Evaluating Melodic Encodings for Use in Cover Song Identification

David D. Wickland
University of Guelph
wickland@uoguelph.ca

David A. Calvert
University of Guelph
dcalvert@uoguelph.ca

James Harley
University of Guelph
jharley@uoguelph.ca

ABSTRACT

Cover song identification in Music Information Retrieval (MIR), and the larger task of evaluating melodic or other structural similarities in symbolic musical data, is a subject of much research today. Content-based approaches to querying melodies have been developed to identify similar song renditions based on melodic information. But there is no consensus on how to represent the symbolic melodic information in order to achieve greater classification accuracy. This paper explores five symbolic representations and evaluates the classification performance of these encodings in cover song identification using exact matching of local sequences. Results suggest the more lossy encodings can achieve better overall classification if longer melodic segments are available in the data.

1. INTRODUCTION

The landscape of todays digital music exploration paradigm has shifted greatly in recent years, and will likely continue to change. With the growth in popularity of subscription-based collections, people are discovering and consuming music in vast and varied ways on a number of devices and platforms. With such an increase in access, there is greater demand for users to explore, interact with, and share music. To this end, there is continued demand for novel and efficient ways to index, retrieve, manipulate, etc. digital music.

Symbolic melodic similarity, as a content-based approach to MIR, can be considered a sub-discipline of music similarity. The goal of melodic similarity is to compare or communicate structural elements or patterns present in the melody. Where vast efforts towards music discovery and recommender systems have historically focused on music similarity, by employing low-level feature extraction and clustering or other classification schemes, there has been comparatively less focus on melodic similarity. Many applications of similarity analysis for title retrieval, genre classification, etc., do not require the additional effort to process and interpret melodic content. Instead, relying on timbral descriptors, tags, etc. is considerably more efficient, and can often achieve equal if not better performance. However, there are numerous applications of digital musical analysis that cannot be performed without exploring the melodic content directly.

Symbolic melodic similarity research can be largely categorized into monophonic and polyphonic melodies, and into sequence similarity, or harmonic/chordal similarity. Melodies are often extracted from MIDI, MusicXML, or other digital music transcription formats, into representations such as: vectors, contours, text or numerical strings, or graphs. While considerable efforts have been made to create and evaluate melodic similarity measures with different symbolic representations, there has been little attention paid to the behaviours of these approaches with different representations.

This article explores the behaviour of local exact matching of melodies with five symbolic representations of varying information. The lengths of the local matches are used to perform cover song identification, and their classification performance is discussed.

2. LAKH MIDI DATASET

2.1 Preprocessing

The Lakh MIDI dataset was acquired for use in this research. There are many varieties of the Lakh dataset; in particular, this work employs the Clean MIDI subset, which contains MIDI files with filenames that indicate both artist and song title [1, 2]. MIDI files were scraped for track info, and any tracks titled “Melody” were parsed to acquire the melodic information. Any melody that contained two notes overlapping for greater than 50% of their duration was considered polyphonic, and was discarded. All remaining monophonic melodies were transcribed to text, including artist, song title, tempo, meter, and all melodic information (i.e. notes and durations).

Key signature data from MIDI files is unreliable. Consequently, key signatures were estimated for each melody using the Krumhansl-Schmuckler key-finding algorithm, which uses the Pearson correlation coefficient to compare the distribution of pitch classes in a musical segment to an experimental, perceptual “key-profile” to estimate which major or minor key a melody most closely belongs to [3]. The Krumhansl-Schmuckler algorithm works well for melodic segments or pieces that do not deviate from a tonal center; however, pieces that modulate or shift keys will affect the accuracy of the algorithm.

Deduplication was first handled in the original Lakh dataset, where MD5 checksums of each MIDI file were compared, and duplicates were removed. This approach is quite robust but unfortunately still requires further deduplication.

Since MIDI file or track names, or other meta data can be altered without affecting the melodic content, a further step to compare the transcribed melodies and remove duplicates was applied. This ensured that while cover songs with the same or a different artist and same song title were permitted, their transcribed melodies could not match identically.

In total, 1,259 melodies were transcribed, which gives 793,170 melodic comparisons. Of these melodies, the shortest was 14 notes long and the longest was 949 notes long. Within the 1,259 melodies, there were 106 distinct songs that had one or more corresponding cover(s) in the dataset. In total there were 202 covers present in the dataset.

2.2 Ground Truth Cover Songs Data

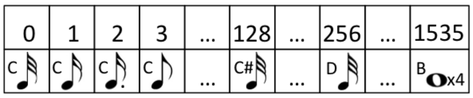
Using the transcribed melodies dataset, a bit map was created to annotate which melodies were covers or renditions. No consideration was given to which melody was the original work and which was the cover. The bit map was constructed such that an annotation of 1 indicated the melodic comparison was between two covers, and 0 indicated the melodies were unique (i.e. non-covers). Melodies were annotated as covers if they had the same song title and artist name, or the same song title and a different artist. Duplicate song titles by different artists were individually inspected to identify if they were genuine covers or unique songs.

3. MELODIC ENCODINGS

Melodies were encoded into five different symbolic representations of varying information loss. These encodings are: Parsons code, Pitch Class (PC), Interval, Duration, and Pitch Class + Duration (PCD). Parsons is a contour representation that ignores any intervalic or rhythmic information and only expresses the relationship between notes as $\{Up, Down, Repeat\} = \{0, 1, 2\}$. PC notation describes notes belonging to one of 12 unique pitch classes: $\{C, C\sharp, ..., B\} = \{0, 1, ..., 11\}$. The Interval representation encodes each note by its intervalic distance from the previous note (e.g. $C \uparrow G = +7$, $B \downarrow G\sharp = -3$). Interval encoding does not apply modulo operations by octave in either the positive or negative direction (i.e. intervals greater than ±12 are permitted). Duration encoding ignores all melodic information and alphabetizes notes based on their quantized duration. Notes were quantized down to 32$^{\text{nds}}$ using Eq. (1), where d_i is the duration of the note, tpb and met are the ticks per beat and time signature meter of the MIDI file, and $|\Sigma|$ is the size of the encoding's alphabet (i.e. $|\Sigma| = 128$ for Duration).This provides 128 possible durations up to a maximum duration of 4 bars at $\frac{4}{4}$ time. Tuples were not supported, and compound signatures were reduced to simple time signatures before quantization.

$$q_i = \left\lfloor \frac{d_i}{tpb \times met} \times \frac{|\Sigma|}{4} \right\rfloor = \left\lfloor \frac{d_i}{tpb \times met} \times 32 \right\rfloor \quad (1)$$

PCD encodes both duration and pitch class information by combining the alphabets of each encoding. Values $[0, 127]$ represent all possible durations of pitch class C, $[128, 255]$ are all possible durations of $C\sharp$, and so on. Figure 1 illustrates the PCD encoding. Both PC and PCD encodings use absolute representations of pitch values, as opposed to relative (e.g. interval). In order to compare melodies accurately, they were transposed to the same key, or the harmonic major/minor equivalent, prior to comparison.



0	1	2	3	...	128	...	256	...	1535
C	C	C	C	...	C#	...	D	...	B x4

Figure 1. Examples of the Pitch Class Duration Encoding Alphabet

4. EXACT MATCHING FOR MELODIC SIMILARITY

Evaluating melodic similarity by solving for local exact matches between musical segments often involves solving the Longest Common Substring (LCS) problem. The LCS solves for the longest string(s) that are a substring of two or more input strings. In the context of this work, melodies are encoded into strings and then compared by solving the LCS. There are two common approaches to solving the LCS: generalized suffix trees, and dynamic programming. This work employs suffix trees because of their computational efficiency.

A suffix tree is a compressed trie that represents all possible suffixes of a given input string [4]. The keys store the suffixes and the values store the positions in the input text. Constructing suffix trees was done using Ukkonen's algorithm, which constructs a suffix tree in $\mathcal{O}((n + m))$ time, where n and m are the lengths of the two input strings [4]. Similarly, the LCS can be solved in $\mathcal{O}((n + m))$ time by traversing the suffix tree.

Generalized suffix trees (GST) are created for a set of input strings as opposed to a single string. The input strings are each appended with a unique character, and then concatenated together to form one aggregate input string. In this work, each pair of melodies being compared were used to create a GST to solve for the LCS of the two melodies. Once constructed, the GST is traversed to annotate nodes as *X* for suffixes belonging to the first melody, *Y* for suffixes belonging to the second melody, and *XY* for suffixes common to both melodies. The path from root to the deepest *XY* node represents the LCS. Figure 2 shows the GST of input strings "*ABAB*" and "*BABA*", such that the concatenated input string is "*ABAB$BABA#*". Paths denoting substrings "*ABA*" and "*BAB*" are both solutions to the LCS.

5. SEQUENCE COMPLEXITY

Shannon entropy measures the average amount of information generated by a stochastic data source, and is calculated by taking the negative logarithm of the probability mass function of the character or value [5]. Shannon entropy is given by H in Eq. (2) where b is the base of the logarithm and p_i is the probability of a character number i occurring

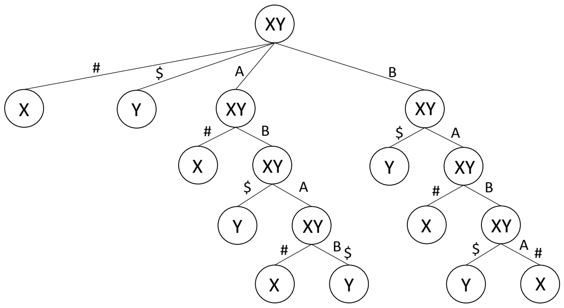


Figure 2. Annotated Generalized Suffix Tree for Input String *ABAB$BABA#* to solve for the LCS

in the input string [6]. In this work, $b = 2$, such that the units of entropy are bits.

$$H = -\sum_{i=1}^{n} p_i \log_b p_i \tag{2}$$

Shannon entropy establishes a limit on the shortest possible expected length for a lossless compression that encodes a stream of data [5]. For a given input string, when a character with a lower probability value occurs, it carries more information than a frequently occurring character. Generally, entropy reflects the disorder or uncertainty in an input, and is used in this work as an approximation to the complexity of an encoded melodic segment.

All non-cover song melodies (i.e. unique) were traversed with a sliding window to calculate the average entropy for a given window length. Figure 3 shows the average entropy, H, as a function of window length for each of the five melodic encodings. The encodings with the smallest alphabet plateau at the lowest average entropy, whereas the encoding with the largest alphabet grows toward a much larger average entropy value.

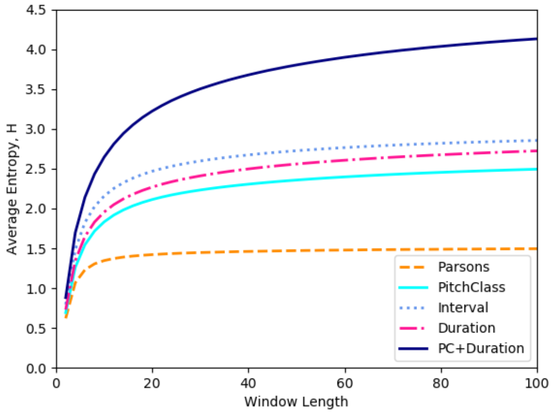


Figure 3. The average entropy, H of unique melodic segments as a function of window length

From the cover songs dataset, the exact matches for each melodic comparison were transcribed for all encodings. All match segments were categorized by their length to compute the average entropy by match length for each of the five encodings. Figure 4 shows the average entropy, H, of the exact match melodic segments as a function of their match length.

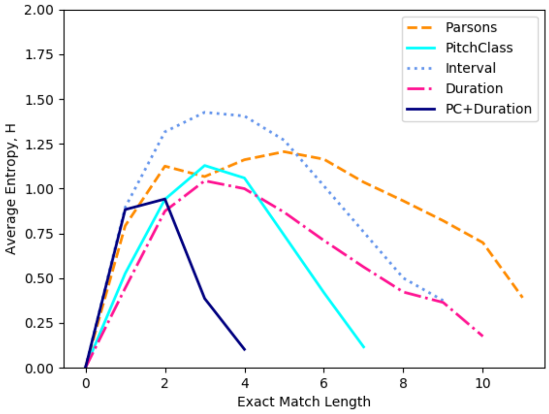


Figure 4. The average entropy, H of exact match melodic segments as a function of match length

Interval encoding achieves the greatest average entropy at a match length $l = 3$, and Parsons has greater average entropy values for the longer melodic segments (i.e. $l > 5$). PCD exhibits the lowest average entropy for nearly all match lengths. This may suggest that while larger alphabet encodings can preserve more information, exact matching techniques such as solving the LCS often discover short, repeating patterns, of comparatively low complexity.

6. COVER SONG IDENTIFICATION

6.1 Binary Classification

Binary classification is the technique of classifying the elements of a given set into two groups on the basis of a predicting or classification rule [7]. In the context of cover song identification, we are interested in identifying which melodies are unique and which are covers. With the ground truth annotated data, we can set a threshold for the length of the LCS between two melodies to predict whether they are unique or covers. Melodies with a LCS shorter than this threshold are predicted to be unique, whereas melodies with a LCS of this length or greater are predicted as covers. A confusion matrix, shown in Table 1 illustrates the four possible outcomes of these predictions: true positive (tp), false positive (fp), true negative (tn), and false negative (fn).

The Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) score are commonly used in binary classification to represent the quality of an automatic classification scheme or rule [8]. The ROC curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various classification thresholds. TPR and FPR are calculated using Eq. (3) and Eq. (4) respectively. The further the curve deviates from the diagonal midline (i.e. extending from $(0, 0)$ to $(1, 1)$), the better the

		Predicted	
		p	**n**
Actual	**p′**	true positive	false negative
	n′	false positive	true negative

Table 1. Confustion Matrix for Binary Classification Scheme

quality of the classifier, assuming the positive prediction is more desired than the negative prediction.

$$TPR = \frac{tp}{tp + fn} \tag{3}$$

$$TPR = \frac{fp}{tp + tn} \tag{4}$$

The AUC score is a normalized measure of the predictive quality of a classifier. An area of 1 represents a perfect classifier, and an area of 0.5 implies the classifier is no better than random guessing.

6.2 Classification Performance

The five melodic encodings were used to compare all melodies against each other to solve for the LCS in every comparison. The lengths of the exact matches were used to predict if the two melodies being compared were covers or unique songs. Figure 5 shows the ROC curves for the five melodic encodings for all exact match length thresholds.

Parsons is the most lossy encoding (i.e. preserves the least information) but achieves the greatest AUC score of all the encodings. The PCD encoding preserves the greatest amount of information of all the encodings and achieves the lowest AUC score. It is notable that while PC is the second-most lossy encoding, its AUC score is lower than Interval and Duration, both of which have considerably larger alphabets and preserve more information. The poor performance of PC and PCD encodings may be due in part to some inaccuracy in the key-finding algorithm; however, it is unlikely these encodings would perform notably better with a perfect key-finding algorithm.

The top left corner at position $(0, 1)$ of the ROC plot represents the perfect classification scheme, with a TPR of 100% and a FPR of 0% [9]. One common approach to selecting a classifier threshold in practice is to identify the point on the curve closest to $(0, 1)$. Table 2 shows the closest point on each of the five encodings' ROC curves to $(0, 1)$, and the exact match threshold at this point. There are circumstances where a greater emphasis on TPR or FPR may be desired, and so a trade-off can be made by selecting a threshold that better suits the application of the classifier. The ability to select the classification threshold for a desired performance is an important aspect of the ROC curve.

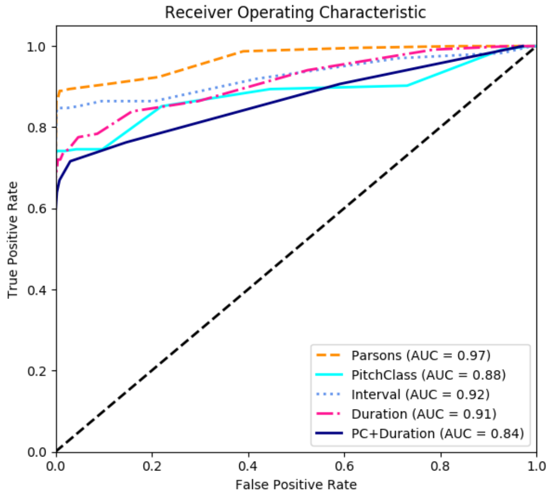


Figure 5. Receiver Operating Characteristic for the five melodic encodings using exact matching

Encoding	**FPR**	**TPR**	**Dist. to (0, 1)**	**Ex. Match Length**
Parsons	0.028	0.894	0.109	14
Pitch Class	0.043	0.746	0.258	9
Interval	0.005	0.847	0.153	13
Duration	0.159	0.839	0.226	8
PC+Duration	0.147	0.763	0.279	4

Table 2. Closest points on ROC Curves for Each Melodic Encoding and the Corresponding Exact Match Length Threshold

7. CONCLUSIONS

In this work, the behaviour of local exact matching as a measure of melodic similarity is applied to melodies encoded with five symbolic representations of varying information. Generalized suffix trees were used for each melodic comparison to solve for the longest common substring between two melodies. The lengths of these local exact matches were used to predict cover songs in a dataset of both unique and cover song melodies.

Parsons code achieves the best overall classification performance at any exact match length threshold, and it is most discriminant at an exact match length threshold of 14. Large alphabet encodings such as PCD achieve poorer classification performance. Results suggest lossy encodings such as Parsons, achieve their best classification rates with longer exact match lengths than encodings that preserve more information.

The average entropy of unique melodies in the dataset grows with the window length of the melodic segment, and with the size of the alphabet of the encoding. The average entropy results from the exact matches of cover song melodies suggests encodings that drive higher complexity exact matches are beneficial; however, ultimately the longer melodic segments are better at differentiating cover song melodies from unique song melodies.

In future work we would like to explore the effects of more granular quantization on the Duration and PCD encodings. A non-repeating contour representation should be compared to Parsons to illustrate the effects of repeating notes in exact matching and to determine if even lossier symbolic representations can achieve as good or better classification performance. It would be advantageous to compare Shannon entropy results to a practical approximation of Kolmogorov complexity such as one or more lossless compression algorithms. Lastly, an investigation of complexity and classification performance with inexact matching similarity measures, such as edit distance, could illuminate the benefits and drawbacks of the faster exact matching approach.

8. REFERENCES

[1] C. Raffel, "Lakh MIDI Dataset v0.1," http://colinraffel.com/projects/lmd.

[2] ——, *Learning-based methods for comparing sequences, with applications to audio-to-MIDI alignment and matching*. Columbia University, 2016.

[3] D. Temperley, "What's key for key? the krumhansl-schmuckler key-finding algorithm reconsidered," *Music Perception: An Interdisciplinary Journal*, vol. 17, no. 1, pp. 65–100, 1999.

[4] D. Gusfield, *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge university press, 1997.

[5] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, p. 379?423, 1948.

[6] P. Grunwald and P. Vitányi, "Shannon information and kolmogorov complexity," *arXiv preprint cs/0410002*, 2004.

[7] A. Ng, K. Katanforoosh, and Y. Bensouda Mourri, "Neural networks and deep learning: Binary classification," https://www.coursera.org/learn/neural-networks-deep-learning/lecture/Z8j0R/binary-classification.

[8] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[9] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve." *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.