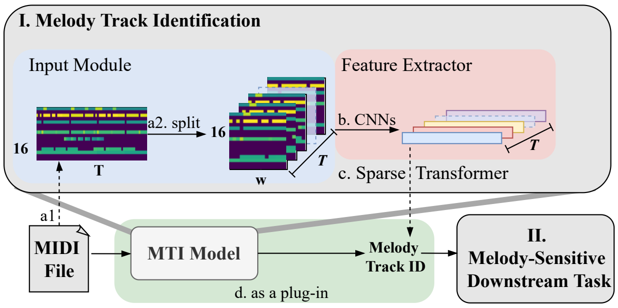


Figure 1. Dataflow of our method. (a) MIDI quantization and split. (b) CNN feature extractor. (c) Global information aggregator. (d) Plug in downstream tasks.

$tr_i = [e_0, e_1, ..., e_M]$ where $e_j \in [0, 127]$. Given an N-track MIDI file $\mathcal{S}$, our task is to predict the ID y^i of one single melody track, formally, $Y = \{y^i|\mathcal{S}\}$. Since a MIDI file has at most 16 sound output channels and one channel can contain several tracks, our MTI model can be seen as a supervised 16-way multirun classifier. When the melody track occupies an entire channel, the initial prediction result is the melody track ID; When the melody track shares a channel with tracks of other musical components, the model reruns a second prediction to further distinguish the melody track from the initially identified channel. Therefore, the prediction process is definitively convergent in finite time.

As shown in Figure 1, the data flow of our proposed robust MTI model is divided into several modules. First, an input MIDI file goes through the input module for preprocessing: it is quantized and serialized as a matrix with a resolution of sixteenth note (Figure 1-a1). Then the matrix is split into overlapped frames with step-wise scanning (Figure 1-a2). Second, a frame level CNN extracts the local features of a frame (Figure 1-b) and a sparse Transformer based classification module predicts the melody track ID based on the aggregated global features (Figure 1-c). Taking in a MIDI and outputting its melody track ID, the whole MTI module (Figure 1-I) can serve as a preprocessing plug-in for other melody-sensitive MIR tasks (Figure 1-II).

3.2 Input Module

Our input module converts an input MIDI file $\mathcal{S}$ into a 2D matrix $\boldsymbol{M} = [m_{ct}]_{C\times T}$, where m_{ct} denotes the pitch value in the c^{th} channel at the t^{th} time step, and T denotes the total length of the MIDI. One time step corresponds to a sixteenth-note length so that the encoding is tempo independent. If there are multiple tracks containing different pitch notes in the same channel c at time t, m_{ct} is temporarily set to the highest pitch value. The module also ensures that all input MIDI files have a full sixteen channels by padding any files with channels of zeros. This encoding provides following advantages against the piano roll representation: 1) It has a fixed dimension of channels; 2) It omits the velocity which can confuse the MTI model; 3) The tracks sharing the same channel can be temporarily fused into one representative pitch sequence and further distinguished in the rerun of the model later.

Inspired by Simonetta et al. [2], we apply step-wise scanning to split the 2D representation of a MIDI file into overlapping frames of fixed window size, w, equivalent to one bar. Each resultant frame is thus a $w \times w$ matrix $\boldsymbol{m}^{(i)}_{t-l:t+w-l}$, where $\boldsymbol{m}^{(i)}_{t-l:t+w-l}$ represents the frame centers around the t^{th} time step in the i^{th} MIDI file, and l represents the preceding l time steps.

3.3 Architecture Design

We propose a CNN + Sparse Transformer structure to learn local context information of each frame while also aggregating global information over the entire sequence. We also propose to overcome the memory bottleneck that a vanilla Transformer imposes. This enables the system to efficiently process longer music by reducing the density of self-attention connections. We adopt a hierarchical training strategy: First, we pretrain the CNN-based feature extractor by predicting the melody track ID at frame level; then, we fine-tune the whole model by predicting a single melody track ID of the input MIDI.

3.3.1 Local Context-Aware Feature Extraction

We use a CNN as our feature extractor to learn the local context information of given frames provided by the input module. The architecture for this portion of the system is shown in Figure 2. The feature extractor ϕ first goes through pretraining. Specifically, we add a temporary MLP layer with weights $\boldsymbol{W}$ and bias b to each frame to predict its fame-level melody track ID. The predicted probability for a certain frame $\boldsymbol{m}^{(i)}_{t-l:t+w-l}$ can be denoted as

$$\hat{y} = \text{softmax}(\boldsymbol{W}\phi(\boldsymbol{m}^{(i)}_{t-l:t+w-l}) + b) \tag{1}$$

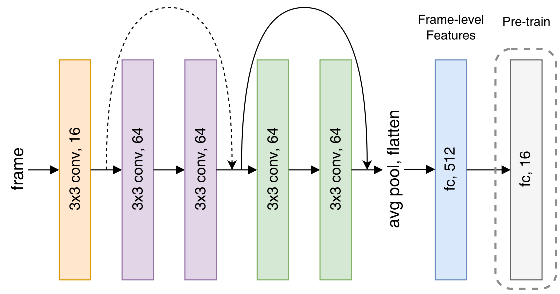


Figure 2. The architecture of our feature extractor. Shortcuts represent skipped paths. Dotted shortcut represents dimension increases. The last fully connected layer (grey) is only used in pre-training.

3.3.2 Global Classification Module

With the pre-trained feature extractor ϕ mentioned above, each MIDI file can now be transformed and concatenated into a context-aware embedding

$$\boldsymbol{M}^{(i)} = \mathop{\Big|\Big|}_{t\in[1,T]} \phi(\boldsymbol{m}^{(i)}_{t-l:t+w-l}) \tag{2}$$

where $\boldsymbol{M}^{(i)} \in \mathbb{R}^{T\times d}$, $||$ denotes concatenation over time steps, and d denotes the dimension of feature extractor ϕ.

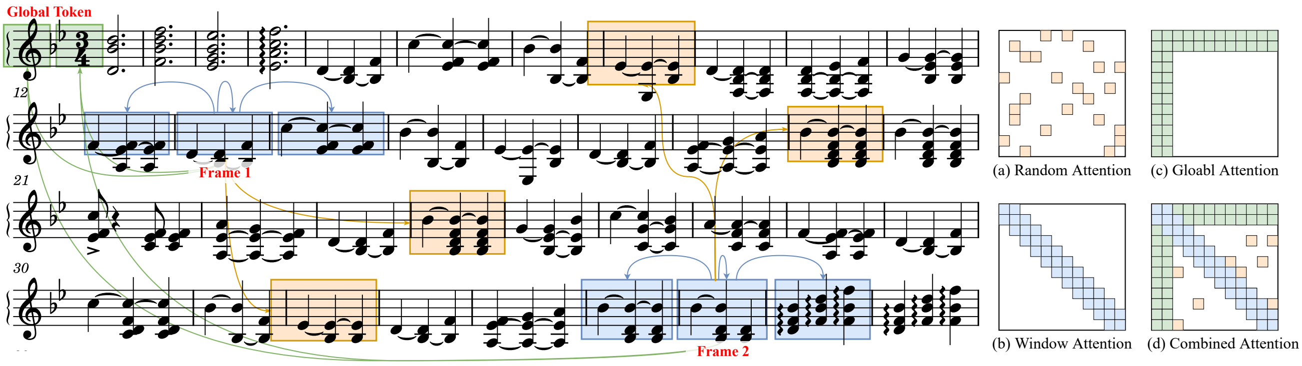


Figure 3. Left: an example of a sparse self-attention mechanism adapted in symbolic music, where frames 1 and 2 attend to three neighboring frames (blue), two random frames (orange) and a global frame (green) respectively. Right: an illustration of each sparse self-attention mechanism proposed by Big Bird, where $g = 2$, $w = 3$, $r = 2$.

We then apply and fine-tune the global feature aggregators (Sparse Transformer) upon the feature extractor by predicting a single melody track ID based on the aggregation of the frame-level features. Taking the sequence of the concatenated CNN features, the Sparse Transformer computes a sequence of embeddings with the same length and then aggregates to one single label, paying attentions to other positions in its computation. The computation of the multi-head self-attention is the same as the typical Transformer:

$$\mathcal{A}(x_i) = x_i + \phi_{attn}(\mathop{||}_{h} \delta(\frac{\mathcal{Q}_h(x_i)\mathcal{K}_h(x_i)^T}{\sqrt{d}})\mathcal{V}_h(x_i)) \quad (3)$$

where $\mathcal{Q}_h(\cdot)$, $\mathcal{K}_h(\cdot)$, $\mathcal{V}_h(\cdot)$ respectively denote the query, key and value function of x_i, $\delta(\cdot)$ denotes the scoring function, $\mathop{||}_{h} \cdot$ denotes concatenation over attention heads, and ϕ_{attn} denotes the projection applied after concatenation. Especially, the sparse self-attention reduces the density of self-attention connections to accommodate longer inputs.

3.3.3 Sparse Self-Attention In Symbolic Music

In a typical Transformer, the full self-attention mechanism requires a quadratic dependency on sequence length. However, the length of MIDI files is generally long; directly applying a full self-attention operation on it can lead to memory overflow. Rather than simply splitting the embedding $M^{(i)}$ into shorter segments [29], which can cause a loss of global information, we leverage Big Bird [23], a transformer with sparse self-attention mechanism. Specifically, the following sparse self-attention mechanisms are proposed: 1) **Random Attention**, all frames attend to r random frames in the sequence, allowing the global information aggregator to have better generalization ability. 2) **Window Attention**, all frames attend to w neighboring frames in the sequence, functioning as an information gateway of a music bar. 3) **Global Attention**, g newly introduced global frames (e.g., a classification token [CLS] preceding the sequence) attend to all frames in the sequence, collecting the global information. By combining these three sparse attention mechanisms, we manage to approximate the effectiveness of a full self-attention meanwhile reducing the time and space complexity from $O(n^2)$ to $O(n)$. Figure 3 gives an example of how frames sparsely attend in symbolic music.

4. EXPERIMENTS AND RESULTS

This section contains the experiment design and the analysis of the results. First, we detail the construction of the datasets. Then we compare our proposed model with several baselines and alternative architectures so as to evaluate if our proposed MTI model can attain SOTA accuracy. Finally, we evaluate our model on three melody-sensitive downstream tasks, including melody segregation, music embedding learning, and genre classification.

4.1 Dataset

We create four different datasets, one per experiment, to train and evaluate our models. Each dataset is split into a training set and a validation set with an 8:2 ratio unless otherwise noted.

4.1.1 For MTI Model

We collect MIDI songs from three widely used MIDI datasets: LMD [30], Reddit MIDI dataset [1], and Free-MIDI [2]. We then eliminate samples with no obvious melody track, such as MIDIs of percussion instruments and chord progression patterns, and we also eliminate MIDI files where the melody changes tracks. Next, we manually annotate the melody track of the remaining songs and shuffle their melody track IDs for label distribution balance. The resultant dataset contains 11,625 samples for model training and validation (**D-MTI**). Additionally, the trained MTI model automatically annotates another 1,475 files in the Free-MIDI dataset, providing an overall dataset of 13,100 MIDI files (**D-Full**) [3] after manual checking. The proposed datasets are desirable for training and evaluating a robust MTI model because the data covers a variety of genres (17 in total), track configurations (track orders and names), as well as MIDIs with multi-track channels.

4.1.2 For Downstream Task Evaluations

To evaluate our MTI model's utility for downstream MIR tasks, we build three more datasets based on Free-MIDI: a dataset of 1,100 randomly selected MIDIs, including audio

[1] https://www.reddit.com/r/datasets/
[2] https://github.com/josephding23/Free-Midi-Library
[3] https://github.com/maxichu/MelodyTrackIdentification

renderings of the MIDIs and of their automatically identified melody tracks (denoted as **D-Audio**), a dataset of 5,000 genre-labeled MIDI files of 17 genres (**D-Genre**), and a dataset of 500 MIDI files of pop songs (**D-POP**).

4.2 Melody Track Identification

To evaluate the effectiveness, efficiency and robustness of our proposed MTI model, we select the following three baseline models: the Skyline algorithm, a Bayesian probability model with dynamic programming [28], and a 1D-CNN model with clustering [3]. The CNN-based melody segregation model [2] is not selected for comparison because it seeks to directly extract the melodic line rather than predict the melody track. We also selected three alternative architectures: a BiLSTM architecture, a CNN + MLP architecture and a CRNN architecture. We then compare our proposed CNN + Sparse Transformer (CNN-ST) architecture with all the above models.

Table 1. Accuracy and efficiency of MTI models. The running time is the total time of infering 1,000 samples. (The frame-level accuracy of the basic CNN model is 49.72%)

Model	Accuracy(%)	Running Time(s)
Skyline	14.7	252.52
Bayesian	40.98	1094.15
1D-CNN	5.51	14818.89
BiLSTM	10.23	12.37
CNN-MLP	83.65	36.12
CRNN	81.60	32.74
CNN-ST	**84.40**	29.71

As shown in Table 1, our proposed model achieves SOTA results without incurring extra computational costs. Both Skyline and the 1D-CNN model suffer sharp reductions in accuracy when dealing with samples which are not highly normative or formative. We believe that Skyline's dependency on music composition convention hinders its generalization, while the clustering phase of 1D-CNN model might be the bottleneck of the capability to accommodate complicated data sources. The Bayesian approaches obtains a lower accuracy of 40.98% on the proposed dataset than the reported 89% accuracy on a standard MIDI dataset. Its definitions of scoring algorithms targeting at only classical music could account for the accuracy drop. These issues can cause their respective algorithms to have low accuracy when processing long music with many different arrangements and genres. Worse still, the high time complexity of the 1D-CNN model can make it too costly to be incorporated into other models or applications.

As shown in Table 1, the comparison between BiLSTM and our proposed CNN-ST shows the CNN's effectiveness in capturing local musical features. The accuracy improvement over frame-level CNN indicates the ability of aggregating global information from frames of the proposed self-attention module. In the case study, our model handle the following occasions better than the baselines: 1) There are other components having a higher average pitch than the melody; 2) The music notes are dense and in large quantity; 3) There are tracks echoing the high-pitch part of the melody. Therefore, our proposed model leads in accuracy, efficiency, and robustness for the MTI task.

4.3 Downstream Tasks

We test our MTI model in three downstream tasks of different types to evaluate its benefits to the MIR community: 1) Improving melody segregation of audio music by providing more easily-obtained training samples; 2) Improving musical embedding learning by allowing more rigorous assumptions on melody; 3) Improving genre classification by emphasizing the impacts of the melody.

4.3.1 Melody Segregation

As a typical melody-as-a-target task, melody segregation requires pairs of {audio music, melody} for training. However, such data is laborious to obtain–labeling large numbers of melodic lines requires a significant amount of time and effort from trained music specialists. Training with small datasets may lead to underfitting issues, such as a recent melody segregation work whose dataset only contained 108 samples [4]. Furthermore, existing paired datasets are often collected from just a few genres, or even one genre, of music. Models trained by those datasets thus often fail to generalize to music from other genres.

This experiment aims to test whether our MTI model can contribute to the melody segregation task by both automatically labeling the melody tracks of MIDI files in order to rapidly expand the amount of audio, including audio from various genres, formats and compositional rules, which can be used to train these systems.

Table 2. Accuracy of different training data settings for the melody segregation task. All results are computed on an independent 100-sample validation subset of D-Audio. OA is short for overall accuracy.

ID	Pretrain Data	Train Data	OA (%)
1	/	MedleyDB	37.7
2	/	D-Audio-1K	39.91
3	/	MedleyDB ∪ D-Audio-1K	**41.1**
4	D-Audio-1K	MedleyDB	**40.23**

We trained four melody segregation models with the same architecture [4] but different datasets. Our control model was trained with the 108 samples of the original MedleyDB [31]. Our three other models were trained with D-Audio-1K (1,000 samples in D-Audio), a combined dataset of MedleyDB and D-Audio-1K, and D-Audio-1K (pretraining) followed by MedleyDB (fine-tuning), respectively. Validation was done with the set of the remaining 100 samples from D-Audio, denoted as D-Audio-100.

As shown in Table 2, not only does an increased amount of training data improve melody segregation accuracy but pretraining with the D-Audio-1K dataset provides superior results to using either of those datasets on their own. The pretraining, however, does worse than the setting where all the data is used for training. This can be attributed to the bias of data distribution between the two datasets.

4.3.2 *Music Embedding Learning*

Music embedding learning is a melody-as-a-requirement task. Many music embedding learning models are trained by predicting the melody notes according to other music components. However, these models generally rely on the Skyline algorithm or track names for data preparation, which can result in errors if they do not find the melody correctly [15, 24].

We conduct an experiment with the PiRhDy embedding model [15] to evaluate how well our proposed MTI can improve music embedding learning models. We run all of PiRhDy's three subtasks for performing music embedding (token modeling, next context modeling, and accompaniment context modeling) on both the D-POP and D-Gen datasets. Before executing those subtasks, we run the Skyline and the proposed CNN-ST+ model to identify melodic lines for the tracks being analyzed.

As shown in Table 3, all three subtasks achieve higher accuracy when using melody tracks identified by our proposed model as compared to Skyline. The best improvement occurs in the subtask of token modeling on the D-Gen dataset, which indicates that our system better handles unusual genres than the existing Skyline algorithm too.

Table 3. Accuracy of models using different methods for PiRhDy embedding learning.

/	Skyline+ D-POP(%)	CNN-ST+ D-POP(%)	Skyline+ D-Gen(%)	CNN-ST+ D-Gen(%)
Token Modeling	60.49	**74.61**	74.77	**96.74**
Context Modeling-Next	91.43	**92.05**	91.89	**94.58**
Context Modeling-Acc	67.45	**78.74**	55.37	**56.53**

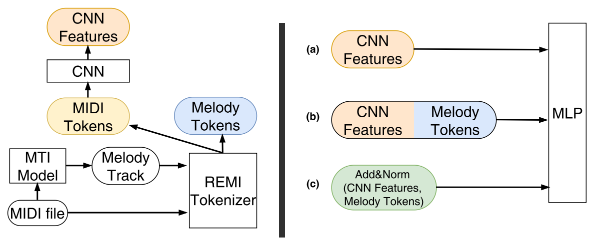


Figure 4. The three compared architectures for music genre classification with different inputs of the MLP layer. The inputs of the MLP are: (a) only the CNN features; (b) concatenation of CNN features and the melody tokens; (c) CNN features Add-Norm with the melody tokens.

4.3.3 *Genre Classification*

As a melody-as-a-dependency task, genre classification can be enhanced by explicitly emphasizing the melody over other musical components. This experiment is conducted on D-Gen. As shown in Figure 4, the full-track MIDIs and the identified melody track are first both tokenized into REMI [32] sequences with MidiTok toolkit [33]. Next, genre prediction is attempted with three different architectures. All of the architectures are CNN + MLP, and their input layers are just CNN features, CNN features appended with tokenized melody note sequences, and the CNN features Add-Norm with the tokenized melody note sequence respectively.

The results in Table 4 show that the accuracy increases when feeding the MLP layer with extra information about the melody. Therefore, highlighting the melody by either appending or blending it with the CNN features makes the music genre more distinguishable.

Table 4. Accuracy of models with different inputs to MLP layers for genre classification.

Inputs to MLP	Accuracy(%)
CNN features	41.51
CNN features ‖ Melody Tokens	**42.75**
Add&Norm (CNN features, Melody Tokens)	**45.76**

5. FUTURE WORK

We notice that emphasizing the melody track could dilute models for certain tasks where melody is not decisive, or at least is not the only decisive factor. For instance, we trained systems for key recognition [34, 35] with full MIDI files, just the melody tracks of MIDI files, and just the non-melody tracks of MIDI files. Whether using a rule-based algorithm [34] or a deep learning-based model [35], the systems trained with full MIDI files outperformed the others, and the systems trained with non-melody tracks outperformed those trained with just melody tracks. We thus estimate that accompaniments may matter more than melody for this task and are looking into ways to tweak our algorithm to account for this.

Also, the samples in our collected dataset contain only one melody track, and so for simplicity, we neglect scenarios such as multi-track melodies and melodies which switch tracks. In the future, we will develop a stronger model which can account for these scenarios even though they make the global musical structure of the song more challenging, and will also create datasets with more detailed annotations that can account for the position of the melody on a frame-by-frame basis.

6. CONCLUSION

Melody track identification is the first step towards music understanding and benefits melody-sensitive MIR tasks. This paper addresses the challenges of identifying the melody track accurately, efficiently, and robustly for any input MIDI. Our experiments show that our proposed model achieves SOTA performance and increases accuracy for many downstream tasks. We are optimistic that further research in this direction will not just enhance the ability of computers to identify melody tracks but will also improve many other aspects of MIR.

7. ACKNOWLEDGEMENTS

This project is funded in part by a grant (R-252-000-B75-112) from Singapore Ministry of Education.

8. REFERENCES

[1] E. Selfridge-Field, "Conceptual and representational issues in melodic comparison," *Computing in musicology: a directory of research*, no. 11, pp. 3–64, 1998.

[2] F. Simonetta, C. Cancino Chacón, S. Ntalampiras, and G. Widmer, "A convolutional approach to melody line identification in symbolic scores," pp. 924–931, 2019.

[3] Y.-W. Hsiao and L. Su, "Learning note-to-note affinity for voice segregation and melody line identification of symbolic music data," in *22nd International Society for Music Information Retrieval Conference*, 2021.

[4] T.-H. Hsieh, L. Su, and Y.-H. Yang, "A streamlined encoder/decoder architecture for melody extraction," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 156–160.

[5] Y. Yu, A. Srivastava, and S. Canales, "Conditional lstm-gan for melody generation from lyrics," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 1, pp. 1–20, 2021.

[6] J. Wu, C. Hu, Y. Wang, X. Hu, and J. Zhu, "A hierarchical recurrent neural network for symbolic melody generation," *IEEE transactions on cybernetics*, vol. 50, no. 6, pp. 2749–2757, 2019.

[7] Y. Chen and A. Lerch, "Melody-conditioned lyrics generation with seqgans," in *2020 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2020, pp. 189–196.

[8] X. Ma, Y. Wang, M.-Y. Kan, and W. S. Lee, "Ai-lyricist: Generating music and vocabulary constrained lyrics," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 1002–1011.

[9] O. Angelini, A. Moinet, K. Yanagisawa, and T. Drugman, "Singing synthesis: With a little help from my attention," in *Interspeech 2020*, 2020. [Online]. Available: https://www.amazon.science/publications/singing-synthesis-with-a-little-help-from-my-attention

[10] P. Lu, J. Wu, J. Luan, X. T. 0003, and L. Zhou, "Xiaoicesing: A high-quality and integrated singing voice synthesis system," in *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, 2020, pp. 1306–1310.

[11] E. Dervakos, N. Kotsani, and G. Stamou, "Genre recognition from symbolic music with cnns," in *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*. Springer, 2021, pp. 98–114.

[12] F. Simonetta, F. Carnovalini, N. Orio, and A. Rodà, "Symbolic music similarity through a graph-based representation," in *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, 2018, pp. 1–7.

[13] T. Endo, S.-i. Ito, Y. Mitsukura, and M. Fukumi, "The music analysis method based on melody analysis," in *2008 International Conference on Control, Automation and Systems*. IEEE, 2008, pp. 2559–2562.

[14] J. Salamon, B. Rocha, and E. Gómez, "Musical genre classification using melody features extracted from polyphonic music signals," in *2012 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2012, pp. 81–84.

[15] H. Liang, W. Lei, P. Y. Chan, Z. Yang, M. Sun, and T.-S. Chua, "Pirhdy: Learning pitch-, rhythm-, and dynamics-aware embeddings for symbolic music," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 574–582.

[16] M. R. Jones, "Music perception: Current research and future directions," *Music perception*, pp. 1–12, 2010.

[17] D. Bahdanau, K. Cho *et al.*, "Neural machine translation by jointly learning to align and translate. arxiv preprint arxiv: 1409.0473," 2014.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[19] S. Dai, X. Ma, Y. Wang, and R. B. Dannenberg, "Personalized popular music generation using imitation and structure," *arXiv preprint arXiv:2105.04709*, 2021.

[20] D. Rizo, P. J. P. De Leon, A. Pertusa, C. Pérez-Sancho, and J. M. I. Quereda, "Melody track identification in music symbolic files." in *FLAIRS Conference*, 2006, pp. 254–259.

[21] Z. Wang, K. Chen, J. Jiang, Y. Zhang, M. Xu, S. Dai, X. Gu, and G. Xia, "Pop909: A pop-song dataset for music arrangement generation," in *ISMIR*, 2020.

[22] Q. Xi, R. M. Bittner, J. Pauwels, X. Ye, and J. P. Bello, "Guitarset: A dataset for guitar transcription." in *ISMIR*, 2018, pp. 453–460.

[23] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang *et al.*, "Big bird: Transformers for longer sequences," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 283–17 297, 2020.

[24] T. Hirai and S. Sawada, "Melody2vec: Distributed representations of melodic phrases based on melody segmentation," *Journal of Information Processing*, vol. 27, pp. 278–286, 2019.

[25] 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 the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 791–800.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[27] A. Uitdenbogerd and J. Zobel, "Melodic matching techniques for large music databases," in *Proceedings of the Seventh ACM International Conference on Multimedia (Part 1)*, ser. MULTIMEDIA '99, New York, NY, USA, 1999, p. 57–66. [Online]. Available: https://doi.org/10.1145/319463.319470

[28] Z. Jiang and R. B. Dannenberg, "Melody identification in standard midi files," in *16th Sound & Music Computing Conference*, 2019, pp. 65–71.

[29] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. H. Engel, "Sequence-to-sequence piano transcription with transformers," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, 2021, pp. 246–253.

[30] C. Raffel, *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Columbia University, 2016.

[31] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "Medleydb: A multitrack dataset for annotation-intensive mir research." in *ISMIR*, vol. 14, 2014, pp. 155–160.

[32] Y.-S. Huang and Y.-H. Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 1180–1188.

[33] N. Fradet, J.-P. Briot, F. Chhel, A. El Fallah-Seghrouchni, and N. Gutowski, "Miditok: A python package for midi file tokenization," in *22nd International Society for Music Information Retrieval Conference*, 2021.

[34] D. Temperley, "What's key for key? the krumhansl-schmuckler key-finding algorithm reconsidered," *Music Perception*, vol. 17, no. 1, pp. 65–100, 1999.

[35] F. Foscarin, N. Audebert, and R. Fournier-S'Niehotta, "Pkspell: Data-driven pitch spelling and key signature estimation," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, 2021, pp. 197–204.

TRACKING THE EVOLUTION OF A BAND'S LIVE PERFORMANCES OVER DECADES

Florian Thalmann, Eita Nakamura, Kazuyoshi Yoshii
Graduate School of Informatics, Kyoto University, Japan
thalm007@umn.edu, {eita.nakamura, yoshii}@i.kyoto-u.ac.jp

ABSTRACT

Evolutionary studies have become a dominant thread in the analysis of large audio collections. Such corpora usually consist of musical pieces by various composers or bands and the studies usually focus on identifying general historical trends in harmonic content or music production techniques. In this paper we present a comparable study that examines the music of a single band whose publicly available live recordings span three decades. We first discuss the opportunities and challenges faced when working with single-artist and live-music datasets and introduce solutions for audio feature validation and outlier detection. We then investigate how individual songs vary over time and identify general performance trends using a new approach based on relative feature values, which improves accuracy for features with a large variance. Finally, we validate our findings by juxtaposing them with descriptions posted in online forums by experienced listeners of the band's large following.

1. INTRODUCTION

Music information retrieval has proven essential for the *analysis of large audio corpora*, especially ones for which traditional music analysis methods are limited. Such cases include large audio collections for which there is no appropriate symbolic transcription, such as ones containing non-western music or improvised music [1–5].

In recent years numerous studies have characterized the temporal evolution of musical characteristics in such corpora. Serra et al identified a restriction of pitch transitions, homogenization of timbral palette, and growing loudness levels in Western popular music [6]. On a similar corpus, Mauch et al discovered three stylistic revolutions between 1960 and 2010, based on topics identified via latent Dirichlet allocation of harmonic and timbral features [7]. Deruty and Pachet determined the 'loudness war' in popular music production to have peaked in 2007 [8]. Weiss et al found harmonic complexity to be gradually increasing in Jazz solos between the 1920s and the 2000s [4]. Weiss et al also confirmed common hypotheses concerning the evolution of chord transitions, intervals, and tonal complexity in Western classical music [9]. Parmer and Ahn measured information-theoretic complexity of pitch, loudness, timbre, and rhythm in a popular music dataset and identified trends over decades [10]. [1]

All of these studies look at music at a social or cultural level and use audio corpora that consist of material by various composers or musicians, as well as of different genres, subgenres, instrumentations, etc. In this paper we present an analogous study which however focuses on the *music of a single band*, the Grateful Dead, who are well-known for their ever-evolving performances. We use a dataset that consists of audio recordings of 2617 performances of 15 songs spanning three decades. Although this may be a sizable dataset for a band, we find that musical characteristics show a large variance over the relatively short timespan. For example, the tempos of the performances in the set range from 50 to 160, which over the whole time span results in a relatively sparse cloud of data points.

Previous approaches are relatively limited when dealing with such diverse data. They all consider the corpus as a whole and simply plot the evolution of audio features against time. This may work reasonably well for some musical characteristics such as tonal complexity, timbre, or loudness. However, we show how subdividing the corpus into subsets, in our case songs, and considering feature data relative to these subsets before integrating them into a whole can improve accuracy and confidence for the detection of overall trends. We identify such trends in various performance characteristics and juxtapose them with observations by experienced listeners from the band's large following. In particular, despite working with a relatively small subset of only 15 of the band's songs, we are able to identify generally perceived trends in tempo, song duration, and dynamic, spectral and harmonic content with promising accuracy.

2. DATASET

The cultural impact and decades long performance history of the Grateful Dead has led to continued interest in the band's music by both fans and scholars [14]. The band is especially known for their free and inclusive approach to music, their unwillingness to bow to the conventions of popular music, and their aspiration to provide their fans

[1] Similar trend analyses were recently done with non-audio music collections, such as with lyrics [11] or symbolic data [12, 13].

with a new experience at every concert. The performances of their songs often vary greatly, even from day to day, and they often engage in long improvisational parts or jams. Almost all of their over 2000 concerts between 1965 and 1995 have been recorded, often multiple times, and most of these recordings are public domain and included in the Live Music Archive's (LMA) largest subcollection. [2] To musicologists, this catalog may be intimidating for these reasons [15], but this make it all the more interesting for studies using MIR methods.

We use a dataset introduced by [5] [3] which includes 2617 performed versions of a set of 15 songs from the Grateful Dead collection of the Live Music Archive. According to the authors, the songs were selected based on two criteria: a large number of versions with soundboard recordings across the whole time span, as well as a studio recording as a potential reference. Many of these recordings contain crowd noise, which may affect the quality of features, and many are out of tune due to varying tape speed during recording. The dataset comes with a script that downloads the files from the LMA and automatically resamples them based on tuning ratios determined from chroma vectors. Figure 1 (a) shows a chronological distribution of the files. We observe very low counts for the first two years which may be due to a lower number of available recordings, and 1975 when the band retired for a year. A relative distribution of songs across the years (Figure 1 (b)) shows that the first two years of the dataset only contain one song, which may be problematic for an evolutionary analysis. A more systematic generation of such a dataset may prove useful in the future, but we chose to use it here without modifications.

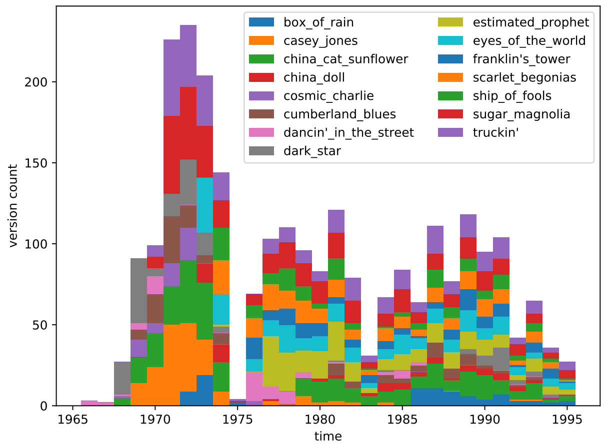
3. METHOD

The fact that we have subsets of identical or similar musical pieces or recordings can be leveraged at different points in the process. First, a large number of audio features are extracted for each recording in the corpus, now tuned as described above. We then validate these features relatively for each song, which allows us to detect outliers, i.e. wrongly classified songs, as well as adjust wrongly extracted features such as double-time beats. Our statistical analysis includes two steps. We first analyze the feature distribution and evolution for each song independently, which allows us to characterize the relative evolution of normalized feature values for each song. These relative evolutions are then collated into a global evolution curve for each feature, which we validate using bootstrapping, i.e. by alternately leaving out each song.

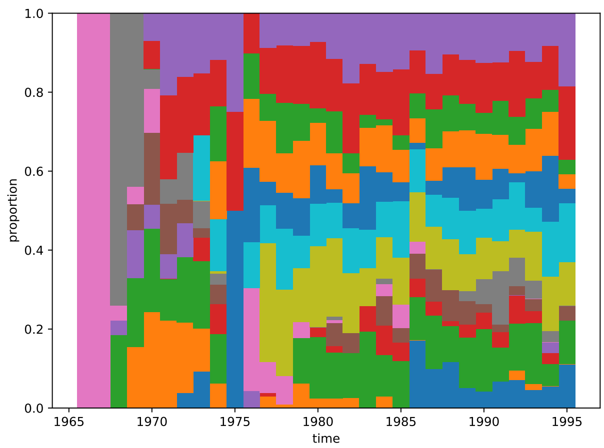
3.1 Feature Extraction

The set of audio features used in our study were inspired by previous work on other corpuses referenced in Section 1 and extracted using madmom [4] (*beats*, from which we de-

[2] https://archive.org/details/GratefulDead
[3] https://github.com/grateful-dead-live/fifteen-songs-dataset
[4] https://madmom.readthedocs.io



(a)



(b)

Figure 1: (a) Chronological distribution of the recordings for each song in the dataset. (b) Relative distribution of songs by year, order and colors correspond to (a).

rived tempo), librosa [5] (*chroma* and *mfcc*, summarized to the madmom beats), as well as the essentia freesound extractor [6] (for *dynamic* and *spectral* summary features). We used standard settings for all extractors, except for with madmom's DBNBeatTrackingProcessor where we used a much higher transition lambda of 2000 rather than the suggested maximum of 300 in the documentation. This was to reduce the probability of the processor jumping between tempos within one audio file, which may have been occurring due to the many long versions (median over 10min, sometimes over 30min), as well as the amount of crowd noise.

We also calculated a few additional features, inspired by other studies. *Tonal complexity* as defined by Weiss et al as the angular deviation or spread of pitch-classes on a circular chroma vector, permuted to correspond to the circle of fifths [4,9,16]. To measure pitch content agnostic of tonality, we devised an analogous *pitch complexity* feature based

[5] https://librosa.org
[6] https://essentia.upf.edu/freesound_extractor.html