Using Pairwise Link Prediction and Graph Attention Networks for Music Structure Analysis

Morgan Buisson, Brian Mcfee, Slim Essid

USING PAIRWISE LINK PREDICTION AND GRAPH ATTENTION NETWORKS FOR MUSIC STRUCTURE ANALYSIS

Morgan Buisson[1] **Brian McFee**[2,3] **Slim Essid**[1]
[1] LTCI, Télécom Paris, Institut Polytechnique de Paris, France
[2] Music and Audio Research Laboratory, New York University, USA
[3] Center for Data Science, New York University, USA

ABSTRACT

The task of music structure analysis has been mostly addressed as a sequential problem, by relying on the internal homogeneity of musical sections or their repetitions. In this work, we instead regard it as a pairwise link prediction task. If for any pair of time instants in a track, one can successfully predict whether they belong to the same structural entity or not, then the underlying structure can be easily recovered. Building upon this assumption, we propose a method that first learns to classify pairwise links between time frames as belonging to the same section (or segment) or not. The resulting link features, along with node-specific information, are combined through a graph attention network. The latter is regularized with a graph partitioning training objective and outputs boundary locations between musical segments and section labels. The overall system is lightweight and performs competitively with previous methods. The evaluation is done on two standard datasets for music structure analysis and an ablation study is conducted in order to gain insight on the role played by its different components.

1. INTRODUCTION

Music structure analysis consists of locating segments that compose a track and grouping them into semantic categories, referred to as musical sections [1]. Approaches to solve this task have significantly been advanced in the past few years, notably due to the creation of large audio datasets along with their structural annotations [2–4]. These annotated corpora have allowed researchers to leverage recent progress in deep learning and design systems that learn signal representations to predict song structures.

1.1 Related work

One crucial aspect when analyzing musical structures is the strong temporal dependency among different events within a track. A musical observation at a given time can impact other observations at any other point in time, and this, at different scales. This multi-level dependency still poses a significant challenge when training music segmentation systems [1]. Recent methods successfully relied on modelling these temporal connections through the use of self-attention mechanism [5–8]. In these cases, the model is equipped with multiple self-attention layers so as to automatically learn to identify such dependencies. While these proved to be effective, they do not rely on any prior knowledge about musical structure and therefore tend to require large training sets or multiple input audio representations (*e.g.* multiple audio features [8], separated instrument stems [7]) so as to better characterize mutual relationships between time instants in the input track. The method introduced in this work proposes to explicitly model such temporal dependencies by leveraging the natural geometry of a track's self-similarity matrix.

Self-similarity representations have been a useful tool to predict the structure of a track [9–12]. A line of work has for example focused on improving this representation through the use of contrastive learning [13–15]. By extracting better audio features, the resulting self-similarity matrices carry more meaningful patterns that can ease structure prediction performed by downstream segmentation methods [10, 16]. In the proposed approach, the self-similarity representation is not used as direct input to a segmentation system but rather to extract structural link features between time frames within the input track. This step is jointly performed with the audio feature extraction stage, the prediction of segment boundaries and section labels, allowing each task to benefit from the others.

1.2 Contributions

In this work, a supervised approach to segmentation of western popular music is proposed that effectively combines the three music structure principles which have been identified in previous studies [1]: *homogeneity*, *repetition* and *regularity*. To this end, the segmentation task is formulated as a graph partitioning problem where links (*i.e.* edges) between musical audio observations taken at any two time instants (*i.e.* nodes) are first characterized as whether connecting elements from the same segment, section or distinct structural entities (different segment or sec-

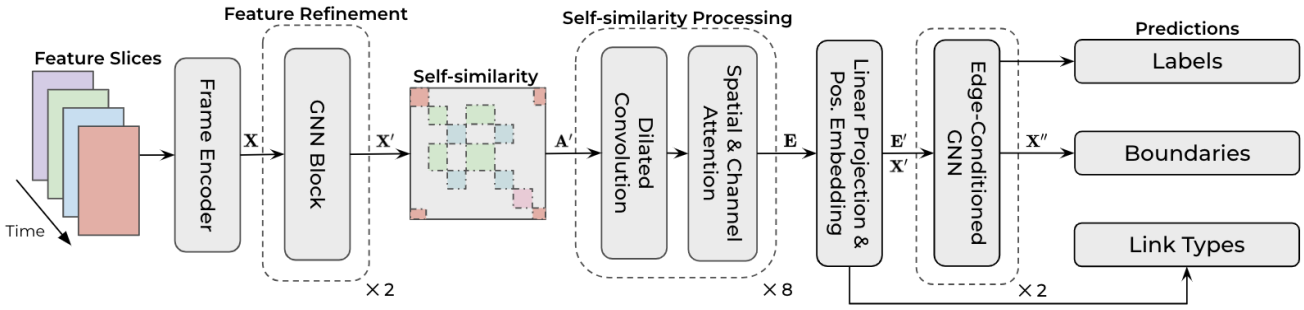

Figure 1. Model overview. Input feature patches are first processed through a frame-level encoder and a first GNN block. A self-similarity of the output features is passed through a 2-dimensional dilated convolutional network to extract link features. Node and link features are finally combined with graph attention network layers to predict boundary locations and section label assignments.

tion). These extracted link features condition a subsequent analysis block based on attention graph neural networks to further refine node features. The system outputs final predictions composed of boundary locations and frame-wise section label likelihoods. Overall, the main contributions of this work are the following: (1) we demonstrate that music segmentation can be modelled as a pairwise link prediction task, which offers a flexible framework that is inspired by well-identified structure principles ; (2) we use graph attention networks to allow frames in the track to dynamically exchange information between each other and we successfully inform this process by the learned link features ; (3) we demonstrate in an ablation study that link features provide some useful structural information about the input track, which significantly improves segmentation performance.

2. METHOD

2.1 Overall approach

The segmentation method proposed in this work proceeds in three main steps, depicted in Figure 1. First, the input track is passed through a frame encoder to obtain a sequence of frame-wise feature vectors. These are further smoothed by a graph neural network (GNN) block, allowing each individual frame to aggregate and combine information from all other time instants in the track. A self-similarity matrix is calculated from these features and fed as input to a 2-dimensional convolutional neural network. A spatial learnable bias is added to the output feature map to inform about each component's source and destination frames' relative positions. The link features, along with the smoothed frame features, are effectively combined through an edge-conditioned graph attention module. The updated frame features finally serve to predict segment boundaries and section labels.

2.2 Audio representation

2.2.1 Input features

For a given track, we start by estimating probable beat positions using an off-the-shelf beat tracking algorithm so as to reduce the length of the feature sequence to be analyzed. Following previous work [14, 15, 17], the input signal is then converted into a log-scaled Mel-spectrogram representation, from which slices centered around each detected beat position are extracted.

2.2.2 Frame encoder

The sequence of mel-spectrogram slices is passed through an encoder to obtain a sequence of feature vectors $\mathbf{X} \in \mathbb{R}^{N \times d}$ where N is the number of detected beats (*i.e.* slices) and d is the embedding dimension. The objective of this step is to extract relevant spectro-temporal information from each slice. The architecture of the encoder is inspired from the work by Won *et al.* [18] for music tagging. It consists of three convolutional blocks to extract low-level features, followed by two transformer encoder layers which temporally summarize the content of each slice. To obtain more robust audio representations, the pre-training strategy proposed by Buisson *et al.* [15] is followed. It uses a contrastive loss to learn an embedding space in which frames from repeating sequences over the whole track are close. In this work, the self-supervised pre-training stage is performed on $20,000$ unlabelled tracks, covering various music genres such as rock, popular, rap, jazz, electronic or classical.

2.3 Feature refinement

The sequence of feature vectors $\mathbf{X}$ is processed by a first GNN block. The objective is to further refine local discontinuities by allowing each frame to exchange information with all other frames in the track. To this end, a self-similarity matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ is calculated from $\mathbf{X}$ such that its elements $A(i,j)$ are defined as:

$$A(i,j) = \exp\left(-\gamma \left\| \frac{x_i}{\|x_i\|_2} - \frac{x_j}{\|x_j\|_2} \right\|_2^2\right), \qquad (1)$$

where the bandwidth parameter γ is simply set as $\gamma = \frac{1}{2s}$, with $s = \text{std}\left(\left\| \frac{x_i}{\|x_i\|_2} - \frac{x_j}{\|x_j\|_2} \right\|_2^2\right)$ and $\|.\|_2$ denotes the ℓ-2 norm. The matrix $\mathbf{A}$ can be regarded as the weighted adjacency matrix of a complete graph $G = (V, E)$, where

the set of nodes V corresponds to each frame contained in the track, and its edges E represent the strength of their mutual connections (*i.e.* similarity). However, each feature slice was transformed independently by the frame encoder (see Section 2.2.2). To improve both segment homogeneity and discriminability, two graph convolution layers [19] are applied to smooth the node features $\mathbf{X}$, of which the update rule for an arbitrary layer l is expressed as:

$$x_i^{(l+1)} = \sigma\left(\sum_{1\leq j\leq N} \frac{A(j,i)}{N} x_j^{(l)} \mathbf{W}^{(l)} + b^{(l)}\right), \quad (2)$$

where $\mathbf{W}^{(l)}$ and $b^{(l)}$ are learnable weight parameters and σ is an activation function: Exponential Linear Unit (ELU) in this work. Equation (2.3) shows that each frame in the sequence $\mathbf{X}$ receives a weighted combination of all other frames in the track and is then linearly transformed before applying a non-linear activation. A common issue encountered with graph neural networks is the over-smoothing phenomenon [20], where all points end up having the same representation after passing through several layers. To limit this effect, the output features are further processed by a multi-layer perceptron (MLP) [21], yielding the refined node features $\mathbf{X}' \in \mathbb{R}^{N\times d}$.

2.4 Link feature extraction

2.4.1 Motivations

Recognizing the structure of a song can be achieved by learning to fully characterize the mutual relationships between time frames from its beginning to its end. In other words, if for any pair of time points (*i.e.* audio frames), one can successfully predict if they belong to the same musical segment or section, then the overall structure of the song can be easily recovered (*e.g.* through a simple graph traversal). Figure 2 shows a visual representation of this link prediction task.

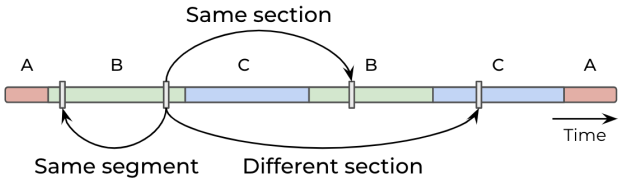

Figure 2. Schematic representation of the link characterization task. For each time instant, the goal is to classify its mutual relationship with all other instants in the input track as either, from the same segment, section or a different section.

It is interesting to notice that each of the structure principles somehow translates into specific characteristics of the self-similarity matrix. The *homogeneity* of musical segments can be observed through the appearance of block-like structures on the main diagonal. Similarly, *repetitions* of sequences can be spotted by diagonal stripes whereas repeating homogeneous segments will appear as off-diagonal blocks. The notion of *regularity* is visible as the relative size of these patterns, which tends to be consistent within a track and in specific genres such as western popular music. Therefore, the self-similarity representation of a track yields crucial information on its structural organization and can be exploited to extract link-related information. Additionally, it provides an efficient information bottleneck which can improve generalization across different songs.

2.4.2 Self-similarity processing

The refined features $\mathbf{X}' \in \mathbb{R}^{N\times d}$ returned by the first GNN block are used to build a self-similarity matrix $\mathbf{A}' \in \mathbb{R}^{N\times N}$, in the same fashion as in Section 2.3. The goal of the link feature extraction step is to classify each component of the input matrix $\mathbf{A}'$ into three categories: "same-segment", "same-section" or "different section" links (see Figure 2). To this end, a 2-dimensional convolutional neural network is used, which is composed of blocs as shown in Figure 3. The kernels' dilation rate is increased expo-

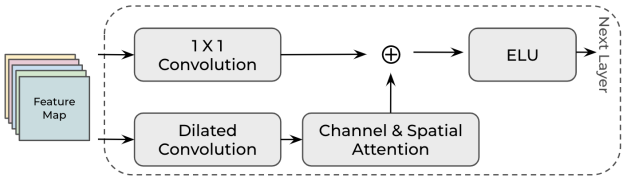

Figure 3. CNN block containing dilated convolution, channel & spatial attention and a residual connection. Each layer contains $C = 16$ channels and has an exponentially increasing dilation rate ranging from 2^0 to 2^7 to efficiently combine structural features at different scales.

nentially at each layer to enlarge the receptive field of the network and capture structural patterns at different scales. Because the goal is to classify each pixel (*i.e.* link), no pooling is applied in-between layers. To further enhance the intermediate feature maps, each convolution is followed by a multi-scale attention module [22] that leverages both spatial and channel interactions. A residual connection is added to the output of each attention block before applying ELU activation. We denote the output feature map as $\mathbf{E} \in \mathbb{R}^{C\times N\times N}$, with C being the number of convolution channels.

2.4.3 Positional embedding

If two frames are similar, then determining whether the pair is "same-segment" or "same-section" may be difficult without information about their position in the piece. To address this ambiguity, a learnable relative positional embedding $\mathbf{B} \in \mathbb{R}^{C\times N}$ is added to $\mathbf{E}$. Formulated as a function of $|i-j|$, it aims to provide each element in $\mathbf{E}$ information on its relative distance to the main diagonal. We opt for a simple strategy which consists, for a given link between nodes i and j, in adding the $(|i-j|)$th vector from a learnable embedding matrix $\mathbf{B}$ to $e_{i,j}$. After doing so for every possible links, the result of this addition denoted as $\mathbf{E}'$, is fed to a linear layer with softmax activation. The link feature extraction network is optimized

using a cross-entropy loss function $\mathcal{L}_{\text{Link}}$ between the link-wise predictions $\hat{y}_{\text{link}}$ and the ground-truth y_{link} obtained from structural annotations.

2.5 Edge-conditioned graph attention sub-network

The refined node features $\mathbf{X}'$ and the edge features $\mathbf{E}'$ provide a detailed representation of the input track. The former contains relevant acoustic information, which has been exchanged between frames for better discriminability across musical segments, while the latter provides information about their pairwise links. To efficiently combine these complementary views of the graph, we propose the use of edge-conditioned graph attention networks [23]. Node features are further improved by aggregating information from all other nodes in the graph, weighted by some learnable attention coefficients. These attention coefficients depend on each node's features and the edge features that link them. For a given node x'_i, the update rule is defined as:

$$x''_i = \mathbf{W}_{\text{s}} \cdot x'_i + \sum_{j \in \mathcal{N}(x'_i)} \alpha_{j,i} \left(\mathbf{W}_{\text{n}} \cdot x'_j + \mathbf{W}_{\text{e}} \cdot e'_{j,i} \right), \quad (3)$$

where $\mathbf{W}$ is used to denote learnable weight matrices for the transformation the node features to update (s="self"), neighboring nodes (n="neighbor") and edge features (e="edge"). The attention coefficients $\alpha_{j,i}$ are obtained as follows:

$$\alpha_{j,i} = \text{softmax}_i \Big(\sigma \big(a^T [\mathbf{W}_{\text{n}} \cdot x'_i || \mathbf{W}_{\text{n}} \cdot x'_j || \mathbf{W}_{\text{e}} \cdot e'_{j,i}] \big) \Big), \quad (4)$$

with a corresponding to a learnable vector, σ to a LeakyRelu activation, $||$ denotes the concatenation operation and $e'_{j,i}$ is the refined link features going from node j to node i. The softmax_i operation normalizes all incoming edges of node i. The forward-pass formulation from Equation (3) closely resembles that of the transformer, but additionally introduces edge features to calculate attention maps and output node features. We use a series of two graph attention layers with residual connections and ELU activation in-between. Both layers use 8 attention heads, the outputs of all heads are concatenated after the first layer and averaged after the second. The output node features are denoted as $\mathbf{X}'' \in \mathbb{R}^{N \times d}$.

2.6 Boundary and label predictions

The output of the overall system consists in boundary locations, expressed in beat indices, along with frame-wise section-label likelihoods. For boundary prediction, consecutive node features x''_i and x''_{i+1} are first concatenated, along with the corresponding link features $e'_{i,i+1}$ between them. The result of this concatenation is transformed through a linear layer with sigmoid activation to output the probability $\hat{y}_{\text{bound}}$ of a segment boundary between these frames. For section-label predictions, we simply feed each frame to a linear layer with softmax activation, resulting in a predicted class assignment matrix $\mathbf{S} \in [0,1]^{N \times K}$, where K corresponds to the number of section labels. We derive a boundary curve by concatenating the boundary predictions $\hat{y}_{\text{bound}}$ over time. To obtain the final boundary locations, we use the peak picking method after Ullrich *et al.* [24] without any thresholding on the RWC-Pop dataset, and the one from Kim *et al.* [7] for Harmonix. For the section label assignment, a simple majority vote is applied within each detected segment to determine its structural label. Due to the imbalance between boundary and non-boundary points, we use a dice loss $\mathcal{L}_{\text{Bound}}$ to optimize the boundary predictions, as it has proven useful in many segmentation tasks before [25]. We use a cross-entropy loss $\mathcal{L}_{\text{Label}}$ for section label predictions.

2.7 MinCut regularization

The objective of the proposed segmentation system is to assign each frame of the input track to one of K possible section labels. Ideally, we want this assignment to be equal for nodes in the graph that are either in the same segment or section, and orthogonal in the remaining cases. From the perspective of graph theory, given the input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, this problem comes down to partitioning the set of nodes $\mathcal{V}$ into K disjoint subsets by removing a minimum volume of edges, which is equivalent to maximizing:

$$\frac{1}{K} \sum_{k=1}^{K} \frac{\text{links}(\mathcal{V}_k)}{\text{degree}(\mathcal{V}_k)} = \frac{1}{K} \sum_{k=1}^{K} \frac{\sum_{i,j \in \mathcal{V}_k} \mathcal{E}_{i,j}}{\sum_{i \in \mathcal{V}_k, j \in \mathcal{V} \setminus \mathcal{V}_k} \mathcal{E}_{i,j}}, \quad (5)$$

where the numerator corresponds to the volume of edges within each cluster, and the denominator counts the edges between nodes in a cluster and the rest of the graph. This task is referred to as the *K-way normalized MinCut* problem. Spectral clustering provides an optimal solution of this problem by projecting the nodes into the Laplacian eigenspace [10, 26]. However, calculating the spectrum of the Laplacian matrix is a costly operation and the final class assignment relies on non-differentiable operations, thus preventing it from being optimized along with the rest of the network.

In order to learn a model that finds an approximate spectral clustering solution in a differentiable manner, we base ourselves on the work by Bianchi *et al.* [27]. They propose a continuous relaxation of the normalized MinCut problem, where a GNN is trained to compute a cluster assignment matrix $\boldsymbol{\mathcal{S}} \in [0,1]^{N \times K}$ by optimizing the objective defined as:

$$\mathcal{L}_{\text{MinCut}} = -\frac{\text{Tr}(\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{A}} \boldsymbol{\mathcal{S}})}{\text{Tr}(\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{D}} \boldsymbol{\mathcal{S}})} + \left\| \frac{\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{S}}}{\|\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{S}}\|_F} - \frac{\mathbf{I}_K}{\sqrt{K}} \right\|_F, \quad (6)$$

where $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N \times N}$ is the graph adjacency matrix, $\boldsymbol{\mathcal{D}}$ is the degree matrix of $\boldsymbol{\mathcal{A}}$, K is the number of classes and $\|.\|_F$ corresponds to the Frobenius norm. The left-hand-side term encourages connected nodes to be clustered together. It reaches its minimum when $\text{Tr}(\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{A}} \boldsymbol{\mathcal{S}}) = \text{Tr}(\boldsymbol{\mathcal{S}}^T \boldsymbol{\mathcal{D}} \boldsymbol{\mathcal{S}})$, meaning that the cluster assignments are equal for all the nodes in the same class and orthogonal to the cluster assignments of nodes from different classes. To avoid degenerate minima (uniform cluster assignments or all nodes being assigned to the same cluster), the right-hand-side term encourages the cluster assignments to be orthogonal

and the clusters to be of similar size. While in practice, it is not always desirable to have a perfectly balanced cluster assignment for music segmentation (due to the variable sizes of musical sections), the loss term $\mathcal{L}_{\text{MinCut}}$ acts as an effective regularizer that helps making the cluster assignment sharper. During training, we use the label agreement matrix $\mathbf{Y}$ of each track as adjacency matrix and the predicted label assignment matrix $\mathbf{S}$ defined in Section 2.6 as $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{S}}$ in Equation (2.7) respectively. The whole system is trained end-to-end in a multi-task fashion, so as to minimize the overall loss function $\mathcal{L}_{\text{total}}$ defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{Bound}} + \mathcal{L}_{\text{Label}} + \mathcal{L}_{\text{Link}} + \mathcal{L}_{\text{MinCut}}. \quad (7)$$

3. EXPERIMENTAL SETTING

3.1 Datasets

The proposed method is assessed on two standard datasets for music structure analysis. To reduce the number of possible section labels, we apply the annotation preprocessing step proposed in the work by Wang *et al.* [5]. We end up with a total of 7 unique section labels for both of the following datasets:

RWC-Pop: the Popular subset of the RWC dataset [28] contains 100 songs with section annotations. The original ones provided by the authors (AIST) are used.

Harmonix: the Harmonix dataset [3] is composed of 912 annotated tracks covering various genres of western popular music such as pop, electronic, hip-hop, rock, country and metal. The audio files were retrieved from YOUTUBE and structural annotations were manually adjusted.

3.2 Evaluation metrics

Common evaluation metrics for automatic structure analysis are employed throughout our experiments. For boundary detection, we report the F-measure [1] of the trimmed [2] boundary detection hit-rate with a 0.5 and 3-second tolerance windows (HR.5F, HR3F respectively). For structural grouping, we report the F-measure of pairwise-frame clustering [30] (PFC) and the F-measure of the normalized conditional entropy (NCE). We additionally measure the weighted label prediction accuracy (Acc), which indicates how well the model predicts frame-wise section labels.

3.3 Implementation details

All tracks are resampled at 22.05 kHz. As input to the frame encoder, we use log-scaled Mel-spectrograms with a window and hop size of 1024 and 256 samples respectively. We compute 64 Mel-bands per frame. The *TorchAudio* library is used for feature extraction [31]. As in previous work [15, 32], beats are estimated for all tracks using the algorithm from Korzeniowski *et al.* [33] implemented in the *madmom* package [34]. Slices of 64 frames ($\simeq 0.75$s) centered at each detected beat location are fed as input to the frame encoder. The frame embedding dimension is set to $d = 32$ and kept fixed throughout the whole system. The number of channels in the link-feature extractor is set to $C = 16$, convolutions use kernels of size $k = 5$. All GNN layers are implemented using the *Deep Graph Library* [35] package. The whole model, including the pre-trained frame encoder, contains less than 330K parameters and is implemented [3] with Pytorch 2.0 [36].

3.4 Experiments

In order to study the impact of each part of our method, we perform an 8-fold cross-validation ablation study on the Harmonix dataset. At each episode, one element from the system is removed: the pre-training stage (Section 2.2.2), the feature smoothing step (Section 2.3), the link features extraction (Section 2.4.2), the positional embedding (Section 2.4.3) and the MinCut regularization (Section 2.7). We use 6 splits for training, one for validation and the remaining one for testing. Then, we perform a cross-dataset evaluation, where one dataset is used for training (split beforehand into training and validation sets) and the other one for testing.

4. RESULTS

4.1 Ablation study

Results from the ablation study given in Figure 4 show the performance of the system when some of its components are discarded during training and inference. The different metrics are averaged over the 8 splits. In the first scenario, the frame encoder is randomly initialized like the rest of the model. We observe a significant decrease on all metrics, showing that the pre-training stage provides robust initial frame representations which are further tuned by the network during training. It is interesting to notice however that without pre-training the frame encoder, the model still predicts a good label assignment matrix, both in terms of pairwise frame clustering and label accuracy. We assume that the impact of this step is rather limited due to the relatively large size of the Harmonix dataset, which provides enough training examples to still learn useful frame features. In the second case, the MinCut regularization is discarded, which negatively impacts all metrics. This tends to confirm that the MinCut regularization enforces sharper cluster assignments, especially around segment boundaries where these can be more evenly distributed.

The most significant variation in segmentation performance is observed when the link feature extraction step is omitted. In this case, pairwise links between nodes are only characterized by their positional embedding and do not contain any structural information. This observation strongly suggests that the model benefits from both perspectives (node and link features) of the input track. When positional embeddings are removed, the link loss $\mathcal{L}_{\text{Link}}$ stops decreasing after several training iterations. Notably

[1] All evaluations are done using the mir_eval package [29].

[2] The first and last boundaries are discarded during evaluation, as they correspond to the beginning and the end of the track and therefore, do not provide any information regarding the system's performance.

[3] Code: `github.com/morgan76/LinkSeg`

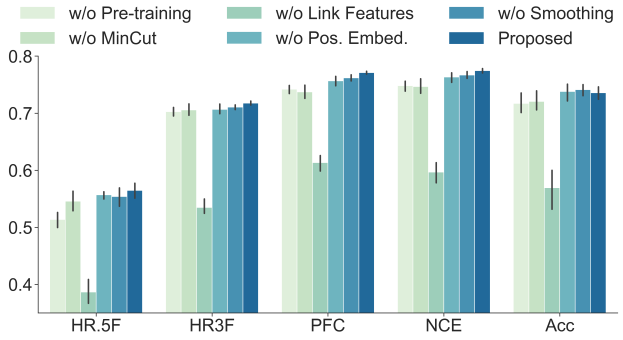

Figure 4. Ablation results on the Harmonix dataset in the cross-validation setting. Metrics are averaged across splits and standard deviation denoted with dark grey vertical bars.

because the network fails to differentiate "same-segment" from "same-section" links, which provides useful structural information near segment boundaries. In the case when the features smoothing step is removed, performance on all metrics, except the label prediction accuracy, is negatively impacted.

4.2 Comparison with previous work

This section compares the performance of our system against recent work for music structure analysis. The first one from Wang *et al.* [14], which we denote as DSF, uses supervised metric learning and spectral clustering [10] for boundary and section label predictions. SpecTNT [5] is based on a spectrogram transformer architecture and directly outputs both a boundary probability curve along with a frame-wise section label assignment. All in One [7] uses demixed spectrograms and several layers of neighborhood attention, operating simultaneously at the instrument and the temporal levels. These three baselines were trained and evaluated on the Harmonix dataset in a cross-validation setting. CBM, for Convolutive Block Matching [37] relies on dynamic programming to find the segmentation that minimizes a cost function. Its parameters were set by cross-validation on the RWC-Pop dataset.

Results on Harmonix and RWC-Pop are given in Table 1. In the cross-validation setting, the proposed method performs worse for boundary detection than the reported baselines. On RWC-Pop, despite being trained on a very small number of tracks (75 at each episode), the model still manages to pick up transitions between structural elements, even so at a high temporal resolution (± 0.5 second). This is to be compared with the first two baselines, namely DSF [14] and SpecTNT [5] which used the whole Harmonix dataset for training, along with additional datasets for the latter. The CBM algorithm [37] shows the strongest performance in this setting, as it explicitly favors musical segments of pre-defined length (which is around 8 bars in most cases for RWC-Pop), whereas our method does not make any assumption on the distribution of section lengths. It is also important to note that our system operates on a rather coarse time resolution (beat level) and only requires a gross discretization of the input track's timeline to function. We argue that better performance could be achieved by providing a more fine-grained time division (tatum level for example) but at a higher computational cost.

	HR.5F	HR3F	PFC	NCE	Acc
Harmonix					
DSF [14]	.497	**.738**	.689	.743	–
SpecTNT$_{24s}$ [5]	.570	–	.700	.714	.701
SpecTNT$_{36s}$ [5]	.558	–	.712	.724	.723
All in One [7]	**.660**	–	.738	.769	–
Cross-val.	.568	.717	**.771**	**.772**	**.742**
Cross-dataset	.462	.664	.660	.671	.530
RWC-Pop					
DSF [14]	.438	.653	.704	.739	–
SpecTNT$_{24s}$ [5]	.623	–	.749	.728	.675
CBM [37]	.644	**.806**	–	–	–
Cross-val.	.585	.750	.785	.802	**.813**
Cross-dataset	**.648**	.786	**.812**	**.812**	.747

Table 1. Boundary detection and structural grouping results on the Harmonix and RWC-Pop datasets. *Cross-val* indicates results that were obtained through cross-validation, averaged across splits. *Cross-dataset* refers to the results obtained when the model is trained on one and tested on the other.

In terms of structural grouping, the method proposed in this work outperforms all baselines on both datasets in most settings. Even though the label prediction method employed is rather simple and directly dependent on the boundary detection results, the model successfully learns to group frames across repetitions of identical musical sections. Finally, the high section label prediction accuracies obtained show that the network not only manages to successfully group frames together, but also predicts the right section label in a vast majority of cases.

Finally, cross-dataset results from RWC-Pop to Harmonix (*Cross-dataset* row) show that the model still generalizes to some extent, despite the very small quantity data used for training. On the other hand, training the model on Harmonix and testing it on RWC-Pop leads to strong performance both in terms of boundary detection and structural grouping, indicating that the network's generalization capacity increases as more annotated data is available for training.

5. CONCLUSION

This work proposes a new approach to music segmentation by learning to characterize pairwise relationships between time instants in a musical recording. The structural view of the input track obtained from this auxiliary task can be combined with local frame information to effectively predict boundary locations between musical segments and section labels. Future research includes the extension of the link prediction task to various levels of segmentation and arbitrary labels semantic.

6. REFERENCES

[1] O. Nieto, G. J. Mysore, C.-i. Wang, J. B. Smith, J. Schlüter, T. Grill, and B. McFee, "Audio-based music structure analysis: Current trends, open challenges, and applications." *Transactions of the International Society for Music Information Retrieval (TISMIR)*, vol. 3, no. 1, pp. 246–263, 2020.

[2] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. De Roure, and J. S. Downie, "Design and creation of a large-scale database of structural annotations." in *ISMIR*, 2011.

[3] O. Nieto, M. C. McCallum, M. E. Davies, A. Robertson, A. M. Stark, and E. Egozy, "The harmonix set: Beats, downbeats, and functional segment annotations of western popular music." in *ISMIR*, 2019.

[4] S. Balke, J. Reck, C. WEIS, J. ABESER, and M. Müller, "Jsd: A dataset for structure analysis in jazz music," *Transactions of the International Society for Music Information Retrieval (TISMIR)*, vol. 5, p. 1, 2022.

[5] J.-C. Wang, Y.-N. Hung, and J. B. Smith, "To catch a chorus, verse, intro, or anything else: Analyzing a song with structural functions," in *ICASSP*, 2022.

[6] G. Peeters, "Self-similarity-based and novelty-based loss for music structure analysis," in *ISMIR*, 2023.

[7] T. Kim and J. Nam, "All-in-one metrical and functional structure analysis with neighborhood attentions on demixed audio," in *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2023, pp. 1–5.

[8] T.-P. Chen, L. Su, and K. Yoshii, "Learning multi-faceted self-similarity for musical structure analysis," in *2023 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2023, pp. 165–172.

[9] O. Nieto and T. Jehan, "Convex non-negative matrix factorization for automatic music structure identification," in *ICASSP*, 2013.

[10] B. McFee and D. Ellis, "Analyzing song structure with spectral clustering." in *ISMIR*, 2014.

[11] T. Grill and J. Schlüter, "Music boundary detection using neural networks on combined features and two-level annotations." in *ISMIR*, 2015.

[12] G. Peeters, A. La Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis," in *ISMIR*, 2002.

[13] M. C. McCallum, "Unsupervised learning of deep features for music segmentation," in *ICASSP*, 2019.

[14] J.-C. Wang, J. B. Smith, J. Chen, X. Song, and Y. Wang, "Supervised chorus detection for popular music using convolutional neural network and multi-task learning," in *ICASSP*, 2021.

[15] M. Buisson, B. McFee, S. Essid, and H.-C. Crayencour, "A repetition-based triplet mining approach for music segmentation," in *ISMIR*, 2023.

[16] J. T. Foote and M. L. Cooper, "Media segmentation using self-similarity decomposition," in *Storage and Retrieval for Media Databases 2003*, vol. 5021. SPIE, 2003, pp. 167–175.

[17] M. Buisson, B. McFee, S. Essid, and H.-C. Crayencour, "Learning multi-level representations for hierarchical music structure analysis," in *ISMIR*, 2022.

[18] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," in *ISMIR*, 2021.

[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[20] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[21] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, "Vision gnn: An image is worth graph of nodes," *Advances in neural information processing systems*, vol. 35, pp. 8291–8303, 2022.

[22] D. Ouyang, S. He, G. Zhang, M. Luo, H. Guo, J. Zhan, and Z. Huang, "Efficient multi-scale attention module with cross-spatial learning," in *ICASSP*, 2023.

[23] T. Monninger, J. Schmidt, J. Rupprecht, D. Raba, J. Jordan, D. Frank, S. Staab, and K. Dietmayer, "Scene: Reasoning about traffic scenes using heterogeneous graph neural networks," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1531–1538, 2023.

[24] K. Ullrich, J. Schlüter, and T. Grill, "Boundary detection in music structure analysis using convolutional neural networks." in *ISMIR*, 2014.

[25] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)*. IEEE, 2020, pp. 1–7.

[26] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[27] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *International conference on machine learning*. PMLR, 2020, pp. 874–883.

[28] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Popular, classical and jazz music databases." in *ISMIR*, 2002.

[29] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "mir_eval: A transparent implementation of common mir metrics," in *ISMIR*, 2014.

[30] M. Levy and M. Sandler, "Structural segmentation of musical audio by constrained clustering," *IEEE transactions on audio, speech, and language processing*, 2008.

[31] Y.-Y. Yang, M. Hira, Z. Ni, A. Chourdia, A. Astafurov, C. Chen, C.-F. Yeh, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Mahadeokar, J. Hwang, J. Chen, P. Goldsborough, P. Roy, S. Narenthiran, S. Watanabe, S. Chintala, V. Quenneville-Bélair, and Y. Shi, "Torchaudio: Building blocks for audio and speech processing," *arXiv preprint arXiv:2110.15018*, 2021.

[32] J. Salamon, O. Nieto, and N. J. Bryan, "Deep embeddings and section fusion improve music segmentation," in *ISMIR*, 2021.

[33] F. Korzeniowski, S. Böck, and G. Widmer, "Probabilistic extraction of beat positions from a beat activation function." in *ISMIR*, 2014.

[34] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "Madmom: A new python audio and music signal processing library," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016.

[35] M. Y. Wang, "Deep graph library: Towards efficient and scalable deep learning on graphs," in *ICLR workshop on representation learning on graphs and manifolds*, 2019.

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, 2019.

[37] A. Marmoret, J. E. Cohen, and F. Bimbot, "Barwise music structure analysis with the correlation block-matching segmentation algorithm," *Transactions of the International Society for Music Information Retrieval (TISMIR)*, vol. 6, no. 1, pp. 167–185, 2023.