the MIDI file into fragments, perform the database search on each of the fragments separately, and average the database match scores from all fragments. This approach ensures that a discontinuity will only affect a single fragment, rather than the entire query. We experiment with two different shingling strategies: (a) deterministic shingling, in which windows of length  $L$  are taken throughout the MIDI bootleg score with a hop size of  $\frac{L}{2}$ , and (b) probabilistic shingling, in which we randomly sample  $K$  windows of length  $L$ , where  $K$  and  $L$  are hyperparameters.

**Figure 6** compares the effect of shingling strategy and hyperparameter settings for the dynamic N-gram system. Here, we use the 160 test MIDI files from the Sheet MIDI Retrieval dataset as queries, and use all solo piano sheet music images in IMSLP for the database.

There are 4 things to notice about **Figure 6**. First, longer fragment lengths (i.e. higher  $L$ ) and more fragments (i.e. higher  $K$ ) yield better results, as we might expect. Second, the benefit of a longer fragment length seems to saturate once the fragment length is in the range 500–1000. Third, the benefit of having more fragments becomes less and less as the fragment length increases. Indeed,



**Figure 6:** Comparing different strategies for handling repeats and jumps in MIDI-sheet image retrieval. The MIDI query is broken up into fragments of length  $L$ , either through deterministic shingling or a fixed number ( $K$ ) of randomly sampled windows.

for longer lengths of 500 or 1000, the effect of having more fragments is marginal. Fourth, the deterministic and probabilistic shingling strategies perform approximately the same for longer fragment lengths.

Based on the above observations, we select the probabilistic shingling with  $K = 8$  fragments and  $L = 500$  fragment length as the optimal strategy. This setting achieved the highest retrieval accuracy of all configurations and had much lower runtimes (average 6.4 seconds) than the deterministic shingling approach (average 12.9 seconds).

### 4.3 Linking LMD and IMSLP

As a practical application of our system, we find correspondences between LMD (Raffel, 2016) and IMSLP. This is a large-scale MIDI-sheet image retrieval problem in which there are no ground truth annotations. Our goal here is not to characterize retrieval accuracy, but rather to identify true matches for the purpose of constructing a rich, multimodal dataset that contains real (non-synthetic) sheet music images and corresponding MIDI data. Below, we describe the data preprocessing, methodology, and summary of the true matches found.

The data consists of all solo piano files in LMD and IMSLP. The IMSLP piano database is the same as in Section 3.1 and contains 29,310 sheet music PDFs. The Lakh MIDI dataset is filtered by removing any MIDI file that contains a note event from a non-piano instrument. This reduces the LMD data from 176,581 to 27,077 files.

Our methodology for identifying LMD-IMSLP matches consists of four steps. First, we use every LMD piano MIDI file as a query to search the IMSLP database. For each MIDI file, we store a ranked list of all IMSLP PDFs based on their match score. Second, we use every IMSLP piano sheet music PDF as a query to search the LMD database. For each sheet music PDF, we store a ranked list of all LMD piano MIDI files based on their match score. Third, we determine a list of (MIDI file, IMSLP PDF) pairs that are in the top  $M$  results in both directions of the search. This provides strong circumstantial evidence that the MIDI

and PDF may be a true match. We experimented with different values of  $M$  and found that the number of (MIDI, PDF) pairs satisfying the bidirectional top  $M$  condition approximately saturates for values of  $M$  between 10 and 12. We used  $M = 11$ , which results in 8,521 (MIDI, PDF) pairs. Fourth, we manually verify the (MIDI, PDF) pairs to determine a set of true matches. In order to reduce the amount of manual work, we merge multiple PDFs from the same piece and select one representative PDF to verify. After collapsing in this way, there are a total of 4,178 pairs to verify manually.

Our end result is a text file containing a list of verified, matching (LMD MIDI file, IMSLP PDF) pairs. Each line of the text file contains an LMD filename and a unique ID number that specifies an IMSLP PDF file. For convenience, we also include the composer and piece name on each line. Note that there are often multiple PDFs for a piece (i.e. multiple sheet music versions), some of which may match the MIDI file and some of which may not. This occurs because of the structure of IMSLP data: a “piece” in IMSLP may actually be a collection of pieces (e.g. Chopin Etudes Op. 25), and the PDFs may contain the entire collection or individual elements in the collection (e.g. only Etude Op. 25 No. 1). We only list one PDF per IMSLP piece, and the listed PDF is guaranteed to contain the notes played in the MIDI file.

**Table 4** summarizes the verified LMD-IMSLP matches that were found. Since we only verified one representative PDF per piece, this table should be interpreted as a summary of (LMD MIDI file, IMSLP piece) matches. The matches are grouped by composer and sorted by composer frequency. The second column indicates the number of LMD MIDI files for which a true match was found in IMSLP, and the third column indicates the number of unique IMSLP pieces for which a true match was found in LMD. In total, there were 1,433 matching (MIDI, PDF) pairs spanning 43 different composers and 349 different IMSLP pieces. The composers with the most matching MIDI files are not surprising: Bach, Chopin, Beethoven, Mozart, Liszt, Debussy, and Brahms. We release our list of LMD-IMSLP matches along with all code in an open-source repository.

**Table 4:** Summary of verified matches between the Lakh MIDI dataset and IMSLP.

Composer	# Lakh MIDI	# IMSLP Pieces
Bach	356	94
Chopin	291	49
Beethoven	186	50
Mozart	116	30
Liszt	77	32
Debussy	48	9
Brahms	41	10
Other	318	75
Total	1433	349



## 5 Conclusion

We present a method for large-scale retrieval of piano sheet music images. Our method combines bootleg score features with a novel dynamic N-gram fingerprinting method that ensures that every fingerprint is discriminative enough to warrant a table lookup. We study the performance of our proposed method on two different tasks: camera-based sheet music identification and large-scale MIDI-sheet image retrieval. Using all piano sheet music images in the IMSLP database, the dynamic N-gram method achieves high retrieval accuracy ( $>0.8$  mean reciprocal rank) and sub-second runtimes. As a practical application, we use our proposed method to find matches between the Lakh MIDI dataset and the IMSLP database. For future work, we plan to expand this approach to include non-piano sheet music.

## 6 Reproducibility

The data and code for reproducing our results can be found at <https://github.com/HMC-MIR/SheetMusicID>.

## Note

- <sup>1</sup> This article is a journal extension of an earlier conference paper (Yang and Tsai, 2020) that only examines the sheet-sheet retrieval task. This article provides more in-depth analysis of the sheet identification problem (Section 3), and then extends the same approach to the MIDI-sheet image retrieval task (Section 4).

## Acknowledgements

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. Large-scale computations were performed with XSEDE Bridges at the Pittsburgh Supercomputing Center through allocation TG-IRI190019.

## Competing Interests

The authors have no competing interests to declare.

## References

- Achankunju, S. P.** (2018). Music search engine from noisy OMR data. In *International Workshop on Reading Music Systems*, pages 23–24.
- Babenko, A., and Lempitsky, V.** (2015). Aggregating local deep features for image retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1269–1297. DOI: <https://doi.org/10.1109/ICCV.2015.150>
- Calvo-Zaragoza, J., Hajič, J., Jr. and Pacha, A.** (2020). Understanding optical music recognition. *ACM Computing Surveys (CSUR)*, 53(4): 1–35. DOI: <https://doi.org/10.1145/3397499>
- Calvo-Zaragoza, J., Toselli, A. H., and Vidal, E.** (2018). Probabilistic music-symbol spotting in handwritten scores. In *IEEE International Conference on Frontiers in Handwriting Recognition*, pages 558–563. DOI: <https://doi.org/10.1109/ICFHR-2018.2018.00103>
- Damm, D., Fremerey, C., Kurth, F., Müller, M., and Clausen, M.** (2008). Multimodal presentation and browsing of music. In *Proceedings of the International Conference on Multimodal Interfaces*, pages 205–208. DOI: <https://doi.org/10.1145/1452392.1452436>
- Dorfer, M., Arzt, A., and Widmer, G.** (2016). Towards score following in sheet music images. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 789–795.
- Dorfer, M., Arzt, A., and Widmer, G.** (2017). Learning audio-sheet music correspondences for score identification and offline alignment. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 115–122.
- Dorfer, M., Hajič, J., Jr. Arzt, A., Frostel, H., and Widmer, G.** (2018a). Learning audio-sheet music correspondences for cross-modal retrieval and piece identification. *Transactions of the International Society for Music Information Retrieval*, 1(1): 22–33. DOI: <https://doi.org/10.5334/tismir.12>
- Dorfer, M., Henkel, F., and Widmer, G.** (2018b). Learning to listen, read, and follow: Score following as a reinforcement learning game. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 784–791.
- Dorfer, M., Schlüter, J., Vall, A., Korzeniowski, F., and Widmer, G.** (2018c). End-to-end crossmodality retrieval with CCA projections and pairwise ranking loss. *International Journal of Multimedia Information Retrieval*, 7(2): 117–128. DOI: <https://doi.org/10.1007/s13735-018-0151-5>
- Fremerey, C., Clausen, M., Ewert, S., and Müller, M.** (2009). Sheet music-audio identification. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 645–650.
- Fremerey, C., Müller, M., and Clausen, M.** (2010). Handling repeats and jumps in score-performance synchronization. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 243–248.
- Fremerey, C., Müller, M., Kurth, F., and Clausen, M.** (2008). Automatic mapping of scanned sheet music to audio recordings. In *Proceedings of the International Conference on Music Information Retrieval*, pages 413–418.
- Hajič, J., Kolárová, M., Pacha, A., and Calvo-Zaragoza, J.** (2018). How current optical music recognition systems are becoming useful for digital libraries. In *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, pages 57–61. DOI: <https://doi.org/10.1145/3273024.3273034>
- Henkel, F., Balke, S., Dorfer, M., and Widmer, G.** (2019). Score following as a multimodal reinforcement learning problem. *Transactions of the International Society for Music Information Retrieval*, 2(1): 67–81. DOI: <https://doi.org/10.5334/tismir.31>
- Izmirli, Ö., and Sharma, G.** (2012). Bridging printed music and audio through alignment using a midlevel score representation. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 61–66.

- Kurth, F., Müller, M., Fremerey, C., Chang, Y.-H., and Clausen, M.** (2007). Automated synchronization of scanned sheet music with audio recordings. In *Proceedings of the International Conference on Music Information Retrieval*, pages 261–266.
- Malik, R., Roy, P. P., Pal, U., and Kimura, F.** (2013). Handwritten musical document retrieval using music-score spotting. In *IEEE International Conference on Document Analysis and Recognition*, pages 832–836. DOI: <https://doi.org/10.1109/ICDAR.2013.170>
- Radenovic, F., Toliás, G., and Chum, O.** (2016). CNN image retrieval learns from BoW: Unsupervised finetuning with hard examples. In *Proceedings of the European Conference on Computer Vision*, pages 3–20. DOI: [https://doi.org/10.1007/978-3-319-46448-0\\_1](https://doi.org/10.1007/978-3-319-46448-0_1)
- Radenovic, F., Toliás, G., and Chum, O.** (2018). Finetuning CNN image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7): 1655–1668. DOI: <https://doi.org/10.1109/TPAMI.2018.2846566>
- Raffel, C.** (2016). *Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching*. PhD thesis, Columbia University. DOI: <https://doi.org/10.1109/ICASSP.2016.7471641>
- Shan, M., and Tsai, T.** (2020). Improved handling of repeats and jumps in audio-sheet image synchronization. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 62–69.
- Tanprasert, T., Jenrungrot, T., Müller, M., and Tsai, T.** (2019). MIDI-sheet music alignment using bootleg score synthesis. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 91–98.
- Thomas, V., Fremerey, C., Müller, M., and Clausen, M.** (2016). Linking sheet music and audio – challenges and new approaches. In Müller, M., Goto, M., and Schedl, M., editors, *Dagstuhl Follow-Ups*, 3: 1–22. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany. DOI: <https://doi.org/10.4230/dfu.vol3.11041.1>
- Thompson, J., Hankinson, A., and Fujinaga, I.** (2011). Searching the liber usualis: Using CouchDB and ElasticSearch to query graphical music documents. In *Proceedings of the International Society for Music Information Retrieval Conference*.
- Toliás, G., Sicre, R., and Jégou, H.** (2016). Particular object retrieval with integral max-pooling of CNN activations. In *Proceedings of the International Conference on Learning Representations*.
- Tsai, T.** (2020). Towards linking the Lakh and IMSLP datasets. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 546–550. DOI: <https://doi.org/10.1109/ICASSP40776.2020.9053815>
- Tsai, T., Yang, D., Shan, M., Tanprasert, T., and Jenrungrot, T.** (2020). Using cell phone pictures of sheet music to retrieve MIDI passages. *IEEE Transactions on Multimedia*. DOI: <https://doi.org/10.1109/TMM.2020.2973831>
- Wang, A.** (2003). An industrial strength audio search algorithm. In *Proceedings of the International Conference on Music Information Retrieval*, pages 7–13.
- Yang, D., Tanprasert, T., Jenrungrot, T., Shan, M., and Tsai, T.** (2019). MIDI passage retrieval using cell phone pictures of sheet music. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 916–923.
- Yang, D., and Tsai, T.** (2020). Camera-based piano sheet music identification. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 481–488.

**How to cite this article:** Yang, D., & Tsai, T. J. (2021). Piano Sheet Music Identification Using Dynamic N-gram Fingerprinting. *Transactions of the International Society for Music Information Retrieval*, 4(1), pp. 42–51. DOI: <https://doi.org/10.5334/tismir.70>

**Submitted:** 29 August 2020

**Accepted:** 23 January 2021

**Published:** 01 April 2021

**Copyright:** © 2021 The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CC-BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. See <http://creativecommons.org/licenses/by/4.0/>.

][

*Transactions of the International Society for Music Information Retrieval* is a peer-reviewed open access journal published by Ubiquity Press.

**OPEN ACCESS** 