TO CATCH A CHORUS, VERSE, INTRO, OR ANYTHING ELSE: ANALYZING A SONG WITH STRUCTURAL FUNCTIONS

Ju-Chiang Wang[1], Yun-Ning Hung[2,], and Jordan B. L. Smith[1]*

[1] ByteDance
[2] Center for Music Technology, Georgia Institute of Technology, Atlanta, GA, USA
`ju-chiang.wang@bytedance.com, amy.hung@gatech.edu, jordan.smith@bytedance.com`

ABSTRACT

Conventional music structure analysis algorithms aim to divide a song into segments and to group them with abstract labels (e.g., 'A', 'B', and 'C'). However, explicitly identifying the function of each segment (e.g., 'verse' or 'chorus') is rarely attempted, but has many applications. We introduce a multi-task deep learning framework to model these structural semantic labels directly from audio by estimating "verseness," "chorusness," and so forth, as a function of time. We propose a 7-class taxonomy (i.e., intro, verse, chorus, bridge, outro, instrumental, and silence) and provide rules to consolidate annotations from four disparate datasets. We also propose to use a spectral-temporal Transformer-based model, called SpecTNT, which can be trained with an additional connectionist temporal localization (CTL) loss. In cross-dataset evaluations using four public datasets, we demonstrate the effectiveness of the SpecTNT model and CTL loss, and obtain strong results overall: the proposed system outperforms state-of-the-art chorus-detection and boundary-detection methods at detecting choruses and boundaries, respectively.

Index Terms— Music structure, segmentation, semantic labeling, Transformer, SpecTNT

1. INTRODUCTION

In Music Structure Analysis (MSA) of audio, two tasks are defined: *segmentation*, where the aim is to divide a recording of a song into non-overlapping segments, and *labeling* (or grouping), where the aim is to label the segments with symbols (e.g., 'A', 'B', etc.) to indicate how they are grouped. However, these bits of information give a limited description of a song. More advanced tasks include *hierarchical analysis*, in which segmentation and grouping must be estimated at multiple timescales, and *semantic labeling*, in which the labels should have meaning, such as 'chorus', 'verse', or 'solo'. While hierarchical MSA has attracted sustained attention [1, 2, 3, 4, 5], semantic MSA, despite having many uses, has rarely been addressed: the most significant work on this task was over a decade ago [6, 7].

Early MSA algorithms (see [8] for a review) were mostly non-supervised, based on processing some version of a self-similarity matrix computed from traditional audio features such as MFCCs or chroma. However, as large annotated datasets became available (e.g., *SALAMI* [9] and *Harmonix Set* [10]), supervised approaches benefitted. For example, the usual way to estimate segment boundaries was once to compute a 'novelty function' and choose the points that maximized it [11]; Ullrich et al. used a subset of [9] to train a model to estimate the 'boundaryness' of each instant given a long context window, and achieved a new state-of-the-art [12].

Similarly, Wang et al. [13] trained a model to estimate 'chorusness' (jointly with boundaries), and achieved state-of-the-art *chorus detection*. (This is another sub-task of MSA, and an early work on this topic [14] is alluded to by our title.) In this paper, we extend [13] and aim to model 'verseness', 'bridgeness', and more, directly from audio. Our system represents the first that is able to assign generic structural semantic labels since [6], and the first in which the audio content directly informs the label prediction.

We build on [13] by introducing two deep neural network (DNN) architectures to MSA, *Harmonic-CNN* [15] and *SpecTNT* (Spectral-Temporal Transformer in Transformer) [16]. Transformer architectures have not been used yet for MSA, but have the potential to improve the temporal modeling at the representation learning stage; they have earned great attention for their strength in modeling sequential data [17, 18]. In particular, we adopt SpecTNT, a hierarchical variant of Transformer that models the spectral and temporal dimensions of an input spectrogram with two Transformer encoders. We also propose to use the Connectionist Temporal Localization (CTL) loss [19] to improve the temporal modeling.

A system capable of estimating semantic labels has clear uses: it would enhance any application that already relies on MSA, such as structure-based navigation systems [20] or automated remixing of sections from different songs [21, 22]. Furthermore, the ability to analyze structure without requiring full context of a song allows MSA to be applied to song fragments, or songs that are very short (e.g., a TikTok song under 30 seconds long). For such inputs, approaches that rely on self-similarity matrices and clustering may fail. An automatic mastering system [23] could be improved by predicting the function and structure of the input clips. Finally, a system not dependent on a full song would be a first step toward a *real-time* MSA system. Such a system could be essential to applications in live contexts, such as a system that controlled the visuals at a concert to match the music (as suggested in [8]).

To sum up, our proposed MSA system differs from most previous work in three ways: it predicts semantic (not abstract) labels; it is supervised; and it is non-context-dependent, meaning it does not rely on detecting repeating sequences or relative novelty changes. The main contributions of this paper are: a mapping used to consolidate information from disparate datasets (Sec. 2); the proposed approaches, including a system that represents the first application of Transformers to MSA (Sec. 3); and the evaluations, including ablation and cross-dataset studies using four public datasets (Sec. 4).

2. STRUCTURAL LABEL CONVERSION

Structural analysis is recognized as an "ill-defined" problem [8], since the "solution" is not unique: listeners usually disagree about

---

*The author conducted this work as an intern at ByteDance.

the exact structure of a piece. The SALAMI dataset was designed to mitigate this: section groupings were annotated separately from function (inspired by [24]), and an "Annotator's Guide" [25] defined a valid set of function labels. The data still contain many instances of conflicting labels, including dozens of cases where one listener's 'chorus' was another's 'verse'. However, there is far more agreement than disagreement: if two annotators agree on the start time of a section, they agree on the function label at least twice as often as they disagree, and this rate only increases if we collapse synonymous groups of labels: e.g., the Annotator's Guide indicates 'coda', 'fade-out,' and 'outro' all have the same basic function. Therefore, we expect the data created by annotators to be useful for this task.

To train a model to predict functions, we must strike a balance between our goal—to model a rich set of semantic meanings—with our available means: a motley set of datasets, collected with different standards, made up of different genres, with functional terms that vary in specificity. Based on our study of existing datasets, there is no standard naming system for structural labels. For example, some use 'refrain' as chorus, some specify section sub-types (e.g., 'verse A' in RWC [26]), and some allow compound functions (e.g., 'instchorus', appearing in Harmonix Set). Before developing a model to classify structural functions, we have to define a fixed taxonomy, and then define rules to map the free-form annotations from several datasets onto the same taxonomy. Any choice here will be a compromise among the utility of the taxonomy, the amount of training data per class, and the validity of the mappings.

We chose a 7-class taxonomy that is suited to Western pop music: 'intro', 'verse', 'chorus', 'bridge', 'inst' (i.e., instrumental), 'outro', and an auxiliary class 'silence'. This is equivalent to the set of 5 basic classes named in the SALAMI Annotator's Guide, plus 'inst' and 'silence'. We then define Algorithm 1, which gives a mapping for any annotated label that does not match the 7-class taxonomy. The function `conversion()` iteratively tests for the existence of substrings in the label. Since multiple substrings can appear in one label, the order of elements in `substrings` determines the priority of the matches; for instance, 'instrumentalverse' will be converted to 'verse', not 'inst', and 'pre-chorus' will be converted to 'verse' to disentangle the build from the true chorus (as in [13]). Note that `"end"` is used to mark the timestamp for the end of a song, and it is not regarded as a function label. We found that `substrings` covers 99.3% of raw labels in the existing datasets. The remaining labels mostly include instrument descriptors (such as 'guitar,' 'gtr,' 'riff,' 'spoken,' 'voice,' and 'groove'); after listening to most examples, we decided to convert them all to `"inst"`.

3. PROPOSED APPROACH

Our proposed system pipeline is depicted in Fig. 1, and is conceptually similar to the pipeline proposed in [13]. A song is first divided into overlapping audio chunks. After audio feature extraction, the DNN model predicts, for each chunk, two types of activation curves: one for boundaries and one for each structural function. We propose two alternative strategies, *instant* and *multi-point*, for the DNN model. In the *instant* model, each chunk leads to a single prediction vector indicating the likelihood of each function label for the instant at the center of the chunk (as in [12]). Then, the sequence of predictions forms the output curves. In the *multi-point* model, predictions are made for every time point of an input chunk. We follow the method of [13] to merge the outputs of all the overlapping chunks into the final prediction curves. Finally, a few simple post-processing steps return the output is a usable format: boundary timestamps and a single function label per segment.

```
substrings = [
  ("silence", "silence"), ("pre-chorus", "verse"),
  ("prechorus", "verse"),  ("refrain", "chorus"),
  ("chorus", "chorus"),    ("theme", "chorus"),
  ("stutter", "chorus"),   ("verse", "verse"),
  ("rap", "verse"),        ("section", "verse"),
  ("slow", "verse"),       ("build", "verse"),
  ("dialog", "verse"),     ("intro", "intro"),
  ("fadein", "intro"),     ("opening", "intro"),
  ("bridge", "bridge"),    ("trans", "bridge"),
  ("out", "outro"),        ("coda", "outro"),
  ("ending", "outro"),     ("break", "inst"),
  ("inst", "inst"),        ("interlude", "inst"),
  ("impro", "inst"),       ("solo", "inst")]

def conversion(label):
    if label == "end":  return "end"
    for s1, s2 in substrings:
        if s1 in label.lower():  return s2
    return "inst"
```
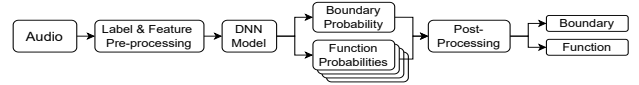
Algorithm 1. The conversion rule for a raw label.

Fig. 1. Pipeline for the proposed system.

3.1. Label and Audio Feature Pre-processing

The steps to create a *chorus activation curve*, the target curve for the prediction of chorus segments, are given in [13]. We use the same steps, but repeat them for the 6 additional function classes to create 7 *function activation curves*. There is still only one *boundary activation curve*, but it now includes all boundaries (not just those between chorus and non-chorus segments), as described in [12].

We smooth the transitions of the function activation curves using a 2-second-wide Hann window: a 1-second ramp from 0 to 1 prior to the onset, and a 1-second ramp down after the offset, as in [13]. Regarding the "boundary section," we set a duration of 0.6 seconds for each boundary (whereas [13] used 0.5 seconds).

As for audio feature extraction, we use harmonic representation (whereas [13] used mel-spectrogram), since trainable harmonic filters help capture harmonic information while preserving spectral-temporal locality. It has proven useful in music auto-tagging [15].

3.2. Harmonic-CNN

Harmonic-CNN [15] is designed to mimic human perception by extracting the spectrogram features through its harmonic representation front-end, as harmonic structure is known to play a key role in the human auditory system. Its network architecture contains several temporal pooling layers with the aim of predicting a single target for each music tag, so we believe it can be a good fit to the goal of the *instant* model in this work. Given the harmonic representation input, it applies seven 2D-convolutional layers in the back-end to extract high-level features, followed by two dense layers to predict the target of a time-step for the boundary and function activations.

3.3. Spectral-Temporal Transformer

Although the Transformer architecture has shown remarkable performance in modeling sequential data such as text [18] and musical scores [27, 28], its data-hungry nature makes it inapt for tasks with little training data. The total number of current annotated songs for
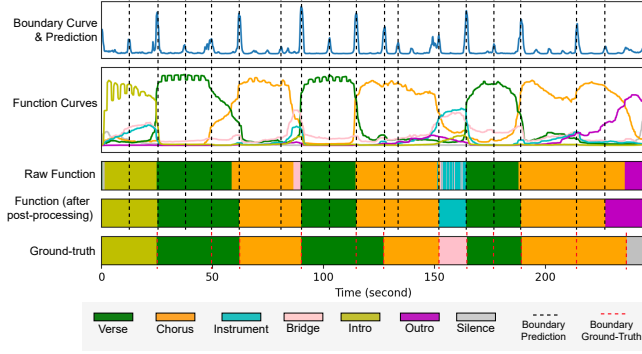
Fig. 2. Example of post-processing for a test song *Complicated* by Avril Lavigne, using the SpecTNT (24s CTL) settings. The top two rows display the raw outputs of the activation curves. "Raw Function" shows the argmax labels at each time-step of the DNN output.

public MSA datasets is less than 3000, which is likely insufficient to train a satisfactory Transformer model.

By contrast, SpecTNT (adopted in this work) has shown good performance in beat and downbeat tracking [29], vocal melody extraction [16], and chord recognition [16] using datasets even smaller than those for MSA. The basic principle of SpecTNT arises from the interaction between two levels of Transformer encoders, namely a *spectral encoder* and a *temporal encoder*. The spectral encoder is responsible for extracting the spectral features via *Frequency Class Tokens* (FCTs) for each time-step, where an FCT is an aggregated embedding that characterizes harmonic and timbral information. The temporal encoder then exchanges local information (i.e., FCTs) along the time axis; this self-attention step can help discover structural patterns related to novelty, homogeneity, and repetition [8]. For example, the self-attention mechanism can allow frames around a boundary to attend to the boundary, and frames with the same function to attend to one another. Owing to its hierarchical design, SpecTNT permits a smaller number of parameters as compared to the original Transformer [17], and we expect this attribute can help improve the generalization for the MSA task.

A SpecTNT model consists of three modules: a two-dimensional residual network (ResNet) [30] at the front-end to extract intermediate information from the input harmonic representation; a stack of SpecTNT blocks; and a linear layer to output the target probability at all time-steps. Therefore, it serves as the *multi-point* model in this work. For ResNet, each convolutional layer uses a kernel size of 3. Then, we use 5 SpecTNT blocks. In each block, we use 96 feature maps with 4 attention heads for the spectral encoder, and 96 feature maps with 8 attention heads for the temporal encoder.

3.4. Post-processing

The raw outputs include a boundary curve and 7 structural function curves. In an end-to-end manner, one can assign the structural function with the largest probability (i.e., arg-max) for each time-step. However, as illustrated in Fig. 2, this 'raw function' output can be inaccurate near boundaries and inconsistent within a segment. To get more usable output, we first interpret the boundary activation curve using the peak-picking method proposed in [12]. Then, we simply choose the function labels with the largest *average* probability in each segment. After this post-processing, we can see a more accurate result (see Fig. 2).

3.5. Training Loss

To jointly model the boundaries and structural functions, we define two types of losses: *boundary loss* and *function loss*. Each is calculated by summing the weighted binary cross-entropy between the prediction and target at each time-step of the corresponding activation curve. Following the heuristic used in [13], we use a weight 0.9 for boundary loss and 0.1 for function loss when combining them, since the boundary curve is sparse and more difficult to learn.

To enhance the coherence of the predictions and reduce the fragmentation, we propose to add the *Connectionist Temporal Localization (CTL) loss* [19] to the function loss. The CTL loss aims to model the sequential order of labels. In popular music, the order of sections follows regular patterns: a song is unlikely to start with an outro or end with an intro—this is the premise of [7]. We use the raw section annotations (i.e., before they are converted to the activation curves) to form the target sequence of section tokens (e.g., ['intro', 'verse', 'chorus', 'inst']). Then, the CTL loss is calculated between the prediction (a $T \times 7$ probability matrix) and the sequence of S tokens, where T and S are the numbers of time-steps and section tokens, respectively, in a training chunk.

4. EXPERIMENTS

4.1. Implementation Details

For STFT before the harmonic representation, we adopt a window length of 1024 with a hop size of 512 on audio signals of 16 kHz sampling rate. The time resolution for the activation curves is about 5.2 per second (i.e., a target per 0.192 seconds). We found using longer duration for the input chunks led to better function prediction, but poorer boundary detection accuracy. Thus, we adopt 24-second chunks and will show result of using 36 seconds in an ablation study.

Data augmentation is vital to training an effective Transformer model. We use the `torchaudio_augmentations` package [31] to randomly add noise, adjust gain, or apply high/low-pass filters for each training sample. We use a random sampling strategy to load a random chunk into a batch, instead of sequentially loading a chunk from the beginning of a training song as adopted in [13]. That is, we first enumerate all the valid training chunks into a list by using a 24-second sliding window with a 3-second hop on every song. If the last chunk exceeds the end of a song, we pad zeros at its end. Then, the data loader draws a chunk uniformly from the list during training. Therefore, a batch can include chunks from different locations of different songs. This technique can lead to faster convergence and better results compared to that in [13].

We use PyTorch 1.8 and Adam optimizer [32] with 0.0005 learning rate, 0.9 weight decay, and 2 epochs of patience. An epoch runs 500 mini-batches with a batch size of 128. We conduct 100 training epochs and use the model with the best validation result for testing.

4.2. Experimental Configurations

We use four public datasets for experiments: *Harmonix Set* [10], *SALAMI-pop* [9], *RWC-Pop* [26], and *Isophonics* [33]. *Harmonix Set* contains 912 western pop songs. Since the original audio was not available, we searched for the correct audio versions and manually refined the annotations. For *SALAMI-pop*, we select a subset of 274 songs (with 445 annotations) in the "popular" genre. In these annotations, the label 'no_function' appears often, but this is the result of a parsing error; we replace every instance of it with the preceding section's function label. *RWC-Pop* is the subset of 100 popular songs in *RWC*. We use the original annotations by AIST. *Isophonics*

contains 277 songs from The Beatles, Carole King, Michael Jackson, and Queen, but we use the *TUT-Beatles* annotations [34] for the 174 Beatles songs.

We conduct evaluations of two types. First, in the ablation study, we use *Harmonix Set* in a 4-fold cross-validation (CV) manner, but include *SALAMI-pop*, *RWC-Pop*, and *Isophonics* in the training set at every iteration. Second, we carry out cross-dataset evaluations: each of *SALAMI-pop*, *RWC-Pop*, and *Isophonics* in turn serves as the test set, and the remaining datasets are used for training. We randomly split 10% of the training set to form the validation set.

Besides regular MSA, we are also interested in the performance of chorus detection, where we only leave the boundary and section annotations involving a 'chorus.' The mir_eval package [35] is used to compute the following evaluation metrics: (1) *HR.5F*: F-measure of hit rate at 0.5 seconds; (2) Accuracy (*ACC*): the frame-wise accuracy between the predicted function label and the converted ground-truth label; (3) *PWF*: F-measure of pair-wise frame clustering; (4) *Sf*: F-measure of normalized entropy score; (5) *CHR.5F*: F-measure of 'chorus' boundary hit rate at 0.5 seconds; (6) *CF1*: F-measure of pair-wise frames for 'chorus' and 'non-chorus' sections [13]. The definitions of (1), (3), (4), and (5) can be found in [8].

4.3. Baseline Methods

We include several baseline methods from prior work. "Scluster" is the spectral clustering algorithm [1] implemented in MSAF [36]. "DSF + Scluster" uses learned deep structure features (DSF) [37] as the inputs for Scluster to predict the MSA results. It employs a Harmonic-CNN trained with metric learning loss in a supervised fashion, and is trained only on *Harmonix Set*. To perform chorus detection for the above two methods, we pick chorus sections using the heuristic in [13], i.e., *Max-dur*, which chooses the segment group that covers the greatest duration of a song as the choruses. "CNN-Chorus" [13] uses a similar approach to this work, but with a different DNN model and training strategy. It was developed to only detect the chorus sections. For *RWC-Pop* and *Isophonics*, which are used in MIREX, we include for comparison the results of three top-performing MIREX submissions: GS3 [3], and SMGA1 and SMGA2 [38].

5. DISCUSSION AND CONCLUSION

Results are presented in Table 1. The main proposed systems are "Harmonic-CNN" (the *instant* method) and "SpecTNT (24s, CLT)" with CTL loss (the *multi-point* method). In the ablation study, we compare these to SpecTNT without CTL; a regular Transformer [39] with CTL; and SpecTNT with CTL and a longer chunk size (36s). The regular Transformer contains only the temporal encoders [18], so it is non-hierarchical and has no spectral encoder. It is clear that this system performs worse than SpecTNT, most likely because the training data is insufficient. Comparing SpecTNT with and without CTL loss, we can see that including the loss improves performance slightly on all metrics, demonstrating its usefulness. We also observe that SpecTNT with longer inputs can improve the function prediction, but the boundary detection performance drops, possibly because the model has over-fit the training data. To prioritize boundary accuracy, we use the shorter window in the next evaluation.

Looking at all the studies, we observe that SpecTNT (24s, CTL) consistently outperforms Harmonic-CNN in all metrics; i.e., the *multi-point* model beats the *instant* model. This may be due to improved temporal coherence in the estimates provided by SpecTNT's hierarchical Transformer. Either way, the multi-point method used

	HR.5F	ACC	PWF	Sf	CHR.5F	CF1
Ablation Study						
Harmonix Set						
Scluster [1]	.263	-	.586	.641	.171	.534
DSF + Scluster [37]	.497	-	.689	**.743**	.326	.611
CNN-Chorus [13]	-	-	-	-	.371	.692
Harmonic-CNN	.559	.680	.670	.682	.462	.784
Transformer (24s, CTL)	.521	.640	.655	.649	.399	.755
SpecTNT (24s)	.565	.690	.687	.702	.491	.813
SpecTNT (24s, CTL)	**.570**	.701	.700	.714	**.501**	.815
SpecTNT (36s, CTL)	.558	**.723**	**.712**	.724	.476	**.831**
Cross-dataset Evaluation						
SALAMI-pop (subset of MIREX 2012 dataset)						
Scluster [1]	.305	-	.545	.572	.196	.418
DSF + Scluster [37]	.447	-	.615	**.653**	.272	.573
CNN-Chorus [13]	-	-	-	-	.308	.602
Harmonic-CNN	.477	.525	.631	.636	.340	.777
SpecTNT (24s, CTL)	**.490**	**.544**	**.651**	.632	**.357**	**.811**
RWC-Pop (MIREX 2010 RWC collection)						
GS3 (2015) [3]	.524	-	.542	.684	-	-
SMGA2 (2012) [38]	.246	-	.688	.733	-	-
DSF + Scluster [37]	.438	-	.704	**.739**	.343	.653
Harmonic-CNN	.571	.646	.719	.694	.396	.800
SpecTNT (24s, CTL)	**.623**	**.675**	**.749**	.728	**.465**	**.847**
Isophonics (MIREX 2009 Collection)						
GS3 (2015) [3]	.564	-	.567	.686	-	-
SMGA1 (2012) [38]	.228	-	**.653**	**.700**	-	-
Harmonic-CNN	.543	.499	.611	.598	.339	.670
SpecTNT (24s, CTL)	**.590**	**.550**	.635	.614	**.401**	**.733**

Table 1. Results on four datasets.

with SpecTNT is 15.6 times more efficient at prediction time than the instant method required by Harmonic-CNN, since the former requires 1 call to the model every 3 seconds (as a chunk hops every 3 sec), whereas the latter requires 5.2 calls per second (i.e., time resolution). The *multi-point* method also significantly outperforms the Scluster-based methods at boundary detection (i.e., HR.5F and CHR.5F), showing that directly modeling the boundaries with an activation curve is better than doing feature clustering. In addition, our *multi-point* model can outperform the state-of-the-art boundary detection approach (i.e., a CNN-based *instant* model, Harmonic-CNN, or GS3) by a wide margin, demonstrating the strength of SpecTNT for boundary detection. We also find that modeling more function types may help the system to recognize chorusness more accurately: Harmonic-CNN outperforms its predecessor CNN-Chorus (which only saw chorus and non-chorus labels) at chorus detection.

Our proposed approach is less successful in terms of PWF and Sf on *Isophonics*. A closer investigation reveals that there are relatively few 'chorus' (or 'refrain') labels in the songs of the Beatles, which is a classic evaluation corpus for MSA, but contains many songs with experimental structure. The mismatch between the Beatles songs and the remainder of the datasets, which have more choruses, may account for this poorer performance. However, inspection of other outputs shows that our model often makes justifiable errors: for example, it often predicts an 'outro' when the annotation says 'chorus', but when the song is in fact fading out, like a 'fade-out chorus'. Fig. 2 shows such an example, but also hints at a solution: allowing multiple labels to be potentially predicted per segment.

We have proposed a Transformer-based MSA system that can predict meaningful segment labels, even without context. However, in this work we only study the performance of full-track prediction; to demonstrate its robustness to song fragments remains future work.

6. REFERENCES

[1] B. McFee and D. Ellis, "Analyzing song structure with spectral clustering," in *ISMIR*, 2014, pp. 405–410.

[2] B. McFee, O. Nieto, and J. P. Bello, "Hierarchical evaluation of segment boundary detection.," in *ISMIR*, 2015, pp. 406–412.

[3] T. Grill and J. Schlüter, "Music boundary detection using neural networks on combined features and two-level annotations.," in *ISMIR*, 2015, pp. 531–537.

[4] C. J. Tralie and B. McFee, "Enhanced hierarchical music structure annotations via feature level similarity fusion," in *ICASSP*, 2019, pp. 201–205.

[5] J. Salamon, O. Nieto, and N. J. Bryan, "Deep embeddings and section fusion improve music segmentation," in *ISMIR*, 2021, pp. 594–601.

[6] J. Paulus, *Signal processing methods for drum transcription and music structure analysis*, Ph.D. thesis, Tampere University of Technology, Tampere, Finland, 2009.

[7] J. Paulus and A. Klapuri, "Labelling the structural parts of a music piece with Markov models," in *Computer Music Modeling and Retrieval: Genesis of Meaning in Sound and Music*, vol. 5493, pp. 166–176. Springer-Verlag, 2010.

[8] O. Nieto, G. J. Mysore, C.-I Wang, J. B. L. Smith, J. Schlüter, T. Grill, and B. McFee, "Audio-based music structure analysis: Current trends, open challenges, and applications," *Trans. ISMIR*, vol. 3, no. 1, 2020.

[9] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. D. Roure, and J. S. Downie, "Design and creation of a large-scale database of structural annotations," in *ISMIR*, 2011, pp. 555–560.

[10] O. Nieto, M. McCallum, M. Davies, A. Robertson, A. Stark, and E. Egozy, "The Harmonix Set: Beats, downbeats, and functional segment annotations of western popular music," in *ISMIR*, 2019, pp. 565–572.

[11] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *IEEE ICME*, 2000, pp. 452–455.

[12] K. Ullrich, J. Schlüter, and T. Grill, "Boundary detection in music structure analysis using convolutional neural networks," in *ISMIR*, 2014, pp. 417–422.

[13] J.-C. Wang, J. B. L. Smith, J. Chen, X. Song, and Y. Wang, "Supervised chorus detection for popular music using convolutional neural network and multi-task learning," in *ICASSP*, 2021, pp. 566–570.

[14] M. A. Bartsch and G. H. Wakefield, "To catch a chorus: Using chroma-based representations for audio thumbnailing," in *Proc. WASPAA*, 2001, pp. 15–18.

[15] M. Won, S. Chun, O. Nieto, and X. Serra, "Data-driven harmonic filters for audio representation learning," in *ICASSP*, 2020, pp. 536–540.

[16] W.-T. Lu, J.-C. Wang, M. Won, K. Choi, and X. Song, "SpecTNT: A time-frequency transformer for music audio," in *ISMIR*, 2021, pp. 396–403.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[19] Y. Wang and F. Metze, "Connectionist temporal localization for sound event detection with sequential labeling," in *ICASSP*, 2019, pp. 745–749.

[20] M. Goto, "SmartMusicKIOSK: Music listening station with chorus-search function," in *ACM Symposium on User Interface Software and Technology*, 2003, pp. 31–40.

[21] M. E. P. Davies, P. Hamel, K. Yoshii, and M. Goto, "AutoMashUpper: Automatic creation of multi-song music mashups," *IEEE Tran. Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1726–1737, 2014.

[22] J. Huang, J.-C. Wang, J. B. L. Smith, X. Song, and Y. Wang, "Modeling the compatibility of stem tracks to generate music mashups," *AAAI Conference*, 2021.

[23] B. De Man, R. Stables, and J. D. Reiss, *Intelligent Music Production*, Focal Press, 2019.

[24] G. Peeters and E. Deruty, "Is music structure annotation multi-dimensional? A proposal for robust local music annotation," in *Int. Workshop on LSAS*, 2009, pp. 75–90.

[25] SALAMI, "Annotator's guide," https://github.com/DDMAL/salami-data-public, Last accessed on Feb 11, 2022.

[26] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, classical and jazz music databases," in *ISMIR*, 2002, vol. 2.

[27] C.-Z. A. Huang et al., "Music Transformer: Generating music with long-term structure," in *ICLR*, 2018.

[28] M. Zeng, X. Tan, R. Wang, Z. Ju, T. Qin, and T.-Y. Liu, "MusicBERT: Symbolic music understanding with large-scale pre-training," in *Findings of ACL*, 2021, pp. 791–800.

[29] Y.-N. Hung, J.-C. Wang, X. Song, W.-T. Lu, and M. Won, "Modeling beats and downbeats with a time-frequency Transformer," *ICASSP*, 2022.

[30] M. Won, A. Ferraro, D. Bogdanov, and X. Serra, "Evaluation of CNN-based automatic music tagging models," in *Sound and Music Computing Conference*, 2020.

[31] J. Spijkervet, "Spijkervet/torchaudio-augmentations," 2021, https://zenodo.org/record/4748582.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[33] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler, "OMRAS2 metadata project 2009," in *ISMIR Late Breaking and Demo*, 2009.

[34] J. Paulus, "Improving Markov model based music piece structure labelling with acoustic information," in *ISMIR*, 2010, pp. 303–308.

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

[36] O. Nieto and J. P. Bello, "Systematic exploration of computational music structure research," in *ISMIR*, 2016, pp. 547–553.

[37] J.-C. Wang, J. B. L. Smith, W.-T. Lu, and X. Song, "Supervised metric learning for music structure features," in *ISMIR*, 2021, pp. 730–737.

[38] J. Serra, M. Müller, P. Grosche, and J. L. Arcos, "The importance of detecting boundaries in music structure annotation," *Proc. MIREX*, 2012.

[39] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," in *ISMIR*, 2021, pp. 769–776.