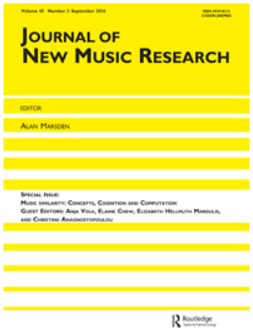


Journal of New Music Research

ISSN: 0929-8215 (Print) 1744-5027 (Online) Journal homepage: http://www.tandfonline.com/loi/nnmr20

Evaluating the Role of Repeated Patterns in Folk Song Classification and Compression

Peter Boot, Anja Volk & W. Bas de Haas

To cite this article: Peter Boot, Anja Volk & W. Bas de Haas (2016) Evaluating the Role of Repeated Patterns in Folk Song Classification and Compression, Journal of New Music Research, 45:3, 223-238, DOI: 10.1080/09298215.2016.1208666

To link to this article: http://dx.doi.org/10.1080/09298215.2016.1208666

Published online: 25 Jul 2016.

Journal of New Music Research, 2016
Vol. 45, No. 3, 223–238, http://dx.doi.org/10.1080/09298215.2016.1208666

Routledge
Taylor & Francis Group

Evaluating the Role of Repeated Patterns in Folk Song Classification and Compression

Peter Boot, Anja Volk and W. Bas de Haas

Utrecht University, The Netherlands

(Received 15 October 2015; accepted 20 June 2016)

Abstract

According to musicological studies on oral transmission, repeated patterns are considered important for determining musical similarity in folk songs. In this paper, we study the relevance of repeated patterns for modelling similarity and compression in a retrieval setting. Using a dataset of 360 Dutch folk songs, we compare the classification accuracy of both humanly annotated patterns and automatically retrieved patterns by means of a pattern discovery algorithm. A framework is proposed to use these patterns for compression and classification in tune families. The annotated patterns allow us to compress the songs by 60% at the expense of a 3 percentage points decrease in classification accuracy. However, none of the automatic pattern discovery algorithms is able to reach a similar combination of compression ratio and retrieval accuracy. We conclude that repeated patterns are relevant for similarity estimation and compression, but that the state of the art in automatic pattern discovery cannot compete with expert annotations in this retrieval setting.

Keywords: melodic similarity, repeated patterns, compression, folk songs, tune families

1. Introduction

This paper investigates the role of repeated patterns for establishing similarity between related folk songs. Similarity is a fundamental concept in the area of Music Information Retrieval (MIR), which researches methods to organize digitized music collections using music similarity, in order to facilitate the retrieval of musical pieces from a large collection that are similar to a given query. Similarity is also a central concept in Cognitive Science, which has introduced several formalized models for the human assessment of similarity (Goldstone & Son, 2005). However, there exists no comprehensive approach to similarity in the domain of music, which poses serious challenges for the modelling of music similarity in MIR. Musicological and cognitive studies suggest that repeated *musical patterns* that are transformed up to a certain extent, yet not beyond recognition, are essential for the human assessment of music similarity (Volk, De Haas, & Van Kranenburg, 2012). Folk music provides a specifically interesting case study of the relationship between musical patterns and similarity. Since this music evolved through oral transmission (Karpeles, 1968), variation to the musical material is introduced in this process. In ethnomusicology, folk songs that are supposed to have a common 'ancestor' in the line of oral transmission are grouped into a *tune family* (Bayard, 1950), which contains similar songs. Cowdery (1984) considers the detection of *shared musical patterns* between songs as an important criterion for determining tune families.

We study the relation between folk song similarity and the existence of shared musical patterns in songs belonging to the same tune family in a *computational manner*. More specifically, we focus on similarity relations in a collection of Dutch folk songs of the *Meertens Tune Collections*.[1] The songs in this collection have been classified by musicological experts into tune families. Since the historical origin of the songs is not known from documentary evidence, the experts relied in the classification process on the perceived musical similarity of the songs, taking a holistic and intuitive approach to assess the similarity between songs. If the similarity between songs was considered high enough to assume a genetic relationship, these songs were placed into the same tune family. No systematic approach has been developed regarding what musical dimensions contribute to this perceived musical similarity

[1] http://www.liederenbank.nl/mtc, accessed 14 October 2015.

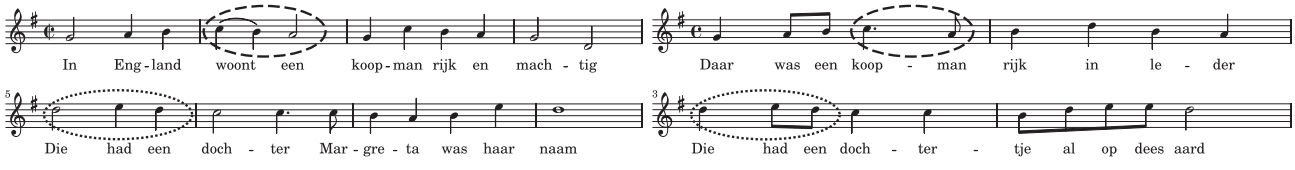
Correspondence: Anja Volk, Utrecht University, Information and Computing Sciences, Buys Ballot Laboratory, Princetonplein 5, Utrecht, 3584 CC Netherlands. E-mail: A.Volk@uu.nl

(Volk & Van Kranenburg, 2012). For understanding these underlying dimensions, Volk and Van Kranenburg (2012) carried out an annotation study with the experts, revealing that melodic contour, rhythm, lyrics and motifs are important ingredients for the similarity between the songs. From a numerical evaluation of 360 songs, it appeared that *motifs* were the most decisive factor for the establishment of similarity between these songs. The experts were asked subsequently to *annotate* the motifs they considered important for the tune family classification. This process resulted in two datasets, the first being a collection of 360 songs divided into 26 tune families as previously used by Volk and Van Kranenburg (2012). This set is called the Annotated Corpus (MTC-ANN, Volk & Van Kranenburg, 2012; Van Kranenburg, de Bruin, Grijp, & Wiering, 2014). The second dataset contains the characteristic motifs for these 360 songs as annotated by the musicological experts, called the *Annotated Motifs*. An example of two such motifs can be seen in Figure 1. The motifs have been annotated in a post hoc manner, hence after the classification of the songs into tune families, in order to elucidate the underlying similarity relations of songs belonging to the same tune family. We therefore want to computationally verify the intuition of the musicologists that the annotated motifs from MTC-ANN are characteristic enough to successfully group these folk songs into tune families.

The musicologists defined a motif as a short fragment of a song that also occurs in other songs of the same tune family. Hence, a motif is *characteristic* of a particular tune family such that it serves as a basic cue to detect the relation between the melodies (Volk & Van Kranenburg, 2012). As Figure 1 illustrates, the corresponding repetitions in the motifs are primarily located in the melodic and rhythmic dimensions, while slight variations between the occurrences are possible. The average length of a motif in the set of Annotated Motifs is four notes. In this context, a motif is defined by its repeating character, hence we can generalize a motif to a *pattern that repeats*. The only difference between a motif and a pattern is that a motif is characteristic of a particular tune family, which is not necessarily the case for a pattern. From a computational point of view, the experts' annotation approach is therefore related to the process of automatically finding characteristic repeated patterns between songs. Over the past decades, many pattern discovery algorithms have been developed that aim to find *perceptually important* patterns: repetitions that are noticeable by a human listener and therefore considered important, mainly in the dimensions of melody and rhythm. Based on the insights provided by Cowdery (1984) and Volk and Van Kranenburg (2012) that motifs or repeated patterns are important for tune family classification, we also study the effect of *automatically* discovered patterns as a means to classify folk songs.

Sequential alignment algorithms have been proven successful for the classification of folk songs into tune families (Hillewaere, Manderick, Conklin, & Ehu, 2012; Van Kranenburg, Volk, & Wiering, 2013). Despite the fact that motifs tend to play a major role in the human assessment of folk song similarity, the use of these patterns in computational settings for tune family classification has not been studied extensively. Therefore, we use annotated or automatically discovered patterns for the classification. Because of the considered importance of these motifs by musicological experts, we assume that repeated patterns are more *stable* in the light of oral transmission than non-repeating parts. We therefore expect that these stable segments contain enough characteristic information to successfully classify songs into the right tune families.

By removing all non-distinctive parts from a song, we construct a *sparse melody* of that song, which contains only those notes that are contained in a repeated pattern. The sparse melody is therefore a compressed representation of the original song and is constructed from the discovered repeated patterns. To test whether these patterns contain enough distinctive information with respect to the original tune family, we use the compressed songs in a *retrieval setting*. This way, the quality of the repeated patterns can be measured by the retrieval accuracy of the tune family classification of folk songs. We assume that the compressed representations contain enough information for a successful classification. Therefore, we test whether the retrieval accuracy for the MTC-ANN folk song dataset is not influenced to a great extent when we use compressed songs instead of the original songs for the classification task. In a research field where more and more data becomes available, compression becomes increasingly important for processing large music collections without computational and retrieval performance loss. Repeated patterns can be useful to reduce the amount of information needed for retrieval in a preprocessing step. Hence, we classify folk songs based on their compressed representation to study the relevance of repeated patterns for modelling similarity between folk songs.



(a) First two phrases of record 70079, strophe 1. (b) First two phrases of record 73788, strophe 1.

Fig. 1. Two excerpts of folk songs from MTC-ANN, both from the tune family *'Daar was een koopman rijk en machtig'*. Two annotated repetitions from the Annotated Motifs are visualized.

1.1 Contribution

In this paper, we study the relevance of repeated patterns in folk songs for similarity estimation and compression in a retrieval setting. We compare patterns extracted by musicological experts to patterns extracted from a symbolic encoding of the folk songs employing six state-of-the-art pattern discovery algorithms, by using them as a means to *compress* a song. We show that annotated patterns allow one to compress the songs by 60% at the expense of a 3 percentage points decrease in tune family classification accuracy. However, none of the automatic pattern discovery algorithms is able to reach a similar combination of compression ratio and retrieval accuracy. These results are placed into context by discussing biases towards the selected discovery algorithms, sparse melody construction, datasets, similarity calculation, and naive retrieval baselines. We conclude that repeated patterns are relevant for similarity estimation and compression, but that the state of the art in automatic pattern discovery cannot compete with expert annotations in this retrieval setting.

The remainder of this paper is structured as follows: Section 2 contains the relevant related work. In Section 3, the notation used in the subsequent sections is discussed. Section 4 describes the framework we developed for using patterns for compression and classification. In Section 5, the experiments and results are explained. The results are discussed in Section 6 and in Section 7 our conclusions are given.

2. Related work

The role of repeated patterns for establishing music similarity has been studied elaborately in music cognition and computational research. According to cognitive studies about the assessment of similarity in music that contains repeated patterns, both novice and expert listeners appeared to be able to cluster these patterns into groups (Mélen & Wachsmann, 2001; Ziv & Eitan, 2007). This outcome confirms the relevance of repeated patterns from the cognitive perspective. For the specific case of capturing the variations introduced through oral transmission, both theoretical studies and computational approaches have investigated the role of repeated patterns (Bayard, 1950; Conklin & Anagnostopoulou, 2001; Cowdery, 1984; Van Kranenburg et al., 2013; Wiora, 1941). Moreover, automatically discovered patterns have been used for the classification of songs in musical genres (Karydis, Nanopoulos, & Manolopoulos, 2006; Lin, Liu, Wu, & Chen, 2004), geographic origin (Conklin, 2009) and tune families (Van Kranenburg, Volk, & Wiering, 2012).

Most pattern discovery algorithms to date aim to find perceptually meaningful repetitions in either a single song or across multiple songs. For the remainder of this paper, we call finding repetitions in a single song *Song Pattern Discovery* and finding repetitions across multiple songs *Tune Family Pattern Discovery*.[2] We refer the reader to Janssen et al. (2013) for a recent overview of pattern discovery algorithms. Since 2014, the yearly MIREX task on the *Discovery of Repeated Themes and Sections* helps in evaluating the output of such algorithms and makes it possible to compare them. The output is evaluated against a *ground truth* consisting of pattern annotations from five (classical) pieces (Collins, 2014a). Metrics are introduced to evaluate the output, which are extensions of the default precision, recall and F_1 score metrics (Manning, Raghavan, & Schütze, 2009).

The use of repeated patterns for compression allows us to evaluate the output of a pattern discovery algorithm differently, namely by measuring how well the discovered patterns are suitable for the classification of folk songs into tune families using compression. By doing so, we test the relevance of the discovered patterns within a specific task, instead of comparing the patterns to an annotated ground truth. This approach has also been suggested by Meredith (2015), proposing a classification method that combines pattern discovery and compression. Different point-set compression algorithms (pattern discovery algorithms that treat a musical piece as a set of points in a multidimensional space) are compared in the context of folk song classification. The similarity between the folk songs is determined by calculating the *Normalized Compression Distance* (Li, Chen, Li, Ma, & Vitanyi, 2004) between each pair of songs. These pattern discovery algorithms, which are also compression algorithms by design, perform reasonably well on classification. However, the resulting accuracy is not as good as aligning and comparing folk songs at the note-to-note level. With the 360 songs of MTC-ANN, Van Kranenburg et al. (2013) achieved a classification accuracy of 99% using Note to Note Alignment.

Where Meredith (2015) solely uses pattern discovery to calculate the compression distance, our approach uses *melodic* information of the compressed songs to determine the similarity between the songs. Moreover, we search for repeated patterns between all songs belonging to one tune family instead of between all pairs of songs. On top of that, this study focuses on the comparison between annotated patterns and automatically retrieved patterns from a wide range of algorithms, whereas the work of Meredith (2015) specifically aims at comparing the performance of point-set compression algorithms.

3. Preliminaries

Let $D = \{s_1, s_2, ..., s_N\}$ be a folk song dataset, consisting of N songs. A function $\mathcal{T}(s) : s \rightarrow t$ returns the tune family of a song s. The set F represents the collection of tune families present in D, such that $\forall s \in D : \mathcal{T}(s) \in F$. The set $F_t = \{s \mid s \in D \wedge \mathcal{T}(s) = t\}$ contains all songs s that are all labelled with tune family t.

Two songs u and v belong to the same family if $\mathcal{T}(u) = \mathcal{T}(v)$. The length of a song (i.e. the number of notes) is

[2] In most literature, these two discovery approaches are referred to as *intra opus* and *inter opus* pattern discovery.

denoted by $|s|$, whereas the size of the dataset is written as $|D|$. Analogous, $|F|$ represents the total number of tune families in the dataset.

A pattern discovery algorithm returns a set of patterns P, given either a single song or a set of songs. This can be written as a function $\mathcal{P}_{song} : s \rightarrow P$ that extracts a set of patterns P_s from a single song s and a function $\mathcal{P}_{family} : \{s \in D\} \rightarrow P$ that returns a set of patterns P_s found in a set of songs. In the case of Tune Family Pattern Discovery, the latter function can be defined as $\mathcal{P}_{family} : F_t \rightarrow P$ for a particular tune family t. Here, P contains patterns that occur in songs of the same tune family. Each pattern $p \in P$ consists of a set of occurrences, denoted by O_p.

4. Method

We assess the relevance of repeated patterns by measuring the classification accuracy on the compressed songs. For this, we extract patterns from a song, compress them based on these patterns, calculate the similarity between the pieces in the dataset and perform the classification. If repeated patterns are indeed important for the classification of folk songs into tune families, the classification of these compressed songs will give comparable results, compared to the classification accuracy on the uncompressed songs. Our testing framework is outlined in Figure 2 and needs a dataset of folk songs as input, as described in Section 4.1. Then, four steps are performed:

Pattern Discovery This incorporates the extraction of repeated patterns from each song in the dataset, either by Song Pattern Discovery or Tune Family Pattern Discovery. The output of this step contains either the motif annotations by the musicologists or automatically discovered patterns using the output of a pattern discovery algorithm. The implemented algorithms for this stage are detailed in Section 4.2.

Sparse Melody Construction This step constructs a sparse melody for each song in the dataset, based on the patterns retrieved in the previous step. Notes that are not contained in a discovered pattern will be removed and therefore not be present in the sparse melody. We can define the process of *compression* as the combination of the pattern discovery step and the sparse melody construction step. We test for different construction techniques that are discussed in Section 4.3.

Similarity Calculation The similarity between each pair of constructed melodies is calculated, in order to classify the songs into tune families in the next step. Two different sequence alignment algorithms are used for this, which are explained in Section 4.4.

Classification In the final step, the actual classification is performed. Since we know the correct class label (tune family) of each song, the classification accuracy can be calculated. The details of the classification method can be found in Section 4.5.

In the first three steps, different approaches are implemented in the framework. For example, six different pattern discovery algorithms are compared. This way, we can measure the impact of a certain choice in each step. We therefore test for various configurations of methods and corresponding parameter values. The performance of such configuration is expressed by two different values: the *coverage* and *accuracy*. The coverage value expresses the measure of compression of the folk songs after the sparse melody construction step. The *classification accuracy* expresses the ratio of correctly classified songs. In the preceding of this section, each step of the framework is explained in more detail.

4.1 Dataset

The dataset we used in this research is MTC-ANN, a set of 360 Dutch folk songs divided into 26 tune families. MTC-ANN stems from the Meertens Tune Collections (Van Kranenburg et al., 2014) that contains a total number of 4830 Dutch folk songs in the symbolic Humdrum `**kern` data format (Huron, 1997). Of the 360 songs in this dataset, 354 of them have one or more annotated motifs that were in retrospect important for the tune family classification. According to the maintainers of the dataset, the experts who were in charge of the annotation process did not find any motif in the remaining six songs. On average, there are 3.4 repeated patterns annotated for each song. We refer to this set of motifs as the *Annotated Motifs*. For this study, we use an updated set with annotated motifs that will be released in version 2.0 of MTC-ANN (Van Kranenburg, Janssen, & Volk, forthcoming) where some repetitions are added that had been missed in the original annotation study. In Figure 1, two excerpts of songs from MTC-ANN are visualized, together with two annotated repetitions.

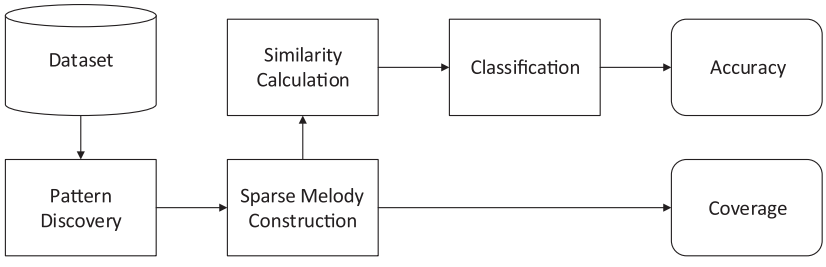


Fig. 2. Schematic overview of the proposed framework.

4.2 Pattern discovery

A pattern discovery algorithm returns for each discovered pattern P a set of pattern occurrences O_p. Each occurrence is described by a set of (onset, pitch) pairs. The algorithms we test are specifically designed for either the discovery of patterns within a single song (Song Pattern Discovery) or pattern discovery across multiple pieces (Tune Family Pattern Discovery). Apart from the pattern discovery algorithms, the manual annotation process of characteristic motifs in Volk and Van Kranenburg (2012) can also be seen as a pattern discovery method, namely as a Tune Family Pattern Discovery. We compare six state-of-the-art pattern discovery algorithms:

- SIATEC (Meredith, Lemström, & Wiggins, 2002)
- COSIATEC (Meredith et al., 2002)
- SIARCT-CFP (Collins, Arzt, Flossmann, & Widmer, 2013)
- PatMinr (Lartillot, 2014a)
- MotivesExtractor (Nieto & Farbood, 2014a, 2014b)
- MGDP (Conklin, 2010)

These algorithms differ from each other in underlying music representation (sequential or symbolic), search domain (Song or Tune Family Pattern Discovery) and discovery approach. For all algorithms except MGDP, the original implementation of the authors was used with slight modifications to support short musical pieces.

SIATEC (Structural Induction Algorithm with Translational Equivalence Classes, Meredith et al., 2002), is a geometric pattern discovery algorithm that treats a melody as a set of points in a multidimensional space. It is developed to return pairs of repetitions within a single song. These pairs are grouped into Translational Equivalence Classes, of which patterns are translational equivalent to each other. A drawback of SIATEC, and also of many other pattern discovery algorithms, is that they yield a large number of patterns. To limit the output, Meredith et al. (2002) proposed a compressed variant of SIATEC, *COSIATEC*. Based on a number of heuristics, only the *best* discovered pattern in the musical piece is selected. Then, all occurrences of this pattern are removed from the piece. This process is repeated until no more pattern can be found. This way, a musical piece can be described by only one occurrence of each pattern, plus all vectors that translate this first occurrence to every other occurrence. Due to the removal step, each note is part of at most one pattern.

Another attempt to limit the output of a pattern discovery algorithm to only *relevant* patterns was the addition of a *compactness trawler* (Collins, Thurlow, Laney, Willis, & Garthwaite, 2010) to the aforementioned algorithms. This resulted in *SIACTTEC* and *COSIACTTEC*. The compactness trawler rejects patterns that are very sparse (i.e. patterns with many gaps) and tries to find better patterns inside the patterns returned by SIATEC or COSIATEC. Another limitation of these algorithms is that they can only handle *exact* repetitions of a pattern; small variations would not be detected. Collins et al. (2013) presented a solution for this with the algorithm *SIARCT-CFP*, where a fingerprinting method was used to categorize inexact pattern occurrences into a single pattern.

Besides these geometric approaches of finding repeated patterns, many pattern discovery algorithms use an underlying *sequential* representation of music. Here, a musical piece is described as a sequence of notes rather than a set of multidimensional vectors. One of such algorithms is *PatMinr* (Lartillot, 2014a), which aims at discovering only patterns that are not subsets of any earlier discovered pattern. On top of that, Lartillot (2014a) proposes a method to prevent the discovery of many nearly identical patterns forced by cyclic structures in the music by explicitly modelling these pattern cyclicities. PatMinr has participated in the *Discovery of Repeated Themes and Sections* MIREX task (Lartillot, 2014b) in 2014.

The *MotivesExtractor* algorithm by Nieto and Farbood (2014a) was initially developed for finding patterns in audio data, but was adapted for symbolic music to participate in the *Discovery of Repeated Themes and Sections* MIREX task. The algorithm makes use of a *key invariant self-similarity matrix*, where pattern occurrences will become visual by diagonal lines in the similarity matrix. By tuning the *strictness* of the diagonal tracing, this algorithm can also discover inexact pattern occurrences.

The last algorithm we discuss is the Maximally General and Distinctive Pattern (MGDP) algorithm by Conklin (2010). All algorithms mentioned earlier were developed for Song Pattern Discovery, whereas the MGDP algorithm is specifically designed to find patterns repeating in a *collection* of songs. It limits the output size by only returning patterns that are *general* in the sense that no other discovered pattern is a subset of another returned pattern and *distinctive* with respect to a second collection of songs, the *anti-corpus*. By choosing the anti-corpus as a set of songs that are distinct from the corpus, only patterns that are characteristic for the corpus will be returned.

Besides the MGDP algorithm, the number of known pattern discovery algorithms that find patterns in *multiple* songs is very low. To overcome this limitation, we propose a method to adapt any Song Pattern Discovery algorithm into a Tune Family Pattern Discovery algorithm without internal modifications to the algorithms. To achieve this, all songs of a tune family are concatenated right before the pattern discovery step in the framework visualized in Figure 2. Then, the concatenated song is used as input for the pattern discovery algorithm, which treats the concatenated piece as a single song. After the discovery, the songs are split into individual pieces again and the pattern occurrences are assigned to the right piece. Pattern occurrences that lie on the boundary of two adjacent songs in the concatenated piece are discarded. This process is shown in Figure 3.

This approach allows us to use all five Song Pattern Discovery algorithms also for Tune Family Pattern Discovery. Due to computational limitations, we discard SIARCT-CFP for this task. Hence, together with the MGDP algorithm, there

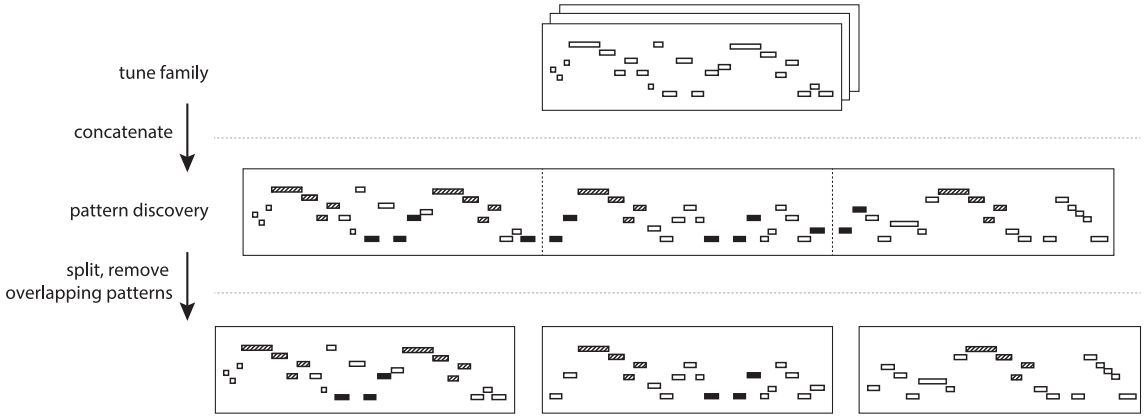


Fig. 3. Visualization of the procedure for Tune Family Pattern Discovery using algorithms only designed for Song Pattern Discovery, two patterns are highlighted. All songs in each tune class are concatenated and used as input to the discovery algorithms. After the pattern discovery, all overlapping patterns (in this example the second and fourth filled pattern) are removed and the concatenated tunes are split into the original pieces again.

are still five algorithms that can be used to find patterns across multiple songs. For the Song Pattern Discovery task, MGDP is the only algorithm that cannot be used. This brings the number of Song Pattern Discovery algorithms also to five. The number of times an algorithm runs in each experiment depends on the discovery task. For Song Pattern Discovery, the algorithm needs to run $|D|$ times, whereas for the Tune Family Pattern Discovery task, the algorithm runs $|F|$ times.

4.3 Sparse melody construction

After the pattern discovery step, a set of discovered patterns is acquired for each song in the dataset. In the sparse melody construction step, for each song $s \in D$ a *sparse melody* is constructed in such a way that only notes that are part of a pattern in P_s will also be part of the constructed melody. This way, each song is compressed in a *lossy* way.

Let $\sigma(s) : s \rightarrow s'$ be a general sparse melody construction function that generates a sparse melody s' of a song s based on the set of repeated patterns P_s. In the following, we introduce three different sparse melody construction approaches used in the framework. Recall that only one of these methods is used in each experimental run of the framework. By varying the construction method, the effect of (high) compression and gaps in the resulting songs can be made clear. These methods are also visualized in Figure 4:

Onset Preserved Construction In this construction method, all notes not belonging to any pattern are replaced by a rest of the same duration. This way, the onset of all notes is preserved in the sparse representation. More formally, the function that constructs the sparse melody using the patterns of a song s to this set of notes can be written as $\sigma_{opr}(s) = \{n \mid n \in s \wedge \exists p \in P(\exists o \in O_p(n \in o))\}$.

Gapless Construction Instead of leaving gaps in parts of the pieces where notes are removed, the notes are shifted in such a way that the notes will follow each other directly without any rest in the constructed melodies. This way, all notes that are present in a pattern are *stitched together* without any gaps caused by the note removal. The same notes are selected as in the Onset Preserved Construction method. The construction function is also the same as in the previous method.

First Pattern Occurrence Construction Some pattern discovery algorithms return a massive amount of patterns, eventually causing the two previous strategies to give less meaningful results. When this is the case, nearly all notes will be present in one of the returned patterns, causing the sparse melody to be nearly the same as the original song. To prevent this, the First Pattern Occurrence Construction method only uses the *first* occurrence of each repeated pattern for the sparse melody construction, instead of *all* occurrences. For the note removal, the same strategy is used as the Gapless Construction method, omitting any gaps caused by the note removal. This function can be written as $\sigma_{for}(s) = \{n \mid n \in s, \exists p \in P_s(n \in o_1, o_1 \in O_p)\}$, where o_1 is the first occurrence of the set of occurrences O_p for a particular pattern p.

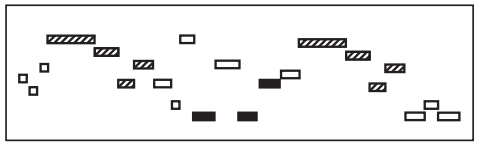
As in the previous step, one of these construction methods can be chosen in an experiment to compress the folk songs based on the discovered patterns. From the original dataset D, a set of sparse melodies D' is created where

$$D' = \{\sigma(s) \mid s \in D \wedge |s| > 0\} \tag{1}$$

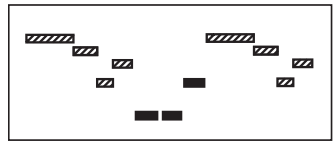
As a result, D' only contains the songs from D that are at least one note long after the sparse melody construction. To describe the measure of compression after this step, the *coverage* of a compressed song is introduced as:

$$\text{Coverage}(s) = \frac{|\sigma(s)|}{|s|} \tag{2}$$

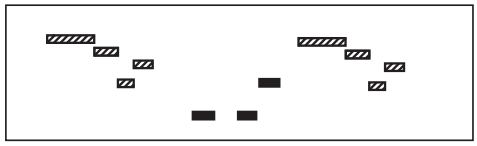
In other words, the coverage is defined as the ratio of notes in the original song that are *covered* by the generated sparse melody. Since the compression ratio can be defined in the



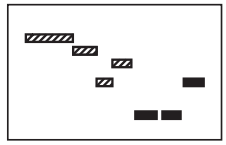
(a) Representation of a song where two patterns are discovered (filled and striped). The white notes will be removed, since they are not part of a pattern.



(b) Onset preserved construction.



(c) Gapless construction.



(d) First occurrence construction.

Fig. 4. The resulting pieces after the sparse melody construction procedure from the first piece of (a). From this figure we can see that the original song length is 22 notes, so the coverage after the construction step is $11/22 = 0.5$ for (b) and (c) and $7/22 \approx 0.32$ for (d).

same way as the coverage in Equation 2, the terms *coverage* and *compression ratio* can be used interchangeably. For the complete dataset, the coverage can be defined as:

$$\text{Coverage}(D, D') = \frac{1}{|D'|} \cdot \left(\sum_{\{s|s\in D \wedge \sigma(s)\in D'\}} \text{Coverage}(s) \right) \tag{3}$$

which is the average of the coverage values for all songs in D'. The songs of which no sparse melody can be constructed (i.e. $|\sigma(s)| = 0$) are not taken into account in the final coverage value since these songs cannot be correctly classified any more. This prevents the coverage to be very low in cases where $|D'|$ is small. The coverage value of the entire dataset is one of the output values from the framework, as can be seen in Figure 2.

4.4 Similarity calculation

For the set of sparse melodies D' we need a measure to determine how similar two songs are. A distance function $d(u, v)$ is defined that represents the distance between the songs u and v. A small distance means a high similarity between the songs, where a large distance corresponds to two non-similar musical pieces.

For each pair of songs in D', this *similarity distance* is calculated using a sequence alignment algorithm (Van Kranenburg, 2010). This method originates from computational biology to find (partial) matches in protein strings. In the musical domain, musical pieces are transformed into a sequential representation (e.g. a sequence of pitch values, intervals, note durations). Then, sequences of two songs are aligned in such a way that the matching score is maximized. Pairs of notes that match according to a predefined *scoring function* result in a positive score, where gaps and mismatches will lead to a penalty. The distance between two songs is then determined by the score of the optimal alignment configuration.

The reason to use this kind of similarity modelling is that sequence alignment has been proven to be an effective similarity measure for monophonic (folk) music (Hillewaere et al., 2012). Two of such alignment methods are implemented in the framework, the first one being *Note to Note Alignment*. Van Kranenburg et al. (2013) have shown that alignment at a note to note level is a good similarity measure for tune family classification of MTC-ANN. They use the Needleman–Wunsch–Gotoh alignment algorithm (Gotoh, 1982; Needleman & Wunsch, 1970) for calculating the optimal alignment. Van Kranenburg et al. (2013) achieved a classification accuracy of 0.99 using a sequential representation of songs with a scoring function that takes into account pitch band, metric weight (a measure of how important a note is for the rhythm) and phrase position (the position of a note in the musical phrase). To reduce the chance of overfitting and to make the effect of the compression more visible, we use Note to Note Alignment on pitch intervals only.

As an alternative to Note to Note Alignment, we also evaluate melodic similarity using higher level information about the musical pieces, such as the melodic shape. This approach has the advantage over Note to Note Alignment that small variations between tunes are not penalized strongly when the melodic contour is roughly the same. This concept is used by Urbano (2013), who developed a global sequence alignment algorithm using B-Splines (De Boor, 1972) to model the melodic shape of a song. This *Melodic Shape Alignment* method has been the best-performing approach for the *Symbolic Melodic Similarity* MIREX task[3] over five years (2010–2014). More specifically, the SHAPEH alignment algorithm is used that treats each pair of three consecutive pitch values as a

[3] http://www.music-ir.org/mirex/wiki/2015:Symbolic_Melodic_Similarity, accessed 14 October 2015.

curve. The actual alignment is done by matching the directions (upwards or downwards) of the begin- and endpoints of the curves.

4.5 Classification

Based on the distances returned by one of the alignment algorithms described in the previous step, all tunes in D' are piecewise compared using one of the sequence alignment methods outlined in the previous section. In the classification step, each song will be assigned to a tune family based on these calculated similarity distances. For this, the k-Nearest-Neighbour classifier (Cover & Hart, 1967) is used. The set of sparse melodies D' is divided into two sets: a test set (of which we want to determine the class labels) and a training set (of which we know the true class labels). For each piece in the test set, the distances to all other pieces in the training set are ranked in such a way that the top k results contain the k closest (most similar) pieces. Since the class labels of all songs are known beforehand, each musical piece will be assigned to the majority class of its k nearest neighbours. We use leave-one-out cross-validation, which means that in each run of the classifier the test set contains only one song and the training set contains all other songs. The cross-validation process consists of $|D|$ rounds, such that each song in D is part of the test set once. In our implementation, $k = 1$, hence it is sufficient to only look at the closest neighbour $\mathrm{NN}(s)$ for each song $s \in D'$:

$$\mathrm{NN}(s) = \left\{ u \mid u \in D' \wedge d(u, s) = \min_{v \in D', v \neq s} d(v, s) \right\}. \quad (4)$$

It is possible that there exists more than one closest neighbour when multiple tunes are at an equal minimum distance from the queried song (i.e. $|\mathrm{NN}(s)| > 1$). In this case, the majority class of the closest neighbours wins. In the event of a tie, the class of a randomly selected closest neighbour is assigned. Since the correct class label of s is known from the original dataset, it can be immediately checked whether the classification is correct (i.e. it is a true positive (TP)) or not:

$$\mathrm{TP}(s) = \begin{cases} 1 & \text{if } \mathcal{T}(s) = \mathcal{T}(\mathrm{NN}(s)) \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

This procedure is repeated for all songs in D'. For each cross-validation run, another song $s \in D'$ is selected. The classification accuracy of the constructed dataset D' can be calculated as follows:

$$\mathrm{Accuracy}(D, D') = \frac{\sum_{s \in D'} \mathrm{TP}(s)}{|D|} \quad (6)$$

Hence, the classification accuracy is the ratio of songs that is correctly classified. To prevent an overly optimistic accuracy score, the number of true positives is divided by $|D|$ instead of $|D'|$. This means that all songs $s \in (D \setminus D')$ (i.e. all songs in which no pattern is found) are treated as wrongly classified.

After this step, both the coverage and classification accuracy are known for the chosen pattern discovery algorithm, sparse melody construction method and similarity calculation algorithm.

5. Experiments and results

The framework outlined in the previous section can be used to extract repeated patterns from a set of songs, compress these songs based on the discovered patterns, classify the compressed representations and calculate the classification accuracy. In this section, we describe a number of experiments to thoroughly understand the role of (automatically) discovered patterns for the classification of folk songs into tune families. First, the classification accuracy on the uncompressed songs is retrieved in Section 5.1. This will give us a baseline of how well the implemented alignment algorithms perform on the complete songs. In Section 5.2, we introduce a number of *naive* compression approaches that are used in the compression step. These naive methods replace the compression process using the discovered repeated patterns. This way, the results of these experiments act as a baseline for the compression using repeated patterns. We call these experiments the *reference experiments*. Then, we describe the experiments where the Annotated Patterns are used for compression and classification (Section 5.3) and in Section 5.4, the experiments and results using automatically discovered patterns are put apart.

5.1 Classification accuracy on uncompressed songs

In this first experiment, we calculate the classification accuracy without pattern discovery or sparse melody construction. To achieve this, both the pattern discovery and the sparse melody construction step in the framework are skipped. This results in a *coverage* of 1.0 since no notes are removed and hence nothing is compressed. This experiment is run for both similarity calculation algorithms (Note to Note Alignment and Melodic Shape Alignment).

A classification accuracy of 0.93 is achieved using Note to Note Alignment and 0.87 for Melodic Shape Alignment. Consequently, both alignment algorithms lead to a high classification accuracy. Although the differences are small, Note to Note Alignment performs better than Melodic Shape Alignment. The classification performance of Note to Note Alignment is in correspondence with the findings of (Van Kranenburg et al., 2013) using this alignment approach on pitch interval sequences.

5.2 Naive compression methods

The compression strategies that are proposed in the sparse melody construction step all make use of the set of discovered patterns: notes that do not belong to a pattern are removed such that a smaller, compressed song is formed. To test the

stability of repeated patterns after compression, a number of naive compression methods are proposed as a baseline. As a result, the sparse melody construction step of the framework is replaced by one of the naive compression methods explained below.

5.2.1 Selecting a consecutive subsequence

The sparse melody construction modifies the internal structure of the songs, which might make it more difficult for an alignment algorithm to successfully align the songs. To measure this effect on the classification of long consecutive sequences, we perform three different experiments where a predefined number of consecutive notes are selected in the sparse melody construction step. This method is based on the observations of Xia, Huang, Ma, Dannenberg, and Faloutsos (2014), that the first or last part of a song is already a good feature for determining similarity. Here, the length n of this section is variable (in their case, n was 200 for large songs). The three different approaches are, for each piece in the corpus:

- select only the first n notes;
- select only the last n notes;
- select only the middle n notes.

Here, 'selecting' means that these notes are kept and all other notes are removed.

5.2.2 Random notes selection

As an absolute baseline, we measure the compression and classification performance of a random selection of notes. In each piece, a number of notes are randomly selected until a predefined coverage ratio is reached. All other notes are removed from the song. The desired coverage ratio, as well as the length l of each random selection, can be varied. If one of these notes is already present in another selection, this selection is rejected and a new random selection of length l is tried. For each coverage value in the range [0.1, 0.9], we average the classification accuracy using these notes over 20 samples. The coverage values are chosen with intervals of 0.1 and $l = 3$.

5.2.3 Random pattern selection

A downside of the previous method is that the note selection is completely random and does not replicate the effect of patterns repeating across *multiple* songs. To solve this, a bit mask is generated with the length of the largest song in the dataset. Then, the same random selection as described above is used to populate the mask with ones, until a predefined coverage ratio is reached. The mask is then used to select and remove notes, depending on the values of the mask. This procedure is the same for each song in the corpus, such that the position of the selected notes is the same for all pieces. This experiment is performed 200 times for random coverage values. Since the coverage is chosen to be random in each run, the coverage values are binned into intervals of 0.05. The values in each bin are averaged to get one accuracy value for each coverage bin.

5.2.4 Results

The results of these naive compression methods on MTC-ANN and using Note to Note Alignment are presented in Figure 5. For clarification purposes, the classification accuracy of the uncompressed songs is represented by a line instead of a single point. The coverage will be 1.0 for this experiment since every note is kept in the sparse melody construction step.

The Random Notes selection leads to the lowest accuracy for all compression methods. We suspect that this is caused by the fact that this approach creates many small gaps in

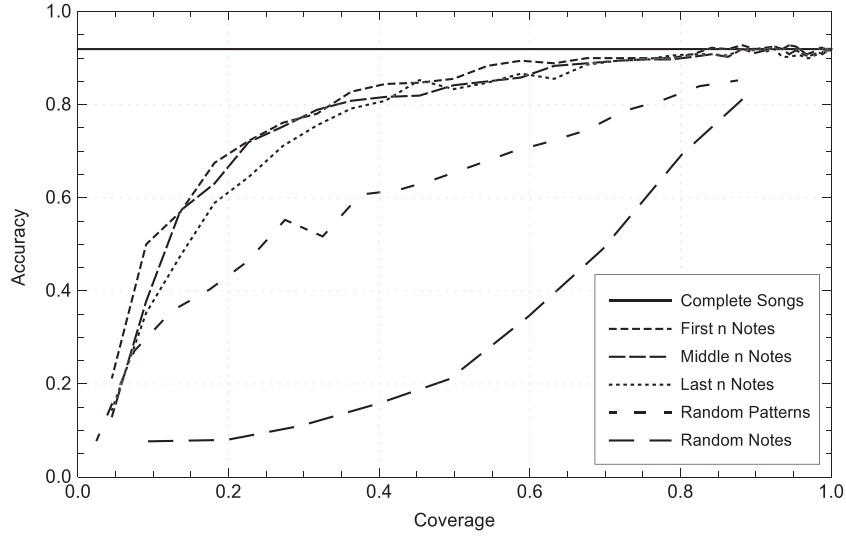


Fig. 5. Results of the reference experiments using Note to Note Alignment.

the songs, which affects the performance of the sequence alignment algorithm. If these algorithms cannot calculate sensible similarity distances between the songs, the resulting classification accuracy will also be low. When random patterns are selected (leading to larger sequences of notes selected), the accuracy is already much better compared to the Random Notes selection method.

The selection of the first, middle or last n notes as compression strategy leads to the highest accuracy. Unlike the Random Note selection, the accuracy increases fast at low coverage values. The selection of the last n notes even outperforms the classification of the complete songs for high values of n, although the differences are small.

5.3 Similarity using the annotated motifs

In contrast to the naive compression methods explained in the previous section, we now use the Annotated Motifs as the output of the pattern discovery step in the framework (see Figure 2). The sparse melody construction is now driven by the motifs that are annotated by the musicological experts. This tests the assumption that these motifs can be used to classify the folk songs of MTC-ANN into the right tune families and that these motifs can be used as a means for compression.

The results of the tune classification using the Annotated Motifs of MTC-ANN are presented in Table 1. The first thing to notice is that the resulting classification accuracy using Onset Preserved Construction/Gapless Construction and Note to Note Alignment (0.89) is only slightly worse than the performance of the uncompressed songs. However, on average only 40% of the notes of the uncompressed songs are used in the classification task. This shows that a high compression ratio is possible at the expense of only a small decrease in accuracy.

Like the results of Section 5.1, the use of Melodic Shape Alignment leads to a slightly lower accuracy than the Note to Note Alignment approach. Onset Preserved Construction and Gapless Construction yield higher accuracy values than First Pattern Occurrence Construction, although the difference is only 0.03 for both alignment algorithms. Compared with the naive compression methods explained in the previous section, the Annotated Motifs work better for compression than any of the naive methods.

The resulting coverage when using the First Pattern Occurrence Construction method is lower than for the other two construction techniques: 0.30. This can be explained by the fact that only the first occurrence of each pattern is used in the generation of a sparse melody. Although this construction method influences the classification accuracy negatively, the decrease in coverage (9 percentage points) is larger than the decrease in accuracy (3 percentage points) for both similarity metrics. The coverage and accuracy for the Onset Preserved Construction and Gapless Construction methods are identical. This has to do with the fact that both alignment algorithms ignore gaps in the songs for the sequence alignment.

To summarize, compared with the classification accuracy on the uncompressed songs, the accuracy has only slightly decreased using the Annotated Motifs for similarity and compression with a coverage of only 40%. Using the First Pattern Occurrence Construction, the sparse melodies even cover just 30% of the original songs at a cost of a slightly lower classification accuracy.

5.4 Similarity using pattern discovery algorithms

We now look at the accuracy and coverage of the classification experiments based on the automatically discovered repeated patterns for compression. Each pattern discovery algorithm can be tuned by a number of parameters. We therefore run each algorithm for a combination of parameters such that a large part of the parameter space is explored. This results in a point set of coverage and accuracy values for all parameter configurations. First, the results of Song Pattern Discovery are presented and then the results of using Tune Family Pattern Discovery.

5.4.1 Song pattern discovery

The results for the Song Pattern Discovery task are shown in Figure 6 using the Gapless Construction method and Note to Note Alignment. The output of the reference experiments and the results on the Annotated Motifs are also plotted.

It can be seen from both the algorithm output and the reference experiments that there exists a positive relation between the coverage and classification accuracy. When the coverage decreases, the amount of information (i.e. the number of notes) to successfully calculate the similarity between the songs also decreases. Since we want to study the effect of compression by pattern discovery algorithms, we define a good performing algorithm as one that results in a low coverage and in a high

Table 1. Classification accuracy and coverage of the compressed songs from MTC-ANN using the Annotated Motifs.

Sparse Melody Construction method	Coverage	Accuracy	
		Note to Note	Melodic Shape
Onset Preserved Construction	0.397	0.89	0.81
Gapless Construction	0.397	0.89	0.81
First Occurrence Construction	0.304	0.86	0.78

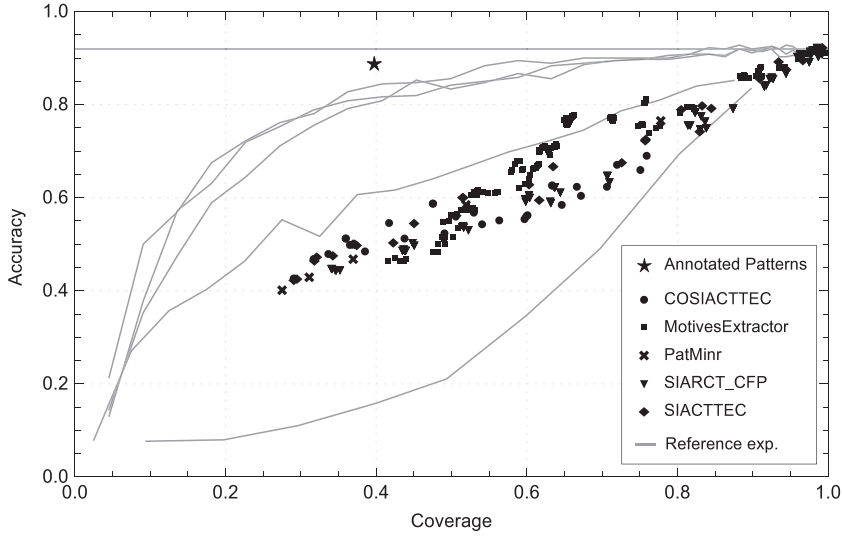


Fig. 6. Song Pattern Discovery using the Gapless Construction method and Note to Note Alignment. Each point in the plot represents a specific Pattern Discovery configuration.

classification accuracy (the top-left part of the graphs). In contrast, algorithms that return too many patterns yield a coverage of almost 1.0 and are therefore not suitable for compression.

Almost every algorithm output performs better than the Random Notes selection. The selection of the first, middle or last n notes works better than any pattern discovery algorithm. By keeping only the first 18 notes of each song (corresponding to an average coverage of 0.41) more than 84% of the pieces are classified correctly. Only the Annotated Motifs yield better results, with an accuracy of 0.89 (see Table 1). The Annotated Motifs also perform much better than *any* of the pattern discovery algorithm output: for similar coverage values, the pattern discovery algorithms only achieve a classification accuracy of around 0.5. The Random Pattern Selection outperforms almost every pattern discovery algorithm output. This means that selecting the same notes in each song works better than selecting notes based on repetition within each song.

These results show that patterns that repeat *inside* a song are not sufficient for explaining to what tune family a song belongs. There is a large gap in performance between the algorithm output and the Annotated Motifs. A reason for this is that these motifs are annotated while keeping the tune families in mind. Therefore, we investigate in the next section whether repetition inside the *entire tune family* is better suitable.

5.4.2 Tune family pattern discovery

The results of the pattern discovery task within tune families are presented in Figure 7. Here, the patterns that are discovered occur in multiple songs of the same tune family. The results of the reference experiments are the same as for the Song Pattern Discovery task since they are not influenced by the discovery task but only by the alignment method we choose.

Compared to the Song Pattern Discovery task, the resulting coverage and accuracy values of the algorithm output are more spread. The positive relation between coverage and accuracy that we observed in the previous experiment, is also visible in these results. Also, many algorithm configurations return very few patterns, which results in a low coverage and accuracy. For some parameter configurations of MotivesExtractor, the classification accuracy and coverage is even 0. This means that either no repeated patterns were found in the tune families, or that each discovered pattern occurrence overlapped with least two songs. The latter situation leads to the removal of these occurrences in the splitting process (Section 4.3). Compared to the Song Pattern Discovery task, the overall maximum coverage is lower: around 0.9 against nearly 1.0 for the Song Pattern Discovery task.

The patterns found with Tune Family Pattern Discovery are more useful in a compression scenario than the patterns found using Song Pattern Discovery. Many algorithm configurations now lead to a higher accuracy than the Random Pattern reference experiment. The selection of the first n notes works still better than any pattern discovery algorithm configuration. For some parameter configurations, however, the MGDP algorithm comes close to the performance of the selection of the middle n and last n notes. This can be explained by the fact that the MGDP algorithm only discovers patterns within a tune family that are distinctive compared to the other tune families.

Still, the Annotated Motifs perform better than any pattern discovery algorithm output that is tested, although the differences are smaller when comparing these results with the Song Pattern Discovery task. To conclude, the pattern discovery results are overall better than for the Song Pattern Discovery task with respect to the reference experiments, but the Annotated Motifs and the selection of the first n notes are still superior in terms of classification accuracy. In the next section, we attempt to explain the discrepancy between the performance of the Annotated Motifs and the pattern discovery algorithm output.

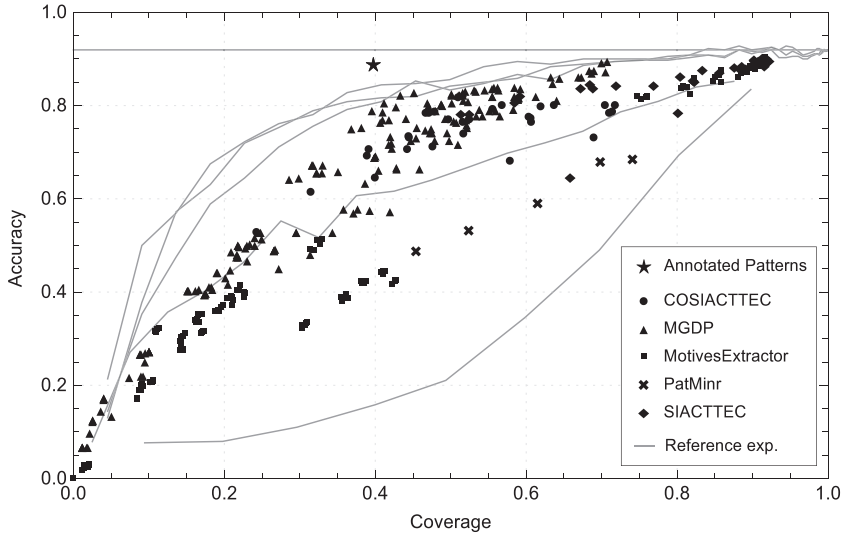


Fig. 7. Tune Family Pattern Discovery using the Gapless Construction method and Note to Note Alignment. Each point in the plot represents a specific Pattern Discovery configuration.

6. Discussion and methodological reflection

We have shown the results for a number of framework configurations in the previous section. In order to rule out possible biases introduced by specific design choices in the framework, we tested each step for different implementations and parameters. In this section, we shortly discuss possible biases towards the dataset, the alignment algorithms and the pattern discovery algorithms. Moreover, we present the differences between the Annotated Motifs and automatically discovered patterns. A more detailed description of these findings can be found in Boot (2015).

To rule out a potential bias of the Note to Note Alignment algorithm towards MTC-ANN, we repeated the experiment from Section 5.1 on *another* dataset. The reason for this is that the Note to Note Alignment algorithm is specifically designed for MTC-ANN (Van Kranenburg et al., 2013). We therefore composed a set of Irish folk tunes in Boot (2015), similar in size as MTC-ANN and also divided into different families based on variation.

This study shows that both *alignment algorithms* also perform very well on this dataset, with a classification accuracy of 0.96 and 0.92 for Note to Note alignment and Melodic Shape Alignment, respectively. This confirms that Note to Note Alignment also works for other folk song datasets, which is in correspondence with the observations by Van Kranenburg (2010) and Hillewaere et al. (2012) that sequence alignment is a good similarity measure for folk song classification. In this domain, the Note to Note Alignment algorithm even outperforms the winning implementation of the *Symbolic Melodic Similarity* MIREX task.[4]

From the results in Section 5.4, we see that Tune Family Pattern Discovery performs better than Song Pattern Discovery. The reason for this is that in the Tune Family Pattern Discovery task, we specifically search for patterns repeating in an entire tune family, rather than in a single song. This makes the classification easier, since there is a higher correspondence between the compressed songs from the same family when only repeated segments that occur within that family are used for classification. We solved the lack of Tune Family Pattern Discovery algorithms by concatenating all songs from the same tune family and use these large songs as input for a Song Pattern Discovery algorithm. This method is favourable against pairwise pattern discovery (where patterns are discovered between all pairs of songs in the dataset), because many pattern discovery algorithms incorporate filtering steps that are dependent on the total length of the song. Performing pairwise pattern discovery instead of pattern discovery on the concatenated songs will therefore lead to inconsistent pattern filtering behaviour between the pieces. A pitfall of the concatenation method is that the complexity increases fast for larger (concatenated) songs and therefore the running time also increases.

The concatenation of complete tune families can lead to a positively biased classification accuracy, due to the fact that we already know the tune family assignments beforehand. This way, the algorithms in the Tune Family Pattern Discovery task will only return patterns that occur within the same tune family. Possible patterns across songs of different tune families will therefore never be discovered. This bias could be solved by performing pattern discovery on the complete dataset, where patterns will be discovered across all pieces in the dataset irrespective of their original tune family. Another approach is to sample from the dataset in order to reduce the size. This is something that should be studied further in future research. Collins (2014b) has already performed an analysis of using SIARCT-CFP for Tune Family Pattern Discovery on

[4]The Note to Note Alignment algorithm never competed in this task.

Table 2. Average number of patterns discovered in a single song. For the Annotated Motifs, the average number of discovered patterns per song is 3.40.

Algorithm/method	Average number of patterns per song	
	Song Pattern Discovery	**Tune Family Pattern Discovery**
COSIACTTEC	1.80	4.70
MGDP	–	47.04
MotivesExtractor	2.53	39.28
PatMinr	2.95	50.64
SIARCT-CFP	19.60	–
SIACTTEC	16.41	313.24

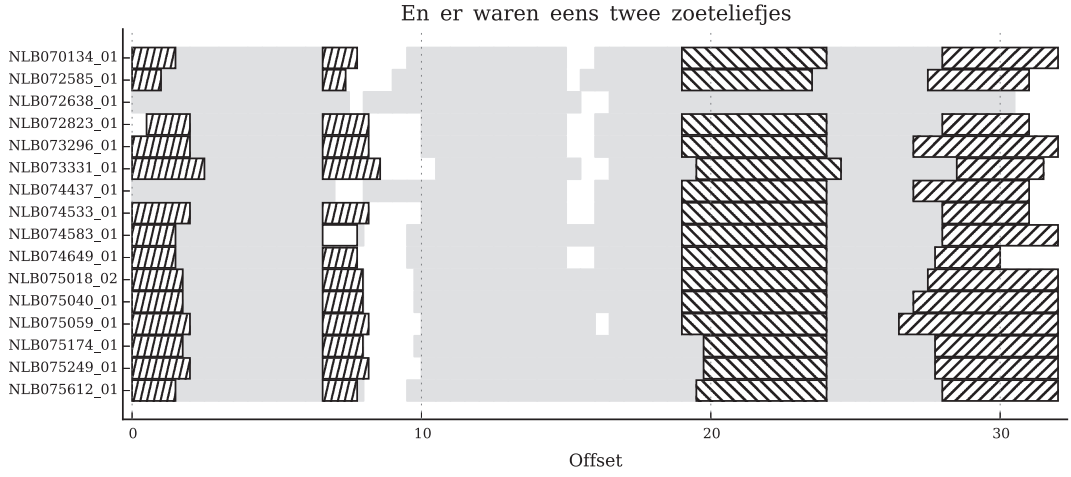


Fig. 8. Position of the Annotated Motifs plotted for the tune family 'En er waren eens twee zoeteliefjes'. In most songs three different motifs are annotated, while the first one repeats within the songs.

Beethoven's piano sonatas. However, in our case, the approach of the Tune Family Pattern Discovery task is justified since it mimics the pattern discovery procedure as followed by the annotation experiment in Volk and Van Kranenburg (2012). Here, the musicologists also knew the right tune families before annotating the characteristic motifs.

To assess the differences between the *automatically discovered* patterns and the *Annotated Motifs*, we carried out a number of additional experiments in Boot (2015) to compare the output of annotated and automatically discovered patterns. Using the evaluation metrics that were defined for the *Discovery of Repeated Themes and Sections* MIREX task, the output of each tested configuration of a pattern discovery algorithm was compared with the Annotated Motifs as *ground truth*. In the best case, at most 70% of the discovered patterns could be matched with at least one occurrence to one of the Annotated Motifs for a configuration of SIACTTEC.

Besides this, the large performance gap between the annotated and automatically discovered patterns can also be explained by the discovery 'strategy' of the musicologists and the algorithms. Compared to classical music, which is mostly used as input for pattern discovery algorithms, folk songs are relatively short with an average length of only 48 notes. Therefore, the annotated motifs are also short in length (four notes on average). One can imagine that short patterns will occur more often than longer patterns, and are more likely to be *unintentional* repetitions. It is therefore crucial for both humans and algorithms to be able to rate a short pattern as being *characteristic* or *non-characteristic* in order to limit the output size. In Table 2, the average number of discovered patterns is shown for each algorithm. In the set of Annotated Motifs, 3.4 motifs are annotated per song on average. In can be seen that, except for COSIACTTEC (which limits the output size very well due to its compression approach), all other algorithms in the Tune Family Pattern Discovery task return much more patterns per song on average. Apparently, these pattern discovery algorithms lack a filtering heuristic that discards non-characteristic short patterns.

Another factor that explains the difference between annotated and discovered patterns is that the occurrences of the Annotated Motifs vary in both melody and rhythm within a single pattern as can be seen in Figure 1. We therefore expect that these variations can also be a reason why pattern discovery algorithms did not find the same pattern occurrences as those

that are annotated, since many of the tested pattern discovery algorithms cannot discover inexact repetitions. This explains the difference in performance between COSIACTTEC and the Annotated Motifs despite the low number of patterns that are returned, since COSIACTTEC only discovers exact repetitions.

To better understand the characteristic property of the Annotated Motifs, we examined the *position* of the patterns in Boot (2015). We found that the position of a pattern in the song is a good heuristic for finding pattern occurrences. Analysis showed that the *order* of which patterns occur in a tune family is very consistent between the songs in the same family. An example of this is illustrated in Figure 8. This is in correspondence with the observations of Van Kranenburg (2010), who stated that phrase position (i.e. the position of a note in a phrase) is a good feature for Note to Note Alignment. Pattern occurrences that are annotated based on their position are often inexact occurrences in terms of melodic and rhythmic similarity because the location of a pattern is assumed more important than the exact similarity of melody or rhythm. This emphasizes the need for more pattern discovery algorithms that are able to discover inexact pattern occurrences, or discovering repetitions primarily based on their position. This is supported by the results of Rodríguez-López and Volk (2015) who have shown that positional information of repetitions is also an important feature for melodic segmentation.

7. Conclusion and future work

In this paper, we studied the relevance of repeated patterns for modelling similarity and compression in folk songs using a computational approach. We developed a framework for using pattern discovery algorithms as compression method to determine the similarity between folk songs belonging to the same tune family. To evaluate how we can leverage repeated patterns for compression and tune classification, we used a number of classification comparisons, namely classification of uncompressed songs, classification of naive lossy compression approaches by selecting the first, middle or last n notes in each song, and classification using random notes and random patterns for compression.

The results for the classification of the songs using uncompressed songs and naive compression methods reveal the following. The classification of the entire songs yields an accuracy between 0.87 and 0.96, showing that the classification accuracy is high using the entire songs. Using a naive lossy compression approach by selecting the first, middle or last n notes in each song, leads to an accuracy that is still high even for lower coverage values. The selection of random patterns and notes as compression method performs worse.

The results for the classification of the songs using patterns obtained by pattern discovery algorithms and using the Annotated Motifs from the set of Dutch folk songs MTC-ANN reveal the following. Using the Annotated Motifs, the classification accuracy was 0.89 using Gapless Construction and Note to Note Alignment, with only a coverage of 0.4. This means that with only 40% of the notes left in each song on average, the classification accuracy has dropped only a few percentage points compared to the classification of the entire songs. For the pattern discovery algorithms, we studied both the effect on the classification accuracy using Song Pattern Discovery and Tune Family Pattern Discovery. None of the algorithms perform comparably to the Annotated Motifs in any of these settings. Moreover, the selection of the first, middle or last n notes was superior to most of the algorithm runs.

Hence, we conclude regarding the relevance of repeated patterns for modelling similarity and compression that the Annotated Motifs performed better than anything else in terms of coverage and compression ratio. Although the accuracy is slightly worse than the classification performance of the uncompressed songs, an almost similar result can be achieved with only 40% of the notes. Moreover, the random selection of notes has shown that these approaches do not lead to a higher classification accuracy than the Annotated Motifs. We can therefore conclude that repeated patterns *do* influence the classification process in a positive way and that these parts are sufficient to describe the tune families. This confirms our hypothesis that stable parts contain enough distinctive information to successfully classify songs. This finding also confirms the musicological hypothesis that patterns are important for the similarity between folk songs belonging to the same tune family as reported in Volk and Van Kranenburg (2012).

Our results show that a selection of state-of-the-art pattern discovery algorithms does not reach the same accuracy as the Annotated Motives at the same coverage when we use these patterns for retrieval. We have tested these algorithms with a large amount of different parameter configurations in order to ensure that the whole search domain was included. Our study shows that there is an important difference between the discovery of characteristic patterns of a tune family, and the discovery of perceptually meaningful patterns. Current pattern discovery algorithms focus mainly on the second use case, which is different from our methodology. We can therefore conclude that we need to modify these algorithms to better support the discovery of characteristic patterns within tune families and the discovery of stable parts in a song. A possible starting point for this is incorporating the position of a pattern into the algorithms, since we have seen that this is an important property of the Annotated Motifs. The evaluation of repeated patterns in the context of classification and compression therefore presents a crucial step towards bridging the gap between manually and automatically discovered patterns.

Acknowledgements

We thank the anonymous reviewers for their constructive comments on earlier drafts of this paper. We thank Peter van Kranenburg and Berit Janssen from the Meertens Institute for providing MTC-ANN and an early version of the Annotated Motifs.

Funding

Anja Volk and W. Bas de Haas are supported by the Netherlands Organization for Scientific Research, NWO-VIDI grant 276-35-001.

References

Bayard, S. (1950). Prolegomena to a study of the principal melodic families of British-American folk song. *Journal of American Folklore, 63*(247), 1–44.

Boot, D.P. (2015). *Using discovered and annotated patterns as compression method for determining similarity between folk songs* (Master's thesis). Utrecht, The Netherlands: Utrecht University.

Collins, T. (2014a). *Discovery of repeated themes & sections*. Retrieved from http://www.music-ir.org/mirex/wiki/2014:Discovery_of_Repeated_Themes_%26_Sections

Collins, T. (2014b). Inter-opus analyses of Beethoven's piano sonatas. Paper presented at the *Eighth European Music Analysis Conference*, Leuven.

Collins, T., Arzt, A., Flossmann, S., & Widmer, G. (2013). SIARCT-CFP: Improving precision and the discovery of inexact musical patterns in point-set representations. In *Proceedings of the 14th International Society for Music Information Retrieval Conference* (pp. 549–554). Canada: International Society for Music Information Retrieval. Retrieved from http://www.cp.jku.at/research/papers/Collins_etal_ISMIR_2013.pdf

Collins, T., Thurlow, J., Laney, R., Willis, A., & Garthwaite, P. (2010). A comparative evaluation of algorithms for discovering translational patterns in Baroque keyboard works. In *Proceedings of the 11th International Society for Music Information Retrieval Conference* (pp. 3–8). Canada: International Society for Music Information Retrieval. Retrieved from http://ismir2010.ismir.net/proceedings/ismir2010-2.pdf

Conklin, D. (2009). Melody classification using patterns. In *Proceedings of the Second International Workshop on Machine Learning and Music* (pp. 37–41). Retrieved from http://www.ehu.es/cs-ikerbasque/conklin/papers/mml09conklin.pdf

Conklin, D. (2010). Discovery of distinctive patterns in music. *Intelligent Data Analysis, 14*(5), 547–554.

Conklin, D., & Anagnostopoulou, C. (2001). Representation and discovery of multiple viewpoint patterns. In *Proceedings of the 2001 International Computer Music Conference* (pp. 479–485). San Francisco: International Computer Music Association.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*(1), 21–27. doi:10.1109/TIT.1967.1053964

Cowdery, J.R. (1984). A fresh look at the concept of tune family. *Ethnomusicology, 28*(3), 495–504. doi:10.2307/851236

De Boor, C. (1972). On calculating with B-splines. *Journal of Approximation Theory, 6*(1), 50–62. doi:10.1016/0021-9045(72)90080-9

Goldstone, R.L., & Son, J.Y. (2005). Similarity. In K.J. Holyoak, & R.G. Morrison (Eds.), *Cambridge handbook of thinking and reasoning* (pp. 13–36). Cambridge: Cambridge University Press.

Gotoh, O. (1982). An improved algorithm for matching biological sequences. *Journal of Molecular Biology, 162*(3), 705–708. doi:10.1016/0022-2836(82)90398-9

Hillewaere, R., Manderick, B., Conklin, D., & Ehu, U.P.V. (2012). String methods for folk tune genre classification. *Proceedings of the 13th International Society for Music Information Retrieval Conference* (pp. 217–222). Canada: International Society for Music Information Retrieval.

Huron, D. (1997). Humdrum and kern: Selective feature encoding. In E. Selfridge-Field (Ed.), *Beyond MIDI: The handbook of musical codes*. Cambridge, MA: MIT Press.

Janssen, B., De Haas, W.B., Volk, A., & Van Kranenburg, P. (2013). Discovering repeated patterns in music: state of knowledge, challenges perspectives. In *Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research* (pp. 225–240). Marseille: Laboratory of Mechanics and Acoustics.

Karpeles, M. (1968). The distinction between folk and popular music. *Journal of the International Folk Music Council, 20*, 9–12.

Karydis, I., Nanopoulos, A., & Manolopoulos, Y. (2006). Symbolic musical genre classification based on repeating patterns. *AMCMM '06 Proceedings of the 1st ACM workshop on Audio and music computing multimedia* (pp. 53–58). New York: ACM.

Lartillot, O. (2014a). In-depth motivic analysis based on multiparametric closed pattern and cyclic sequence mining. *Proceedings of the 15th International Society for Music Information Retrieval Conference* (pp. 361–366). Canada: International Society for Music Information Retrieval.

Lartillot, O. (2014b). Submission to MIREX discovery of repeated themes and sections. In *10th Annual Music Information Retrieval eXchange (MIREX'14)*. Retrieved from http://www.music-ir.org/mirex/abstracts/2014/OL1.pdf

Li, M., Chen, X., Li, X., Ma, B., & Vitanyi, P. (2004). The similarity metric. *IEEE Transactions on Information Theory, 50*(12), 3250–3264. doi:10.1109/TIT.2004.838101

Lin, C., Liu, N., Wu, Y., & Chen, A. (2004). Music classification using significant repeating patterns. In Y.J. Lee, J. Li, K.-Y. Whang, & D. Lee (Eds.), *Database systems for advanced applications* (Lecture Notes in Computer Science, Vol. 2973, pp. 506–518). Berlin: Springer.

Manning, C.D., Raghavan, P., & Schütze, H. (2009). *An introduction to information retrieval*. Cambridge: Cambridge University Press.

Mélen, M., & Wachsmann, J. (2001). Categorization of musical motifs in infancy. *Music Perception, 18*(3), 325–346.

Meredith, D. (2015). Music analysis and point-set compression. *Journal of New Music Research, 44*(3), 245–270. doi:10.1080/09298215.2015.1045003

Meredith, D., Lemström, K., & Wiggins, G.A. (2002). Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research, 31*(4), 321–345. doi:10.1076/jnmr.31.4.321.14162

Needleman, S.B., & Wunsch, C.D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology, 48*(3), 443–453. doi:10.1016/0022-2836(70)90057-4

Nieto, O., & Farbood, M. (2014a). Submission to MIREX discovery of repeated themes and sections. In *10th Annual Music Information Retrieval eXchange (MIREX'14)*. Retrieved from http://www.music-ir.org/mirex/abstracts/2014/NF1.pdf

Nieto, O., & Farbood, M.M. (2014b). Identifying polyphonic patterns from audio recordings using music segmentation techniques. In *Proceedings of the 15th International Society for Music Information Retrieval Conference* (pp. 411–416). Retrieved from http://www.nyu.edu/projects/farbood/pdf/NietoFarbood-ISMIR2014.pdf

Rodríguez-López, M.E., & Volk, A. (2015). Location constraints for repetition-based segmentation of melodies. In T. Collins, D. Meredith, & A. Volk (Eds.), *Mathematics and computation in music* (Lecture Notes in Computer Science, Vol. 9110, pp. 73–84). New York: Springer International Publishing.

Urbano, J. (2013). MIREX 2013 symbolic melodic similarity: A geometric model supported with hybrid sequence alignment. In *Music Information Retrieval Evaluation eXchange*. Retrieved from http://www.music-ir.org/mirex/abstracts/2013/JU2.pdf

Van Kranenburg, P. (2010). *A computational approach to content-based retrieval of folk song melodies* (PhD thesis), Utrecht University, Utrecht, Netherlands.

Van Kranenburg, P., de Bruin, M., Grijp, L.P., & Wiering, F. (2014). The Meertens Tune Collections. In *Meertens Online Reports*. Amsterdam: Meertens Institute.

Van Kranenburg, P., Janssen, B.J., & Volk, A. (forthcoming). The Meertens Tune Collections: Annotated corpus (MTC-ANN) version 2.0. In *Meertens Online Reports*. Amsterdam: Meertens Institute.

Van Kranenburg, P., Volk, A., & Wiering, F. (2012). On identifying folk song melodies employing recurring motifs. In *Proceedings of the 12th International Conference on Music Perception and Cognition and the 8th Triennial Conference of the European Society for the Cognitive Sciences of Music* (pp. 1057–1062). Thessaloniki: School of Music Studies, Aristotle University.

Van Kranenburg, P., Volk, A., & Wiering, F. (2013). A comparison between global and local features for computational classification of folk song melodies. *Journal of New Music Research, 42*(1), 1–18. doi:10.1080/09298215.2012.718790

Volk, A., De Haas, W.B., & Van Kranenburg, P. (2012). Towards modelling variation in music as foundation for similarity. In *Proceedings of the 12th International Conference on Music Perception and Cognition and the 8th Triennial Conference of the European Society for the Cognitive Sciences of Music* (pp. 1085–1094). Thessaloniki: School of Music Studies, Aristotle University.

Volk, A., & Van Kranenburg, P. (2012). Melodic similarity among folk songs: An annotation study on similarity-based categorization in music. *Musicae Scientiae, 16*(3), 317–339. doi:10.1177/1029864912448329

Wiora, W. (1941). Systematik der musikalischen Erscheinungen des Umsingens. *Jahrbuch für Volksliedforschung, 7*, 128–195. doi:10.2307/846331

Xia, G., Huang, T., Ma, Y., Dannenberg, R., & Faloutsos, C. (2014). MidiFind: Similarity search and popularity mining in large MIDI databases. *Sound, Music, and Motion, 8905*, 259–276.

Ziv, N., & Eitan, Z. (2007). Themes as prototypes: Similarity judgments and categorization tasks in musical contexts. *Musicae Scientiae, 11*(1 Suppl), 99–133.