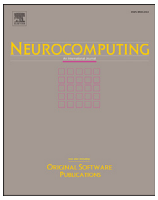
Neurocomputing 666 (2026) 132208

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Symbolic music structure analysis with graph representations and changepoint detection methods

Carlos Hernandez-Olivan*, Jose R. Beltran

Department of Electronics and Telecomunications, University of Zaragoza, Zaragoza, 50010, Spain

ARTICLE INFO

Communicated by C. Chen

Keywords:
Music information retrieval
Music structure analysis
Graph
Symbolic music
Signal processing
Machine learning

ABSTRACT

Music Structure Analysis (MSA), particularly symbolic music boundary detection, is crucial for understanding and creating music, yet segmenting music structure at various hierarchical levels remains an open challenge. In this work, we propose three methods for symbolic music boundary detection: Norm, an adapted feature-based approach, and two novel graph-based algorithms, G-PELT and G-Window. Our graph representations offer a powerful way to encode symbolic music, enabling effective structure analysis without explicit feature extraction. We conducted an extensive ablation study using three public datasets, Schubert Winterreise (SWD), Beethoven Piano Sonatas (BPS) and Essen Folk Dataset, which feature diverse musical forms and instrumentation. This allowed us to compare the methods, optimize their parameters for different music styles, and evaluate performance across low, mid, and high structural levels. Our findings demonstrate that our graph-based approaches are highly effective; for instance, the online and unsupervised G-PELT method achieved an F1-score of 0.5640 with a 1-bar tolerance on the SWD dataset. We further illustrate how algorithm parameters can be adjusted to target specific structural granularities. To promote reproducibility and usability, we have integrated the best-performing methods and their optimal parameters for each structural level into musicaiz, an open-source Python package. We anticipate these methods will benefit various Music Information Retrieval (MIR) tasks, including structure-aware music generation, classification, and key change detection.

1. Introduction

Music Structure (or *form*) Analysis (MSA) is a field of Music Information Retrieval (MIR) consisting of predicting the structure or *form* of music pieces. Music structure is a musical principle [15] that is closely related to other musical principles like harmony. Music presents a hierarchical structure in which sections are ordered in a coherent way. Such sections contain different rhythmic patterns and harmonic progressions that express different ideas. However, in Western classical music such sections can be connected among themselves with the so-called cadences or bridges. From the perspective of music cognition, there are different ways of understanding the hierarchical structure in music: grouping structure, metrical structure, reduction in time-span and prolongational [26].

In terms of computational approaches to MSA, there are different subtasks that have been previously studied, such as *melodic segmentation*, *molecular discovery*, and *structural segmentation* [28]. Structural segmentation can also be divided into two smaller tasks: *boundary detection* which aims to segment a piece by its structural boundaries, and *segment labeling* which aims to label or name the sections of a piece. In this paper, we focus specifically on the task of *boundary detection*, as it is the first and fundamental step in analyzing the form of a music piece. Accurate boundary detection is critical for downstream tasks like structural labeling, music generation, and performance analysis.

Music boundary detection (MBD) has been studied from two different perspectives according to the nature of the input music: the symbolic domain and the audio domain. Whereas the symbolic domain refers to the music data that contain the basic information about the notes or instruments (MIDI or MusicXML files), the audio domain refers to the digital musical signal and transformations that can be computed from it. While MBD in the audio domain often involves complex signal processing to infer high-level musical features, this makes the task harder in the audio domain, since there is no high-level musical information in the digitized music signal.

In the symbolic domain, MBD has been studied for decades [28], specially regarding monophonic melodic segmentation [2,25,36]. However,

* Corresponding author.
Email address: carloshero@unizar.es (C. Hernandez-Olivan).

https://doi.org/10.1016/j.neucom.2025.132208
Received 19 December 2024; Received in revised form 4 November 2025; Accepted 26 November 2025
Available online 28 November 2025

fewer methods exist for online, unsupervised symbolic music structure analysis across multiple instruments and hierarchical levels.

MBD presents the following challenges:

- Music genres or styles, and also pieces of the same genre, have different forms or structures, making universal models difficult. Consequently, the number and duration of sections are unknown a priori, often leading to imbalanced data in supervised learning contexts.
- Boundaries represent a small percentage of the data compared to the total number of notes or events in a piece. This sparsity, combined with variable content, complicates the training of data-driven models for precise boundary identification.
- Music often exhibits a hierarchical structure with different levels, e.g., in Western classical music form we can find the following levels: *high* structure (sections), *mid* structure (themes or musical phrases) and *low* structure (motifs). However, not all music can be divided into those levels, especially in non-Western classical traditions. Capturing this multi-level segmentation remains a challenge.
- There are only a few datasets in MIR of symbolic music that contain structure annotations, limiting large-scale evaluation and supervised training.

MBD not only covers the task of music analysis, but it can also be used in music generation systems to reinforce such models in generating coherent music [15,47].

In this paper, we propose and compare two distinct methods for online and unsupervised symbolic music boundary detection. These methods predict segment boundaries without requiring prior information such as the number of boundaries, time signature, or specific features beyond note onset, offset, and pitch. We validate our methods using two public multi-instrument datasets with different structures and music genres, providing an extensive ablation study.

1.1. Contributions

The main contributions of this work are:

- Two novel, fast, and online graph-based methods (G-PELT and G-Window) for symbolic music structure analysis, which utilize graph representations to capture musical relationships from temporal information without explicit feature extraction (Fig. 1). To our knowledge, this is the first online method to perform Music Structure Segmentation for multi-instrument polyphonic music, and the first to explicitly segment symbolic music by hierarchical structure levels.
- A comparison of the performance of our methods with three different public datasets that contain various forms and instruments. Both datasets include polyphonic multi-instrument music. The ablation study involves the evaluation across three structure levels: low, mid and high.

The algorithms presented in this paper are integrated into musicaiz package to improve reproducibility.[1]

1.2. Paper organization

This paper is organized as follows: in Section 2 we discuss previous work done in MBD, in Section 3 we describe the proposed methods, in Section 4 we compare them, in Section 5 we evaluate our methods on two public datasets, in Section 6 we discuss the results and in Section 7 we provide conclusions and future work that could benefit from our research.

2. Related work

MBD has been a long-standing research area within MIR, with studies spanning several decades. MBD can be categorized by the domain of the input music (audio and symbolic) and by the specific subtask being addressed (e.g., boundary detection, segment labeling, or hierarchical segmentation).

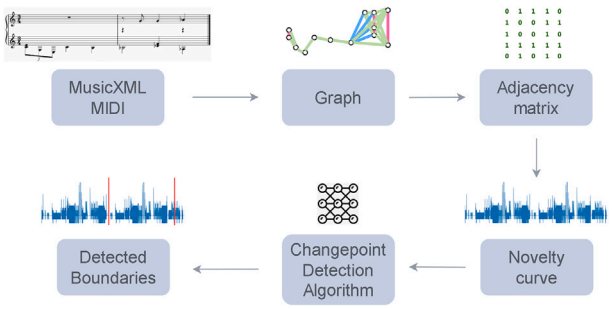


Fig. 1. Our proposed method for symbolic music segmentation.

2.1. Audio music boundary detection

In the audio domain, MBD typically relies on extracting features from the raw audio signal that capture musical similarity and novelty. Common features include Mel-frequency Cepstral Coefficients (MFCCs), chroma features, or beat-synchronous features, which are then used to construct Self-Similarity Matrices (SSMs). Pioneering works include those by Foote [9] who introduced novelty curves for segmentation. Also, l_1-graph representations of audio features have been proposed for MBD [33]. l_1-graphs are a type of graph in which the edges between nodes are weighted based on the absolute difference between the values of their corresponding features. More recent approaches have leveraged deep learning, employing Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) to learn hierarchical features for segmentation directly from spectrograms [17]. While presenting a strong performance, methods from the audio domain often tackle different challenges related to signal processing and perceptual feature extraction, which are not directly applicable to symbolic data.

2.2. Symbolic music boundary detection

Symbolic MBD operates on discrete representations of music, such as MIDI, MusicXML, or custom text-based formats, which explicitly encode musical events like notes, rests, and durations. This domain allows for the direct analysis of musical parameters like pitch, rhythm, and harmony without the complexities of acoustic signal variations.

MBD in symbolic music has been studied for decades [28]. Most of the works in symbolic music focused on for monophonic melodic segmentation [2,25,36]. These methods can be applied to annotate large corpus of data such as the `SIATEC` and `COSIATEC` algorithms [29]. The vast majority of these feature-based methods use rule-based systems or need to extract features such as the *Inter Onset Intervals* (IOIs, Eq. (1)) from the symbolic data to find a structural description of the music [27]. Examples of this are the Local Boundary Detection Model (LBDM) [3] which uses the IOIs, pitches and rests to segment music phrases, or the Grouper Program [42] which uses the *Offset to Offset Intervals* (OOI) in addition to the IOIs. Extensions of these methods use variations of these features to perform the same task. Among these approaches we can find $\triangle$IOI [5] which consists of computing the difference between an IOI and the previous IOI, Meter Finder [43] and $\triangle$IOI OR Meter Finder OR Pause [5]. These approaches were tested and their weights were optimized on monophonic melodies of the Essen Folk dataset [39]. Similarly, the Pattern Boundary Detection Model (PAT) [4] uses symbolic features to perform phrase segmentation. Adaptive melodic segmentation in MIDI files have also been proposed [46]. Rule mining techniques [23] and pseudo-supervised methods [24] are other techniques to perform the

[1] https://github.com/carlosholivan/symbolic-music-structure-analysis

melody segmentation task. For this purpose, there are annotated corpus [37] like TAVERN [7]. The dataset contains 27 sets of themes and variations for piano by Mozart and Beethoven.

Focusing on higher level structures, computational analysis techniques of different musical forms have been proposed. From J. S. Bach fugues [10] to the Sonata form structure that has been studied in W. A. Mozart string quartets [1]. The techniques used in these works are mostly feature-based with harmonic and rhythmic features such as the rhythm break or the triple hammer blow.

2.3. Symbolic music as graphs

Graph representations offer a more holistic approach by capturing not only individual musical events but also their relationships in a flexible and expressive manner. From a theoretical perspective, music is not only a sequence of temporally ordered events but a network of interdependent relationships—temporal, harmonic, and motivic that unfold across multiple structural levels. Traditional feature-based or sequence-based models implicitly assume temporal continuity, where structure arises primarily from the succession of adjacent events. In contrast, a graph formulation assumes relational connectivity, where structural meaning emerges from patterns of interaction among events, regardless of their immediate temporal proximity. Formally, a graph $G(V, E)$ can be defined as a collection of points that have different types of connections. The points are called nodes V and the connections between the nodes are the edges E. Graph representations are becoming popular in applications such as protein discovery [19] and simulating complex physics [38]. In symbolic music, graph representations have been proposed for diverse purposes concerning structural analysis or music classification [30].

The theoretical advantage of using graphs for Music Boundary Detection (MBD) lies in their ability to represent non-local dependencies and multidimensional relations. For example, harmonic closure, motivic recurrence, or contrapuntal overlap often connect events that are distant in time but structurally related—relations that sequence-based models struggle to encode. Jeong et al. [18] leveraged graph encodings to learn contextual note representations, and Simonetta et al. [40] used graph structures to quantify inter-piece similarities. Furthermore, Karystianos and Widmer [20] applied a stochastic Graph Convolutional Network (SGSMOTE) to identify Perfect Authentic Cadences (PACs), a structural phenomenon strongly tied to musical boundaries. In such representations, nodes correspond to notes or rests, while edges capture temporal adjacency, synchronous onset, or note overlap.

The flexibility of graph-based models allows for the simultaneous encoding of multiple structural principles, such as temporal continuity, harmonic cohesion, and textural dependency, within a single representational space. Edges can denote sequential, vertical, or contextual relationships, enabling the emergence of boundaries as topological discontinuities within the network (e.g., changes in connectivity or clustering structure). In contrast, traditional feature-based methods primarily describe local event properties, offering a fragmented view of the structure. By integrating both local and global relational information, graph representations provide a theoretically grounded and structurally coherent framework for modeling musical form and detecting boundaries.

Recent learning-based symbolic segmentation approaches extend these ideas using deep temporal and graph neural architectures that can learn hierarchical representations of musical structure directly from data. Examples include Transformer-based temporal models that capture long-range dependencies across symbolic sequences [13,48], and Graph Neural Networks (GNNs) that propagate relational information between notes or motifs to infer higher-order structure [21]. While these methods achieve impressive segmentation accuracy when trained on large annotated corpora, they rely heavily on supervised learning and extensive parameterization, which can limit interpretability and generalization to unseen styles. In contrast, the present work focuses on an unsupervised and interpretable formulation that operates directly on symbolic features without requiring labeled data, offering complementary advantages in data efficiency and musicological transparency. A systematic empirical comparison with these deep models is left for future work.

3. Proposed methods

In this section, we propose two algorithms for music structure detection in symbolic music: Norm, G-PELT and G-Window. Note that Norm is an implementation of $\triangle$IOI [5] with the addition of a peak picking algorithm on the novelty function. Thus, the method extracts the IOIs and pitch direction of the music to obtain the boundaries of the structural segments. The two novel contributions are the G-PELT and G-Window methods, which are novel graph-based methods that segment different levels in the structure of symbolic music depending on hyperparameter values (see Table 4). None of these methods need the symbolic music to be quantized, nor the time signature or the beats per minute information, which means that we can use both methods in non curated MIDI files or in musicXML files.

3.1. Method 1: Norm

This method is based on the segmentation model proposed by Cenkerová et al. [5]. However, we added a peak-picking strategy based on the novelty function extracted from the SSM built from the normalization of the IOIs and pitch directions. In Algorithm 1 we provide the pseudo-code of the Norm algorithm. The algorithm consists of the following steps:

We sort the notes by their *onset* or `Note ON` (in MIDI notation) and compute the IOIs vector $\mathbf{x} \in R^{N-1}$ where N is the number of notes in the music piece (Eq. 1):

$$x_i = onset_{i+1} - onset_i \quad i = 1 \dots N \tag{1}$$

We define the local direction or pitch contour vector $\mathbf{l} \in R^{N-1}$ similarsimilarly to previous works in which the contour intervals were stored in an array of contour intervals called the COM-matrix [31]. We compute $\mathbf{l}$ as follows: if a note is followed by a note with a higher pitch (ascendent), $l_i = 1$ and $l_i = -1$ if it is lower (descendent). If the pitch of the following note is equal to the current pitch then $l_i = 0$. In Eq. (2) we show the expression of the local direction vector:

$$l_i = \begin{cases} 1 & \forall \quad p_{i+1} > p_i \\ -1 & \forall \quad p_{i+1} < p_i \\ 0 & \forall \quad p_{i+1} = p_i \end{cases} \quad i = 1 \dots N \tag{2}$$

where p_i refers to the pitch of the ith note.

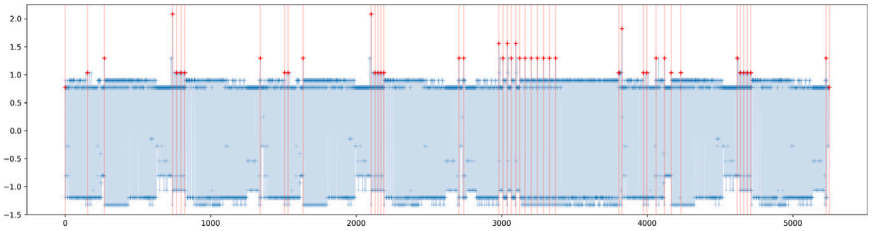
After computing $\mathbf{x}$ and $\mathbf{l}$ we sum them to construct the vector $\tilde{\mathbf{x}}$. We normalize this vector to obtain the $\hat{\mathbf{x}}$ vector by applying the z-score normalization (Eq. 3):

$$\hat{x}_i = \frac{x_i - \mu}{\sigma} \quad i = 1 \dots N \tag{3}$$

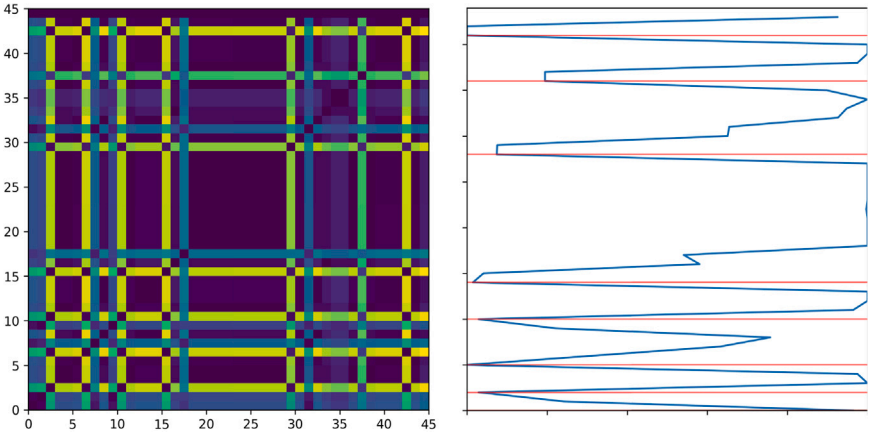
where μ is the mean and σ is the standard deviation of the whole IOIs vector.

With $\tilde{\mathbf{x}}$, we now apply a peak picking strategy. By doing this, we extract the segment boundary candidates $\mathbf{b} = [b_1, \dots, b_S] \in R^C$ with C being the number of boundary candidates. We select a window size w_1 and a threshold τ_1 (standard deviation above the mean). The window size w_1 is defined as a function of the number of notes N so $w = \frac{\alpha}{\hat{n}}$ where $\hat{n} = 15$ and α is a constant parameter. We fixed $\hat{n} = 15$ to optimize the algorithm performance.

After that, we can compute the SSM, $\mathbf{S} \in R^C \times R^C$, by grouping the notes in the obtained boundary candidates. The distance to construct the $\mathbf{S}$ is the Euclidean distance. Since not all the segments have the same



(a) Normalized IOIs and local direction vector with the boundary candidates detected after the peak picking (in red).



(b) SSM and novelty curve with predicted boundaries (in red)

Fig. 2. Boundary candidates (a) and SSM with its novelty curve (b) in the sample n°1 of Beethoven Piano Sonatas Dataset.

length, we add zeros to the segments of lesser length to calculate the distance between all of them.

Once $\mathbf{S}$ is computed, we can obtain its novelty curve $\mathbf{c}$ (Eq. 4) and apply the peak picking strategy again using a sliding window w_2 and a threshold τ_2. This will yield the predicted boundaries $\hat{\mathbf{b}}$.

$$c_i = norm(s_{i+1}, s_i) \tag{4}$$

where *norm* refers to the Euclidean norm.

In Fig. 2(a) we show the identified peaks (boundary candidates) in the normalized $\hat{\mathbf{b}}$ vector, and in Fig. 2(b) the SSM with the novelty curve and the predicted boundaries for the Schubert_D911-01.mid file of the Schubert Winterreise Dataset (SWD).

The procedure of the Norm method is summarized below, and its pseudocode can be found in Algorithm 1.

- Calculate the IOI's $\mathbf{x}$ and the local direction $\mathbf{l}$ vectors of the file and sum them: $\tilde{\mathbf{x}}$.
- Apply z-score normalization to $\tilde{\mathbf{x}}$: $\hat{\mathbf{x}}$.
- Calculate the boundary candidates, $\mathbf{b}$, by peak picking with window w_1 and threshold τ_1.
- Construct a Self-Similarity Matrix, $\mathbf{S}$, with the segments grouped by the boundary candidates obtained in the previous step.
- Get the novelty curve $\mathbf{c}$ from $\mathbf{S}$.
- Calculate the boundaries $\hat{\mathbf{b}}$ by peak picking with window w_2 and threshold τ_2.

Algorithm 1 Norm.

Require: w_1, τ_1, w_2, τ_2 ▷ params
$n \leftarrow read\_file$
▷ calculate candidates b
$x \rightarrow ioi(n)$
$l \rightarrow local\_direction(n)$ $\hat{x} \leftarrow sum(x, l)$
$z = z\_normalization(\hat{x})$
$b \rightarrow peak\_picking(z, w_1, \tau_1)$
▷ group notes by candidates b
$g \leftarrow$ empty list of lists
for i in b **do**
  $g \leftarrow append(n_i, n_{i+1})$
end for
▷ construct S
$S \leftarrow array_{len(b),len(b)}$
for i in g **do**
  $v_i \leftarrow sum(ioi(g_i)||local\_direction(g_i))$
  for j in g **do**
    $v_j \leftarrow sum(ioi(g_j)||local\_direction(g_j))$
    $S_{i,j} = euclidean\_dist(v_i, v_j)$
  end for
end for
$c \leftarrow novelty(S)$
▷ get $b\prime$ and the respective note positions
$b' \rightarrow peak\_picking(n, w_2, \tau_2)$
for k in b' **do**
  $b'\_notes \leftarrow append(b_k)$
end for

3.2. Graph representation

Our graph representation is based on previous works [18,20,41]. Since our representation is based on MIDI files, which might do not contain time signature and tempo information, neither bars nor beats are taken into account in our representation. We only extract the time signatures from the datasets metadata for measuring the performance of our methods in Section 4. Similar to Ref. [20], we create three types of undirected connections only between notes (nodes), however we do not encode rests since MIDI files do not provide them directly. In Eq. (5)

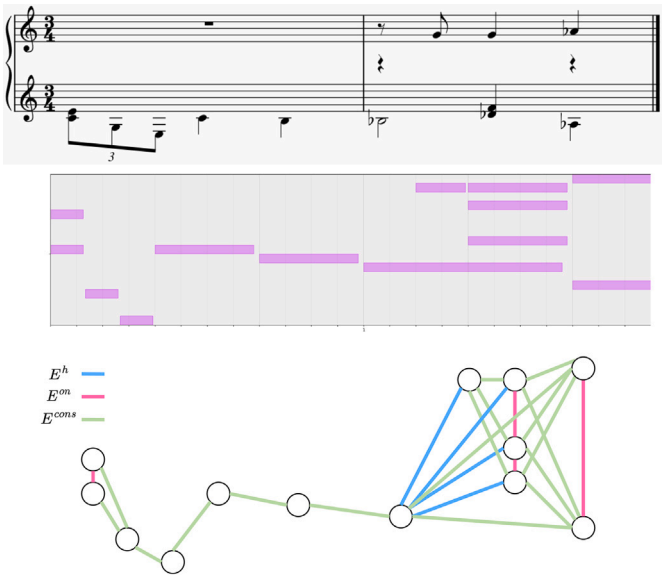


Fig. 3. From top to down: score, pianoroll and our proposed graph representation of 2 bars of symbolic music.

we can see the expression of the edges of our graph representation: E^{on} represents the edges between notes that occur on the same onset; E^{cons} are the edges between consecutive notes in time, and edges E^h represent overlapping notes in time (notes that start when there is already a note being played). The edges of the graph E are defined as $E \subseteq E^{on} \cup E^{cons} \cup E^h$:

$$\begin{aligned} e^{on}_{i,j} &= \{on(v_i) = on(v_j)\} \\ e^{cons}_{i,j} &= \{off(v_i) = on(v_j)\} \\ e^{h}_{i,j} &= \{[off(v_i) > on(v_j)] \wedge [on(v_i) < on(v_j)]\} \end{aligned} \tag{5}$$

where *on* and *off* refer to the notes *onsets* and *offsets*, respectively.

Nodes are not connected to each other. In Fig. 3 we show an example of 2 bars in a score, pianoroll and our graph representation.

3.2.1. PELT algorithm

In this method, we encode symbolic music as a graph as we described previously to then obtain the adjacency matrix $\mathbb{A}$ of the graph. Note that constructing $\mathbb{A}$ has a quadratic complexity with the number of notes n in the file: $\mathcal{O}(n^2)$. We then compute the novelty curve of $\mathbb{A}$ with the expression in Eq. (4). After that, we run the PELT algorithm with `ruptures` package[2] [44].

PELT [22] is a changepoint detection algorithm in time series. It detects the change points of a signal by optimizing a cost function C as defined in Eq. (6).

To calculate the optimal changepoint location, the algorithm starts by initializing the cost of the entire series C_i to infinity, then calculates the cost of each possible changepoint location per step, selects the point i that minimizes the cost and then updates the cost function with the new location i'. In Eq. (6) we show the expressions that correspond to this process. After that, the windows are reduced in size by a factor δ and the process in Eq. (6) is repeated.

$$\begin{aligned} C_i &= \min(C_j) + \lambda(i,j) + p \\ i' &= argmin(C_i) \\ C_i &= \min(C_i, C_{i'}) \end{aligned} \tag{6}$$

[2] https://github.com/deepcharles/ruptures.

where i, j is the segment to be processed, and p is the penalty, which controls the number of the detected changepoints.

The window size w controls the size of the analyzed segments per iteration, and the jump value j is related to the number of changepoints skipped in each iteration. We also define in this case the window size w as a function of the number of notes N so $w = \frac{\alpha}{N}$ where $N = 15$ and α is a constant parameter. We define the jump value $j = \beta \cdot w$ as a function of the window size w and a constant β. We find the optimal values of the parameters α and β in Section 4.

The complexity of the method depends on the cost function used. In our case, we use the Kernelized mean change cost function RBF of Eq. (7) which has a quadratic complexity corresponding to the sum of the computation of the kernel density estimate (KDE), $\mathcal{O}(n^2)$, and the computation of the cost at each candidate is $\mathcal{O}(n)$. This leads to an overall complexity of the PELT method of $\mathcal{O}(n^2 + nlog(n))$, since the number of iterations is determined by the logarithm of the ratio of the window size and the minimum size $\mathcal{O}(log(n))$.

$$C(\theta) = \sum_{t=1}^{T} \left\| y_t - \sum_{j=1}^{k} w_j \phi \left(\frac{\left\| x_t - \mu_j \right\|^2}{\sigma^2} \right) \right\|^2 \tag{7}$$

where $C(\theta)$ is the cost function, $\theta = w_j, \mu_j, \sigma^2$ is a set of parameters, T is the number of points in the signal, y_t is the observed value at time t, x_t is the feature value at time t, w_j is the weight for the j^{th} RBF component, μ_j is the mean of the j^{th} RBF component, σ^2 is the variance of the RBF components, and $\phi(u) = \exp(-u)$ is the RBF function $\|\cdot\|$ is the Euclidean norm.

In Fig. 4 we can see the predicted boundaries in a sample of the Schubert Winterreise Dataset (SWD). We can observe how the algorithm predicts different boundaries at different levels of structure.

In Algorithm 2 we provide the pseudocode of the PELT algorithm re-adapted for our graph representation and renamed to G-PELT.

Algorithm 2 G-PELT.

Require: w, j, p ▷ params
$n \leftarrow read\_file$
$G \leftarrow midi\_to\_graph(n)$
$A \leftarrow adjacency\_matrix(G)$
$c \leftarrow novelty(A)$
$b' \leftarrow$ **PELT**(n, w, j, p)

3.2.2. Window algorithm

In this method, we also encode symbolic music as a graph and compute the novelty **c** from the adjacency matrix **A** of the graph as we do in the G-PELT algorithm. After that, we apply a sliding window algorithm. The algorithm uses two windows $y_{i,j}$ and $y_{j,k}$, that are compared by computing a discrepancy measure with the cost function C of each window as we show in Eq. (8).

$$d(y_{i,j}, y_{j,k}) = C(y_{i,j}) - C(y_{i,k}) - C(y_{j,k}) \tag{8}$$

The window size W defines the segment splits for each signal point i as: $y_{i-w/2,i}, y_{i,i+w/2}$.

The computational complexity of the algorithm is $\mathcal{O}(nw)$. We need to add the complexity of the cost function C. We use the same function defined for G-PELT algorithm (see Eq. (7)), which computational complexity in this case is $\mathcal{O}(nw)$ since the complexity of computing the KDE cost function for a single window is $\mathcal{O}(w^2)$ and since the window is moved along the data, the total time required to compute the cost function for the entire data set is $\mathcal{O}(nw)$. This leads to an overall complexity of the sliding window algorithm to $\mathcal{O}(2nw)$ (Table 1).

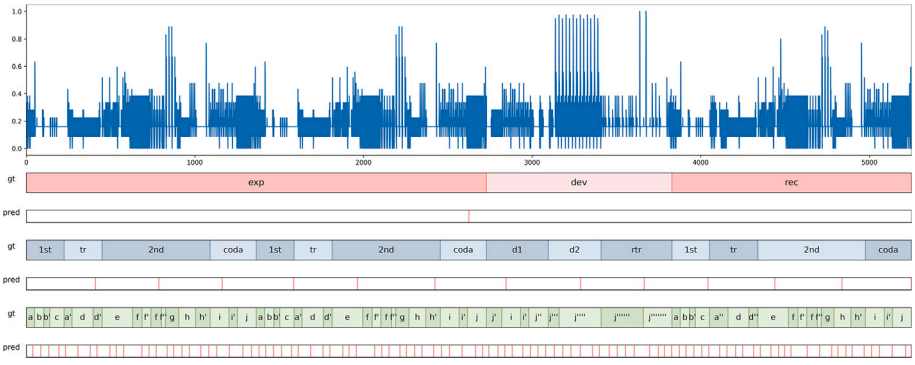


Fig. 4. Adjacency matrix novelty curve of the graph built of the Beethoven piano sonata n11, 1st movement. In the figure, gt and pred refers to the ground truth annotations and predicted boundaries with out G-PELT method, respectively. The x axis refers to the notes in the file and the y axis is the value of the novelty.

Table 1
Parameters and complexity of the algorithms for symbolic music structure segmentation. Note that the complexity of constructing the adjacency matrix is separated from the complexity of PELT and Window algorithms.

Algorithm	Params.	Complexity
Norm	$\alpha_1, \tau_1, w_2, \tau_2$	$\mathcal{O}(n) + \mathcal{O}(b^2)$
G-PELT	α, β, p	$\mathcal{O}(n^2) + \mathcal{O}(n^2 + nlog(n))$
G-Window	α, β, p	$\mathcal{O}(n^2) + \mathcal{O}(2nw)$

Table 2
Multi-instrument dataset analysis.

Dataset	files	TSig: files	Total bound.	Boundaries per file
SWD	23	2/4: 7 3/4: 7 4/4: 4 3/8: 1 6/8: 3 12/8: 1	mid: 192	mid: $8.41_{\pm 2.79}$
BPS	31	2/4: 8 3/4: 6 4/4: 13 3/8: 1 6/8: 2 12/8: 1	low: 1.439 mid: 438 high: 115	low: $46.42_{\pm 21.55}$ mid: $14.13_{\pm 3.46}$ high: $3.8_{\pm 1.05}$

4. Ablation study

In this section we perform an ablation study to compare the methods described in Section 3 with different parameter values. The goal of this comparison is to find the optimal parameter values for each method and dataset, and to determine the method that outperforms the others.

As we mentioned in Section 3, our representation is based on MIDI files where no information about bars or beats is provided. This makes the algorithms more robust since MIDI files may not be quantized, especially files containing expressive performances. Thus, it is easier to convert a MusicXML file to MIDI and use our method for score segmentation rather than converting a MIDI file to MusicXML due to the ornamentations and the lack of symbolic information that a MIDI file provides in comparison to a score.

We convert MIDI files into graphs. The implementation has been added to the `musicaiz` package [16] which uses the `NetworkX` package [3] [14] to perform the conversion. We test the algorithms performance with `mir_eval` package [34].

For the evaluation, we select 2 tolerances: 1 beat and 1 bar tolerances. The 1 beat tolerance has been used previously in the cadences identification task [20]. We added the 1 bar tolerance to provide further insights into the performance of the algorithms.[4]

4.1. Datasets

We use the Schubert Winterreise Dataset (SWD) which contains 24 files (MIDI, MusicXML, PDF and audios) annotated with harmony and form analysis (our task). Since we aim to test our methods for different music forms, we also test the algorithms with the Beethoven Sonatas Dataset (BPS) which contains the annotations of the 32 first movements of Beethoven piano sonatas. However, this dataset provides the symbolic music files as `csv`, so we converted them to MIDI format with `musicaiz` package to be able to read, process and measure them against the structure annotations provided by the dataset. In Table 2 we show the metadata of both datasets.[5] Whereas SWD has only annotations of the structure in the middle level, BPS dataset has annotations for the three levels that we called: low, mid and high. We will provide the evaluation results of Essen Folk Dataset in Section 5, since our ablation study comprehends multi-instrument data.

To compare the performance of our methods, we set the lower bound (renamed to baselines from now on) for each dataset following the procedure introduced in previous works [12]. This limit represents randomly placed boundaries. For the SWD dataset we set five synthetic equidistant boundaries and in the BPS dataset we set 4, 14 and 46 boundaries per file which correspond to the mean of the boundaries per file and level (see Table 2).

4.2. Schubert winterreise dataset (SWD)

In this subsection, we test the performance of the algorithms in the SWD. To be able to process the information and convert the symbolic information to graphs, we use `musicaiz` package [16] which works only with MIDI files, thus, we use the raw MIDI files of SWD.

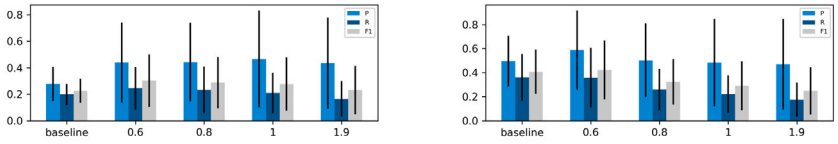
4.2.1. Norm

We test the Norm algorithm varying its parameters α_1, τ_1, w_2, and τ_2 in order to find their optimal values. In Fig. 5(a) we show the performance of the Norm method varying α_1 and in Fig. 5(c) we vary w_2 with

[3] https://github.com/networkx/networkx.

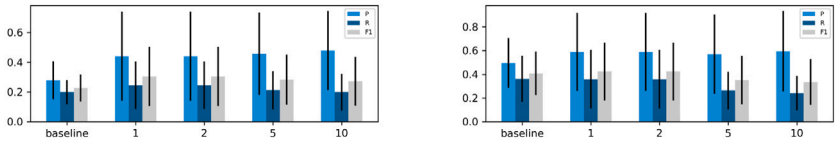
[4] Note that our tolerance values are analogous to the boundary detection task in the audio domain, where there are two tolerance values 0.5 and 3 s. 1 beat tolerance means that in a 3/8 bar, the beat will be the crotchet.

[5] We exclude file 11 since it contains tempo changes.



(a) Norm 1 beat with α_1 variation and τ_1=1, w_2=2, τ_2=0.5.

(b) Norm 1 bar with α_1 variation and τ_1=1, w_2=2, τ_2=0.5.



(c) Norm 1 beat with α_1 variation and τ_1=1, w_2=2, τ_2=0.5.

(d) Norm 1 bar with w_2 variation and τ_1=1, α_1=0.6, τ_2=0.5.

Fig. 5. Comparison of Norm method in the SWD dataset setting different α_1 and w_2 values for 1 beat and 1 bar tolerances.

Table 3
MBD results for the algorithms with optimal parameters and measured with SWD.

	SWD		
	P	R	F_1
	Tolerance: 1 beat		
baseline	$0.2782_{\pm 0.12}$	$0.1980_{\pm 0.08}$	$0.2258_{\pm 0.08}$
Norm **b**	$0.2017_{\pm 0.14}$	$0.4352_{\pm 0.31}$	$0.2556_{\pm 0.17}$
Norm **b**′	$0.4398_{\pm 0.30}$	$0.2450_{\pm 0.15}$	$0.3029_{\pm 0.19}$
G-PELT	$0.2665_{\pm 0.15}$	$0.5265_{\pm 0.28}$	$\mathbf{0.3455_{\pm 0.18}}$
G-Window	$0.3066_{\pm 0.21}$	$0.3372_{\pm 0.29}$	$0.2869_{\pm 0.20}$
	Tolerance: 1 bar		
baseline	$0.4956_{\pm 0.21}$	$0.3619_{\pm 0.19}$	$0.4078_{\pm 0.18}$
Norm **b**	$0.3415_{\pm 0.18}$	$0.7199_{\pm 0.31}$	$0.4298_{\pm 0.18}$
Norm **b**′	$0.5884_{\pm 0.32}$	$0.3588_{\pm 0.24}$	$0.4228_{\pm 0.24}$
G-PELT	$0.4366_{\pm 0.13}$	$0.8473_{\pm 0.17}$	$\mathbf{0.5640_{\pm 0.12}}$
G-Window	$0.5371_{\pm 0.23}$	$0.5527_{\pm 0.29}$	$0.4863_{\pm 0.19}$

the optimal α_1 value for 1 beat tolerance. In Fig. 5(b) and (d) we repeat the same procedure but testing algorithm with 1 bar tolerance.

The results show that there is no huge variation of the metrics when measuring the results with 1 beat tolerance; however, when testing the results with 1 bar tolerance, the sensibility of the results against α and w_2 increases. The best performing parameters for both 1 beat and 1 bar tolerances are $\alpha = 0.6$, $w_2 = 2$, $\tau_1 = 1$, and $\tau_2 = 0.5$ with a $F_1 = 0.3029$ for 1 beat and $F_1 = 0.4228$ for 1 bar tolerances (see Table 3).

4.2.2. G-PELT

We follow the same procedure as done with the Norm method to find the optimal parameter values for the G-PELT method: α and β. We fix the penalty p to 0.7 which we found to perform better than other values. In Fig. 6 we show the results of our experiments with G-PELT method in SWD.

Looking at the results in Fig. 6, we can see that the method is sensitive to the α parameter, especially the Recall metric. The best-performing parameter values are $\alpha = 0.6$, $\beta = 0.15$ and $p = 0.7$ for both 1 beat and 1 bar tolerances with $F_1 = 0.3455$ for 1 beat and $F_1 = 0.5640$ for 1 bar tolerances (see Table 3).

4.2.3. G-window

As we did with the previous methods, we find the optimal parameter values for the G-Window method: α and β. We fix the penalty p to 0.5 in this case. In Fig. 7 we show the results of our experiments with G-Window method in SWD.

The results in Fig. 7 show that the G-Window method is also sensible to its parameter α. We found that the best performing parameter values are $\alpha = 1$ and $p = 0.5$ for both 1 beat and 1 bar tolerances with a $F_1 = 0.2869$ for 1 beat and $F_1 = 0.4863$ (for 1 bar tolerances see Table 3).

4.3. Beethoven sonatas for piano (BPS)

After testing the performance of our algorithms with the SWD, we now follow a similar procedure with the BPS dataset. In spite of the fact that the BPS dataset was originally proposed for symbolic harmonic analysis [6], it also contains structure annotations of the first movements of Beethoven sonatas for piano at 3 levels, which makes it suitable for MSA [11]. The structure annotations for each file are stored in `phrases.xlsx`.[6] To test the performance of our methods with this dataset, we follow the same procedure that we described previously for the SWD. The only difference between this dataset and the SWD is that SWD provides only middle level annotations whereas in the BPS we will analyzed the low, middle and high structure levels since the dataset provides the annotations per level. To show how the novelty of the graph adjacency matrix represents well the structure of the symbolic music, and to show an example of a file, we present in Fig. 4 the predicted boundaries in a sample of the BPS dataset with the G-PELT method.

4.3.1. G-PELT

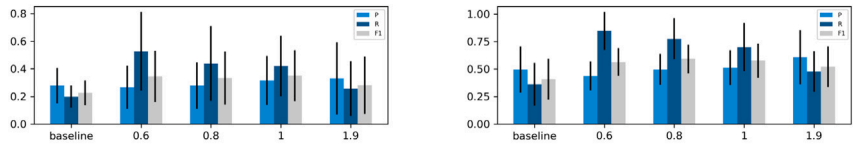
As we did for the SWD, we optimize the parameter values for the BPS dataset. However, since we found that the best performing method in the SWD was G-PELT, we will optimize the values of the parameters for this algorithm for the three structure levels and we will compare the performance in each level with the other methods (Norm and G-Window). In Table 7 we show the results of the ablation of the three methods for each structure level and tolerance.

Similar to the ablation with the SWD, we first vary α and then with the optimal α value we find the optimal β. The penalty is fixed and obtained for each structure level. In Fig. 8 we show a comparison of different parameter values for G-PELT method in the structure levels (low, mid and high) in the BPS dataset.

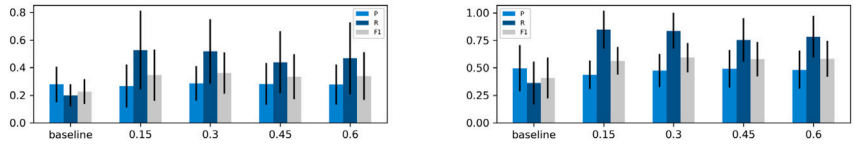
Comparing the results with the SWD in general, we found similarities as for the high Recall values in comparison with the Precision. However, in the mid level the Precision is higher than the Recall for almost all the α and β tested values. This means that the method predicts lower False Positives and higher False Negatives, or in other words, that the method will miss real boundaries (lower Recall) but the predicted boundaries will be more likely to be the real ones (higher Precision).

After the ablation of the parameters for each model and dataset, in Table 4 we show the optimal parameters for both SWD and BPS datasets.

[6] File 31 end has an error since a segment start must be smaller than the end. We fixed it changing the value 246–346 that are the actual beats in the file.

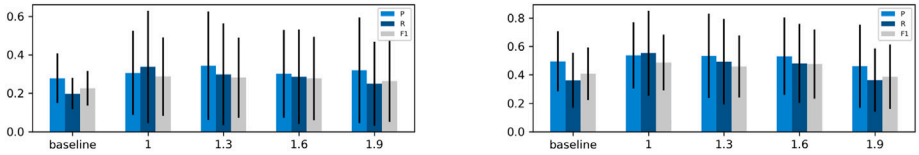


(a) G-PELT 1_beat with α variation and β=0.15. (b) G-PELT 1_bar with α variation and β=0.15



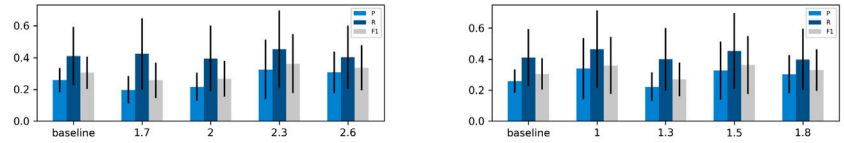
(c) G-PELT 1_beat with β variation and α=0.6 (d) G-PELT 1_bar with β variation and α=0.6

Fig. 6. Comparison of G-PELT method in the SWD dataset setting different α and β values for 1 beat and 1 bar tolerances. The penalty value is fixed to 0.7.

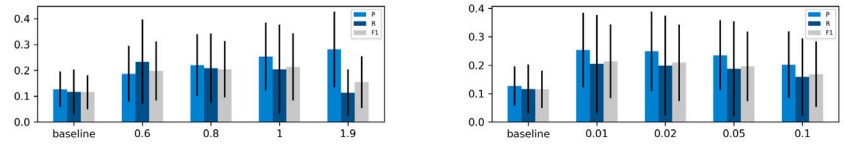


(a) G-Window 1_beat with α variation. (b) G-Window 1_bar with α variation.

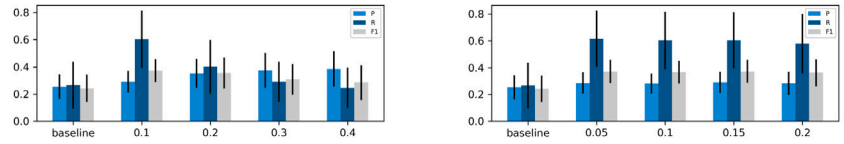
Fig. 7. Comparison of G-Window method in the SWD dataset setting different α values for `1_beat` and `1_bar` tolerances. The penalty value is fixed to 0.5.



(a) High-level G-PELT 1_bar with α variation and β=1.5. (b) High-level G-PELT 1_bar with β variation and α=2.3.



(c) Mid-level G-PELT 1_beat with α variation and β=0.01. (d) Mid-level G-PELT 1_beat with β variation and α=1.



(e) Low-level G-PELT 1_beat with α variation and β=0.15. (f) Low-level G-PELT 1_beat with β variation and α=0.1.

Fig. 8. Comparison of G-PELT method in the BPS dataset setting different α and β values for the high, mid and low levels. High level is tested with 1 bar and mid and low levels with 1 beat tolerances. The penalty value is fixed to 4, 0.5 and 0.1 for the high, mid and low levels, respectively.

The extensive ablation results presented in Figs. 5 through 8 are summarized in Table 5, addressing the need for a clearer understanding of parameter-metric relationships. For all methods, we observe a general tendency where increasing the parameters typically leads to an increase in the metrics. Notably, the `Norm` method exhibited a marked increase in sensitivity to α_1 and w_2 when measured with the **1 bar** tolerance, a trend not as pronounced with the stricter 1 beat tolerance. Conversely, the G-PELT algorithm on the BPS mid-level showed an interesting trade-off where P remained higher than R for most α values, indicating a robust prediction set with fewer false positives, but a tendency to miss real boundaries. This concise summary aids in supporting the model's robustness and enhances the reproducibility of the optimal parameter selection reported in Table 4.

5. Evaluation

In this section, we provide the results of our methods evaluated in the three datasets: SWD, BPS and Essen Folk Dataset. In spite of the fact that the aim of our work is not to segment monophonic melodies, we ran our algorithms in the Essen Folk dataset [39] to compare our models with previous works. We use the sequences provided in [45], which

Table 4
Optimal parameters and for each algorithm and dataset.

Dataset	level	Algorithm	Params
SWD	mid	Norm	$\alpha_1 = 0.6$, $\tau_1 = 1$
			$\alpha_2 = 2$, $\tau_2 = 0.5$
		G-PELT	$\alpha = 0.6$, $\beta = 0.15$, $p = 0.7$
		G-Window	$\alpha = 1$, $p = 0.5$
BPS	high	G-PELT	$\alpha = 2.3$, $\beta = 1.5$, $p = 4$
	mid	G-PELT	$\alpha = 1$, $\beta = 0.01$, $p = 0.5$
	low	G-PELT	$\alpha = 0.1$, $\beta = 0.15$, $p = 0.1$

Table 5
Summary of ablation study: Effect of key parameters on Precision (P) and Recall (R).

Dataset	Algorithm	Parameter	Trend (P, R)
	G-PELT	α	$\uparrow \alpha \Rightarrow \uparrow R$
		β	$\approx$ constant / Low sensitivity (Fig. 6(c,d))
	G-Window	α	$\uparrow \alpha \Rightarrow \uparrow R$
BPS	G-PELT (Low)	α	$\downarrow \alpha \Rightarrow \uparrow R$
	G-PELT (Mid)	α	$\uparrow \alpha \Rightarrow \uparrow P$; $\downarrow \alpha \Rightarrow \uparrow R$
	G-PELT (High)	α	$\approx$ constant

Table 6
MBD results for the algorithms with optimal parameters and measured with Essen Folk Dataset for 1 beat tolerance.

	P	R	F_1
ΔIOI [5]	$0.7165_{\pm 0.12}$	$0.5631_{\pm 0.23}$	$0.6307_{\pm 0.15}$
Compound [5]	$0.8165_{\pm 0.24}$	$0.7631_{\pm 0.11}$	$0.7890_{\pm 0.13}$
Norm **b**	$0.6371_{\pm 0.11}$	$0.4096_{\pm 0.45}$	$0.4985_{\pm 0.34}$
Norm **b′**	$0.6890_{\pm 0.23}$	$0.6231_{\pm 0.12}$	$0.6542_{\pm 0.12}$
G-PELT	$0.7871_{\pm 0.15}$	$0.8898_{\pm 0.28}$	$\mathbf{0.8353_{\pm 0.15}}$
G-Window	$0.8137_{\pm 0.22}$	$0.7981_{\pm 0.24}$	$0.8058_{\pm 0.16}$

contain 6226 monophonic European melodies where phrase boundaries are annotated. For SWD and BPS datasets, after finding their optimal hyperparameter values of the algorithms in Section 4, we provide the overall evaluation results in this section.

First, we test the performance of the algorithms in the Essen Folk Dataset to compare them with the state-of-the-art methods and prove the effectiveness of our methods in monophonic symbolic music. Since the dataset is composed of monophonic melodies instead of full pieces like the SWD and BPS datasets, we use the same hyperparameters as for the low-level structure detection in the BPS dataset, which matches the structure level in the Essen Folk Dataset. The results are shown in Table 6.

The results show that the G-PELT is the best performing method, however, G-Window is the algorithm that achieves better P. This is because the G-Window local sliding windows capture more boundaries per file than the G-PELT method designed for global optimization. This is also why G-PELT has a higher R.

We measured the errors across the datasets to give more insights about the algorithms precision in terms of time tolerance. In Fig. 9 we show the histogram of errors in beats that the G-PELT algorithm makes in relation with the number of boundaries detected. We can observe that the most common error rage is from 0 to 10 beats, which means that when the algorithm makes a prediction, it is relatively close to the real boundary.

The results in Table 3 show the performance in the SWD, however, the analysis can be confusing due to the fact that the SWD does not have annotations per structure level. In order to do a deeper analysis and show how the algorithms perform in the different levels of the structure of music, we also test the algorithms with the BPS dataset. This dataset

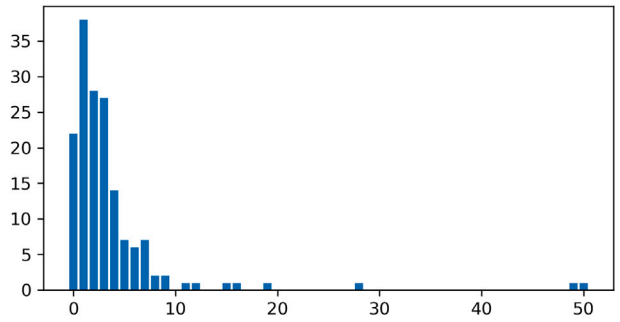


Fig. 9. Number of errors (vertical axis) in beat units (horizontal axis) measured in the predicted boundaries with G-PELT method in the SWD dataset.

provides annotations at the 3 structure levels in Western classical music. Since the dataset contains the Beethoven piano sonatas, the levels are organized as: high-level structure (exposition, development and recapitulation), themes (1st, tr, 2nd, coda, etc.), and phrases or motifs (a, a', etc.). Low level boundaries contain the mid and high level ones, and mid level boundaries also contain high level ones. With these annotations we can measure the algorithms performance in various levels and demonstrate in which level they perform better. In Table 7 we show the performance metrics of the algorithms measured with the BPS dataset (Fig. 10).

Beyond quantitative performance, we conducted a qualitative analysis to examine whether the predicted boundaries correspond to musically meaningful transitions. In Fig. B.12, we show how G-PELT successfully identifies boundaries that align with perceptually relevant musical events such as thematic changes, cadences, and rhythmic contrasts. For instance, the transition around bars 58–59 corresponds to the shift from theme C to D, characterized by changes in harmonic rhythm and texture in the left hand. Similarly, the algorithm detects a cadence at bar 66, a musically plausible segmentation point not reflected in the annotated data. These examples demonstrate that the detected boundaries are not merely algorithmic artifacts but are consistent with structural cues recognized in traditional musical analysis.

In Appendix B we see another example of the G-PELT algorithm performance in a Debussy piece, which differs in period and form from the SWD for which the parameters have been optimized.

6. Discussion

The best-performing algorithm for the SWD is G-PELT as we show in Table 3 with $P = 0.4366$, $R = 0.8473$ and $F_1 = 0.5640$ (optimal parameters are shown in Table 4). The high Recall value means that the algorithm is capable of identifying boundaries in the ground truth and that it does not predict boundaries that are not in the ground truth (low number of False Negatives). However, the low Precision shows that the algorithm does not detect a high number of the real boundaries (high number of False Positives). This is an indicator that the algorithm could be used in scenarios where the recall of the algorithm is a priority.

When testing the performance of the methods in the BPS dataset, we find differences in the results depending on the structure level we are working with. We should clarify that the performance in the high-level structure is more difficult since there are only 3–5 boundaries versus the 40–60 that the low-level presents per file (see Table 2). However, the algorithms detect the end of the file boundary, which is translated as the addition of a True Positive. The impact of this in the high-level leads to a greater effect than in the mid and low levels. The reason why the expectation of a decrease in performance in the high level analysis comes from the fact that if a file has ~800 beats (BPS file n1) and three boundaries in the high level, the boundaries represent less than 0.5 % of the beats, making it difficult for the algorithm to perform more accurately at lower tolerance levels (`1_beat`). It is worth noting that the same applies

Table 7
MBD results for the algorithms with optimal parameter values measured with the BPS dataset.

	High Level			Mid Level			Low Level		
	P	R	F_1	P	R	F_1	P	R	F_1
	Tolerance: 1 beat								
baseline	0.1875	0.3041	0.2233	0.1272	0.1169	0.1155	0.2533	0.2656	0.2417
Norm **b**	0.0579	0.4559	0.0987	0.1036	0.3773	0.1573	0.1829	0.7612	0.2857
Norm **b′**	0.2585	0.3906	0.2880	0.2378	0.1038	0.1308	0.2969	0.2450	0.2489
G-PELT	0.2765	0.3114	0.2686	0.2540	0.2045	0.2137	0.2896	0.6033	0.3716
G-Wind.	0.2092	0.2614	0.2183	0.2436	0.1531	0.1724	0.4816	0.1205	0.1849
	Tolerance: 1 bar								
baseline	0.2578	0.4088	0.3040	0.2611	0.2557	0.2447	0.4415	0.4676	0.4254
Norm **b**	0.0862	0.6432	0.1455	0.1820	0.6347	0.2755	0.2506	0.9711	0.3854
Norm **b′**	0.2708	0.4140	0.3028	0.3674	0.1721	0.2132	0.5904	0.4458	0.4696
G-PELT	0.3437	0.3585	**0.3361**	0.3865	0.3024	**0.3224**	0.4370	0.8434	**0.5473**
G-Wind.	0.2466	0.3192	0.2617	0.3566	0.2345	0.2611	0.7538	0.1989	0.2999

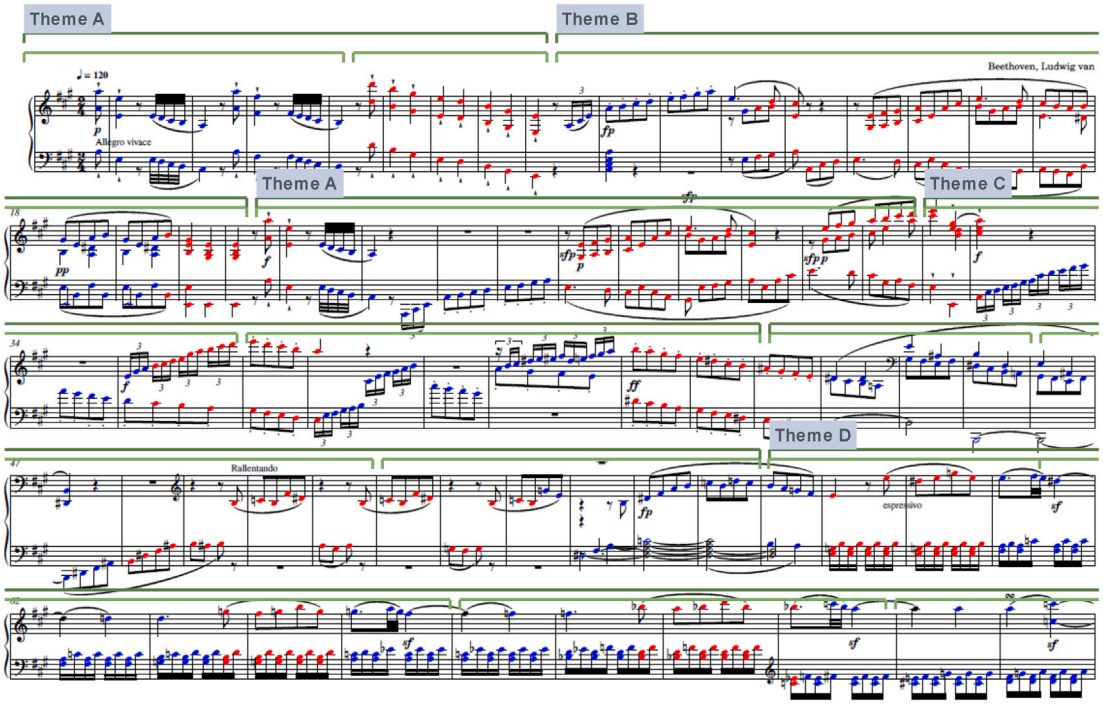


Fig. 10. Beginning of the 1st movement of the Beethoven Piano Sonata n2 segmented by our G-PELT method with optimal parameter values. Predicted segments are colored in red and blue. Ground truth is marked in bright green (low level, 5th column in the csv) and dark green (mid level, 6th column in the csv) lines above the staff.

to the baseline, as the number of boundaries in annotations increases, it becomes more likely to increase the number of true positives because there are also more synthetic boundaries per file.

An example of the performance of the G-PELT method in a Beethoven piano sonata is shown in Fig. B.12. In the figure, we show the predicted sections with colors and the ground truth annotations of the BPS dataset in the green lines above the score.[7] We would like to highlight how the algorithm identifies sections that are not labeled in the dataset annotations, such as cadences. As an example, there is an annotation in bar 58 (theme D start) that, for our understanding, does not correspond to the real boundary. In this case, the method identifies as the real boundary the bar 59 which is more likely to be the real one due to the rhythm changes in the left hand. The same happens in bar 32 (theme C) and in bar 66 where the annotation seem to be displaced from the real boundary. Being that said, we would suggest that the BPS dataset structure annotations should be revised.

Finally, these qualitative examples reinforce that the detected boundaries correspond to musically meaningful transitions, linking our algorithmic evaluation with concrete musicological phenomena such as cadences, thematic contrasts, and phrase demarcations.

7. Conclusion

We presented three methods that aim to segment symbolic music in its structure or form. After measuring the performance of each method in the context of symbolic structure segmentation with two public datasets, we can conclude that the best performing method is G-PELT for both SWD and BPS datasets and structure levels. We provide evidence of how by changing the parameters of the algorithms, music can be segmented into different levels of structure, opening new possibilities to better understand how music is created and enabling new scenarios in

[7] The ground truth has been extracted from https://github.com/MarkGotham/Taking-Form/blob/master/corpus/Beethoven_Sonatas/sonata2op2no2movt1.csv

applications such as music generation or classification. The proposed methods are online and unsupervised which makes them suitable for processing large corpus of data, however, for a deeper analysis of a particular form, a learning algorithm might be needed to improve the results of this work. We suggest revisiting the annotations of the BPS dataset by a group of expert musicians. This could be done with the help of our G-PELT method which is our best performing method, as a guide when analyzing the low level structure.

The presented segmentation method can be used in applications such as music generation or data augmentation. The fast online proposed method would help tokenize the files faster than offline methods. In applications such as fingering [35], data augmentation is difficult due to the fact that common techniques (pitch shifting, etc.) cannot be applied since they change the fingering or difficulty of the task. Being able to segment music into coherent parts might help these models for training purposes. Automatic structural segmentation can also help annotators when it comes to annotating the structure at a large corpus of data.

Future lines of work might be building a deep neural graph network that segments music based on these methods. In addition, having a better performance in the segmentation task might open future works about structure labeling or classification. Apart from that, since we only use temporal information to construct the graph, we propose adding harmonic information in the novelty curve obtained from the adjacency matrix to improve the results of this work. We did not include them due to the fact that these features need to be predicted, as they are not encoded in a MIDI file. However, future work might focus on encoding the chord progression predicted with recent advancements in harmonic analysis (e.g. AugmentedNet [32]) to better capture the structural boundaries, since harmony and structure are closely related to each other.

CRediT authorship contribution statement

Carlos Hernandez-Olivan: Writing – original draft, Visualization, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Jose R. Beltran:** Supervision, Resources, Project administration, Funding acquisition.

Funding

Spanish Science, Innovation and University Ministry by the RTI2018-096986-B-C31 contract. Aragonese Government by the AffectiveLab-T60-20 R project.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Carlos Hernandez-Olivan reports that financial support was provided by the University of Zaragoza. Jose R. Beltran reports a relationship with the University of Zaragoza that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors would like to thank Pedro Ramoneda for his insightful comments and Sonia Rubio for the help.

Appendix A. Structure encoding example

A.1. Applications

The presented segmentation methods can be used in symbolic music generation or data augmentation. In music generation, current tokenizers such as the Multi-Track Music Machine (MMM) [8] do not provide section tokens due to the fact that they are not encoded in MIDI-like or score-like formats. With our method, these section tokens could be included (at beat or bar level) to generate symbolic music and be able to inpaint sections, e.g., readapting the MMMBar tokenization to include section tokens. In Fig. A.11 we show an example based on the MMMTrack tokenizer [8] that includes the section tokens.

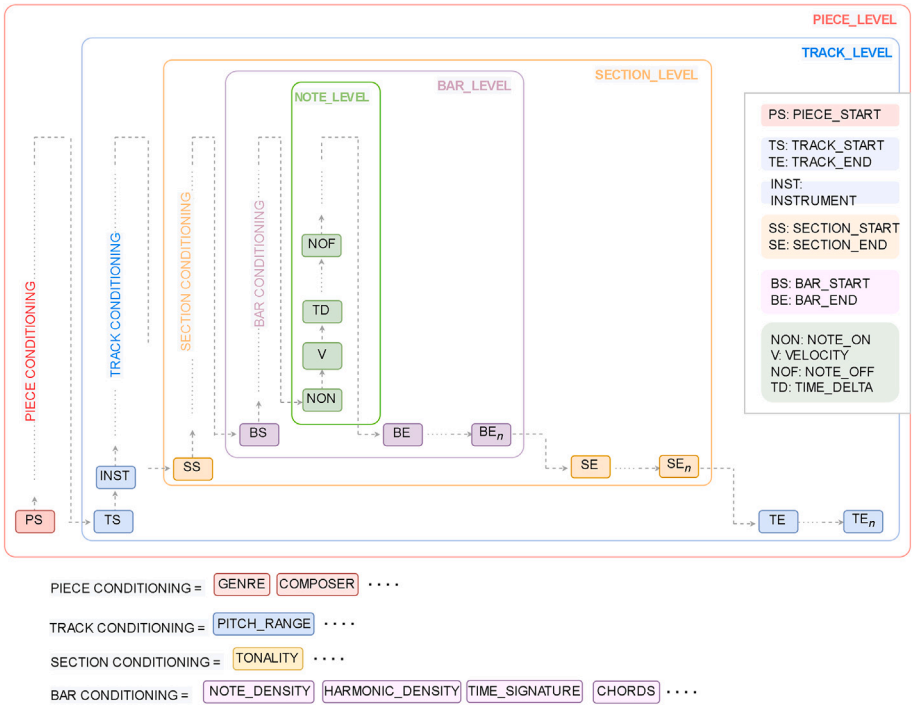


Fig. A.11. A general encoding based on the MMM. The figure shows a general scheme configuration of how the MMMTrack encoding extended for conditioning the generation at different levels. We extend the three levels proposed in the original MMM by adding a structure level.

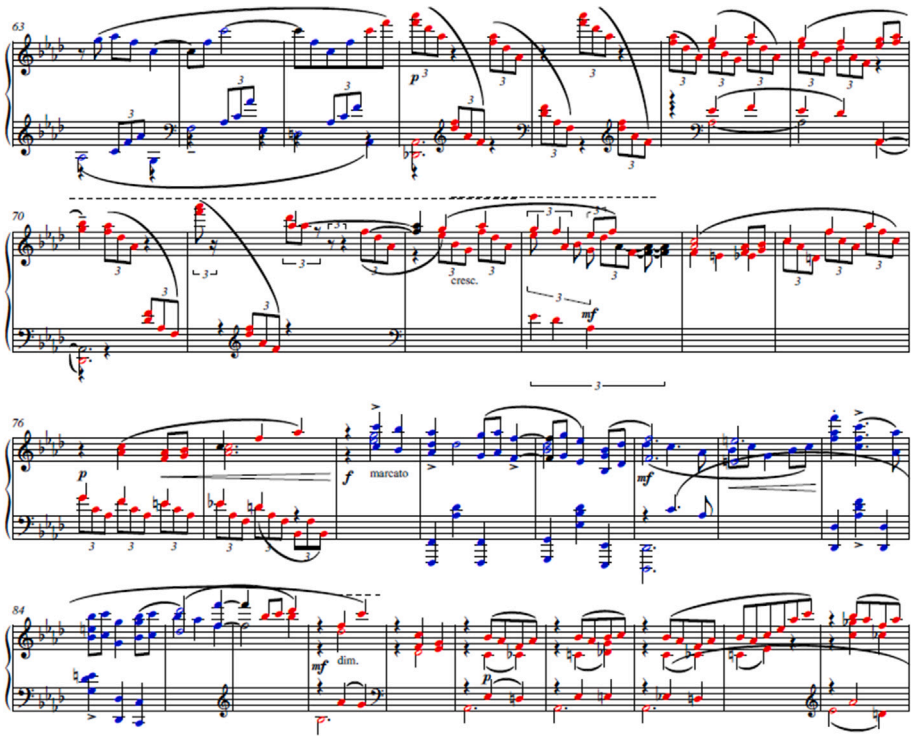


Fig. B.12. Vals romantique by Debussy (bars 63–93) segmented by our G-PELT method with parameters optimized for Shubert Winterreise Dataset.

Appendix B. G-PELT test example

With the G-PELT algorithm with optimized parameters with SWD we show how the algorithm works in other music styles and periods. In Fig. B.12 we show an example of a Vals by Debussy which is a different form and a composer from a different period in the Western classical music than the Schubert's Winterreise songs. In the figure, it can be seen that the algorithm segments coherent sections attending to the rhythm and texture of the piece, despite the parameter values having been optimized for another form and style.

Data availability

We shared the GitHub repository link to improve reproducibility.

References

[1] P. Allegraud, L. Bigo, L. Feisthauer, M. Giraud, R. Groult, E. Leguy, F. Levé, Learning sonata form structure on mozart's string quartets, Trans. Int. Soc. Music Inf. Retr. 2 (2019) 82–96.
[2] S. Bassan, Y. Adi, J.S. Rosenschein, Unsupervised symbolic music segmentation using ensemble temporal prediction errors, Preprint (2022).
[3] E. Cambouropoulos, The local boundary detection model (LBDM) and its application in the study of expressive timing, in: Proceedings of the 2001 International Computer Music Conference, ICMC 2001, Havana, Cuba, September 17–22, 2001, Michigan Publishing, 2001, https://hdl.handle.net/2027/spo.bbp2372.2001.021.
[4] E. Cambouropoulos, C. Tsougras, Influence of musical similarity on melodic segmentation: representations and algorithms, in: Journées D'informatique Musicale, 2004.
[5] Z. Cenkerová, M. Hartmann, P. Toiviainen, Crossing phrase boundaries in music, in: Proceedings of the Sound and Music Computing Conferences, 2018.
[6] T. Chen, L. Su, Functional harmony recognition of symbolic music data with multi-task recurrent neural networks, in: E. Gómez, X. Hu, E. Humphrey, E. Benetos (Eds.), Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23–27, 2018, 2018, pp. 90–97, http://ismir2018.ircam.fr/doc/pdfs/178_Paper.pdf.
[7] J. Devaney, C. Arthur, N. Condit-Schultz, K. Nisula, Theme and variation encodings with roman numerals (TAVERN): a new data set for symbolic music analysis, in: M. Müller, F. Wiering (Eds.), Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26–30, 2015, 2015, pp. 728–734.
[8] J. Ens, P. Pasquier, Mmm: exploring conditional multi-track music generation with the transformer, Preprint (2020).
[9] J. Foote, Visualizing music and audio using self-similarity, in: J.F. Buford, S.M. Stevens, D.C.A. Bulterman, K. Jeffay, H. Zhang (Eds.), Proceedings of the 7th ACM International Conference on Multimedia '99, Orlando, FL, USA, October 30–November 5, 1999, Part 1, ACM, 1999, pp. 77–80.
[10] M. Giraud, R. Groult, E. Leguy, F. Levé, Computational fugue analysis, Comput. Music J. 39 (2015) 77–96, http://www.jstor.org/stable/43829264.
[11] M. Gotham, M. Ireland, Taking form: a representation standard, conversion code, and example corpora for recording, visualizing, and studying analyses of musical form, in: A. Flexer, G. Peeters, J. Urbano, A. Volk (Eds.), Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4–8, 2019, 2019, pp. 693–699.
[12] T. Grill, J. Schlüter, Music boundary detection using neural networks on spectrograms and self-similarity lag matrices, in: 23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, August 31–September 4, 2015, IEEE, 2015, pp. 1296–1300.
[13] X. Guan, Z. Dong, H. Liu, Q. Li, Improving phrase segmentation in symbolic folk music: a hybrid model with local context and global structure awareness, Entropy 27 (2025) 460.
[14] A. Hagberg, P. Swart, D. Schult, Exploring Network Structure, Dynamics, and Function Using NetworkX. Technical Report. 2008. Los Alamos National Lab(LANL), Los Alamos, NM (United States).
[15] C. Hernandez-Olivan, J.R. Beltrán, Music Composition with Deep Learning: A Review, Springer International Publishing, Cham. pp. 2023, pp. 25–50. https://doi.org/10.1007/978-3-031-18444-4_2
[16] C. Hernandez-Olivan, J.R. Beltran, Musicaiz: a Python library for symbolic music generation, analysis and visualization, SoftwareX 22 (2023) 101365.
[17] C. Hernandez-Olivan, J.R. Beltran, D. Diaz-Guerra, Music boundary detection using convolutional neural networks: a comparative analysis of combined input features, Int. J. Interact. Multimed. Artif. Intell. 7 (2021) 78–88.
[18] D. Jeong, T. Kwon, Y. Kim, J. Nam, Graph neural network for music score data and modeling expressive piano performance, in: International Conference on Machine Learning, PMLR, 2019, pp. 3060–3070.
[19] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al., Highly accurate protein structure prediction with alphafold, Nature 596 (2021) 583–589.
[20] E. Karystinaios, G. Widmer, Cadence detection in symbolic classical music using graph neural networks, Preprint (2022).
[21] E. Karystinaios, G. Widmer, Graphmuse: a library for symbolic music graph processing, Preprint (2024).
[22] R. Killick, P. Fearnhead, I.A. Eckley, Optimal detection of changepoints with a linear computational cost, J. Am. Stat. Assoc. 107 (2012) 1590–1598.
[23] P. van Kranenburg, Rule mining for local boundary detection in melodies, in: J. Cumming, J.H. Lee, B. McFee, M. Schedl, J. Devaney, C. McKay, E. Zangerle, T. de Reuse (Eds.), Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11–16, 2020, 2020, pp. 271–278, http://archives.ismir.net/ismir2020/paper/000226.pdf.

[24] S. Lattner, C.E.C. Chacón, M. Grachten, Pseudo-supervised training improves unsupervised melody segmentation, in: Q. Yang, M.J. Wooldridge (Eds.), Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25–31, 2015, AAAI Press, 2015a, pp. 2459–2465.
[25] S. Lattner, M. Grachten, K. Agres, C.E. Cancino Chacón, Probabilistic segmentation of musical sequences using restricted boltzmann machines, in: International Conference on Mathematics and Computation in Music, Springer, 2015b, pp. 323–334.
[26] F. Lerdahl, R. Jackendoff, An overview of hierarchical structure in music, Music Perception (1983) 229–252.
[27] F. Lerdahl, R.S. Jackendoff, A Generative Theory of Tonal Music, Reissue, with a New Preface, MIT Press, 1996.
[28] M.R. López, A. Volk, Automatic segmentation of symbolic music encodings: a survey, Utrecht University (2012).
[29] D. Meredith, K. Lemström, G.A. Wiggins, Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music, J. New Music Res. 31 (2002) 321–345.
[30] B. Mokbel, A. Hasenfuss, B. Hammer, Graph-based representation of symbolic musical data, in: A. Torsello, F. Escolano, L. Brun (Eds.), Graph-Based Representations in Pattern Recognition, Berlin, Heidelberg, Springer Berlin Heidelberg, 2009, pp. 42–51.
[31] R.D. Morris, New directions in the theory and analysis of musical contour, Music Theory Spectr. 15 (1993) 205–228, https://doi.org/10.2307/745814
[32] N. Nápoles López, M. Gotham, I. Fujinaga, Augmentednet: a roman numeral analysis network with synthetic training examples and additional tonal tasks, in: Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR, 2021.
[33] Y. Panagakis, C. Kotropoulos, G.R. Arce, L1-graph based music structure analysis, in: A. Klapuri, C. Leider (Eds.), Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24–28, 2011, University of Miami, 2011, pp. 495–500, http://ismir2011.ismir.net/papers/PS4-4.pdf.
[34] C. Raffel, B. McFee, E.J. Humphrey, J. Salamon, O. Nieto, D. Liang, D.P. Ellis, C.C. Raffel, Mir_eval: a transparent implementation of common MIR metrics, in: In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR, 2014.
[35] P. Ramoneda, D. Jeong, E. Nakamura, X. Serra, M. Miron, Automatic piano fingering from partially annotated scores using autoregressive neural networks, in: J. Magalhães, A.D. Bimbo, S. Satoh, N. Sebe, X. Alameda-Pineda, Q. Jin, V. Oria, L. Toni (Eds.), MM '22: the 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10–14, 2022, ACM, 2022, pp. 6502–6510, https://doi.org/10.1145/3503161.3548372
[36] M. Rodríguez López, B. de Haas, A. Volk, et al., Comparing repetition-based melody segmentation models, in: Proceedings of the 9th Conference on Interdisciplinary Musicology (CIM14), SIMPK and ICCMR, 2014, pp. 143–148.
[37] M.E. Rodríguez-López, A. Volk, Symbolic segmentation: a corpus-based analysis of melodic phrases, in: M. Aramaki, O. Derrien, R. Kronland-Martinet, S. Ystad (Eds.), Sound, Music, and Motion - 10th International Symposium, CMMR 2013, Marseille, France, October 15–18, 2013. Revised Selected Papers, Springer, 2013, pp. 548–557, https://doi.org/10.1007/978-3-319-12976-1_33
[38] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, P.W. Battaglia, Learning to simulate complex Physics with graph networks. Preprint.
[39] H. Schaffrath, The Essen folksong collection, Database Containing 6 (1995).
[40] F. Simonetta, F. Carnovalini, N. Orio, A. Rodà, Symbolic music similarity through a graph-based representation, in: S. Cunningham, R. Picking (Eds.), Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion, Wrexham, United Kingdom, September 12–14, 2018, ACM, 2018, pp. 26:1–26:7, https://doi.org/10.1145/3243274.3243301
[41] W.M. Szeto, M.H. Wong, A graph-theoretical approach for pattern matching in post-tonal music analysis, J. New Music Res. 35 (2006) 307–321.
[42] D. Temperley, The Cognition of Basic Musical Structures, MIT Press, 2004.
[43] P. Toiviainen, T. Eerola, Autocorrelation in meter induction: the role of accent structure, J. Acoust. Soc. Am. 119 (2006) 1164–1170.
[44] C. Truong, L. Oudre, N. Vayatis, Selective review of offline change point detection methods, Signal Process. 167 (2020) https://doi.org/10.1016/j.sigpro.2019.107299
[45] P. Van Kranenburg, Melodic features for Meertens Tune collections and essen folksong collection (1.1) (2019) https://doi.org/10.5281/zenodo.3551003
[46] G. Wilder, Adaptive melodic segmentation and motivic identification, in: ICMC, 2008.
[47] G. Wu, S. Liu, X. Fan, The power of fragmentation: a hierarchical transformer model for structural segmentation in symbolic music generation (2022).
[48] G. Wu, S. Liu, X. Fan, The power of fragmentation: a hierarchical transformer model for structural segmentation in symbolic music generation, IEEE/ACM Trans. Audio Speech Lang. Process. 31 (2023) 1409–1420.

Author biography

Carlos Hernandez-Olivan completed his PhD in Electronics at the Universidad de Zaragoza in 2025 under the supervision of Dr. José R. Beltrán. He received the B.E. and M.Sc. degrees in Industrial Engineering in 2017 and 2019, respectively. He studied viola at the Professional Conservatory of Zaragoza where he received his professional certificate in 2013. He interned at SONY AI, SONY CSL and NTT CSL in Japan, and at Universal Music Group in the UK. In 2023, he was awarded with the VULCANUS Grant. He visited C4DM in Queen Mary University of London in 2025. His research interests are focused on Music Information Retrieval and acoustics, in particular, in the music analysis and generation systems with Artificial Intelligence.

Jose R. Beltran received the M.Sc. and Ph.D. degrees in Physics from the University of Zaragoza, Zaragoza, Spain, in 1988 and 1994, respectively. He is an Associate Professor with the Department of Electronic Engineering and Communications, University of Zaragoza. He has been involved in different research and development projects on Audio Analysis and Processing. His research interests are focused on the study of Automatic Learning Systems for the analysis, processing and synthesis of the musical signal. In 2008, he was a promoter of an academic spin-off: ARSTIC Audiovisual Solutions S.L. devoted to the use of technologies for the artistic and audiovisual fields. Prof. Beltrán is a member of the Aragon Institute for Engineering Research (I3A), Reseach Group in Advanced Interfaces (AffeciveLab).