for each level $\ell \in \{1; \ldots; L-1\}$, 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 means that the representations get more refined to detect short-term musical patterns. The modified sampling method is summarized in Figure 2. The process is then repeated by starting over from level 0, going down the hierarchy with the same anchor index, transferring the negative example from the current to the next level, and uniformly sampling the positive ones using the right δ parameters. At the end, the whole training set for all levels of the hierarchy is given by combining every set of triplets T_ℓ level-wise: $T = \{T_\ell\}_{\ell=0}^{L-1}$.

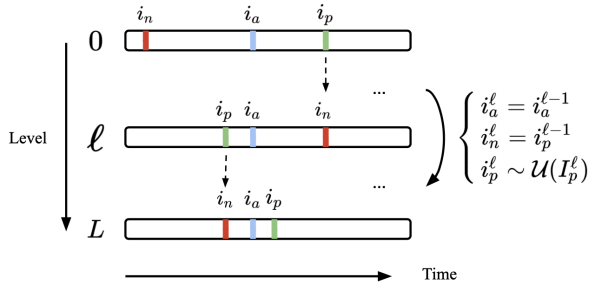

Figure 2. Modified triplet sampling, moving downwards in the hierarchy.

2.2 Disentangled hierarchy levels

During training, the model is shown triplets 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. [12], called Conditional Similarity Networks. This method has already proven to be efficient in the context of multi-dimensional music similarity learning [11], 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 d_ℓ, a set of L masking functions $m_\ell \in \{0,1\}^n$ that are applied to the embedding space of size n is defined. 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. For a given triplet (x_a, x_p, x_n) at level ℓ, the training objective becomes:

$$\mathcal{L}(x_a, x_p, x_n) = [D_\ell(x_a, x_p) - D_\ell(x_a, x_n) + \alpha]_+, \quad (2)$$

$$D_\ell(x_i, x_j) = \| \, m_\ell \circ [f(x_i) - f(x_j)] \, \|_2^2 \quad (3)$$

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 3, where $L = 3$ and $\ell = 1$. 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 were evenly distributed within the range $[0.05, 0.1]$ so that for each level $\ell \in \{0; \ldots; L-2\}$, we have $\alpha_\ell > \alpha_{\ell+1}$.

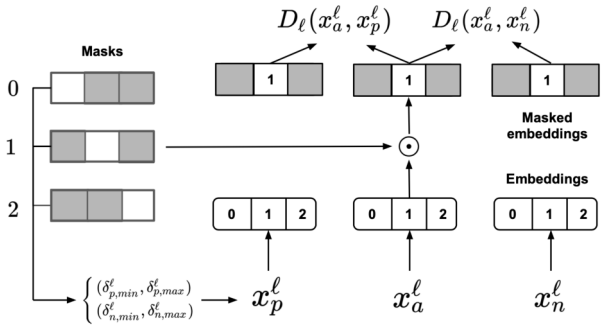

Figure 3. Training pipeline for $\ell = 1$ and $L = 3$. At each iteration, the current hierarchy level defines the set of δ parameters to sample the positive example. The mask here conserves the sub-region corresponding to level $\ell = 1$.

3. EXPERIMENTS

The evaluation of our method is divided into three distinct parts. First, we consider the problem of boundary detection on flat annotations using the SALAMI dataset. Second, we verify if the learned hierarchical representations improve multi-level segmentation predictions on that same dataset using the two-level structural annotations available. We finally demonstrate the flexibility of our approach and provide additional results on other commonly used datasets for music structure analysis where their original flat annotations have been automatically expanded beforehand [13].

3.1 Datasets

We use five different datasets in our evaluation:

SALAMI: the Structural Annotations for Large Amounts of Music Information (SALAMI) [8] 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. We use a subset of 884 songs labelled by two different annotators. Therefore, for each track contained in this subset, 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 as SALAMI.

BeatlesTUT: a revised version of 174 annotated Beatles songs, originally released in the Isophonics dataset [14] and corrected by researchers from Tampere University of Technology.

RWC-Pop: the Popular subset of the RWC dataset [15] contains 100 songs with section annotations. Note that two versions of these annotations are available online; here the ones originally provided by the authors (AIST) are used.

RWC-Jazz: the Jazz subset of the RWC dataset [15] is composed of 50 songs from various Jazz sub-genres such as Vocal, Big Band, Modal, Funky, Free or Fusion Jazz.

JAAH: the Audio-aligned jazz harmony dataset (JAAH) [16] is composed of 113 tracks selected from "The Smithsonian Collection of Classic Jazz" and "Jazz: The Smithsonian Anthology", covering various performers, sub-genres and historical periods.

3.1.1 Obtaining multi-level annotations:

For all datasets but SALAMI, we apply automatic hierarchy expansion [13] before evaluating multi-level segmentation. As can be seen in the descriptive statistics from Table 1, both the distributions of section labels and segment durations vary from one dataset to another. This difference can either be explained by the style of annotations (*i.e.* label taxonomy, desired level of detail...) or the music genre. As a consequence, the average number of levels obtained after automatic hierarchy expansion is dependent on the repetition of section labels and their semantic structure, which varies with the annotation process as well.

Dataset	N	Uni	Seg	Dur	Levels
SALAMI^0 (upper)	884	5.3	10.9	63.5	2.0
SALAMI^0 (lower)	884	10.0	33.1	18.4	2.0
SALAMI^1 (upper)	884	5.0	11.2	61.1	2.0
SALAMI^1 (lower)	884	9.2	34.1	18.0	2.0
BeatlesTUT	174	5.6	10.1	36.1	2.5
RWC-Pop	100	8.9	16.4	28.5	2.9
RWC-Jazz	50	14.2	19.9	32.1	2.8
JAAH	113	6.2	8.0	63.1	2.0

Table 1. Datasets descriptive statistics. N: number of annotated songs. Uni: average number of unique section labels per song. Seg: average number of segments per song. Dur: average duration of each section per song (in beats). Levels: average number of annotation levels per song after automatic hierarchy expansion. SALAMI^i: ith annotator.

3.1.2 Training data:

Since this work falls under the scope of unsupervised learning, a non annotated external audio collection is used for training. It is composed of $23,725$ 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 testing datasets.

3.2 Evaluation metrics

3.2.1 Flat segmentation:

For boundary detection, we report the F-measure [1] of the trimmed boundary detection hit-rate with a 3-second tolerance window (F_3) on the original annotations. We also report the F-measure of frame pairwise clustering [18] (F_{pairwise}), which gives another view on flat segmentation performance in terms of frame-wise section assignment.

[1] All evaluations are done using the mir_eval package [17].

3.2.2 Multi-level segmentation:

The second part of the evaluation on multi-level segmentation is carried out using the L-measure [7]. 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) such that i and j receive the same label deeper in the hierarchy than i and k is retrieved. 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 [5, 10], hierarchies estimated with greater depth than reference annotations can make the L-precision metric unreliable. Therefore, our evaluation focuses on the L-recall, indicating how much of the reference hierarchy is retrieved in the estimated one. For this part of the evaluation, the expanded version of each dataset is used except for SALAMI, where for comparison purposes, the reference hierarchy only comprises both of the original annotation levels provided by each annotator (*upper* and *lower*).

3.3 Input features

All tracks are resampled at 22.05 kHz. Previous work has demonstrated that homogeneous regions and sharp changes of timbral content can be a good indicator of section transitions [19]. Therefore, we use log-scaled mel-spectrograms, with a window and hop size of 2048 and 256 respectively. We compute 60 mel-bands per frame. Beats are estimated for all tracks using the Librosa [20] implementation of the beat tracking algorithm from Ellis [21]. For both feature types, patches of 512 frames ($\simeq 5.94$s) are observed, centered at each detected beat location.

3.4 Implementation details

3.4.1 Network architecture:

We use a basic convolutional neural network architecture composed of 3 convolutional blocks, each comprising a convolutional and a max-pooling layer and Relu activation, followed by two fully-connected layers with Relu activations and a third fully-connected layer with linear activation. All convolutional layers use a kernel size of $(6, 4)$. A common practice in contrastive learning is to constrain the learned representations to lie within the unit hypersphere [22]. Therefore, the output embeddings are L2-normalized prior to distance calculations. The models were implemented with Pytorch 1.7.1 [23]. The RMSProp optimizer with default parameters is used. All models are trained on the non-annotated external audio collection described in Section 3.1 for a maximum of 200 epochs. The learning rate is set to 10^{-4} and dropout [24] is applied with probability 0.1 after each convolutional block and 0.2 after each fully-connected layer. All models [2] return embeddings of dimension $n = 128$.

[2] Code: `github.com/morgan76/HE`

3.4.2 Masks design:

In previous work, it was found beneficial to learn the masks during training to promote information sharing across similarity dimensions [12]. As in the method proposed by Lee et al. [11], we found that this did not bring any major improvement. Since information is already shared implicitly among the different hierarchy levels by the sampling strategy detailed in Section 2.1, the masks are kept disjoint from one another with equal length. After some preliminary experiments, the number of hierarchy levels $L = 4$ has been found as a good compromise between the diversity of triplets at each level and the temporal scale between the top and bottom ones.

3.4.3 Batch sampling scheme:

During training, mini-batches of size 120 are composed of 10 anchor points uniformly sampled from one song, and from which 12 triplets are derived (3 for each level). To choose good sampling parameters, we used both annotation levels of the held-out subset of SALAMI and measured the amount of true positive and true negative examples while varying $\delta_{p,min}$ and $\delta_{p,max}$ of level $\ell = 0$. It was found that setting $\delta_{p,min} = 32$ and $\delta_{p,max} = 64$ provided a good balance between the true positives rate at level $\ell = 0$ and the true negatives rate at level $\ell = 1$. For the case where $L = 4$, the rest of the parameters were set such that each level spans the same duration in beats (i.e. 16 beats) under the maximum value of 64 beats. All sampling parameters δ used for each level are summarized in Table 2.

L	ℓ	$\delta_{p,min}$	$\delta_{p,max}$	$\delta_{n,min}$	$\delta_{n,max}$
1	0	1	16	1	128
4	0	48	64	64	128
	1	32	48	48	64
	2	16	32	32	48
	3	1	16	16	32

Table 2. Sampling parameters (in beats) used in our experiments for $L = 1$ and $L = 4$ hierarchy levels.

3.5 Downstream algorithms and baselines

A common way of evaluating deep representations for music structure analysis is to measure the improvement made when combined with downstream segmentation methods. While there exists a variety music segmentation algorithms in the literature [1, 2], the one employed in these experiments was chosen to facilitate comparison against previous work. Boundary detection and section grouping on flat annotations as well as multi-level segmentation are performed with spectral clustering [9], as it remains the only unsupervised method that can output multiple levels of segmentation while being competitive. Additionally, it appears as a well-suited downstream method for hierarchical features since it operates on a graph decomposition of the audio signal. The proposed triplet sampling method forces the learned features to discriminate frames temporally close to one another at different levels in the hierarchy. Consequently, each sub-region in the embeddings learns one possible decomposition of the song. Applied on each of these sub-regions, spectral clustering can take advantage of the graph sub-structures proper to each level in order to efficiently retrieve the overall structure of the song. The original algorithm [9] takes two distinct audio features as input (MFCC and CQT), here, both features are replaced by the representations proposed in this work. Results obtained with the whole embedding matrices are denoted by HE (Hierarchical Embeddings). As an upper-bound of the proposed system, section grouping and multi-level segmentation are also performed using each individual sub-region of the embeddings (*i.e.* hierarchy levels), and the best results obtained across levels (denoted by HE_{best}) are reported. In a use case scenario, this can be seen as selecting the most adapted level of representation for each track in the testing set given a desired amount of granularity. For SALAMI, boundary detection is performed per annotator. For each, the scores obtained on both annotation levels (*upper* and *lower*) are computed both separately and combined together (best score between both levels per annotator is kept, noted *combined*). As an example, "$\text{HE}_{0,\text{best}}$" corresponds to the score obtained for the first annotator, selecting for each track the embedding level which maximizes the metric considered. In addition to results from previous work [5,9], those obtained here are compared against the method proposed by McCallum [3] (which comes down to setting $L = 1$ as described in Table 2), it is denoted as FE (Flat Embeddings).

4. RESULTS

4.1 Flat segmentation

Flat segmentation results on SALAMI are given in Table 3. The representations proposed in this work yield competitive results against the reported baselines on all the metrics considered. This trend is accentuated when the best embedding sub-region is selected. For *lower* annotations, the learned representations improve over traditional features. However, they do not perform better than those from McCallum [3], since this method uses sampling parameters that are more adapted to this level of annotation. The best-level scenario shows that the smallest temporal scales used during training (levels $\ell = 2, 3$) allow for the detection of very small regions of homogeneous timbral content, which helps detecting section changes at this level of annotation. The higher pairwise clustering scores indicate that these small detected regions are homogeneous enough to be identically labelled with spectral clustering (k-means step).

For the *upper* annotations, the results for boundary detection and pairwise clustering constantly improve over the reported baselines, indicating that for higher levels in the hierarchy, the proposed representations improve homogeneity inside annotated sections. Long-term similarities are implicitly captured by the highest embedding levels ($\ell = 0, 1$), yielding discriminative features able to separate consecutive musical sections at that level.

Finally, for both annotation levels combined, all models

perform better than when considering each level independently. The fact that difficult examples at the *lower* level are better managed at the *upper* one and vice-versa indicates that the representations learned are not specific to any particular annotation level. The very small performance gap across annotators also shows that these same representations capture relevant structure characteristics that are shared between them.

Level	Method	F_3	$F_{pairwise}$
lower	LSD [9]	0.525 ± 0.19	0.561 ± 0.16
	FE_0 [3]	0.624 ± 0.14	0.561 ± 0.14
	HE_0	0.611 ± 0.16	0.580 ± 0.15
	$HE_{0,best}$	0.643 ± 0.15	0.580 ± 0.15
	FE_1 [3]	0.611 ± 0.14	0.563 ± 0.14
	HE_1	0.600 ± 0.15	0.581 ± 0.14
	$HE_{1,best}$	0.635 ± 0.15	0.580 ± 0.14
upper	SNF [10]	0.456	0.567
	DEF [5]	0.564	0.600
	LSD [9]	0.579 ± 0.15	0.652 ± 0.13
	FE_0 [3]	0.568 ± 0.17	0.694 ± 0.14
	HE_0	0.597 ± 0.18	0.714 ± 0.14
	$HE_{0,best}$	0.627 ± 0.16	0.719 ± 0.14
	FE_1 [3]	0.559 ± 0.17	0.697 ± 0.14
	HE_1	0.595 ± 0.18	0.718 ± 0.14
	$HE_{1,best}$	0.625 ± 0.16	0.720 ± 0.14
combined	HE_0	0.665 ± 0.13	0.730 ± 0.14
	$HE_{0,best}$	0.711 ± 0.12	0.733 ± 0.14
	HE_1	0.662 ± 0.13	0.731 ± 0.14
	$HE_{1,best}$	0.707 ± 0.12	0.731 ± 0.14

Table 3. Boundary detection and section grouping results on SALAMI.

4.2 Multi-level segmentation

The results obtained for multi-level segmentation are reported in Table 4. When employing the full embedding representation, the performance on multi-level segmentation is competitive in terms of L-recall with previous work. As well as for boundary detection, selecting the best embedding sub-region leads to even further improvement. The importance of keeping inter-annotator agreement as a reference for comparison in multi-level segmentation has previously been argued [10]. It is found that the proposed representations result in multi-level segmentations that adapt to both annotators, within the range of the inter-annotator agreement reported by Tralie and McFee [10].

Method	L-precision	L-recall	L-measure
Inter-annot	0.664	0.664	0.654
LSD [7]	0.419	0.636	0.498
SNF [10]	0.431	0.668	0.517
DEF [5]	0.435	0.673	0.520
FE_0 [3]	0.412 ± 0.10	0.677 ± 0.13	0.505 ± 0.11
HE_0	0.413 ± 0.11	0.680 ± 0.13	0.507 ± 0.11
$HE_{0,best}$	0.432 ± 0.11	0.694 ± 0.13	0.527 ± 0.11
FE_1 [3]	0.413 ± 0.10	0.663 ± 0.12	0.503 ± 0.10
HE_1	0.418 ± 0.11	0.671 ± 0.13	0.509 ± 0.11
$HE_{1,best}$	0.423 ± 0.11	0.686 ± 0.13	0.517 ± 0.11

Table 4. Multi-level segmentation results on SALAMI. Inter-annot denotes the inter-annotator agreement.

4.3 Additional evaluation

In Table 5, the results obtained for boundary detection, section grouping and multi-level segmentation on additional datasets using the whole embedding matrices are summarized. The boundary detection scores obtained for BeatlesTUT and RWC-Pop fall within the same range, where more specifically, the score on RWC-Pop is higher than the one obtained by Wang et al. [4] with representations learned via supervised contrastive learning. However, a significant drop is observed for the two remaining datasets: RWC-Jazz and JAAH. If the music genre might play a role in this performance gap, it is also worth considering some statistics of these datasets summarized in Table 1. For RWC-Jazz, the high number of unique section labels compared to the total number of segments might cause some errors during the section grouping step done at the frame level with k-means (last step of the spectral clustering method). Regarding the JAAH dataset, given the low average number of segments per track, the segmentation method returns more boundaries than those originally annotated, therefore reducing the hit-rate precision and F-measure. For all metrics considered, other experiments have shown that hierarchical representations also performed better than their flat counterparts [3], of which due to space constraints, the results are not reported here.

Dataset	F_3	$F_{pairwise}$	L-P	L-R	L-M
BeatlesTUT	71.77	72.25	49.32	75.25	59.37
RWC-Pop	68.07	65.35	47.02	77.06	58.30
RWC-Jazz	55.05	58.51	32.89	81.80	45.76
JAAH	55.57	76.72	46.49	81.18	58.55

Table 5. Boundary detection, section grouping and multi-level segmentation results on additional datasets (in percentage) with the whole embedding matrix. L-P: L-precision, L-R: L-recall, L-M: L-measure.

The L-recall values obtained across datasets remain within the same range, regardless of the performance achieved on flat segmentation or section grouping. The temporal notion induced during sampling helps adapting to different musical genres or annotation sources. Even though the learned representations may not always fit with one specific level in the annotations, most of the reference structure hierarchies are captured and more refined levels of segmentation are discovered.

5. CONCLUSION

In this work, unsupervised contrastive learning of deep representations for music structure analysis at different time-scales has been explored. By leveraging time information and the hierarchical aspect of music structure, the resulting representations facilitate single and multi-level segmentation while being robust against different types of annotations. Future work includes searching for better-suited architectures to detect musical patterns at different time scales and automatically combine them to accommodate specific annotation styles or levels.

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*, 2020.

[2] O. Nieto and J. P. Bello, "Systematic exploration of computational music structure research." in *ISMIR*, 2016.

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

[4] J.-C. Wang, J. B. L. Smith, W. T. Lu, and X. Song, "Supervised metric learning for music structure features," in *ISMIR*, 2021.

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

[6] J. B. Smith and E. Chew, "Using quadratic programming to estimate feature relevance in structural analyses of music," in *Proceedings of the 21st ACM international conference on Multimedia*, 2013.

[7] B. McFee, O. Nieto, M. M. Farbood, and J. P. Bello, "Evaluating hierarchical structure in music annotations," *Frontiers in psychology*, 2017.

[8] 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.

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

[10] C. J. Tralie and B. McFee, "Enhanced hierarchical music structure annotations via feature level similarity fusion," in *ICASSP*, 2019.

[11] J. Lee, N. J. Bryan, J. Salamon, Z. Jin, and J. Nam, "Disentangled multidimensional metric learning for music similarity," in *ICASSP*, 2020.

[12] A. Veit, S. Belongie, and T. Karaletsos, "Conditional similarity networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[13] B. McFee and K. M. Kinnaird, "Improving structure evaluation through automatic hierarchy expansion," in *ISMIR*, 2019.

[14] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler, "Omras2 metadata project 2009," in *ISMIR*, 2009.

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

[16] V. Eremenko, E. Demirel, B. Bozkurt, and X. Serra, "Audio-aligned jazz harmony dataset for automatic chord transcription and corpus-based research." in *ISMIR*, 2018.

[17] 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.

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

[19] F. Kaiser and G. Peeters, "A simple fusion method of state and sequence segmentation for music structure discovery." in *ISMIR*, 2013.

[20] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015.

[21] D. P. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, 2007.

[22] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *ICML*, 2020.

[23] 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.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, 2014.

MULTI-INSTRUMENT MUSIC SYNTHESIS WITH SPECTROGRAM DIFFUSION

Curtis Hawthorne **Ian Simon** **Adam Roberts** **Neil Zeghidour**
Josh Gardner* **Ethan Manilow*** **Jesse Engel**
Google Research, Brain Team
{fjord,iansimon,adarob,neilz,jesseengel}@google.com
jpgard@cs.washington.edu, ethanm@u.northwestern.edu

ABSTRACT

An ideal music synthesizer should be both interactive and expressive, generating high-fidelity audio in realtime for arbitrary combinations of instruments and notes. Recent neural synthesizers have exhibited a tradeoff between domain-specific models that offer detailed control of only specific instruments, or raw waveform models that can train on any music but with minimal control and slow generation. In this work, we focus on a middle ground of neural synthesizers that can generate audio from MIDI sequences with arbitrary combinations of instruments in realtime. This enables training on a wide range of transcription datasets with a single model, which in turn offers note-level control of composition and instrumentation across a wide range of instruments. We use a simple two-stage process: MIDI to spectrograms with an encoder-decoder Transformer, then spectrograms to audio with a generative adversarial network (GAN) spectrogram inverter. We compare training the decoder as an autoregressive model and as a Denoising Diffusion Probabilistic Model (DDPM) and find that the DDPM approach is superior both qualitatively and as measured by audio reconstruction and Fréchet distance metrics. Given the interactivity and generality of this approach, we find this to be a promising first step towards interactive and expressive neural synthesis for arbitrary combinations of instruments and notes.

1. INTRODUCTION

Neural audio synthesis of music is a uniquely difficult problem due to the wide variety of instruments, playing styles, and acoustic environments. Among current approaches, there is often a trade-off between interactivity and generality. Interactive models, such as DDSP [3,4], offer realtime synthesis and fine-grained control, but only for specific types of instruments with domain-specific training information. On the other hand, models like Jukebox [5] are much more general and allow for training on any type of music, but are several orders of magnitude slower than realtime and offer more limited and global control, such as lyrics and global style (though with some early MIDI conditioning experiments).

Natural language generation and computer vision have seen dramatic recent progress through scaling generic encoder-decoder Transformer architectures [6, 7]. MT3 [8, 9] demonstrated that such an approach can be adapted to Automatic Music Transcription, transforming spectrograms to variable-length sequences of notes from arbitrary combinations of instruments. This generic approach enables training a single model on a wide variety of datasets: anything with paired audio and MIDI examples.

In this paper, we explore the inverse problem: finding a simple and general method to transform variable-length sequences of notes from arbitrary combinations of instruments into spectrograms. By pairing this model with a spectrogram inverter, we find that we are able to train on a wide variety of datasets and synthesize audio with note-level control over both composition and instrumentation.

To summarize, our contributions include:

- Demonstrating that the generic encoder-decoder Transformer approach used for multi-instrument transcription can likewise be adapted for multi-instrument audio synthesis.
- Realtime synthesis enabling interactive note-level control over both composition and instrumentation, by pairing a MIDI-to-spectrogram Transformer model with a GAN spectrogram inverter, and training on a wide range of paired audio/MIDI datasets (synthetic and real) with diverse instruments.
- Adapting DDPM decoding for arbitrary length synthesis without edge artifacts, through additional autoregressive conditioning on segments ($\sim$5 seconds) of previously generated audio.
- Quantitative metrics and qualitative examples demonstrating the advantages of segment-wise diffusion decoders over frame-wise autoregressive decoders for continuous spectrograms.

* Work done as a Google Brain Student Researcher.

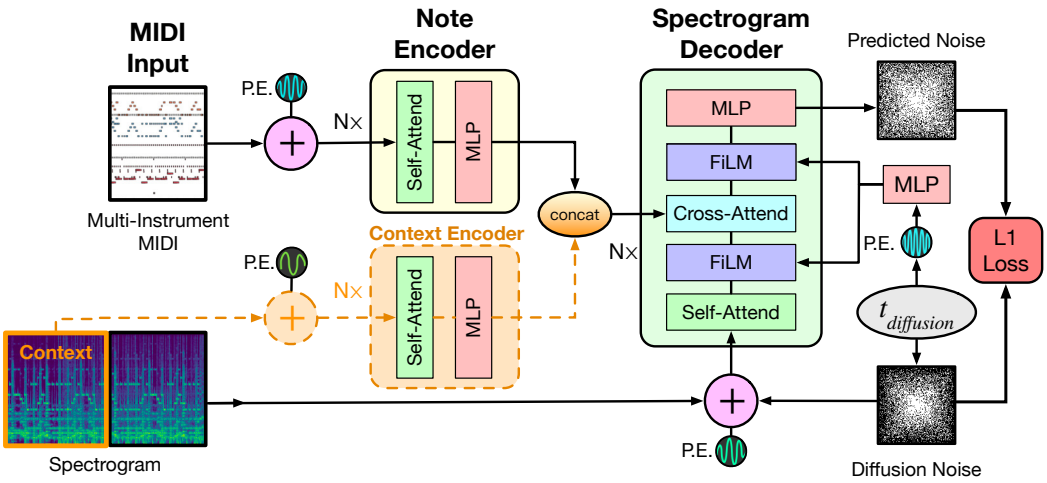

Figure 1. Training configuration for our spectrogram diffusion model. The architecture is an encoder-decoder Transformer that takes a sequence of note events as input and outputs a spectrogram. We train the decoder stack as a Denoising Diffusion Probabilistic Model (DDPM) [1], where the model learns to iteratively refine Gaussian noise into a target spectrogram (Figure 2). We generate ∼5 second spectrogram segments, and to ensure a smooth transition between these segments we (optionally) encode the previously generated segment in a second encoder stack. At inference time, a generated spectrogram is inverted to a waveform using a model similar to MelGAN [2]. "P.E." means Positional Encoding.

- Source code and pretrained models [1] .

2. RELATED WORK

Neural audio synthesis first proved feasible with autoregressive models of raw waveforms such as WaveNet and SampleRNN [10, 11]. These models were adapted to handle conditioning from latent variables [12, 13] or MIDI notes [14–17], but suffer from slow generation due to needing to run a forward pass for every sample of the waveform. Real-time synthesis often requires the use of specialized CPU and GPU kernels [18, 19]. These models also have limited temporal context due to the high sample rate of audio (e.g., 16kHz or 48kHz), where even the largest receptive fields (thousands of timesteps) can equate to less than a second of audio.

Approaches to overcome these speed limitations of waveform autoregression have focused on generating audio directly with a single forward pass. Architectures commonly either use GANs [20–23], controllable differentiable DSP elements such as oscillators and filters [3,4,24–27], or both [28, 29]. These models often focus on limited domains such as generating a single instrument/note/voice at a time [4, 15, 16, 30]. Here, we focus our search on architectures capable of both realtime synthesis, note-level control, and synthesizing mixtures of multiple polyphonic instruments at once.

Researchers have also overcome temporal context limitations of waveform autoregression by adopting a multi-stage approach, first creating coarser audio representations at a lower sample rate, and then modeling those representations with a predictive model before decoding back into audio. For example, Jukebox and Soundstream [5,31,32] use a Transformer to autoregresively model the discrete vector-quantized codes [33] of a base waveform autoencoder.

Tacotron architectures [34–36] have demonstrated that straightforward spectrograms can be an effective audio representation for multi-stage generation, first autoregressively generating continuous-valued spectrograms, and then synthesizing waveforms with a neural vocoder. The success of this approach led to a flurry of research into spectrogram inversion models, including streamlined waveform autoregression, GANs, normalizing flows, and Denoising Diffusion Probabilistic Models (DDPMs) [1, 2, 18, 37–39]. Our approach here is inspired by Tacotron. We use an autoregressive spectrogram generator paired with a GAN spectrogram inverter as a baseline, and further improve upon it with a DDPM spectrogram generator.

DDPMs and Score-based Generative Models (SGMs) have proven well-suited to generating audio representations and raw waveforms. Speech researchers have demonstrated high-fidelity spectrogram inversion [38, 39], text-to-speech [40, 41], upsampling [42], and voice conversion [43, 44]. Musical applications include singing synthesis of individual voices [45, 46], drums synthesis [47], improving the quality of music recordings [48], symbolic note generation [49], and unconditional piano generation [50]. Similar to singing synthesis, here we investigate DDPMs for spectrogram synthesis, but focusing on arbitrary numbers and combinations of instruments.

3. ARCHITECTURE

We approach the problem of audio synthesis using a two-stage system. The first stage consists of a model that produces a spectrogram given a sequence of MIDI-like note events representing any number of instruments. We then use a separate model to invert those spectrograms to au-

[1] https://github.com/magenta/music-spectrogram-diffusion