PIANO SHEET MUSIC IDENTIFICATION USING MARKETPLACE FINGERPRINTING

Kevin Ji **Daniel Yang** **TJ Tsai**
Harvey Mudd College
kji,dhyang,ttsai@hmc.edu

ABSTRACT

This paper studies the problem of identifying piano sheet music based on a cell phone image of all or part of a physical page. We re-examine current best practices for large-scale sheet music retrieval through an economics perspective. In our analogy, the runtime search is like a consumer shopping in a store. The items on the shelves correspond to fingerprints, and purchasing an item corresponds to doing a fingerprint lookup in the database. From this perspective, we show that previous approaches are extremely inefficient marketplaces in which the consumer has very few choices and adopts an irrational buying strategy. The main contribution of this work is to propose a novel fingerprinting scheme called marketplace fingerprinting. This approach redesigns the system to be an efficient marketplace in which the consumer has many options and adopts a rational buying strategy that explicitly considers the cost and expected utility of each item. We also show that deciding which fingerprints to include in the database poses a type of minimax problem in which the store and the consumer have competing interests. On experiments using all solo piano sheet music images in IMSLP as a searchable database, we show that marketplace fingerprinting substantially outperforms previous approaches and achieves a mean reciprocal rank of 0.905 with sub-second average runtime.

1. INTRODUCTION

This paper tackles the problem of identifying piano sheet music based on a cell phone picture of all or part of a physical page. This is the camera-based sheet music identification task. Such a system could be used to conveniently retrieve Youtube videos of relevant performances, compare different scores for a particular passage of music, or – more generally – explore representations of sheet music that are useful for alignment and retrieval.

Previous work on retrieval tasks involving sheet music fall into three groups. The first group of related works study audio–sheet alignment and retrieval. The earliest works used Optical Music Recognition (OMR) systems to convert sheet music to a symbolic format like MIDI, extracted chroma features from both MIDI and audio, and then performed alignment or retrieval using dynamic time warping (DTW). This approach has been used to synchronize audio and sheet music [1–4] and perform audio–sheet retrieval [5, 6]. More recent works have explored the use of convolutional neural networks (CNNs) to project both sheet music and audio into an embedding space where similarity can be computed directly. This approach has been applied to various forms of audio–sheet music alignment [7–11] and audio–sheet retrieval [7, 12, 13]. The second group of related works study symbolic–sheet retrieval. Several recent works studying MIDI-sheet retrieval have used the bootleg score feature representation [14], which encodes the positions of noteheads relative to staff lines. This representation has been used for MIDI–sheet passage retrieval [14, 15] and to find matches between the Lakh MIDI dataset and IMSLP sheet music [16, 17]. Other approaches find matches with symbolic queries by performing OMR or object recognition on the sheet music, and then doing an n-gram lookup [18,19], string matching [20], or keyword spotting [21]. The third group of related works — and the works that are most directly relevant to our present study — explore sheet–sheet retrieval. Hajic et al. [22] use OMR to convert sheet music to MIDI and then use DTW on the pitch sequences. Waloschek et al. [23] use a CNN to project entire measures into an embedding space to align different sheet music editions of the same piece. Yang and Tsai [24] propose a dynamic n-gram fingerprint derived from bootleg score features to identify sheet music based on cell phone images of physical pages.

This paper re-examines current practices for large-scale sheet–sheet retrieval from an economics perspective. This perspective makes clear what the weaknesses of current approaches are and suggests ways to improve them. We will focus our analysis on the dynamic n-gram approach [24], since it is the largest-scale study (using all solo piano sheet music images in IMSLP) and achieves robust sub-second retrieval (0.85 MRR). The dynamic n-gram approach has three steps: (1) it extracts a sequence of bootleg score features $x_1, x_2, \ldots, x_L$ from the query image, (2) it constructs either a 1-gram (x_i), 2-gram (x_i, x_{i+1}), 3-gram (x_i, x_{i+1}, x_{i+2}), or 4-gram $(x_i, x_{i+1}, x_{i+2}, x_{i+3})$ fingerprint at each offset $i = 1, 2, \ldots, L$, where the size of the n-gram at offset i is selected at runtime to ensure that the number of fingerprint matches in the database is below a certain threshold, and (3) the n-gram fingerprints are used

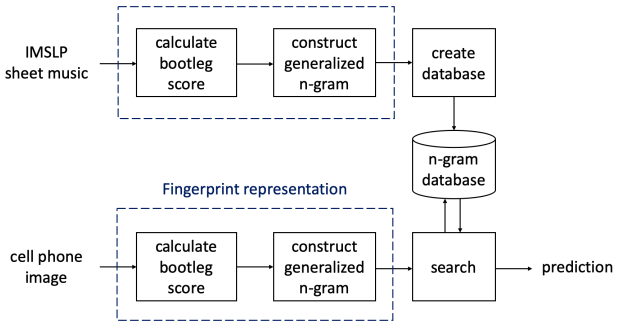

Figure 1. Overview of the marketplace fingerprinting approach. The upper half describes the offline process of constructing the database, and the lower half describes the online process of performing a real-time search.

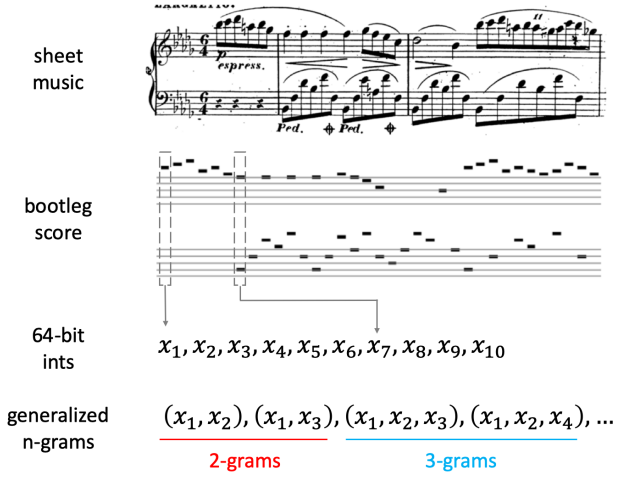

Figure 2. Description of the generalized n-gram fingerprint representation. The sheet music is first converted to a bootleg score, each column of the bootleg score is represented as a 64-bit integer, and n-grams are constructed from various groupings of integers.

with an inverted file index to identify the database item containing the most fingerprint matches.

Consider the following analogy. Imagine that the runtime search is like a consumer shopping in a store. Every offset $i = 1, 2, \ldots, L$ in the bootleg score sequence is like an aisle in the store. The 1-gram, 2-gram, 3-gram, and 4-gram fingerprints are like items on the shelves. In the dynamic n-gram approach, the store has exactly four items on every aisle, and the consumer always purchases the most expensive item in each aisle that is below a maximum acceptable price. From an economics perspective, this setup is horrible for the consumer, and the consumer's purchasing strategy is irrational.

The main contribution of this paper is to propose a novel fingerprinting scheme called marketplace fingerprinting. Marketplace fingerprinting is the result of redesigning the system above using principles of economics to produce a more efficient marketplace. One of the key principles in this approach is that more options and choices are good for the consumer. We generalize the notion of an n-gram in order to produce a much larger set of n-gram types, which corresponds to offering many more items in each aisle. Furthermore, we adopt a much more rational purchasing strategy in which the consumer explicitly considers the cost and the expected utility of each item, purchases the items with the highest utility-to-cost ratio, and is allowed to purchase multiple items in each aisle as long as they stay under budget. We also show that the database design problem (i.e. which n-gram fingerprints to include in the database) presents a type of minimax problem in which the store and the consumer have competing interests. On experiments involving all solo piano sheet music images in IMSLP, we show that the marketplace fingerprinting method substantially improves retrieval accuracy compared to the dynamic n-gram method. [1]

2. SYSTEM DESCRIPTION

Figure 1 shows an overview of our proposed approach. We will describe the system in three parts: computing the fingerprint representation (Section 2.1), creating the database (Section 2.2), and searching the database (Section 2.3). The bootleg score representation is adopted from previous work, but the generalized n-gram, database construction, and search mechanism are novel contributions.

2.1 Fingerprint Representation

Figure 2 shows how n-gram fingerprints are computed. This process is described in the next two paragraphs.

The first step is to extract bootleg score features. The bootleg score is a mid-level feature representation that encodes the positions of filled noteheads relative to staff lines in piano sheet music [14]. The bootleg score itself is a $62 \times N$ binary matrix, where 62 indicates the total number of distinct staff line positions in both the left and right hand staves and N indicates the number of grouped note events in the sheet music (e.g. a chord containing four notes played simultaneously would constitute a single grouped note event). Note that this representation throws away a significant amount of information such as key signature, time signature, accidentals, note duration, octave markings, clef changes, and non-filled noteheads. Nonetheless, it has been successfully used in several applications involving sheet music, including sheet music identification [16, 24], sheet–MIDI retrieval [14, 15, 17], and sheet–audio alignment [25, 26]. We represent each column (containing 62 bits) as a 64-bit integer, so that the bootleg score is encoded as a sequence of integers $x_1, x_2, \ldots, x_N$.

The second step is to construct generalized n-grams. The concept of n-grams comes from linguistics [27], where the frequency of word sequences in a large corpus was historically used for language modeling. Many previous works have likewise used n-grams for language modeling with music data (e.g. [28, 29]). Here, we use a generalization of n-grams that is specifically useful for indexing and retrieval. When constructing generalized n-grams at

[1] Code can be found at `https://github.com/HMC-MIR/ImprovedSheetID`.

offset i for a bootleg score sequence $x_1, x_2, \ldots, x_N$, we consider any combination of n elements that satisfies two conditions: (1) the leftmost element must be x_i, and (2) the elements must be selected from a fixed context window of length C. For example, when $C = 4$ and we consider up to 3-grams, there is one 1-gram $\{(x_i)\}$, three 2-grams $\{(x_i, x_{i+1}), (x_i, x_{i+2}), (x_i, x_{i+3})\}$, and three 3-grams $\{(x_i, x_{i+1}, x_{i+2}), (x_i, x_{i+1}, x_{i+3}), (x_i, x_{i+2}, x_{i+3})\}$, resulting in a total of 7 generalized n-grams. In our experiments, we consider up to 3-grams with $C = 6$, which results in a total of $T = 16$ different generalized n-gram types at each offset i. We will denote the generalized n-grams at offset i as $y_{i1}, y_{i2}, \ldots, y_{iT}$.

2.2 Database Construction

If we had an infinite amount of RAM available, the database construction problem would be trivial: we would simply include all of the generalized n-grams in the database. However, because the IMSLP dataset is large and we have many different types of n-grams, the total amount of memory required to store everything in the database quickly becomes exorbitant. This forces us to be strategic in choosing which n-grams to include in the database and which to exclude.

The database is simply a reverse index in which the key is the n-gram fingerprint and the value is a list of all instances in the IMSLP dataset where the n-gram occurs. This list consists of (PDF, offset) tuples that specify the IMSLP PDF and bootleg score offset where the fingerprint occurs. Note that the n-gram *type* and n-gram *value* must both match in order to be included in the reverse index (i.e. a (x_i, x_{i+1}) 2-gram and (x_i, x_{i+2}) 2-gram will never collide).

We approach this problem from the perspective of a store manager who has a limited amount of shelf space and wants to fill the shelves with products that maximize some utility function. There are two different types of resources in this scenario. The first resource is shelf space, which in our case corresponds to the amount of RAM available. In our experiments, we work with a machine that has 128 GB of RAM. The second resource of interest is the utility function. Our utility function is the expected number of match points that will be added to the true matching PDF at runtime. Consider a single n-gram fingerprint y_{ij} at offset i in a query. If y_{ij} is not in the database or if the bootleg score representation has errors (e.g. it fails to detect a notehead in the sheet music), the true matching item in the database will accumulate 0 match points. On the other hand, if y_{ij} is included in the database and the bootleg score computation is correct, the true matching item in the database will accumulate 1 match point. Therefore, the expected (added) utility for including y_{ij} in the database is the probability that its constituent bootleg score representation is correct. We estimated this probability of correctness as a fixed constant for each of the 16 generalized n-gram types based on the training data, as shown in Figure 3. Unsurprisingly, the 1-gram has the highest probability of correctness, and the 3-grams had the lowest (4-grams were even lower). Using

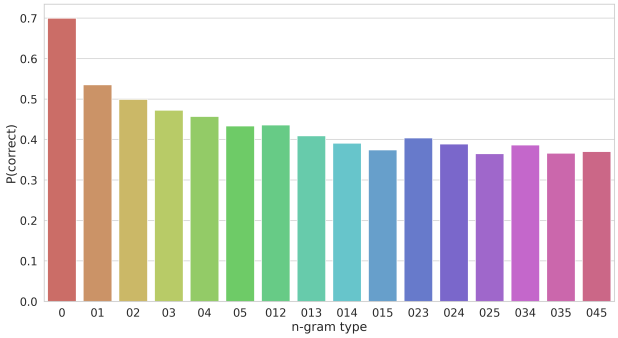

Figure 3. Probability of correctness for various generalized n-gram types, as estimated on training data. Each bar indicates the fraction of n-grams of that type in the training queries whose underlying bootleg score representation matched the database.

this approach, the total utility of an n-gram fingerprint y_{ij} is $U(y_{ij}) = N(y_{ij})P_{correct}(j)$, where $N(y_{ij})$ indicates the total number of times the fingerprint occurs in the IMSLP data and where $P_{correct}(j)$ indicates the probability that the underlying bootleg score representation is correct (as shown in Figure 3).

At this point, we *could* simply sort all of the unique n-gram fingerprints by their utility (largest to smallest), and then add them to the database in order until the memory is used up. However, this ignores the fact that the store manager (i.e. the database construction problem) and the consumer (i.e. the runtime search problem) have competing interests. Note that the utility function rewards fingerprints that occur very frequently. [2] For example, the n-gram with the highest utility is a 1-gram (i.e. single bootleg score column) with a single notehead present, which occurs more than a million times in the database. This fingerprint offers the store manager the highest return, but is an awful proposition for the consumer. From the search perspective, the primary constraining resource is runtime, not memory. Given two different n-grams to choose from — one that occurs 1 million times in the database and one that occurs once in the database — the latter is far more desirable at runtime: both offer the possibility of adding 1 match point to the true matching item in the database, but one requires processing 1 million matches and the other only requires processing 1 match. Therefore, the database construction problem is a type of minimax problem, in which the store manager wants to offer the most frequently occurring fingerprints but the consumer wants to purchase the least frequently occurring fingerprints.

Based on these considerations, the database is constructed in the following way. All of the unique n-gram fingerprints in IMSLP are sorted by their utility value $U(y_{ij})$ from largest to smallest. We discard any fingerprints that occur more than $\gamma = 10,000$ times in the database. This value of γ is set conservatively to ensure that all n-gram

[2] Because there is a memory overhead for adding a new entry to the reverse index, it is possible to fit more matches in memory by adding frequently occurring fingerprints than by adding lots of rarely occurring fingerprints.

fingerprints that are actually selected at runtime on the training data are included in the database. We then add the remaining fingerprints to the database in order of utility value until the memory has been used up.

At the end of the database construction step, we have a reverse index that contains a subset of the generalized n-grams found in the IMSLP data. This subset excludes extremely common n-grams (that occur $> \gamma$ times) as well as extremely rare n-grams (due to memory limits).

2.3 Search

In our analogy, the runtime search is akin to a consumer shopping in the store. We first process the cell phone image query to compute a sequence of bootleg score features $\tilde{x}_1, \ldots, \tilde{x}_L$, and then construct a set of $T = 16$ generalized n-grams $y_{i1}, \ldots, y_{iT}$ at each offset $i = 1, \ldots, L$. This set of (approximately) TL candidate n-grams are the items on the store shelves that the consumer can purchase. Each offset i corresponds to an aisle in the store, and each aisle contains T items. [3] The consumer uses a strategy (described in the next few paragraphs) to purchase selected candidate n-grams from each aisle of the store. These selected n-grams are then used to search the database. The consumer purchasing strategy and the search mechanism are described in the next three paragraphs.

The consumer adopts a disciplined greedy purchasing strategy. They first decide on a budget for the whole store (B_{tot}), divide it by the number of aisles to determine a budget per aisle ($B_{aisle} = B_{tot}/L$), and then purchase as many items in each aisle as they can in order to maximize utility while staying under their budget for the aisle. Any unused funds from an aisle carry over to the next aisle. During a search, the primary constrained resource is runtime – we do not want a query to take too long to process. The runtime is directly correlated with the number of fingerprint matches in the database that are processed. The total budget B_{tot} is therefore specified in terms of the maximum total number of matches we are willing to process for each query. B_{tot} is a hyperparameter that can be selected to achieve a desired runtime. In our experiments, we set $B_{tot} = 65000$ in order to achieve ≈ 1 second average runtime per query on the training set.

How should the consumer decide which items (i.e. which n-grams) to purchase in each aisle? We assume that the consumer is rational and wants to maximize their own utility. We define the utility as the expected number of match points added to the true matching item in the database, which (as explained in Section 2.2 paragraph 3) is the probability that the underlying bootleg score representation is correct. We again approximate this probability based on the n-gram type and the statistics on the training data (as shown in Figure 3). The runtime cost for each item is proportional to the number of fingerprint matches in the database, since processing many fingerprint matches will require more runtime. Putting this all together, the consumer tries to maximize utility by adopting the following strategy: they purchase items in each aisle in decreasing order of their utility-to-cost ratio $R(y_{ij}) = P_{correct}(j)/N(y_{ij})$ until the budget for the aisle (B_{tot}/L plus any carryover from the previous aisle) has been spent. Note that $P_{correct}(j)$ can be determined based only on the n-gram type and $N(y_{ij})$ can be determined quickly without needing to actually process the fingerprint matches. This information allows the system to dynamically adjust which n-grams to select in an informed and rational manner.

The search is performed using the histogram of offsets method [30]. This method provides an efficient way to search a large database in order to find a sequence of matching fingerprints aligned in time. For each fingerprint y_{ij} (i.e. the n-gram of type j at offset i in the query) that is purchased by the consumer, the system processes the list of fingerprint matches in the database from the reverse index. Each fingerprint match is specified by two pieces of information: the PDF and the offset k in the bootleg score where the fingerprint occurs. Processing a fingerprint match (i.e. a (PDF, k) tuple) means adding the relative offset $k - i$ to the PDF's histogram. Note that a sequence of matching fingerprints aligned in time will result in a histogram with a large spike at the true relative offset (i.e. where the query occurs in the PDF). Therefore, we can use the maximum bin count in each histogram as a match score for the PDF. Because IMSLP often contains multiple PDFs for a single piece, we calculate the *piece* match score as the maximum score among its constituent PDFs. Finally, we sort all pieces in the database by their piece score. The resulting ranked list is the final output of the system.

2.4 Comparison to Dynamic N-gram

It is instructive to compare the proposed marketplace fingerprinting approach to the dynamic n-gram approach. We will compare these two along the three axes described above: the fingerprint representation, the database construction, and the search mechanism.

Fingerprint representation. The dynamic n-gram approach considers four types of standard n-grams at each offset i: (x_i), (x_i, x_{i+1}), (x_i, x_{i+1}, x_{i+2}), and $(x_i, x_{i+1}, x_{i+2}, x_{i+3})$. The marketplace fingerprinting approach generalizes the notion of an n-gram and offers a much wider selection of n-gram types to the consumer.

Database construction. The dynamic n-gram approach is to simply add all 1-grams to the database, then all 2-grams, then all 3-grams, etc. until memory runs out. For 128 GB of RAM, this approach maxes out at 4-grams. This results in a database that has complete representation of four different types of n-grams. The marketplace fingerprinting approach, on the other hand, considers 16 different types of n-grams and selects a subset of the most useful fingerprints from each n-gram type in a principled way.

Search. In our analogy, the dynamic n-gram approach corresponds to a store in which every aisle has four items, and the consumer purchases exactly one item per aisle. The consumer purchasing strategy is to set a fixed budget for

[3] The last few aisles may contain less than T items due to a lack of context.

each aisle and to purchase the most expensive item in each aisle that is under the budget. The marketplace fingerprinting approach corresponds to a store in which every aisle has 16 (or more) items, and the consumer can purchase multiple items per aisle. The consumer purchasing strategy is to set a fixed total budget for the whole store, determine a budget per aisle to control their total spending, and purchase the items in each aisle with highest utility-to-cost ratio until the aisle budget has been spent. The key idea behind the marketplace fingerprinting approach is that options and choices are good for the consumer. Once the consumer has purchased a set of fingerprints, both approaches use the histogram of offsets method to search the database.

3. EXPERIMENTAL SETUP

The experimental setup is identical to [24] for fair comparison with the dynamic n-gram approach. We describe the data and evaluation metrics below for completeness.

The queries in our system are cell phone images taken of physical pages of piano sheet music. There are 10 cell phone images of 200 different piano pieces across 25 different composers, resulting in a total of 2000 queries. The pictures were taken with four different cell phone models. The cell phone pictures are spread across the length of the piece and are taken in a variety of different physical locations, lighting conditions (including both flash and no flash), and levels of zoom (between 1 and 5 lines of sheet music). The proposed marketplace fingerprinting system and the baseline systems do not have any trainable parameters (only hyperparameters), so we use only 40 pieces for training (400 queries) and the remaining 160 pieces for testing (1600 queries).

The database consists of all solo piano sheet music in IMSLP. In total, there are 31,384 PDFs, 29,310 pieces, and 374,758 pages of sheet music. [24] provides a precomputed dataset of bootleg score features on all solo piano sheet music in IMSLP, and we use this dataset without modification.

We evaluate system performance along two axes: retrieval accuracy and runtime. Because each query matches exactly one unique *piece* in IMSLP (with multiple different PDF versions), we use mean reciprocal rank (MRR) as a measure of retrieval accuracy. MRR is computed as

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{R_i} \tag{1}$$

where N indicates the number of queries and R_i indicates the rank of the true matching piece. For the IMSLP piano dataset, R_i ranges between 1 and $29,310$. Note that MRR ranges between 0 and 1, where 1 corresponds to perfect performance. We also measure the runtime required to process each query and report the average and standard deviation of the runtimes. All experiments were performed on a 2.1 GHz Intel Xeon processor with 128 GB RAM.

System	MRR	Runtime	
		avg	std
1-gram	.709	21.5s	12.5s
2-gram	.845	2.76s	1.11s
3-gram	.808	1.99s	.36s
4-gram	.755	1.12s	.25s
5-gram	.688	1.07s	.13s
dynamic n-gram	.853	.98s	.12s
marketplace	.905	.95s	.14s

Table 1. Comparison of system performance on the camera-based piano sheet music identification task. The middle column indicates the retrieval accuracy in terms of mean reciprocal rank (MRR), and the rightmost column indicates the average and standard deviation of runtimes. The bottom row shows the performance of the proposed marketplace fingerprinting system.

4. RESULTS

We compare the marketplace fingerprinting approach to six other baseline systems. The first five baselines are fixed n-grams with $n = 1, 2, 3, 4, 5$. These approaches use a single fixed-size n-gram as the fingerprint representation. For example, given a sequence of bootleg score features $x_1, \ldots, x_L$, the fixed 2-gram baseline would construct fingerprints of the form (x_i, x_{i+1}). The fixed n-gram approach for large-scale retrieval was explored in [17]. The sixth baseline is the dynamic n-gram method [24], which represents the state-of-the-art in sheet music identification. For a given sequence of query bootleg score features $x_1, \ldots, x_L$, the dynamic n-gram constructs one fingerprint at each offset i of the form $(x_i, x_{i+1}, \ldots, x_{i+L_i})$ where $0 \leq L_i \leq 3$. The size of the n-gram is selected dynamically for each offset i in order to ensure that the number of fingerprint matches in the database is below a specified threshold.

Table 1 shows the retrieval accuracy and runtime for all 7 systems. We can see that the marketplace fingerprinting system has the highest retrieval accuracy by a large margin (.905 vs .853). We can roughly interpret this gap as a reduction in "errors" by about $\frac{1}{3}$ compared to the dynamic n-gram approach. Furthermore, this improvement in retrieval accuracy does not come at the expense of runtime: the runtime budget (B_{tot}) for the marketplace system was selected to achieve < 1 second average runtime per query on the training set. The improvement in retrieval accuracy instead comes from utilizing the runtime more *efficiently* – processing many more fingerprints with a higher utility-to-cost ratio compared to the dynamic n-gram approach.

5. ANALYSIS

In this section we explore two questions of interest to gain deeper intuition into the performance of the marketplace fingerprinting system.

The first question of interest is, "What is the effect of the runtime budget (B_{tot})?" This is a hyperparameter specify-

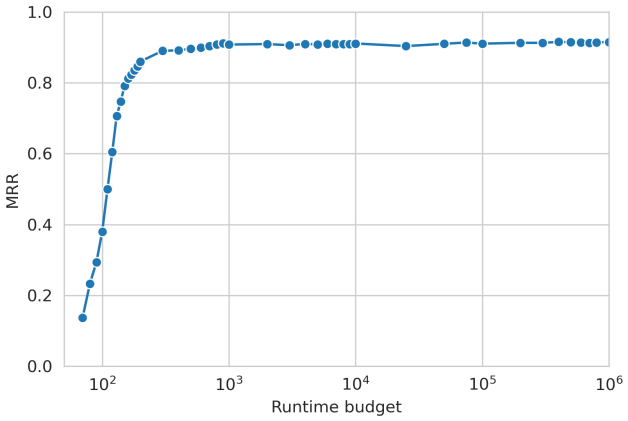

Figure 4. Relationship between runtime budget (B_{tot}) and retrieval accuracy. The runtime budget is a hyperparameter that specifies the maximum number of fingerprint matches in the database that can be processed for a single query.

ing the total number of fingerprint matches in the database that we are willing to process for each query. The total runtime budget determines the budget for each aisle, which in turn determines how many n-gram fingerprints will be processed. By setting B_{tot} appropriately, we can thus trade off runtime for retrieval accuracy: if we're willing to wait longer to process a query, we can get higher quality results.

Figure 4 shows the retrieval accuracy of the marketplace fingerprinting system across a range of runtime budget values. The retrieval accuracy improves dramatically as the runtime budget increases from 1 to 500, but then reaches a plateau and remains approximately constant for runtime budget values greater than 1000. This plateau strongly suggests that we have reached an upper bound on performance through the use of redundancy in the fingerprint lookups. Thus, we would not expect that adding more n-gram types would improve results further. Any significant additional improvements to retrieval accuracy would likely need to come from a more accurate bootleg score (or alternative) representation. Perhaps the most surprising finding in Figure 4 is how *low* the runtime budget is when it reaches the plateau. With only a budget of 1000 matches – chosen strategically, of course – it is possible to already achieve a MRR of .908 with an average runtime of 0.71 seconds. This is quite remarkable considering that the dynamic n-gram model performs lookups on fingerprints with up to $10,000$ matches in the database, so that a single lookup would likely use up the entire runtime budget.

The second questions of interest is, "What is the distribution of fingerprints in the database?" Figure 5 shows the distribution of fingerprints in the databases for all seven systems. [4] The fingerprints are sorted from most frequent (left) to least frequent (right), and their frequency of occurrence in the IMSLP dataset is shown on the y-axis. Note that both axes are shown on a log scale. The ideal distribution for optimal hashing performance is a uniform (flat)

[4] The curves for the fixed n-gram systems match those of Figure 3 in [24]. However, we have confirmed that the curve for the dynamic n-gram system in Figure 3 of [24] is incorrect. The brown curve in Figure 5 above shows the corrected distribution.

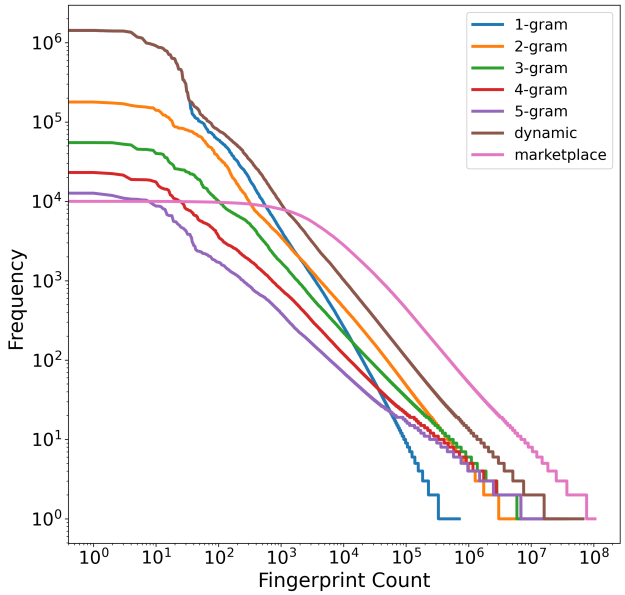

Figure 5. Fingerprint distribution for different systems. For each system, the set of unique fingerprints in the database are sorted from most frequent (left) to least frequent (right), and the y-axis indicates the frequency of occurrence. Note that both axes are on a log scale.

distribution, in which all fingerprints occur the same number of times. We can see that the marketplace fingerprinting system has the flattest distribution, avoiding extremely common fingerprints and minimizing extremely rare fingerprints. By considering many more types of n-grams, it is able to achieve a flatter distribution that is closer to the ideal uniform distribution.

6. CONCLUSION

This paper proposes a way to identify sheet music using a novel fingerprinting scheme called marketplace fingerprinting. Our approach considers the retrieval problem through the lens of an economic marketplace in which a consumer (the search) with a finite budget (runtime) purchases items (fingerprints) in a store (the database). Building off of previous work that uses n-grams of bootleg score features as fingerprints, we generalize the notion of n-grams to greatly expand the number of different types of fingerprints in order to give the consumer more options to choose from. We show that choosing which fingerprints to include in the database presents a type of minimax problem in which the consumer (the runtime search problem) and the store (the database design problem) have competing interests. At runtime, the consumer (the search) is presented with many different options (fingerprint types) to choose from and tries to maximize utility by purchasing the items (i.e. doing database lookups on fingerprints) that have maximum utility-to-cost ratio while staying under a fixed budget (runtime). With experiments using all solo piano sheet music images in IMSLP as a searchable database, we show that the marketplace fingerprinting approach substantially outperforms previous approaches.

7. REFERENCES

[1] V. Thomas, C. Fremerey, M. Müller, and M. Clausen, "Linking sheet music and audio – challenges and new approaches," in *Dagstuhl Follow-Ups*, M. Müller, M. Goto, and M. Schedl, Eds. Schloss Dagstuhl–Leibniz- Zentrum für Informatik, Dagstuhl, Germany, 2016, vol. 3, pp. 1–22.

[2] C. Fremerey, M. Müller, , and M. Clausen, "Handling repeats and jumps in score-performance synchronization," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2010, pp. 243–248.

[3] D. Damm, C. Fremerey, F. Kurth, M. Müller, and M. Clausen, "Multimodal presentation and browsing of music," in *Proceedings of the International Conference on Multimodal Interfaces*, 2008, pp. 205–208.

[4] F. Kurth, M. Müller, C. Fremerey, Y.-H. Chang, and M. Clausen, "Automated synchronization of scanned sheet music with audio recordings," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2007, pp. 261–266.

[5] C. Fremerey, M. Müller, F. Kurth, and M. Clausen, "Automatic mapping of scanned sheet music to audio recordings," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2008, pp. 413–418.

[6] C. Fremerey, M. Clausen, S. Ewert, and M. Müller, "Sheet music-audio identification," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2009, pp. 645–650.

[7] M. Dorfer, A. Arzt, and G. Widmer, "Learning audio-sheet music correspondences for score identification and offline alignment," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2017, pp. 115–122.

[8] ——, "Towards score following in sheet music images," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2018, pp. 784–791.

[9] M. Dorfer, F. Henkel, and G. Widmer, "Learning to listen, read, and follow: Score following as a reinforcement learning game," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2018, pp. 784–791.

[10] F. Henkel, S. Balke, M. Dorfer, and G. Widmer, "Score following as a multi-modal reinforcement learning problem," *Transactions of the International Society for Music Information Retrieval*, vol. 2, no. 1, pp. 67–81, 2019.

[11] F. Henkel, R. Kelz, and G. Widmer, "Learning to read and follow music in complete score sheet images," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2020, pp. 780–787.

[12] M. Dorfer, J. Hajič, A. Arzt, H. Frostel, and G. Widmer, "Learning audio-sheet music correspondences for cross-modal retrieval and piece identification," *Transactions of the International Society for Music Information Retrieval*, vol. 1, no. 1, pp. 22–23, 2018.

[13] M. Dorfer, J. Schlüter, A. Vall, F. Korzeniowski, and G. Widmer, "End-to-end cross-modality retrieval with cca projections and pairwise ranking loss," *International Journal of Multimedia Information Retrieval*, vol. 7, no. 2, pp. 117–128, 2018.

[14] D. Yang, T. Tanprasert, T. Jenrungrot, M. Shan, and T. Tsai, "Midi passage retrieval using cell phone pictures of sheet music," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2019, pp. 916–923.

[15] T. Tsai, D. Yang, M. Shan, T. Tanprasert, and T. Jenrungrot, "Using cell phone pictures of sheet music to retrieve midi passages," *IEEE Transactions on Multimedia*, vol. 22, no. 12, pp. 3115–3127, 2020.

[16] D. Yang and T. Tsai, "Piano sheet music identification using dynamic n-gram fingerprinting," *Transactions of the International Society for Music Information Retrieval*, vol. 4, no. 1, pp. 42–51, 2021.

[17] T. Tsai, "Towards linking the lakh and imslp datasets," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 546–550.

[18] J. Thompson, A. Hankinson, and I. Fujinaga, "Searching the liber usualis: Using CouchDB and ElasticSearch to query graphical music documents," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2011.

[19] S. P. Achankunju, "Music search engine from noisy omr data," in *International Workshop on Reading Music Systems*, 2018, pp. 23–24.

[20] R. Malik, P. P. Roy, U. Pal, and F. Kimura, "Handwritten musical document retrieval using music-score spotting," in *IEEE International Conference on Document Analysis and Recognition*, 2013, pp. 832–836.

[21] J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal, "Probabilistic music-symbol spotting in handwritten scores," in *IEEE International Conference on Frontiers in Handwriting Recognition*, 2018, pp. 558–563.

[22] J. Hajič, M. Kolárová, A. Pacha, and J. Calvo-Zaragoza, "How current optical music recognition systems are becoming useful for digital libraries," in *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, 2018, pp. 57–61.

[23] S. Waloschek, A. Hadjakos, and A. Pacha, "Identification and cross-document alignment of measures in music score images," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2019, pp. 137–146.

[24] D. Yang and T. Tsai, "Camera-based piano sheet music identification," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2020, pp. 481–488.

[25] M. Shan and T. Tsai, "Improved handling of repeats and jumps in audio-sheet image synchronization," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2020, pp. 62–69.

[26] ——, "Automatic generation of piano score following videos," *Transactions of the International Society for Music Information Retrieval*, vol. 4, no. 1, pp. 29–41, 2021.

[27] P. F. Brown, V. J. Della Pietra, P. V. Desouza, J. C. Lai, and R. L. Mercer, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, no. 4, pp. 467–480, 1992.

[28] M. Hontanilla, C. Pérez-Sancho, and J. M. Inesta, "Modeling musical style with language models for composer recognition," in *Iberian Conference on Pattern Recognition and Image Analysis*, 2013, pp. 740–748.

[29] J. Wołkowicz and V. Kešelj, "Evaluation of n-gram-based classification approaches on classical music corpora," in *International Conference on Mathematics and Computation in Music*, 2013, pp. 213–225.

[30] A. Wang, "An industrial strength audio search algorithm," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2003, pp. 7–13.