Self-Supervised Learning of Multi-level Audio Representations for Music Segmentation

Morgan Buisson, Brian Mcfee, Slim Essid, Hélène Crayencour

Self-Supervised Learning of Multi-level Audio Representations for Music Segmentation

Morgan Buisson, Brian McFee, Slim Essid, Hélène C. Crayencour

***Abstract*—The task of music structure analysis refers to automatically identifying the location and the nature of musical sections within a song. In the supervised scenario, structural annotations generally result from exhaustive data collection processes, which represents one of the main challenges of this task. Moreover, both the subjectivity of music structure and the hierarchical characteristics it exhibits make the obtained structural annotations not fully reliable, in the sense that they do not convey a "universal ground-truth" unlike other tasks in music information retrieval. On the other hand, the quickly growing quantity of available music data has enabled weakly supervised and self-supervised approaches to achieve impressive results on a wide range of music-related problems. In this work, a self-supervised learning method is proposed to learn robust multi-level music representations prior to structural segmentation using contrastive learning. To this end, sets of frames sampled at different levels of detail are used to train a deep neural network in a disentangled manner. The proposed method is evaluated on both flat and multi-level segmentation. We show that each distinct sub-region of the output embeddings can efficiently account for structural similarity at their own targeted level of detail, which ultimately improves performance of downstream flat and multi-level segmentation. Finally, complementary experiments are carried out to study how the obtained representations can be further adapted to specific datasets using a supervised fine-tuning objective in order to facilitate structure retrieval in domains where human annotations remain scarce.**

***Index Terms*—Music structure analysis, structural segmentation, representation learning.**

I. Introduction

STRUCTURE plays a central role in music and the way it is perceived by human listeners. However, automatically discovering the structure of a music piece remains a challenging task in the field of music information retrieval. The highly complex and multi-dimensional organization of music lies in the numerous interactions occurring between its different composing elements at various temporal scales. The problem of audio-based music structure analysis consists in predicting, for an audio track, the location between different musical segments. Then, these same segments are attributed labels based on some similarity criteria [1].

A major challenge in the task of automatic music structure analysis is related to the evaluation process. Because structural annotations are provided by human listeners, potential disagreements among them about how a particular music piece is organized [2, 3] might appear. Various musical characteristics such as rhythm, harmony or melody can influence the perception of structure, and this, at different points in time. Meanwhile, it has been observed that human listeners tend to isolate some of these features from the audio while providing structural annotations, thus resulting in different ground-truth for the same music piece [4]. Because of this inherent ambiguity in the annotation process, the task of audio-based music structure analysis remains an ill-posed problem.

Musical structures also exhibit a hierarchical organization, as short-term musical events can be recursively integrated into bigger segments, where the common level of segmentation attended by human listeners, often called the *functional* level, contains segments that carry a certain role within the music piece (intro, chorus, verse,...). Most segmentation algorithms explicitly target this very level of detail, as it is also the most represented in existing datasets for music structure analysis. However, a considerable amount of information might be lost by discarding more refined levels of structure such as phrases, bars or beats. Additionally, practical applications such as intra-track navigation [5], music similarity [6], automatic music composition [7] or cover detection [8] could benefit from these various levels of detail at which a track can be decomposed. The multi-level aspect of structure also greatly impacts the final evaluation of segmentation algorithms. Using annotations proceeding from different data sources might become unreliable, in cases where predictions and annotations do not lie at the same level of detail for a given music piece. Recent work has shed light on this problem [1] and new evaluation metrics have been proposed to limit this temporal mismatch between predictions and reference annotations [9]. However, very few works have so far directly addressed the problem of multi-level music segmentation [10, 11, 12], leaving plenty of room for new methods to decompose music pieces at different levels of granularity.

This work proposes a self-supervised contrastive learning-based approach to build robust audio representations prior to structural segmentation. A deep neural network is trained on audio feature patches to learn an embedding space in which frames belonging to the same musical section are close to one-another, thus facilitating the structure discovery performed by any segmentation algorithm applied on top of the output representations. The proposed method aims at encompassing the multi-level organization of structure by building disentangled representations, where each sub-region of the obtained embeddings targets its own specific level of detail, thus providing various views of the same piece, differing by the temporal scales at which they operate. Aligned with our previous findings [13], we show that such an approach can be beneficial both for single-level and multi-level music segmentation tasks, this time using only two representation levels. The evaluation is performed on three substantial datasets for music structure with different genres and annotation styles.

We additionally study a simple method to further optimized the learned representations on a small portion of annotated data. Results suggest that features learned at the pre-training stage are rather general and can be easily biased even on homogeneous music genres and annotation styles during fine-tuning. Thus, highlighting the advantage of relying on self-supervised methods in the first place.

The remainder of this paper is organized as follows. Section II discusses relevant previous work in music structure analysis. More particularly, a focus is put on the multi-level version of the task and existing methods for learning audio representations prior to music segmentation. Section III describes the proposed approach and discusses its technical aspects. Section IV introduces the experimental setting in which the proposed method is assessed, Section V compiles and analyses the results obtained and Section VI details the additional fine-tuning experiment.

II. Related work

Up to recently, numerous music structure analysis methods would generally follow a three-step process. First, features are extracted from the input track, then, a boundary detection algorithm is applied to retrieve transitions between musical segments. Finally, these segments are grouped together given certain similarity criteria to obtain musical sections. The method proposed in this work aims at improving the very first stage of this pipeline, that is, the feature extraction stage. This section gives an overview of previous work addressing music structure analysis. Specifically, an emphasis is put on the feature extraction step of the task, along with its multi-level reformulation.

A. Multi-level segmentation

A single-level or *flat* structural analysis comprises a set of non-overlapping time intervals, each assigned a specific label. The concatenation of these segments covers the entire audio track. While music simultaneously operates at different timescales (bars, beats or onsets), very few approaches have explored the task of jointly segmenting music at these different temporal levels. The goal of multi-level segmentation is to predict a set of segmentations, grouped into a structure called hierarchy where the very first segmentation spans the entire duration of the audio track and the subsequent ones provide an increasing amount of detail (see the illustration in Figure 1). McFee and Ellis [10] regard the problem of music structure with graphs and apply spectral clustering to decompose an enhanced self-similarity matrix employing a combination of hand-crafted features. Besides requiring no annotations, this method also produces segmentations at different temporal levels by considering or discarding principal components of the Laplacian matrix. To better approximate the optimal graph representing the song structure, Tralie and McFee [11] combine different features, including harmonic embeddings, using similarity network fusion before spectral clustering. Salamon et al. [12] further extend this method by integrating two types of deep embeddings along with Constant Q transform (CQT) features. They capture local timbral patterns with few shot-learning and long-term similarities with disentangled deep metric learning [14]. The method proposed here shares the same objective of improving multi-level segmentation by enriching the input audio representation. However, the one proposed here is directly optimized so as to improve downstream multi-level segmentation.

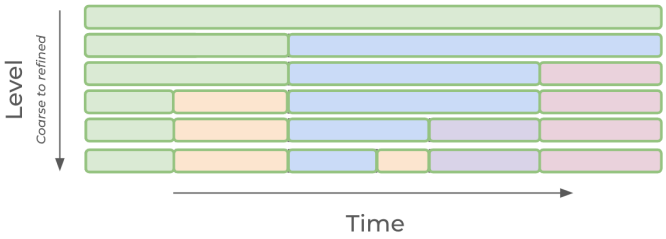

Fig. 1: Schematic view of a multi-level segmentation. Colours represent frame cluster assignment (level-wise) across a 6-level hierarchy.

B. Learning audio representations for music segmentation

The task of structural music segmentation is usually posed as a distance-based problem, where numerous methods rely on the analysis of self-similarity matrices to retrieve homogeneous regions inside a music track. Therefore, coming up with representations simultaneously maximizing the inter-section variance and minimizing intra-section variance is appealing to improve downstream segmentation. Contrastive learning has recently appeared as a method that is well-suited to finding such representations, given that the training process directly aims at optimizing such a constraint in the latent space. When structural annotations are available, finding such representations can be modeled as a supervised learning problem, where the selection of relevant sets of frames used for training is done using section labels. Wang et al. [15] adopt such an approach, along with a multi-similarity loss [16]. Despite allowing them to learn effective structure features, this method is only trained and evaluated on relatively homogeneous datasets, mostly comprising pop and rock music tracks.

In the context of self-supervised learning, finding appropriate sets of frames during training requires to use specifically designed heuristics. The work by McCallum [17] proposes such a method to learn deep features using a triplet-based approach. It relies on the assumption that frames temporally close to each other are more likely to belong to the same musical section than those separated by a certain amount of time. The method proposed in this work can be viewed as a multi-level extension of that of McCallum [17], where the representations learned are optimized for various levels of structural segmentation.

III. Method

Our method aims at building audio representations which facilitate the decomposition of a song at different levels. These should provide strong discriminative capabilities for time frames belonging to different musical sections and separated by a large amount of time. Conversely, they should be more

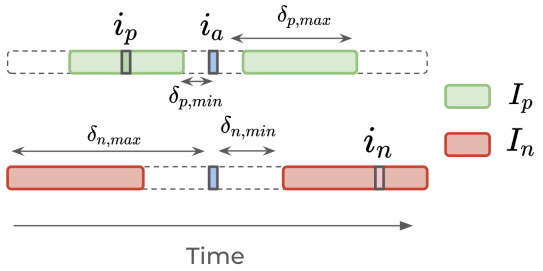

Fig. 2: Initial triplet sampling method at an arbitrary level ℓ of granularity.

homogeneous for frames belonging to the same section and happening within a short time interval. As section lengths might vary from one annotator to another due to the ambiguity of the task [1], the aforementioned constraint is imposed at different temporal scales. Additionally, most datasets for music structure analysis come with only one level of annotations, which motivates us to learn such representations in a self-supervised fashion, taking advantage of large quantities of unlabelled data. A deep neural network is trained to output embeddings which are divided into multiple sub-regions. Each of them is optimized independently using specific sets of frames efficiently sampled to encode the temporal structure of the song at different levels. We propose two different training objectives: the first one is a triplet loss, which directly extends previous work on single-level segmentation [17] to the multi-level case. The second one, similar to the contrastive loss, is a generalization of the former to handle various positive and negative examples for a given point. The input to the network corresponds to audio feature patches which are beat-synchronized: they are centered around the locations of detected beats. This discretization of time drastically reduces the total number of frames while providing relevant prior information about section boundaries (as these usually occur at beat or downbeat positions [1]). From now on, input feature patches are sampled among detected beat positions only. Therefore, a track t corresponds to a sequence of beat indices $\{i_k\}_{k=1}^{N}$ where N is the number of beats in t. We denote by x_k the feature patch centered around beat i_k.

A. Triplet loss

The triplet loss has been used in various music applications of deep metric learning, including music similarity [18, 14], cover song identification [19] or tag-based retrieval [20]. Previous work [17] has shown that the triplet loss can also be used in the context of music structure analysis. It requires beforehand to find triplets of audio feature patches (x_a, x_p, x_n) where x_a is the anchor, x_p is a positive example from the same musical section and x_n the negative example sampled from a different one. For a given triplet $\mathcal{T} = (x_a, x_p, x_n)$, the triplet loss is expressed as:

$$\mathcal{L}(\mathcal{T}) = [d(f_a, f_p) - d(f_a, f_n) + \alpha]_+ \,, \tag{1}$$

where $d(x, y)$ is a pre-defined distance metric (usually the Euclidean distance), $[.]_+$ denotes the Hinge loss, α the margin parameter and f_x is the projection of x into the embedding space by a deep neural network f_θ.

B. Constrastive loss

We propose a further generalization of the approach by McCallum [17] by introducing another training objective. Here, the mutual similarity (i.e. distance) between an anchor and its positive example is not only compared with a single negative example, but a set of negative ones. Formally, for an anchor point i_a, its positive example i_p and a set of negative examples $N(i_a)$, we can write the pairwise loss as:

$$\mathcal{L}^{\text{pair}}(i_a, i_p) = \log \frac{\exp(f_a \cdot f_p/\tau)}{\exp(f_a \cdot f_p/\tau) + \sum\limits_{n \in N(i_a)} \exp(f_a \cdot f_n/\tau)} \tag{2}$$

where $\tau \in [0, 1]$ is a temperature parameter, $\cdot$ denotes the scalar product between ℓ-2 normalized embeddings, and f_a, f_p and f_n are the projections of the anchor, positive and negative input patches in the embedding space by the neural network f_θ. This loss, also known as the N-pair loss [21], is averaged across multiple positive examples $P(i_a)$ of the same anchor, giving the sample-wise loss term defined as:

$$\mathcal{L}^{\text{sample}}(i_a) = \frac{-1}{|P(i_a)|} \sum_{i_p \in P(i_a)} \mathcal{L}^{\text{pair}}(i_a, i_p) \tag{3}$$

The final loss for a given track is finally obtained by summing over the whole set $A(t)$ of anchors selected for track t:

$$\mathcal{L}^{\text{track}} = \frac{1}{|A(t)|} \sum_{i_a \in A(t)} \mathcal{L}^{\text{sample}}(i_a) \tag{4}$$

A possible interpretation of this training objective is, for a given point, to classify its positive example among a set of distractors (negative examples). In our case, this comes down to recognizing the frames that are located within a pre-determined temporal neighborhood around the anchor. Unlike the conventional contrastive loss where positive examples correspond to augmented views of the anchor, positives here are simply given by neighboring frames, thus, we use intra-section variations as an implicit data augmentation process. Additionally in the original contrastive framework, negatives correspond to all remaining data points contained in the batch, potentially containing other samples with the same label as the anchor. In our case, we design a sampling strategy where positive and negatives sets are meant to be mutually exclusive.

C. Single-level sampling

For an arbitrary granularity level of segmentation (a given row in Figure 1), positive and negative sets of beat indices are sampled within respective time intervals I_p and I_n. Intuitively, these intervals rule how "close" or "far away" from the anchor the positive and negative examples will be selected. More specifically, for a given anchor index i_a, positive and negative examples respectively located at beat indices i_p and i_n are uniformly sampled from the interval I_p defined by $\delta_{p,min}$ and $\delta_{p,max}$ and I_n specified by $\delta_{n,min}$ and $\delta_{n,max}$: $i_p \sim \mathcal{U}(I_p)$ and $i_n \sim \mathcal{U}(I_n)$. An example for an arbitrary

level is shown in Figure 2. The original work from McCallum [17] targets only one segmentation level, usually referred to as the "functional" level (or coarse). The author experimentally found the parameters $\delta_{p,min} = 1$, $\delta_{p,max} = 16$, $\delta_{n,min} = 1$ and $\delta_{n,max} = 96$ to yield good results. This method is used as a baseline (denoted as *Flat*) and the original sampling values suggested above are retained in the remainder of this work.

D. Multi-level extension

The sampling method proposed in this work can be viewed as the multi-level extension of that of McCallum [17]. We define a hierarchy H of L representation levels $\ell \in \{0; \dots; L-1\}$ we wish to obtain. For each single level ℓ, sets of beat indices are sampled, this time using level-specific parameters $\delta^\ell = \{\delta^\ell_{p,min}, \delta^\ell_{p,max}, \delta^\ell_{n,min}, \delta^\ell_{n,max}\}$. In order for the learned hierarchy levels to remain consistent with one another, monotonicity is encouraged by further modifying the initial triplet mining technique. In addition to the time constraint imposed on frames of the same level, their probability of being sampled is restricted from one level in the hierarchy to the next. A more detailed description of the sampling process is given below, according to the loss function employed.

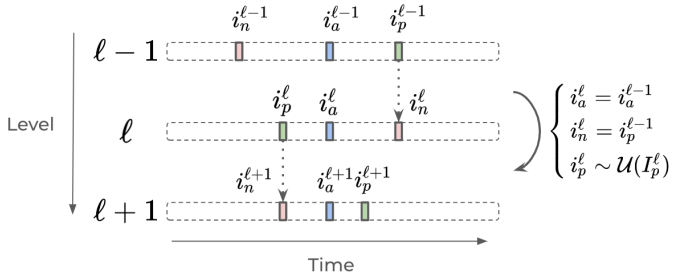

Fig. 3: Modified triplet sampling, moving downwards in the hierarchy.

1) Multi-level triplet loss: For a randomly sampled anchor index at level $\ell = 0$, a complete triplet (i^0_a, i^0_p, i^0_n) is built only using time proximity (*i.e.* δ^0 parameters). Then for each level $\ell \in \{1; \dots; L-1\}$, the positive example is sampled closer and closer to the same anchor (*i.e.* $\delta^\ell_{p,min}$ and $\delta^\ell_{p,max}$ decrease), whereas the negative is obtained by selecting the positive example from level $\ell - 1$. This way, going deeper into the hierarchy enforces the representations to get more refined and detect short-term musical patterns. Turning the positive example at a given level into the negative one at the subsequent level while keeping the anchor point unchanged further encourages the model to learn more level-specific features. The modified sampling method is summarized in Figure 3. At the very first level $\ell = 0$, the positive and negative examples are uniformly sampled from the intervals I^0_p and I^0_n respectively (similar to the single-level case). Then, for each level $\ell \in \{1; \dots; L-1\}$, the negative example is transferred from the current to the next level, and the positive example is uniformly sampled from the interval I^ℓ_p.

2) Multi-level contrastive loss: In the case of the multi-level contrastive loss, the sampling process essentially remains the same as in the triplet case, with a few minor differences.

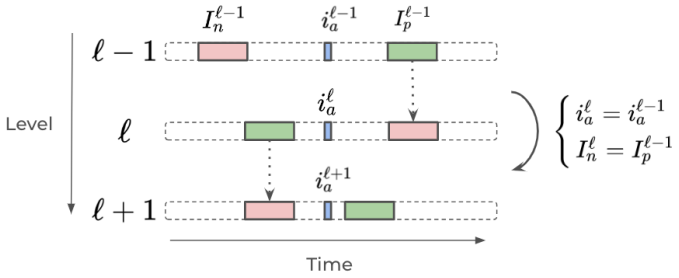

Fig. 4: Multi-level sampling for contrastive loss, moving downwards in the hierarchy.

For a given track t, a number N_a of anchor points are uniformly sampled across the whole track, yielding the anchor set $A(t)$. Then, for each anchor point $i_a \in A(t)$ at level $\ell = 0$, the sets of positive $P(i_a)^0$ and negative frames $N(i_a)^0$ are obtained by sampling uniformly N_a and N_p points from the intervals I^0_p and I^0_n. Then, for each level $\ell \in \{1; \dots; L-1\}$, the current negative sampling interval is the same as the positive one from level $\ell - 1$: $I^\ell_n = I^{\ell-1}_p$ and the new positive sampling interval I^ℓ_p is shifted closer to the anchor, according to the corresponding δ^ℓ parameters. In other words, the triplet sampling approach directly transfers positive indices from the current level to the next, while the contrastive one transfers the whole positive sampling interval (see Figure 4).

E. Disentangling hierarchy levels

During training, the model is shown frames sampled at different hierarchy levels and should optimize the corresponding sub-regions of the output embeddings. We adapt the method introduced by Veit et al. [22], called conditional similarity networks. This method has already proven to be efficient in the context of multi-dimensional music similarity learning [14], where a joint model learns compact representations of music audio signals complying with different similarity criteria, namely genre, mood, instrumentation and tempo. We propose to extend it to the hierarchical case: to model the different temporal distances, a set of L masking functions $m_\ell \in \{0, 1\}^d$ are applied to the embedding space of size d. Each mask can be interpreted as an element-wise gating function selecting the relevant dimensions of the embedding corresponding to a particular level of the hierarchy.

1) Triplet loss: For a given triplet (x_a, x_p, x_n) at level ℓ, the training objective becomes:

$$\mathcal{L}^{\text{triplet}}(x_a, x_p, x_n, \ell) = [D_\ell(x_a, x_p) - D_\ell(x_a, x_n) + \alpha_\ell]_+, \quad (5)$$

where

$$D_\ell(x_i, x_j) = \|m_\ell \circ [f_{x_i} - f_{x_j}]\|^2_2 \quad (6)$$

where $\circ$ is the Hadamard product, $[.]_+$ denotes the Hinge loss, α the margin parameter and f_x is the projection of x into the embedding space by the convolutional neural network. An example is illustrated in Figure 5, where $L = 2$ and $\ell = 0$. Since going deeper into the hierarchy results in triplets of frames getting temporally closer to each other, it is unnecessary for the model to separate samples by the same distance margin at all levels. Therefore, margin values are

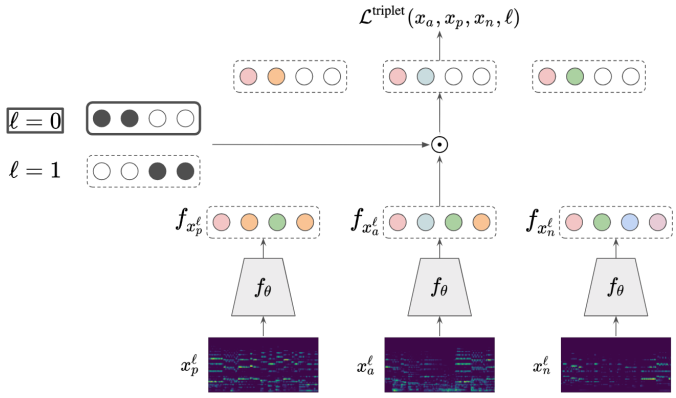

Fig. 5: Training pipeline for $\ell = 0$ and $L = 2$ embedding levels. At each iteration, the current hierarchy level defines the set of triplets used for training according to δ^ℓ parameters. The mask here conserves the sub-region corresponding to level $\ell = 0$.

set such that for each level $\ell \in \{0; \dots; L-2\}$, we have $\alpha_\ell > \alpha_{\ell+1}$.

2) Contrastive loss: For a given anchor point i_a, a positive example i_p and a set of negative examples $N(i_a)$, the pairwise contrastive loss becomes:

$$\mathcal{L}^{\text{pair}}(i_a, i_p, \ell) = \log \frac{\exp(f_a^\ell \cdot f_p^\ell / \tau)}{\exp(f_a^\ell \cdot f_p^\ell / \tau) + \sum\limits_{n \in N(i_a)} \exp(f_a^\ell \cdot f_n^\ell / \tau)} \tag{7}$$

where $f_i^\ell = m_\ell \circ f_i$ corresponds to the masked projection of sample i at level ℓ. The overall training pipeline is similar to the triplet case, depicted in Figure 5, with the difference that several positives and negatives are used for a given anchor point.

IV. Experiments

The experiments carried out in this work are four-folded. First, we consider the problem of single and multi-level boundary detection and section grouping. The next experiment consists in assessing how well the learnt representations adapt to a particular level of annotations. To do so, these are fine-tuned on a held-out training set, after which downstream segmentation is performed. Finally, we provide a qualitative analysis of these representations and illustrate through different examples how each element of our method impacts them.

A. Datasets

Since this work falls under the scope of self-supervised learning, a non annotated external audio collection is used for training. It is composed of $40,000$ tracks, spanning various musical genres such as rock, popular, rap, jazz, electronic or classical. These were retrieved from publicly available playlists and the audio obtained from YOUTUBE. Care has been taken to discard any track from this external collection also present in one of the following testing datasets.

SALAMI: the Structural Annotations for Large Amounts of Music Information (SALAMI) [23] is the most substantial dataset for music structure analysis. It contains $1,359$ tracks ranging from classical, jazz, popular to world and live music. Each track is provided with two levels of structural annotations. The first one denoted as *coarse* level, refers to the functional segmentation level (verse, chorus ...). The second one, *lower* level, provides a more fine-grained segmentation of the track closer to the motif scale. For evaluation, we use a subset of 884 songs labelled by two different annotators. Therefore, for each track, we end up with a total of 4 segmentation ground-truths (2 annotators $\times$ 2 levels of granularity). In the rest of this work, this subset is referred to as $\text{SALAMI}_{\text{test}}$. The remaining tracks of the dataset are used for validation and denoted as $\text{SALAMI}_{\text{val}}$.

JSD: the Jazz Structure Dataset [24] gathers 340 jazz recordings that were manually annotated by 3 semi-professional musicians with a background in jazz music. A key advantage of this dataset resides in the consistency of its annotations. These were built assuming that jazz recordings usually follow a fixed schema that includes the introduction of the main melody (theme), followed by alternating solos from the different musicians and a final return towards the main theme at the end of the track. Additionnally, each music piece is provided with two-level annotations: the chorus level (a full cycle of the harmonic schema) and a solo level, consisting of one or more choruses. We managed to retrieve the right audio version for 300 tracks out of the 340 composing the dataset and use this subset for evaluation.

Harmonix: the Harmonix dataset [25] 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.

B. Evaluation metrics

Common evaluation metrics for automatic structure analysis are employed throughout our experiments. In the case where the test dataset has more than one annotator, the best score across annotators is kept, as the goal of the evaluation process is to measure how close to human-truth the predicted segmentations are. Besides reporting the average score obtained per metric across tracks and its standard deviation, the statistical significance of the results against the flat representation baselines is assessed using a Wilcoxon signed-ranks test with $p < 0.05$. More precisely, the performance of each model variant is individually compared to that of the single level embeddings (*Flat*) in a track-wise manner.

1) Homogeneity: The notion of homogeneity plays a crucial role in the quality of the representations learned. Ideally, these should be very homogeneous inside annotated musical sections and highly heterogeneous across musical sections. To evaluate this aspect, we introduce a metric which compares the embeddings variance within and across annotated sections. This quantity is independent from downstream segmentation performance and therefore, provides a direct way to qualitatively evaluate the representations learned. Let us denote the embeddings matrix by Z of shape $d \times T$ where d and T refer

to the embeddings' features and time dimensions respectively. The total features variance V_{total} across the track is defined as:

$$V_{\text{total}} = \text{tr}(cov(Z)) \times T \tag{8}$$

For each musical section (or segment) i, the intra-section variance of features V_i using the sub-patch Z_i of Z corresponding to the T_i concatenated frames belonging to section (or segment) i is calculated as:

$$V_i = \text{tr}(cov(Z_i)) \times T_i \tag{9}$$

The overall statistics S is given by the following ratio:

$$S = \sum_i \frac{V_i}{V_{\text{total}}} \tag{10}$$

where $i \in \{0, \ldots, N\}$ with N being the number of segments (or sections) in the track considered. The metric S helps understanding how the learned representations vary with respect to the annotations. The lower the value of S, the more aligned the learned representations are with the annotated section boundaries and labels. We calculate this metric with and without section labels, thus yielding two homogeneity metrics S_{seg} and S_{sec} operating at the segment and section levels respectively.

2) Boundary detection: 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) on the original annotations.

3) Structural grouping: We report the F-measure of pairwise-frame clustering [27] (PFC), which gives another view on flat segmentation performance in terms of frame-wise section assignment. Additionally, the V-measure (V) [28] is also calculated, in order to indicate from a probabilistic perspective the amount of information shared between predicted label distributions and their corresponding reference annotations.

4) Multi-level segmentation: The evaluation on multi-level segmentation is carried out using the L-measure [9]. This metric allows for comparing hierarchies of segmentations operating at different scales. First, the reference hierarchy H^R is decomposed into a finite number of time instants (*i.e.* frames). Then, the set $A(H^R)$ of all triplets of frames (i, j, k) is retrieved such that i and j receive the same label deeper in the hierarchy than i and k. The same process is repeated with the same set of time instants for the estimated hierarchy H^E to obtain $A(H^E)$. Finally, the L-precision, L-recall and L-measure are derived by comparing $A(H^R)$ against $A(H^E)$. As noted in previous work [11, 12], hierarchies estimated with greater depth than reference annotations can make the L-precision metric uninformative. Therefore, our evaluation focuses on the L-recall, indicating how much of the reference hierarchy is retrieved in the estimated one.

[1]All evaluations are done using the mir_eval package [26].

[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.

C. Input features

All tracks are resampled at 22.05 kHz. As input to our deep neural network, we use log-scaled Mel-spectrograms with a window and hop size of 1024 and 256 respectively. We compute 64 Mel-bands per frame. The TorchAudio library is used for feature calculation [29]. As in previous work [12], beats are estimated for all tracks using the algorithm from Korzeniowski *et al.* [30] implemented in the *madmom* package [31]. It has been observed that in most cases, the beat tracker returns half the actual number of beats. Therefore, the number of detected beat locations is artificially doubled by linear interpolation. Patches of 128 frames ($\simeq 1.48$s) centered at each detected beat location are fed as input to the network.

D. Network architecture

The architecture of the encoder f_θ follows recent work in automatic music tagging [32], it consists of a CNN front-end and a transformer module. The front-end comprises three residual units with max-pooling and ELU activation to aggregate local spectro-temporal information from the input patch. The resulting low-dimensional feature map is denoted as $S = [S_0, S_1, \ldots, S_{T-1}] \in \mathbb{R}^{C \times F \times T}$ where C, F and T refer to channel, frequency and time dimensions respectively. It is reshaped so as to obtain one single vector per time-step, $S_i \in \mathbb{R}^{C \times F}$, $i \in \{0, \ldots, T-1\}$. The patch-wise sequence of 1D vectors (corresponding to approximately 46 ms each) is then processed by the transformer back-end, consisting of two transformer encoder layers with 8 attention heads each. After average pooling over the time dimension, the embedding vector of the input patch is finally obtained after applying a linear projection and ℓ_2-normalization, resulting in a vector of size $d = 128$. The model contains 1.1M parameters and is implemented with Pytorch 2.0 [33]. The SGD optimizer with 10^{-4} weight decay is used with 0.9 momentum. The model is trained for a maximum of 50 epochs and we use early stopping if the training loss has no decreased for 10 consecutive epochs.

E. Masks design

The output embeddings are divided into $L = 2$ distinct hierarchy levels, as the goal is to target both coarse and refined segmentations (similar to the annotation levels of the SALAMI dataset). In previous work, it was found beneficial to learn the masks during training to promote information sharing across similarity dimensions [22]. Here, we use disjoint masks with a constant value either equal to 0 or 1. The intuition is that the first half of the output embeddings should be used to discriminate coarse musical segments, and by adding more information (*i.e.* expanding the mask), finer-grained details are included and allow for discriminating between time-frames at smaller time scales. For the triplet loss, the margin parameters are set to $\alpha_0 = 0.1$ and $\alpha_1 = 0.05$ for the upper and lower representation levels respectively. For the contrastive loss, we use a unique temperature parameter $\tau = 0.25$ across levels.

F. Batch sampling scheme

1) Triplet model: During training, mini-batches of size 256 are composed of 8 anchor points uniformly sampled from 4

different songs, and from which 8 triplets are derived (4 for each representation level). To choose good sampling parameters, we use both annotation levels of $\text{SALAMI}_{\text{val}}$ and measure the amount of true positive and true negative examples while varying $\delta_{p,min}$ and $\delta_{p,max}$ using *upper* annotations. It was found that setting $\delta_{p,min} = 16$ and $\delta_{p,max} = 32$ provided a good balance between the true positives rate at level $\ell = 0$ and the true negatives rate at level $\ell = 1$. Positive examples at level $\ell = 1$ are sampled using $\delta_{p,min} = 1$ and $\delta_{p,max} = 16$, that is, at most 4 bars away from the anchor (in a 4/4 time signature).

2) Contrastive model: In the contrastive case, a batch comprises all the frames contained in a single song. The loss is computed over $N_a = 16$ anchors per track. The number of positives and negatives per anchor are set to $N_p = 16$ and $N_n = 32$ respectively. These values were found by trial and error and could be further tuned for better performance. However, we found that these already provide a good trade-off between the diversity of patches the model is exposed to during training, and the potential conflicts in positives at each hierarchy level. The positive and negative sampling intervals I_p^ℓ and I_n^ℓ use the same δ^ℓ parameters than for the triplet case.

G. Downstream segmentation

For all experiments, the embeddings returned by each model are fed as input to spectral clustering [10]. The main motivation for such choice resides in the fact that this algorithm jointly performs both boundary detection and structural grouping at multiple levels in an self-supervised manner. This allows one to compare the influence of each of the tested representations into a single unified framework. The original algorithm takes two distinct beat-synchronized audio features as input (MFCC and CQT), we denote this version by LSD (Laplacian Structural Decomposition) and consider it as one of our baselines. When feeding deep representations to this algorithm, we simply replace both input features by the whole embedding matrix multiplied by its corresponding mask. Finally, because this algorithm outputs multiple levels of segmentation, only the one maximizing the considered metric is reported (HR.5F and HR3F for boundary detection, PFC and V-measure for structural grouping).

V. Results

In this section, we provide results of the experiments performed with each model on the three datasets presented in Section IV-A. For clarity, we list here the abbreviations used in the following tables:

- LSD: stands for Laplacian Structural Decomposition [10], which corresponds to our first baseline.
- DEF: corresponds to the work by Salamon *et al.* [12] where segmentation with spectral clustering is improved using pre-trained audio embeddings.
- *Flat*: refers to representations learned with only one level, with a similar sampling strategy than that of McCallum [17], considered as a third baseline.
- *M-level*: denotes the multi-level representations proposed in this work. Because they contain two distinct sub-levels, *M-level*$_{coarse}$, *M-level*$_{refined}$, *M-level*$_{whole}$ and *M-level** respectively correspond to taking the upper embedding level, the lower embedding level, the whole embedding matrix (both levels together) and the best level between both.
- *Flat*/*M-level*$_{triplet/contr.}$: indicates whether the network was trained using the triplet or the contrastive loss.

A. Qualitative analysis

As a first evaluation step, we study the variability of the learned multi-level representations using the S metric defined in Section IV-B1. In Figure 6, we give the value of S across the three test datasets used in this work. The values are calculated at the segment level (S_{seg}) and at the section level (S_{sec}) after the model is trained on two representations levels using the contrastive loss. We observe that for each dataset, the variability within musical sections and segments is consistently lower for the upper representation level. This result was to be expected, as this embedding level is trained to group frames of temporally wider neighborhoods than its lower counterpart.

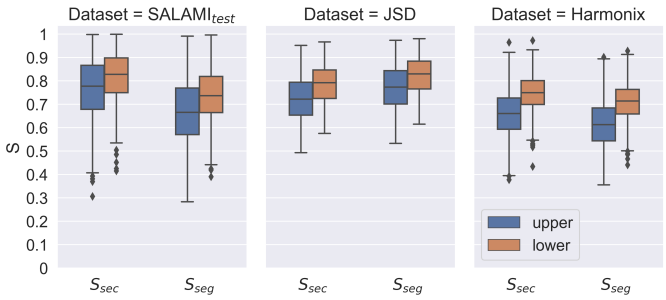

Fig. 6: Average S metric across tracks from $\text{SALAMI}_{\text{test}}$, Harmonix and JSD datasets. S_{sec} and S_{seg} stand for section and segment level respectively. Upper and lower denote the different embedding levels learned.

For $\text{SALAMI}_{\text{test}}$ and Harmonix, the variability within segments is smaller than within sections, indicating that the representations learned tend to vary more across occurrences of a same musical section, possibly due to musical variations between them, or an implicit temporal continuity encoded in the embeddings (the training objective only compares frames relatively close in time). However we observe the inverse phenomenon for the JSD dataset, probably due to the low number of section repetitions at this particular annotation level (*solo*) or the dependency of musical sections with the instrumentation, which turns out to be well captured by the network during training.

We give an example illustrating the proposed multi-level representations in Figure 7 where self-similarity matrices of a track are plotted. On the left-hand side, the self-similarity matrix was obtained using the upper representation level, supposedly the most adapted to a coarse segmentation. One can clearly see the prominent block-like structures indicating regions of high homogeneity. These same blocks also align well with the reference structural annotations, where the transition from one block to the next corresponds to

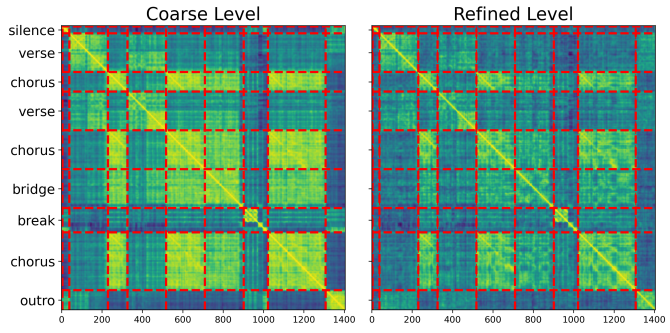

Fig. 7: Example of self-similarity matrices obtained for *Thirty Seconds To Mars - The Kill* obtained for upper (left) and lower (right) representation levels. Red dotted lines denote section boundary locations.

boundaries between musical segments. Similarly on the right-hand side, we plotted the same matrix this time using the lower representation level. Despite still having clear block structures, the overall similarity within those is generally lower than for the upper level, allowing for the appearance of shorter homogeneous sequences within these blocks, which are not as visible at the upper representation level. Therefore, combining both representation levels could allow to differentiate frames at a small and larger temporal scales simultaneously. Thus, the overall graph of the song is given a hierarchical organization that can be leveraged by spectral clustering during downstream segmentation.

B. Flat segmentation

1) Boundary detection: Results for boundary detection on $\text{SALAMI}_{\text{test}}$ and JSD are reported in Table I and II. We first observe that in both cases, replacing hand-crafted features by the representations learned improves the performance of spectral clustering, regardless of the model version considered. From a general perspective, the contrastive loss seems to perform better than the triplet configuration for both representation and annotation levels. As the contrastive loss simultaneously compares various positive and negatives, more general features can be learned during training, thus reducing noise in the resulting self-similarity matrix used for spectral clustering. Learning representations in a multi-level fashion also positively impacts results at a tolerance window of 0.5s, which suggests that there is an advantage of targeting both representation levels with the same backbone network, as these two tasks might benefit from each other.

Concerning the performance on SALAMI reported in Table I, most embedding configurations outperform the DEF baseline [12] for both tolerance windows of 0.5 and 3 seconds. For the triplet case, the performance between upper and lower representation levels is somewhat equivalent on both annotation levels (see lines $\textit{M-level}_{\textit{coarse, triplet}}$, $\textit{M-level}_{\textit{refined, triplet}}$ and $\textit{M-level}_{\textit{whole, triplet}}$) indicating that both representation levels are not disentangled enough to learn level-specific features. On the other hand, the contrastive loss seems to provide a better level disentanglement, where the upper representation level better matches the upper annotations and conversely, the lower one seems more efficient on more fine-grained boundary annotations (see lines $\textit{M-level}_{\textit{coarse, contr.}}$, $\textit{M-level}_{\textit{refined, contr.}}$ and $\textit{M-level}_{\textit{whole, contr.}}$). It is also interesting to see that using only the lower representation level generally leads to poorer performance than when combined with the upper one (see lines $\textit{M-level}_{\textit{refined, contr.}}$ and $\textit{M-level}_{\textit{whole, contr.}}$). This might imply that features learned at the lower level are complementary to those learned at the upper one, and that there exists a hierarchical relationship between them.

	SALAMI *upper*		SALAMI *lower*	
	HR.5F	HR3F	HR.5F	HR3F
DEF [12]	.338	.564	-	-
LSD [10]	.341±.17	.596±.18	.284±.14	.626±.16
$\textit{Flat}_{\textit{triplet}}$ [17]	.378±.16	.680±.17	.353±.13	.718±.12
$\textit{M-level}_{\textit{coarse, triplet}}$	.376±.16	.684±.17	.312±.13	.656±.14
$\textit{M-level}_{\textit{refined, triplet}}$	.376±.17	.684±.17	.310±.13	.660±.14
$\textit{M-level}_{\textit{whole, triplet}}$	.378±.16	.685±.17	.310±.13	.661±.14
$\textit{M-level*}_{\textit{triplet}}$	**.420** ±.16	.706±.17	**.383** ±.12	.737±.12
$\textit{Flat}_{\textit{contr.}}$ [17]	.372±.15	.695±.16	.334±.12	.727±.13
$\textit{M-level}_{\textit{coarse, contr.}}$	**.398** ±.16	.701±.17	**.353** ±.12	.724±.12
$\textit{M-level}_{\textit{refined, contr.}}$	**.380** ±.16	.690±.17	**.347** ±.12	.730±.12
$\textit{M-level}_{\textit{whole, contr.}}$	**.395** ±.16	.698±.17	**.355** ±.13	.726±.12
$\textit{M-level*}_{\textit{contr.}}$	**.450** ±.16	**.738** ±.16	**.394** ±.12	**.751** ±.12

TABLE I: Flat segmentation results on $\text{SALAMI}_{\text{test}}$. Results in bold denote statistically significant improvement over *Flat* representations.

Boundary detection results on JSD given in Table II show that this dataset is much more challenging than SALAMI for any configuration of our system, as can be seen by the severe performance drops on all metrics. However, our results are still on-par with the baselines experimented in the original paper [24]. We can also observe that multi-level representations tend to outperform the single-level ones on a majority of boundary detection metrics (see rows *M-level* and *Flat*). More precisely, the upper representation level seems to be more adapted to these annotations (see rows $\textit{M-level}_{\textit{coarse}}$ and $\textit{M-level}_{\textit{refined}}\textit{Flat}$). Given that the average segment duration in the JSD dataset is around 35 seconds, segment should have an approximate duration of 60 beats (at a typical tempo of 120 bpm). Therefore, positives sampled at the upper representation level approximately span half of their current segment, thus improving the homogeneity of representations at this level. To summarize, the proposed multi-level representations perform as well as their single-level equivalent for boundary detection at a 3s scale, and generally improve results for smaller temporal scales (0.5s).

2) Structural grouping: Results for structural grouping on SALAMI (Table III) and JSD (Table IV) follow the same trends. First, there is no significant difference between using one or two representation levels, either for training or inference (similar results between *Flat* and *Multi-level*) on SALAMI. On JSD, we can observe that using multiple representation levels can actually hurt performance (see *Flat* and *Multi-level* rows). One explanation is that adding embedding levels might make the representations too sensitive for the scale of the *solo* and *chorus* sections of JSD. It has been shown that the pairwise frame clustering metric can be highly sensitive to

	JSD *solo*		JSD *chorus*	
	HR.5F	HR3F	HR.5F	HR3F
LSD [10]	.126±.09	.289±.14	.145±.09	.343±.13
$Flat_{triplet}$ [17]	.266±.16	.537±.21	.243±.15	.495±.18
$M\text{-}level_{coarse,\ triplet}$	.267±.17	**.549** ±.21	.245±.15	.502±.18
$M\text{-}level_{refined,\ triplet}$	.249±.15	.542±.21	.226±.13	.498±.17
$M\text{-}level_{whole,\ triplet}$	.272±.17	**.548** ±.21	.247±.14	.502±.18
$M\text{-}level^{*}_{triplet}$	**.308** ±.16	**.576** ±.21	**.283** ±.14	**.526** ±.17
$Flat_{contr.}$ [17]	.253±.16	.572±.20	.234±.13	.540±.17
$M\text{-}level_{coarse,\ contr.}$	**.279** ±.18	**.581** ±.21	**.250** ±.15	.528±.18
$M\text{-}level_{refined,\ contr.}$	.270±.16	.569±.21	**.245** ±.14	.529±.17
$M\text{-}level_{whole,\ contr.}$	**.280** ±.18	**.581** ±.21	**.255** ±.14	.533±.18
$M\text{-}level^{*}_{contr.}$	**.345** ±.18	**.628** ±.21	**.307** ±.15	**.569** ±.18

TABLE II: Flat segmentation results on JSD. Results in bold denote statistically significant improvement over *Flat* representations.

	JSD *solo*		JSD *chorus*	
	PFC	V	PFC	V
LSD [10]	.488±.11	.457±.12	.382±.12	.496±.12
$Flat_{triplet}$ [17]	.732±.12	.708±.13	.577±.14	.680±.11
$M\text{-}level_{coarse,\ triplet}$	.731±.12	.708±.13	.580±.14	.684±.10
$M\text{-}level_{refined,\ triplet}$	.731±.12	.706±.13	.575±.13	.677±.10
$M\text{-}level_{whole,\ triplet}$	.734±.12	.710±.13	.579±.13	.682±.10
$M\text{-}level^{*}_{triplet}$	**.749** ±.11	**.724** ±.12	**.594** ±.14	**.695** ±.10
$Flat_{contr.}$ [17]	.749±.11	.736±.12	.625±.13	.726±.09
$M\text{-}level_{coarse,\ contr.}$	.751±.12	.733±.12	.606±.14	.706±.10
$M\text{-}level_{refined,\ contr.}$	.740±.12	.723±.13	.593±.14	.697±.10
$M\text{-}level_{whole,\ contr.}$	.751±.12	.733±.12	.603±.14	.703±.10
$M\text{-}level^{*}_{contr.}$	**.771** ±.11	**.751** ±.12	.620±.14	.718±.11

TABLE IV: Section grouping results on JSD. Results in bold denote statistically significant improvement over *Flat* representations.

precise boundary locations [34], which might explain how this metric varies between annotation levels (see columns *upper/lower*, *solo/chorus*). The V-measures obtained however, are more consistent and indicate that the frame assignment returned by spectral clustering is generally better for JSD than SALAMI. While we hypothesize that harmony-related information is encoded into the representations, this result leads us to think that they are also strongly dependent to timbre variations, which would align well with the succeeding solo sections from the JSD dataset.

	SALAMI *upper*		SALAMI *lower*	
	PFC	V	PFC	V
LSD [10]	.782±.14	.626±.17	.664±.16	.645±.13
$Flat_{triplet}$ [17]	.786±.14	.639±.17	.658±.16	.658±.14
$M\text{-}level_{coarse,\ triplet}$	.784±.14	.643±.17	.657±.16	.644±.15
$M\text{-}level_{refined,\ triplet}$	.783±.14	.642±.17	.658±.16	.644±.15
$M\text{-}level_{whole,\ triplet}$	.785±.14	.643±.17	.657±.16	.646±.15
$M\text{-}level^{*}_{triplet}$	**.797** ±.14	**.661** ±.17	**.671** ±.16	**.669** ±.14
$Flat_{contr.}$ [17]	.786±.13	.647±.17	.662±.15	.664±.14
$M\text{-}level_{coarse,\ contr.}$	.788±.14	.649±.17	.664±.16	.666±.14
$M\text{-}level_{refined,\ contr.}$	.788±.13	.648±.17	.664±.15	.664±.14
$M\text{-}level_{whole,\ contr.}$	.791±.13	.651±.17	.667±.15	.668±.14
$M\text{-}level^{*}_{contr.}$	**.810** ±.13	**.677** ±.17	**.684** ±.15	**.682** ±.14

TABLE III: Section grouping results on $\text{SALAMI}_{\text{test}}$. Results in bold denote statistically significant improvement over *Flat* representations.

As for SALAMI, we notice slight improvements when both representation levels are used jointly ($M\text{-}level_{whole}$ rows), probably due to the multi-level learning procedure. Employing the contrastive loss generally results in higher structural grouping performance (see rows $M\text{-}level_{triplet}$ and $M\text{-}level_{contr.}$), confirming our hypothesis that comparing more frames at once during training may lead to less noisy features.

C. Multi-level segmentation

The evaluation of the learned representations on multi-level segmentation is summarized in Table V for SALAMI and Table VI. Here we can clearly see that training a model on more than one representation level brings improvements in terms of L-recall in most cases (see rows *Flat* and *M-level*). This indicates that both high and low-level structures contained in annotations are well captured by the predicted hierarchies.

For SALAMI, all the configurations of the proposed method outperform the DEF baseline [12] by a large margin. Improvements over single-level representations are only achieved in the contrastive loss case, where the most refined representation level includes more than 75% of the annotated hierarchies. Conversely on JSD, the best representations are those that include the coarse level ($M\text{-}level_{coarse,\ contr.}$ and $M\text{-}level_{whole,\ contr.}$). This aligns with the previous results on structural grouping, suggesting that using the lower representation level only might be too specific for the hierarchies annotated in this dataset.

Method	L-precision	L-recall	L-measure
Inter-annot	0.664	0.664	0.654
DEF [12]	.435	.673	.500
LSD [10]	.450±.13	.689±.14	.534±.14
$Flat_{triplet}$ [17]	.449±.12	.740±.12	.549±.13
$M\text{-}level_{coarse,\ triplet}$	.449±.13	.738±.13	.548±.13
$M\text{-}level_{refined,\ triplet}$	.448±.13	.741±.12	.548±.13
$M\text{-}level_{whole,\ triplet}$	.449±.13	.741±.12	.549±.13
$M\text{-}level^{*}_{triplet}$	.457±.13	**.758** ±.12	**.559** ±.13
$Flat_{contr.}$ [17]	.451±.12	.745±.12	.553±.12
$M\text{-}level_{coarse,\ contr.}$	.449±.13	.751±.12	.552±.13
$M\text{-}level_{refined,\ contr.}$	.451±.12	**.755** ±.12	.555±.13
$M\text{-}level_{whole,\ contr.}$	.452±.13	**.754** ±.12	.555±.13
$M\text{-}level^{*}_{contr.}$	**.465** ±.13	**.775** ±.12	**.571** ±.13

TABLE V: Multi-level segmentation results on $\text{SALAMI}_{\text{test}}$. Inter-annot corresponds to the inter-annotator agreement. Results in bold denote statistically significant improvement over *Flat* representations.

D. Comparison to supervised baselines

In this section, we provide an additional performance comparison of our approach against recent fully supervised baselines for music structure analysis. The first one is the work from Wang *et al.* [15] which uses supervised metric learning to learn audio representations and spectral clustering to perform

Method	L-precision	L-recall	L-measure
LSD [10]	.310±.08	.635±.14	.412±.10
Flat$_{triplet}$ [17]	.376±.08	.829±.10	.513±.08
M-level$_{coarse,\ triplet}$	.380±.08	**.841** ±.10	.519±.08
M-level$_{refined,\ triplet}$	.375±.08	.837±.10	.514±.08
M-level$_{whole,\ triplet}$	.377±.08	**.840** ±.10	.516±.08
*M-level**$_{triplet}$	.381±.08	**.851** ±.09	.522±.08
Flat$_{contr.}$ [17]	.391±.08	.849±.09	.531±.08
M-level$_{coarse,\ contr.}$	.381±.08	**.857** ±.09	.523±.08
M-level$_{refined,\ contr.}$	.377±.08	.849±.09	.518±.08
M-level$_{whole,\ contr.}$	.378±.08	**.857** ±.09	.521±.08
*M-level**$_{contr.}$	.385±.08	**.870** ±.08	.530±.08

TABLE VI: Multi-level segmentation results on JSD. Results in bold denote statistically significant improvement over *Flat* representations.

boundary detection and structural grouping. This baseline is denoted as Harmonic CNN, derived from the backbone architecture employed in their work. Comparison against this work is the most straightforward among all the listed baselines, as it is essentially the same music segmentation pipeline as the one followed in our method, with the difference that their network was trained in a fully supervised fashion. The second baseline, denoted as SpecTNT is from subsequent work by Wang *et al.* [35] where a transformer-based architecture is employed to jointly predict the boundary probability of each time frame, along with the label of the musical section it belongs to. Finally we also report a recent method by Kim *et al.* [36] (All in One) which applies a temporal convolutional network on demixed audio sources to simultaneously predict beats, downbeats, section boundaries and section labels. Results for the former two were obtained via 4-fold cross-validation on Harmonix, while results for the latter are obtained from 8-fold cross-validation. In Table VII, the performance on flat segmentation is reported on the original label taxonomy of the dataset. For multi-level segmentation evaluation in Table VIII, we apply hierarchy expansion beforehand [37] by using the label pre-processing step proposed in [35] as the coarsest annotation level.

Method	F.5	F3	PFC	Sf
Harmonic CNN [15]	.497	.738	.684	.743
SpecTNT [35]	.558	-	.712	.724
All in One [36]	.660	-	.738	.769
LSD [10]	.401	.648	.666	.630
Flat$_{triplet}$ [17]	.461	.786	.755	.742
M-level$_{coarse,\ triplet}$	.461	**.799**	.760	.745
M-level$_{refined,\ triplet}$	.461	**.797**	.762	.746
M-level$_{whole,\ triplet}$	**.470**	**.802**	**.762**	**.750**
*M-level**$_{triplet}$	**.520**	**.816**	**.775**	**.755**
Flat$_{contr.}$ [17]	.456	.801	.753	.741
M-level$_{coarse,\ contr.}$	**.487**	.804	**.763**	.750
M-level$_{refined,\ contr.}$	**.468**	.803	**.773**	**.753**
M-level$_{whole,\ contr.}$	**.485**	.808	**.770**	**.754**
*M-level**$_{contr.}$	**.548**	**.834**	**.794**	**.773**

TABLE VII: Comparison with supervised baselines on Harmonix dataset, flat segmentation and section grouping. Results in bold denote statistically significant improvement over *Flat* representations.

Flat segmentation results on the Harmonix dataset are given in Table VII. The contrastive loss yields better audio representations than the triplet one (see lines *M-level*$_{triplet}$ and *M-level*$_{contr.}$). When compared against the supervised baselines listed at the top of the table, we observe that our approach outperforms its fully supervised equivalent from the work by Wang *et al.* [15] for boundary detection at a 3s tolerance window and both structural grouping metrics. At a 0.5s tolerance window, the configuration *M-level*$_{coarse,\ contr.}$ of our system is only 1% below the performance of this baseline. However, our system largely falls behind the two remaining methods for this same tolerance window. The training objective of our approach is not dependent on the segmentation performance as in a supervised setting. Therefore, it is impossible for the network to learn fine-grained structural elements near segment boundaries or acquire any information about section labels assignment, which could indirectly inform the boundary detection task.

Yet, our approach seems to perform well at structural grouping. Because it can leverage large quantities of music data, we hypothesize that more robust structural features are learned by the network, which improves the clustering step during downstream segmentation. Additionally, the label pre-processing proposed in [35] was meant to reduce the number of possible section labels. Despite reducing some ambiguities in the annotation process (merging "chorus" and "refrain" sections), it may still regroup musical segments having different functions within a given track (for example "break" and "impro" are grouped into the "instrumental" category), which could make the section grouping task harder to solve for the model or wrongly penalize label predictions in some cases. As an example, Figure 8 shows the reference annotations for a particular track of the Harmonix dataset (left-hand side). Here, the "prechorus" section is highly similar to the following "chorus" section, as can be seen by brighter values in the self-similarity matrix returned by the proposed model. In this context, attributing the label "verse" to the former, as suggested in [35], would likely push the model to learn an irrelevant label assignment. It is also to be reminded that the clustering results reported in Table VII are obtained by selecting the segmentation prediction level that maximizes the considered metric. Consequently, the proposed system has a higher chance of predicting a label assignment that fits best with the annotated test data among all the predicted levels returned by spectral clustering. This is a major advantage compared to the former two baselines previously introduced, which only formulate one single level of segmentation. All in all, despite not reaching the performance of supervised approaches for boundary detection, our approach can still provide a strong initialization for subsequent systems that would directly optimize segmentation results.

Finally, we conducted a result analysis of the proposed segmentation method. By looking more closely at failure cases, we noticed that most errors made are related to the third aspect of music structure, which has not been considered in this work: the regularity of musical sections. In our scenario, the training objective employed to learn audio representations is mainly motivated by the homogeneity principle: neighbouring frames

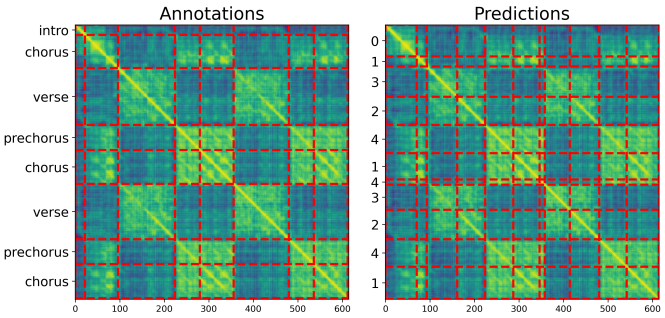

Fig. 8: Example of the self-similarity matrix obtained for *Jay Sean - Down* along with the reference structural annotations (left) and the best predictions from the proposed method (right). Red dotted lines denote section boundary locations.

in time are likely to belong to the same musical segment, and should therefore be close in the embedding space. By applying this heuristic across each training track, the network manages to pick repeating segments. However, there is no explicit constraint formulated with respect to the length of these segments. Similarly, the k-means step performed by spectral clustering does not incorporate any knowledge about section durations either. Figure 8 illustrates a typical failure case where predicted segments (right-hand side) tend to have irregular durations (see predicted segments labelled as 1 and 4 for example). These durations usually differ by a handful of beats from the annotations, so their influence on segmentation performance for larger tolerance windows (3 seconds for example) is relatively low. However, they have a greater impact at more fine-grained resolutions (0.5 second for example), which can partially explain the important performance drop across both tolerance windows of our system.

Method	L-precision	L-recall	L-measure
DEF [12]	.432	.810	.560
LSD [10]	.383±.09	.730±.13	.499±.10
Flat$_{triplet}$ [17]	.425±.08	.829±.10	.559±.09
M-level$_{coarse,\ triplet}$	.426±.08	.832±.10	.560±.09
M-level$_{refined,\ triplet}$	.427±.08	.834±.10	.562±.09
M-level$_{whole,\ triplet}$	.426±.08	.833±.10	.561±.09
*M-level**$_{triplet}$	**.434** ±.08	**.850** ±.09	**.571** ±.09
Flat$_{contr.}$ [17]	.422±.08	.831±.10	.556±.09
M-level$_{coarse,\ contr.}$	.424±.08	.836±.10	.560±.09
M-level$_{refined,\ contr.}$	**.433** ±.08	**.851** ±.10	**.571** ±.09
M-level$_{whole,\ contr.}$	.428±.08	**.845** ±.10	**.565** ±.09
*M-level**$_{contr.}$	**.442** ±.08	**.868** ±.09	**.582** ±.09

TABLE VIII: Multi-level segmentation results on Harmonix. Results in bold denote statistically significant improvement over *Flat* representations.

VI. Fine-tuning

As a last experiment, we explore a simple fine-tuning strategy of the pre-trained multi-level model using a small quantity of annotated data, which relates to a practical scenario in which a user might want to perform structural segmentation on a set of tracks at a specific level of detail, of which only a small fraction was annotated accordingly. The encoder is pre-trained using the contrastive loss and $L = 2$ embedding levels. For fine-tuning, we use a set of tracks from the SALAMI dataset labelled as "Popular" (corresponding to 274 tracks) as training set and 100 randomly chosen remaining tracks of SALAMI, denoted as SALAMI$_{\text{val}}$ for validation. Training is monitored by the average boundary detection score ($\frac{1}{2}$HR3F+$\frac{1}{2}$HR.5F) on SALAMI$_{\text{val}}$. We use the Adam optimizer with 10^{-4} weight decay, the initial learning rate is set to 1×10^{-4} for the last linear layer and 1×10^{-5} for the rest of the encoder. It is multiplied by 0.8 if no segmentation improvement is observed for 10 consecutive epochs.

The training objective used, similar to that of Peeters *et al.* [38], measures how much a predicted self-similarity matrix deviates from an ideal affinity matrix, obtained for each track using its structural annotations. This allows for refining the existing representations by forcing them to be similar or dissimilar if they belong to the same section or not and this, for any arbitrary number of musical sections. We define as mini-batches all the frames comprised in a single song. Given a mini batch $Z = \{X_k\}_{k=1}^{N}$ with N being the number of frames in a track (*i.e.* beats), the predicted self-similarity $Y \in [0,1]^{N \times N}$ is such that:

$$Y(i,j) = 1 - \frac{1}{4}\|Z_i - Z_j\|_2^2, \tag{11}$$

where $i, j = 1 \dots N$. Let $Y^* \in \{0,1\}^{N \times N}$ be the corresponding ideal affinity matrix obtained from structural annotations, the training objective is expressed as:

$$\mathcal{L}(Y, Y^*) = \frac{1}{N^2}\|Y - Y^*\|_F^2, \tag{12}$$

where $\|A\|_F^2$ is the squared Frobenius norm. For conciseness, the evaluation is performed only on Harmonix for boundary detection and structural grouping and the results are reported in Table IX.

Level	HR.5F	HR3F	PFC	V
Init.	.485 ± .17	.808 ± .12	.770 ± .11	.754 ± .10
upper	.479 ± .17	**.817** ±.12	.769 ± .11	.757 ± .10
lower	.487 ± .17	.813 ± .12	.765 ± .11	.752 ± .10

TABLE IX: Boundary detection and structural grouping results on Harmonix after fine-tuning on SALAMI popular subset (using *upper* and *lower* annotations). Results in bold denote improvement over pre-trained representations (Init.).

Results in Table IX show that the simple supervised training objective is not well adapted to the task, where only slight improvements are made when fine-tuning on upper annotations. We have identified several possible explanations for this. First, it is unclear how to predict what the network has learned during its pre-training phase. Therefore, the training objective from Equation (12) might yield very high values when the reference annotations used to build the ground-truth self-similarity matrix does not align with the representations. For example, frames from multiple occurrences of a given section could be grouped together in the embedding space, but receive a different label in the annotations (in case of

refined annotations for example). This would force the network to separate them and therefore lose potentially valuable information on what they had in common. Additionally, the network used does not include any positional information about frames. However in structure, timing information is crucial, as the absolute and relative positions between frames within a track provide strong cues on whether they should be grouped or separated. Finally, one could also change the fine-tuning paradigm, so as to predict frame-wise boundary and section labels probabilities such as in the supervised baselines listed above.

VII. Conclusion

In this work a method to build multi-level audio representations for the task of hierarchical music structure analysis has been proposed. By combining a sampling approach at different temporal scales and a disentangled contrastive learning approach, multi-level representations for music segmentation can be optimized without explicit supervision from structural annotations.

The resulting embedding space provides distinct views of the input music signal, and is shown to encode frame similarity at various levels of detail. The experimental validation performed on three substantial datasets for music structure analysis show that the proposed audio representations improve both single-level and multi-level segmentation predictions after spectral clustering and can in some cases perform in a comparable range than that of fully supervised methods. Further improvements are systematically achieved by selecting the optimal representation level of each segmented track, which further illustrates the advantage of relying on different notions of similarity for music segmentation and incorporating them into the feature extraction stage.

Because this method is self-supervised, it can leverage large quantities of unlabeled data and reduce the effect of bias induced by specific annotation styles or music genres. It could also be further improved by more musically inspired downstream segmentation algorithms. Future work includes the exploration of such methods, in particular, the inclusion of timing as prior information to further enhance the decomposition of musical signals at different levels of detail.

VIII. Acknowledgements

This work was performed using HPC resources from GENCI-IDRIS (Grant 2022-AD011013255R1).

References

[1] Oriol Nieto et al. “Audio-based music structure analysis: Current trends, open challenges, and applications”. In: *Transactions of the International Society for Music Information Retrieval* 3.1 (2020).

[2] Michael J Bruderer, Martin F Mckinney, and Armin Kohlrausch. “The perception of structural boundaries in melody lines of Western popular music”. In: *Musicae Scientiae* 13.2 (2009), pp. 273–313.

[3] Cheng-i Wang, Gautham J Mysore, and Shlomo Dubnov. “Re-Visiting the Music Segmentation Problem with Crowdsourcing.” In: *ISMIR*. 2017.

[4] Jordan Smith. “Explaining listener differences in the perception of musical structure”. PhD thesis. Explaining Listener Differences in the Perception of Musical Structure, 2014.

[5] Markus Schedl. “Intelligent user interfaces for social music discovery and exploration of large-scale music repositories”. In: *Proceedings of the 2017 ACM Workshop on Theory-Informed User Modeling for Tailoring and Personalizing Interfaces*. 2017.

[6] Juan P Bello. “Measuring structural similarity in music”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.7 (2011), pp. 2013–2025.

[7] Shuqi Dai, Huiran Yu, and Roger B Dannenberg. “What is missing in deep music generation? A study of repetition and structure in popular music”. In: *ISMIR*. 2022.

[8] Katherine M Kinnaird. “Aligned sub-hierarchies: A structure-based approach to the cover song task”. In: *ISMIR*. 2018.

[9] Brian McFee et al. “Evaluating hierarchical structure in music annotations”. In: *Frontiers in psychology* (2017).

[10] Brian McFee and Dan Ellis. “Analyzing Song Structure with Spectral Clustering.” In: *ISMIR*. 2014.

[11] Christopher J Tralie and Brian McFee. “Enhanced hierarchical music structure annotations via feature level similarity fusion”. In: *ICASSP*. 2019.

[12] Justin Salamon, Oriol Nieto, and Nicholas J. Bryan. “Deep Embeddings and Section Fusion Improve Music Segmentation”. In: *ISMIR*. 2021.

[13] Morgan Buisson et al. “Learning Multi-Level Representations for Hierarchical Music Structure Analysis”. In: *ISMIR*. 2022.

[14] Jongpil Lee et al. “Disentangled multidimensional metric learning for music similarity”. In: *ICASSP*. 2020.

[15] Ju-Chiang Wang et al. “Supervised Metric Learning For Music Structure Features”. In: *ISMIR*. 2021.

[16] Xun Wang et al. “Multi-similarity loss with general pair weighting for deep metric learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5022–5030.

[17] Matthew C McCallum. “Unsupervised learning of deep features for music segmentation”. In: *ICASSP*. 2019.

[18] Laure Prétet, Gaël Richard, and Geoffroy Peeters. “Learning to rank music tracks using triplet loss”. In: *ICASSP*. 2020.

[19] Guillaume Doras and Geoffroy Peeters. “A prototypical triplet loss for cover detection”. In: *ICASSP*. 2020.

[20] Minz Won et al. “Multimodal metric learning for tag-based music retrieval”. In: *ICASSP*. 2021.

[21] Kihyuk Sohn. “Improved deep metric learning with multi-class n-pair loss objective”. In: *Advances in neural information processing systems* 29 (2016).

[22] Andreas Veit, Serge Belongie, and Theofanis Karaletsos. “Conditional similarity networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[23] Jordan Bennett Louis Smith et al. “Design and creation of a large-scale database of structural annotations.” In: *ISMIR*. 2011.

[24] Stefan Balke et al. “JSD: A Dataset for Structure Analysis in Jazz Music”. In: *Transactions of the International Society for Music Information Retrieval* 5.1 (2022).

[25] Oriol Nieto et al. “The Harmonix Set: Beats, Downbeats, and Functional Segment Annotations of Western Popular Music.” In: *ISMIR*. 2019.

[26] Colin Raffel et al. “mir_eval: A transparent implementation of common MIR metrics”. In: *ISMIR*. 2014.

[27] Mark Levy and Mark Sandler. “Structural segmentation of musical audio by constrained clustering”. In: *IEEE transactions on audio, speech, and language processing* 16.2 (2008), pp. 318–326.

[28] Andrew Rosenberg and Julia Hirschberg. “V-measure: A conditional entropy-based external cluster evaluation measure”. In: *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[29] Yao-Yuan Yang et al. “TorchAudio: Building Blocks for Audio and Speech Processing”. In: *arXiv preprint arXiv:2110.15018* (2021).

[30] Filip Korzeniowski, Sebastian Böck, and Gerhard Widmer. “Probabilistic Extraction of Beat Positions from a Beat Activation Function.” In: *ISMIR*. 2014.

[31] Sebastian Böck et al. “Madmom: A new python audio and music signal processing library”. In: *Proceedings of the 24th ACM international conference on Multimedia*. 2016.

[32] Minz Won, Keunwoo Choi, and Xavier Serra. “Semi-supervised music tagging transformer”. In: *ISMIR*. 2021.

[33] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* (2019).

[34] Oriol Nieto and Juan Pablo Bello. “Music segment similarity using 2d-fourier magnitude coefficients”. In: *ICASSP*. 2014.

[35] Ju-Chiang Wang, Yun-Ning Hung, and Jordan BL Smith. “To catch a chorus, verse, intro, or anything else: Analyzing a song with structural functions”. In: *ICASSP*. 2022.

[36] Taejun Kim and Juhan 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.

[37] Brian McFee and Katherine M Kinnaird. “Improving structure evaluation through automatic hierarchy expansion”. In: *ISMIR*. 2019.

[38] Geoffroy Peeters. “Self-Similarity-Based and Novelty-based loss for music structure analysis”. In: *ISMIR*. 2023.