PATTERN CLUSTERING IN MONOPHONIC MUSIC BY LEARNING A NON-LINEAR EMBEDDING FROM HUMAN ANNOTATIONS

Timothy de Reuse, Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology, McGill University
{timothy.dereuse, ichiro.fujinaga}@mcgill.ca

ABSTRACT

Musical pattern discovery algorithms find instances of repetition in symbolic music, allowing for some user-specifiable amount of variation between identified repetitions; however, they can yield an intractably large number of discovered patterns when allowing for even small amounts of variation. This is commonly addressed by defining some heuristic notion of pattern significance, and returning only the most significant patterns. This paper develops a method of pattern discovery that models human judgement of what constitutes a significant pattern by incorporating annotations of repeated patterns, avoiding the need to design heuristics.

We take pattern discovery as a clustering task, where the input is a set of passages of monophonic music, represented as vectors of extracted features, and the output clusters correspond to discovered patterns. The human annotations are used to train a neural network to learn a low-dimensional embedding of the feature space that maps passages of music close together when they are occurrences of the same ground-truth pattern. The results of this approach match up with the annotations significantly better than the results of an approach using clustering without subspace learning. We provide examples of the types of patterns that this method tends to discover and discuss its feasibility and practicality as a tool for extracting useful information about repetitive structure in music.

1. INTRODUCTION

To discover patterns in a piece of music is, loosely speaking, to find passages that are similar to other passages, and to cluster these passages into inter-related groups. This is done with the goal of producing motivic analyses [34], automated composition systems [6], or as a single step to provide information for some larger Music Information Retrieval application [17]. Humans can perceive repetition between two musical passages even when the passages vary somewhat, so it is important to incorporate some allowable margin of variation between members of a single pattern; unfortunately, this causes a combinatorial explosion, returning many more patterns than are useful for any application. Inexact repetition simply occurs in music too often by chance; as [29] put it, "Most repetitions in music are not interesting." For this reason, discovery of patterns cannot be reduced to the problem of finding passages of music within a single piece that are similar to one another. Limiting the number of patterns found by a pattern discovery algorithm in a systematic way means explicitly or implicitly defining some measure by which one pattern can be judged as "more significant" than another; we refer to these as *pattern significance* measures.

Previous work by Collins et al. [10] suggests that analysts judge pattern significance consistently, based on quantifiable features of the musical surface. By analyzing ground-truth significant patterns, we can simultaneously learn a sense of what metric of melodic similarity the annotators used (by comparing occurrences of a single pattern) and a sense of the metric of pattern significance used (by comparing significant patterns to insignificant patterns), and use these findings to inform a pattern discovery method. We use these insights to model pattern discovery as a cluster analysis problem, where the input data points are a set containing all possible passages of music from the piece under investigation, and the output clusters correspond directly to discovered patterns. There are parallels between common issues in cluster analysis and pattern discovery that make this a sensible choice of technique: it is rarely known in advance how many clusters (or patterns) are present in a dataset (or piece), and validity measures for particular clusterings (or sets of discovered patterns) are the subject of ongoing investigation [15]. The goal here is that our clustering approach learns some criteria of what makes the particular patterns in the ground truth significant, and uses that criteria to return a limited number of patterns. Ideally, the model should be able to rediscover all the ground truth patterns from the songs that contain them, and not too many more.

The terms used to refer to sets of repeating musical passages vary widely throughout the literature. We will define a *pattern* as any set of musical excerpts, and refer to these excerpts as the pattern's *occurrences*. Each pattern is labelled as either *significant* or *trivial*. We ascribe no external meaning to a pattern's significance; if an analyst deems a pattern "significant," we assume that to mean nothing more and nothing less than "this analyst would consider this pattern to be of particular interest in this piece."

2. RELATED WORK

Early musical pattern discovery algorithms were string-based, dealing primarily with monophonic inputs [3, 12, 16, 22, 33, 34]. Geometric pattern discovery approaches were developed to deal with polyphonic music, most often representing music as sets of (onset time, MIDI pitch) or (onset time, morphetic pitch) pairs [5,9,21,29,36].

There are many proposed methods of reducing the number of patterns that are discovered. An early heuristic was based on the concept that patterns are more significant when they are unexpected, and so should be given more weight if their occurrences are statistically unlikely given the distribution of pitches and duration values elsewhere in the piece [11]. Other heuristics take inspiration from information theory, reasoning that a pattern is significant if it can be said to explain a great deal of the redundancy of the piece containing it [**?**, 1, 23, 26–28]. Research by Lartillot has attempted to emulate the cognitive processes that cause a listener to associate present material with past material stored in memory [18–20]. Velarde et al. use the Haar wavelet transform to analyze symbolic music for purposes of pattern discovery by considering the pitch of a monophonic melody as a signal; implicit in this approach is the assumption that this transform allows access to a higher-level musical property that is more relevant to perception of melodic similarity than raw pitch data [39, 40]. Clustering-based approaches are relatively uncommon, often using the clustering aspect for some form of visualization [2,4,17]. Directly comparing the performance of these algorithms is not straightforward. Evaluation against a set of human annotations is standard for the MIREX task in this area [8], but the ground truth used for evaluation there is sparse and taken from several different areas, making it difficult to extrapolate the accuracy of a single method to the method's quality as a whole.

The largest study that investigated human annotations of pattern importance was performed by Collins et al. [10], who asked 20 musically trained subjects to classify patterns in one of Chopin's Mazurkas based on how likely they would be to discuss each pattern if asked to write an essay analysing the whole piece. Principal findings from this study were that a small number of features could be used to explain 70% of the variation between the patterns' importance ratings. The results from this study do not constitute a pattern discovery method in and of themselves, but they speak to the possibility that human significance judgements might be consistent enough to inform a pattern discovery method.

3. APPROACH

This section describes the setup necessary to define our proposed approach. Three main steps are necessary: assembly and feature extraction on the data set, defining a training method that learns an embedding of our feature space, and clustering on the embedded data set.

3.1 Dataset Assembly

The dataset under investigation is the Meertens Tune Collection Annotated Corpus (MTC-ANN) [38], which comprises 93 patterns among 360 monophonic Dutch folk songs in 26 tune families. Each pattern has, on average, 17.8 occurrences, and the average length of an occurrence is 4.14 notes. The small size of the occurrences in this corpus is worth concern; out of the 1657 total occurrences identified across all patterns, 433 of them are three notes long, and 323 of them contain only two notes. It is not obvious that such short snippets of music permit the extraction of useful information about pattern significance. Additionally, these patterns were found as part of a larger annotation process which emphasized finding features of songs that are useful in separating the songs into tune families [37, 41]; our approach uses the assumption that the same implicit significance measure was used for all annotations, but the annotators may have changed their criteria depending on the tune family under consideration. Still, the high number of occurrences per pattern is beneficial for our clustering method, since it is in general easier to detect a denser cluster than a sparser one. Since MTC-ANN contains only segment-like occurrences (i.e., occurrences that contain every note within a single time interval) we will deal only with this type of occurrence. Section 5 discusses how more general subsets of notes might be used instead.

We must find some trivial patterns to compare with this set of significant patterns. We operate under the assumption that any pattern not specified as significant in MTC-ANN is implicitly judged to be trivial. The SIARCT-C (Structure Induction Algorithm for R superdiagonals with Compactness Trawling and Categorization) algorithm, described in [9] and distributed in the PattDisc software toolkit [7], will be used to generate a set of negative examples. In MTC-ANN, patterns are discovered between songs that lie in the same tune families; to match this, and to avoid dealing with the gargantuan number of patterns that would be found looking in 360 songs at once, we find our trivial patterns strictly within the bounds of each tune family. In this process, SIARCT-C will likely find some of the motifs that were identified in MTC-ANN. We remove any pattern from our collection of trivial patterns if it matches one of the significant patterns too closely, where a match is registered if at least half of the occurrences are identical.

We stated previously that pattern discovery could be defined as a clustering problem by considering the input to be "all possible passages of music" extracted from a given input. The occurrences of these discovered trivial patterns will serve as a stand-in for this set of all possible passages to avoid having to work with it directly. This should not cause any loss of generality, since the huge number of patterns returned by SIARCT-C does not significantly help us narrow down the search space. This run of SIARCT-C can be interpreted as a pre-processing step on the set of all possible musical passages from the dataset, where passages are removed if they do not repeat often enough to have a chance at being part of a significant pattern.

3.2 Feature Extraction

Instead of representing each occurrence of our patterns as an ordered sequence of notes and durations, we represent them as feature vectors, to allow the inclusion of information about the context of each occurrence; for example, we quantify how much its pitch range differs from that of the song containing it. Most of the features we extract are defined in the documentation of jSymbolic2, a software tool for extraction of features from symbolic music files [24, 25], but others were devised in previous work specifically for the purpose of extracting useful information from very short passages of music [13]. They fall into five broad categories, here listed with the number of features each contains:

- **Pitch-Related:** Features relating solely to the ordered pitch values of each note (n = 21).
- **Rhythm-Related:** Features relating solely to the ordered duration values and metrical positions of each note (n = 10).
- **Contour-Related:** Features using both pitch and duration values to describe how the sequence changes over time (n = 6).
- **Histograms:** Multi-valued features indicating the raw number of notes with a particular pitch class or duration (n = 31).
- **Context-Related:** Features comparing properties of the occurrence to properties of the song containing it (n = 38).

All together, this yields a feature set of size 106. A detailed list of features and their definitions is provided in the supplementary material to this paper.

3.3 Learning an Embedding

We now have a dataset containing 11,471 short passages of music, each represented as a 106-dimensional feature vector. 1,657 of these are categorized into one of 93 significant patterns, where all members of a single pattern are considered similar to one another and dissimilar to members of any other pattern. 9,814 of our data points are from trivial patterns, and we consider each of these to be dissimilar to every other data point.

The neural network used to learn the embedding is a fully-connected feed-forward network with two hidden layers, each containing 100 nodes, using dropout and batch normalization, and an output layer of size five. The training process for this network takes pairs of data points as input. We use three types of pairs, in equal measure: pairs where both points are members of the same ground-truth pattern (labelled as similar), where both points are members of different ground-truth patterns (labelled as dissimilar), and where one point is from a ground-truth pattern and one is from a trivial pattern (labelled as dissimilar). Both data points are fed separately through the hidden layers, transforming them both into lower-dimensional vectors of length 5, and then their difference is taken; the L1 norm of this difference is the output of the network. Training implements a hinge loss that encourages the output to be near zero if the two input data points are labelled as similar, and encourages the output to be above some margin value (here set to 1) if the points are labelled as dissimilar. Training is halted when the loss on a validation set does not decrease for 1,000 epochs.

The effect of this process is to train the network to learn an embedding of the 106-dimensional feature space into a 5-dimensional space where occurrences of significant patterns are clustered together, and all clusters are placed far away from one another.

3.4 Clustering

We use the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm to cluster our transformed data points [14]. DBSCAN labels sparser areas of the dataset as containing unlabelled noise, which is ideal for this application, as we expect that most data points will represent musical content not part of significant patterns.

DBSCAN takes two parameters; a minimum cluster size, which we set to 3 based on the size of the smallest patterns in our ground truth, and a value ϵ which, roughly, characterizes how close together points must be near the centers of the discovered clusters. Choosing an optimal ϵ is not straightforward; too high and the data will be partitioned into a few large clusters, but too low and most points in the dataset will be taken as noise. The designers of the algorithm recommend the use of a *k-dist graph* to estimate an optimal value. The k-dist graph of a dataset is formed by finding the k-th nearest neighbor of each data point, calculating the distance between each point and its identified neighbor, and sorting these distances in descending order. On a dataset well-suited to clustering, this sequence of distances should form a curve with a single sharp bend; to one side of this bend, where the distances are high, most points are noise points not part of any cluster, and on the other side of the bend, most points are close to their k-nearest neighbor and are likely members of well-defined clusters. The recommended value for ϵ is the value of the graph at this point of bending. Since most points in our dataset might be considered noise, we will evaluate our method over a range of values for ϵ that are further down the curve than this bending point, which will force clusters to be more tightly packed.

An image of the k-dist graph for one of the experiments run in Section 4 is included in the supplementary material to this paper.

4. EVALUATION

We use cross-validation to evaluate the method's performance. The data is split into training / validation / testing sets based on tune family, to ensure that we train the clustering method on the patterns contained in a particular set of songs and then test that method by discovering patterns in a totally separate set of songs. Each set includes all pat-

	Num. Clusters Ratio	Median Cluster Size	All Points		Significant Points	
			Homogeneity	Completeness	Homogeneity	Completeness
Embedding						
ϵ_5	**2.14 ± 0.39**	**6.50 ± 0.40**	0.25	0.12	0.37	0.62
ϵ_{10}	4.76 ± 1.02	5.90 ± 0.21	0.44	0.13	0.63	**0.67**
ϵ_{15}	6.68 ± 1.37	6.00 ± 0.28	0.53	0.13	**0.69**	0.66
ϵ_{20}	8.66 ± 1.67	5.20 ± 0.18	0.54	0.12	**0.69**	0.63
ϵ_{25}	10.15 ± 2.02	5.10 ± 0.08	**0.56**	0.12	0.61	0.64
PCA						
ϵ_5	2.70 ± 0.76	4.50 ± 0.20	0.15	0.13	0.21	0.53
ϵ_{10}	4.97 ± 1.23	4.60 ± 0.21	0.28	0.13	0.37	0.58
ϵ_{15}	6.94 ± 1.39	4.40 ± 0.22	0.37	**0.14**	0.47	0.60
ϵ_{20}	8.42 ± 1.70	4.40 ± 0.21	0.45	**0.14**	0.55	0.61
ϵ_{25}	10.96 ± 2.33	4.40 ± 0.22	0.52	0.12	0.63	0.62

Table 1: Results of the experiments described in Section 4. Plus-minus signs (±) indicate the standard error of a statistic over the five test sets. Standard errors for the homogeneity and completeness statistics are consistently $\ll 0.05$ and are omitted for readability.

terns, significant and trivial, that lie in its designated tune families.

The training and validation sets must be assembled into pairs before the neural network can take them as input. We generate three sets of pairs corresponding to the categories defined in section 3.3. We take all possible unique pairs of the first category that our training set permits—that is, all possible pairs involving two distinct occurrences of the same significant pattern—and reduce the next two categories to the same size as the first through random sampling without replacement. The total number of data point pairs generated via this process varies depending on which tune families are selected from MTC-ANN for validation and training, since every tune family has a different number of identified patterns, but in practice this number lies in the range from $20,000$ to $40,000$.

Once the network has been trained on this set of pairs, we use it to reduce the test set into vectors that lie in the learned subspace. We run DBSCAN with five different values of ϵ, which are estimated by building a k-dist graph on the test set with k=3. For all test sets, this graph has a very sharp bend near the 5th percentile. Denoting the value at the *n*th percentile of the k-dist graph as ϵ_n, we test DBSCAN with values of $\epsilon_5, \epsilon_{10}, \epsilon_{15}, \epsilon_{20}$, and ϵ_{25}. Each of these clusterings can finally be compared directly with the patterns in the test set.

We contrast this method with one that uses Principal Component Analysis (PCA) to reduce the dimensionality of the dataset instead of a learned embedding. The test sets are processed with PCA and the five components of highest magnitude are retained. DBSCAN is used to cluster the result, with the same procedure for estimating ϵ as before, using a k-dist graph built on the PCA-reduced data.

Testing proceeds with 5-fold cross-validation. This does not evenly divide the number of tune families (26), but each tune family has a different number of patterns, and each pattern has a different number of occurrences, so no truly equitable division is possible anyway. The neural network is implemented using PyTorch 1.0 [30], while the implementation of DBSCAN is from scikit-learn [31]. Code for running these experiments is available at `https://github.com/timothydereuse/musical-pattern-clustering`.

4.1 Results

In Table 1, the *Num. Clusters Ratio* column compares the number of clusters obtained in each experiment to the number of ground-truth patterns in each test set. The *Median Cluster Size* represents the median number of occurrences within each pattern, averaged over all test sets. This somewhat awkward metric is necessary because using a straight average would skew too high to accurately represent the data; on every test set, both the PCA and embedding approaches tend to return one or two large patterns with hundreds of “noisy” occurrences, a phenomenon further discussed in Section 4.2.

Traditional metrics of classification accuracy such as precision and recall are not applicable in a clustering task where the number of classes is itself being predicted. The metrics used here are *homogeneity* and *completeness*, two complementary measures of clustering validity which compare a clustering to a ground-truth [35]. The homogeneity of a clustering measures the degree to which clusters in the output are comprised of a single class, assigning a score of 1 if every pattern in the output contains data points from only a single ground-truth pattern, even if some patterns are split apart. Similarly, the completeness measures the degree to which classes in the input are mapped to a single cluster of the output, assigning a score of 1 if every pattern in the input stays unbroken when mapped to patterns in the output, even if some patterns get merged together. We evaluate these two metrics against the ground truth in two different ways: first, considering all points in the test set, and then considering only the points

Figure 1: These four two-note occurrences are part of different patterns in MTC-ANN, but the embedding method merges all of their patterns into one.

(a) Three occurrences from a pattern in MTC-ANN erroneously included in a large, noisy cluster by the embedding method.

(b) Three occurrences from the same pattern in MTC-ANN correctly clustered together by the embedding method.

Figure 2

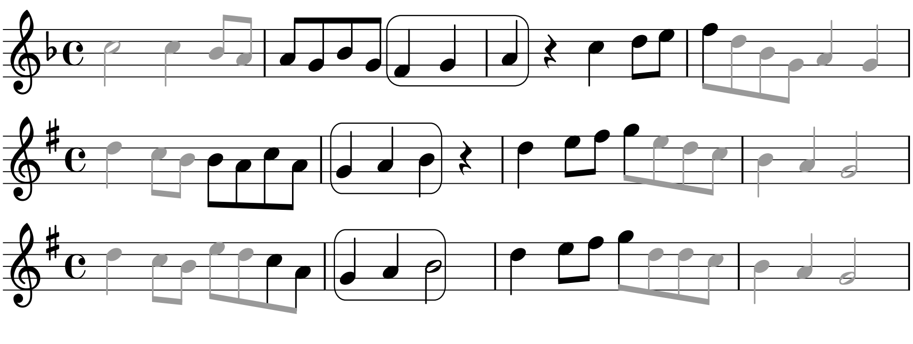

Figure 3: This pattern found by the embedding method is not present in MTC-ANN; however, there is a short three-note pattern (marked by boxes) included in MTC-ANN whose occurrences lie *within* this pattern.

that correspond to significant patterns in the ground truth. Note the very low score for completeness across both the embedding and PCA methods; because the majority of the points in the dataset are labelled as noise, misclassifying any of them into clusters effectively "breaks up" the noise class and lowers the completeness. If we ignore the noise and focus solely on where significant patterns are clustered, we note that they are mostly preserved. The supplementary material to this paper contains an additional table showing the effects of excluding individual feature categories (as defined in Section 3.2) on the clustering.

4.2 Examples

We pick one of the test sets from the row ϵ_{15} of Table 1 to investigate further. This particular clustering finds 21 patterns within six tune families that together contain a total of 77 songs; within these six tune families, MTC-ANN notes 18 significant patterns. Most of the 21 patterns in this clustering do not correspond to patterns in MTC-ANN. In particular, two of the patterns have over 800 occurrences each, almost entirely containing longer occurrences from trivial patterns, with some longer occurrences from significant patterns scattered in as well. In each figure, occurrences are marked with black notes, whereas greyed-out notes show the context in which each occurrence lies.

Only six of the 21 identified patterns have significant overlap with the patterns in MTC-ANN. One reason for this low number is that the clustering method has merged some of the identified patterns together. Figure 1 shows four occurrences from one of these six clusters; all four of these occurrences lie in different patterns of the ground truth, and the full cluster (not shown here, for lack of space) comprises the union of these four original ground-truth patterns. Since all of these occurrences contain only two notes, it is likely that these particular patterns were designated as significant due to their metrical placement in their original songs. It is incorrect that these patterns were merged together, but the fact that *only* these patterns were merged together is notable. The trivial patterns found by SIARCT-C have no shortage of descending intervals that the algorithm might have added to this particular cluster, and yet it contains only descending intervals that were marked as significant by human annotators. This suggests that our subspace-learning neural network has learned something from the "context-related" features mentioned in Section 3.2 that relates to how the human annotators decided which two-note intervals in the original songs merited significance: not enough to separate these four patterns from each other, but enough to separate them from the rest of the dataset as a group.

Figure 2b shows another notable error made in this clustering. One of the ground-truth patterns in this test set is quite large and heterogeneous, comprising 20 occurrences each containing eight or nine notes. Where the occurrences have a relatively simple contour, the clustering correctly groups them together, but it groups the more complicated occurrences with other unclassifiable, long passages in the size-800 clusters mentioned above. It is likely that, given the small size of the dataset, the learned subspace does not encode a particularly complex conception of melodic similarity, which means that longer patterns are unlikely to cluster together unless their similarities are quite pronounced. Compare the three occurrences in Figure 2a to those in Figure 2b, which were successfully clustered into a single pattern, likely as a result of their more uniform contour, and those in Figure 3, where similarity in contour appears to have caused the embedding method to *extend* a pattern existing in MTC-ANN.

5. CONCLUSIONS

We have demonstrated an approach to discovering patterns in symbolic music that maps passages of music onto a low-dimensional subspace where significant patterns form clusters, using an embedding learned from human annotations of repeated patterns. This method outperforms a traditional dimensionality reduction algorithm on common metrics used to validate clustering results against ground truth. There is evidence that the method is capable of learning some notions of pattern significance from the human annotations; though the agreement is far from perfect, and the number of returned patterns is still high, the current state of the art in pattern recognition struggles to agree with human annotations at all [32]. To more rigorously validate this approach in future research, it would be informative to compare a clustering learned from human annotations with a clustering that uses a distance measure derived from an existing melodic similarity metric.

If we continue to restrict ourselves to segment-like occurrences, then extending this approach to polyphonic music would require only a feature set capable of encoding information about polyphonic occurrences. However, to be able to find patterns within polyphonic sources more generally, we must consider occurrences as subsets of notes instead, which is combinatorially infeasible; for a piece of music with n note onsets, there are $O(n^2)$ possible segments, but $O(2^n)$ possible subsets. To address this, it would be necessary to impose limits on the time-extent and number of notes in each occurrence, or to use an existing polyphonic pattern discovery algorithm as a p re-processing step, as we do here with SIARCT-C.

More ground-truth annotations would undoubtedly increase the accuracy of this approach, but annotations of repeated patterns are expensive to acquire, and information from one set of annotations might not generalize well to other genres. The ability to extract useful information from small sets of repeated patterns would be a more valuable tool for development of practical pattern significance measures. A hypothetical use case for this research would be an interactive pattern-discovery interface where, in lieu of changing the parameters of a manually designed heuristic, users could view a list of patterns and mark some as significant, whereupon the algorithm would re-train on that small set and use its findings to reduce the number of patterns returned to the user.

6. REFERENCES

[1] Teppo E. Ahonen, Kjell Lemström, and Simo Linkola. Compression-based similarity measures in symbolic, polyphonic music. In *Proc. of the 12th International Society for Music Information Retrieval Conference*, pages 91–96, Miami, FL, 2011.

[2] Chantal Buteau and John Vipperman. Melodic clustering within motivic spaces: Visualization in OpenMusic and application to Schumann's Träumerei. In *Proc. of the International Conference on Mathematics and Computation in Music*, Communications in Computer and Information Science, pages 59–66, 2009.

[3] Emilios Cambouropoulos. Musical pattern extraction for melodic segmentation. In *Proceedings of the XII Colloquium of Musical Informatics*, Gorizia, Italy, 2003.

[4] Emilios Cambouropoulos and Gerhard Widmer. Automated motivic analysis via melodic clustering. *Journal of New Music Research*, 29(4):303–317, 2000.

[5] Shih-Chuan Chiu, Man-Kwan Shan, Jiun-Long Huang, and Hua-Fu Li. Mining polyphonic repeating patterns from music data using bit-string based approaches. In *2009 IEEE International Conference on Multimedia and Expo*, pages 1170–1173, 2009.

[6] Tom Collins. *Improved methods for pattern discovery in music, with applications in automated stylistic composition*. PhD Thesis, The Open University, 2011.

[7] Tom Collins. PattDisc-Jun2014. Available online at http://www.tomcollinsresearch.net/publications.html. Accessed 18 November 2017, 2014.

[8] Tom Collins. 2017: Discovery of repeated patterns and sections. http://www.music-ir.org/mirex/wiki/2017:Discovery_of_repeated_themes_\%26_sections. Accessed 3rd March 2019, 2017.

[9] Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. SIARCT-CFP: Improving precision and the discovery of inexact musical patterns in point-set representations. In *Proc. of the 14th International Society for Music Information Retrieval Conference*, pages 549–554, Taipei, Taiwan, 2013.

[10] Tom Collins, Robin Laney, Alistair Willis, and Paul H. Garthwaite. Modeling pattern importance in Chopin's mazurkas. *Music Perception: An Interdisciplinary Journal*, 28(4):387–414, 2011.

[11] Darrell Conklin and Christina Anagnostopoulou. Representation and Discovery of Multiple Viewpoint Patterns. In *Proc. of the International Computer Music Conference*, Havana, Cuba, 2001.

[12] Darrell Conklin and Christina Anagnostopoulou. Segmental pattern discovery in music. *INFORMS Journal on Computing*, 18(3):285–293, 2006.

[13] Timothy de Reuse. *A machine learning approach to pattern discovery in symbolic music*. Master's Thesis, McGill University, Montreal, Canada, 2019.

[14] Martin Ester, Hans-Peter Kriegel, Xiaowei Xu, and Jörg Sander. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[15] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[16] Jia-Lien Hsu, Chih-Chin Liu, and A.L.P. Chen. Discovering nontrivial repeating patterns in music data. *IEEE Transactions on Multimedia*, 3(3):311–325, 2001.

[17] Yong-Kyoon Kang, Kyong-I Ku, and Yoo-Sung Kim. Extracting theme melodies by using a graphical clustering algorithm for content-based music information retrieval. In Albertas Caplinskas and Johann Eder, editors, *Advances in Databases and Information Systems*, Lecture Notes in Computer Science, pages 84–97. Springer: Berlin Heidelberg, 2001.

[18] Olivier Lartillot. A musical pattern discovery system founded on a modeling of listening strategies. *Computer Music Journal*, 28(3):53–67, 2004.

[19] Olivier Lartillot. Automated motivic analysis: An exhaustive approach based on closed and cyclic pattern mining in multidimensional parametric spaces. In David Meredith, editor, *Computational Musical Analysis*, pages 273–302. Springer: Switzerland, 2016.

[20] Olivier Lartillot and Petri Toiviainen. Motivic matching strategies for automated pattern extraction. *Musicae Scientiae*, 11(1):281–314, 2007.

[21] Kjell Lemström and Jorma Tarhio. Searching monophonic patterns within polyphonic sources. In *Proc. of the 6th Conference on Content-based Multimedia Information Access*, pages 1261–1279, 2000.

[22] Chih-Chin Liu, Jia-Lien Hsu, and Arbee LP Chen. Efficient theme and non-trivial repeating pattern discovering in music databases. In *Proc. of the International Conference on Data Engineering*, pages 14–21. IEEE, 1999.

[23] Corentin Louboutin and David Meredith. Using general-purpose compression algorithms for music analysis. *Journal of New Music Research*, 45(1):1–16, 2016.

[24] Cory McKay. *Automatic genre classification of MIDI recordings*. Master's Thesis, McGill University, Montreal, Quebec, 2004.

[25] Cory McKay, Tristano Tenaglia, and Ichiro Fujinaga. jSymbolic2: Extracting features from symbolic music representations. In *Proc. of the 17th International Society for Music Information Retrieval Conference*, Late-Breaking Session. New York City, NY, 2016.

[26] David Meredith. Music analysis and Kolmogorov complexity. In *Proceedings of the 19th Colloquio di Informatica Musicale*, 2012.

[27] David Meredith. Analysis by compression: Automatic generation of compact geometric encodings of musical objects. In *Proc. of The Music Encoding Conference*, Trieste, Italy, 2013.

[28] David Meredith. COSIATEC and SIATECCompress: Pattern discovery by geometric compression. In *Proc. of the 14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, 2013.

[29] David Meredith, Kjell Lemström, and Geraint A. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in Pytorch. In *Proc. of the 31st Conference on Neural Information Processing Systems*, Long Beach, CA, 2017.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[32] Iris Yuping Ren, Hendrik Vincent Koops, Anja Volk, and Wouter Swierstra. In Search of the Consensus Among Musical Pattern Discovery Algorithms. In *Proceedings of the 18th International Symposium on Music Information Retrieval*, Suzhou, China, 2017.

[33] Pierre-Yves Rolland. Discovering patterns in musical sequences. *Journal of New Music Research*, 28(4):334–350, 1999.

[34] Pierre-Yves Rolland and Jean-Gabriel Ganascia. Automated motive oriented analysis of musical corpuses: A jazz case study. In *Proc. of the International Computer Music Conference*, pages 240–243, 1996.

[35] Andrew Rosenberg and Julia Hirschberg. V-Measure: A conditional entropy-based external cluster evaluation measure. In *Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420, Prague, Czechia, 2007.

[36] Esko Ukkonen, Kjell Lemström, and Veli Mäkinen. Geometric algorithms for transposition invariant content-based music retrieval. In *Proc. of the 4th International Conference on Music Information Retrieval*, pages 193–199, Baltimore, MD, 2003.

[37] Peter van Kranenburg. *A computational approach to content-based retrieval of folk song melodies*. Doctoral Thesis, Utrecht University, 2010.

[38] Peter van Kranenburg, Martine de Bruin, Louis P. Grijp, and Frans Wiering. The Meertens tune collection. *Meertens Online Reports*, 1, 2014.

[39] Gissel Velarde and David Meredith. A wavelet-based approach to the discovery of themes and sections in monophonic melodies. *Music Information Retrieval Evaluation Exchange (MIREX 2014), Taipei, Taiwan*, 2014.

[40] Gissel Velarde, Tillman Weyde, and David Meredith. An approach to melodic segmentation and classification based on filtering with the Haar-wavelet. *Journal of New Music Research*, 42(4):325–345, 2013.

[41] Anja Volk and Peter van Kranenburg. Melodic similarity among folk songs: An annotation study on similarity-based categorization in music. *Musicae Scientiae*, 16(3):317–339, 2012.