ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447902

SYNTHTAB: LEVERAGING SYNTHESIZED DATA FOR GUITAR TABLATURE TRANSCRIPTION

Yongyi Zang, Yi Zhong*, Frank Cwitkowitz, Zhiyao Duan*

Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA

ABSTRACT

Guitar tablature is a form of music notation widely used among guitarists. It captures not only the musical content of a piece, but also its implementation and ornamentation on the instrument. Guitar Tablature Transcription (GTT) is an important task with broad applications in music education, composition, and entertainment. Existing GTT datasets are quite limited in size and scope, rendering models trained on them prone to overfitting and incapable of generalizing to out-of-domain data. In order to address this issue, we present a methodology for synthesizing large-scale GTT audio using commercial acoustic and electric guitar plugins. We procure SynthTab, a dataset derived from DadaGP, which is a vast and diverse collection of richly annotated symbolic tablature. The proposed synthesis pipeline produces audio which faithfully adheres to the original fingerings and a subset of techniques specified in the tablature, and covers multiple guitars and styles for each track. Experiments show that pre-training a baseline GTT model on SynthTab can improve transcription performance when fine-tuning and testing on an individual dataset. More importantly, cross-dataset experiments show that pre-training significantly mitigates issues with overfitting.

Index Terms— guitar tablature transcription dataset, string-accurate, timbre-rich, sample-based synthesis, music transcription

1. INTRODUCTION

Automatic Music Transcription (AMT) is the task of converting music audio into some kind of music notation, with broad applications in music education, search, and analysis [1]. Guitar Tablature Transcription (GTT) is an instrument-specific characterization of AMT [2], aiming to transcribe guitar audio into guitar tablature. Beyond musical content, tablature specifies the string for each note and any guitar-specific playing techniques that should be employed. With only minimal musical training needed, tablature is highly intuitive for guitarists of all levels. Consequently, it has become the primary means to represent and communicate information regarding guitar performances for both educational and practical purposes[1]. Due to the additional information specified in tablature notation, GTT poses several unique challenges over standard AMT tasks.

Although the utility and necessity of GTT is duly recognized, its progress has been slower relative to other instrument-specific transcription tasks such as piano transcription. One of the main reasons for this is the lack of large-scale annotated datasets. Some existing datasets, such as MedleyDB [3], include a substantial amount of guitar audio but omit specific string-level annotations. Other datasets that include string-level annotations, such as IDMT-SMT-Guitar [4], GuitarSet [5], or EGDB [6], are limited in size and diversity due to significant labor costs in recording and tablature annotation. This issue of data scarcity makes GTT models trained on such datasets susceptible to overfitting issues and limits the development of novel and more advanced transcription methods. In order to continue progressing GTT, innovative and scalable strategies for significantly expanding guitar audio-tablature datasets are imperative.

In this paper, we propose a methodology for synthesizing guitar audio directly from tablature using virtual instrument software. The tablature used for synthesis intrinsically represents the annotations for the resulting audio, allowing us to produce data perfectly suitable for GTT. We leverage the symbolic tablature dataset DadaGP [7] and employ our methodology to realize SynthTab, a large-scale, string-accurate, and timbre-rich GTT dataset. Our work bears resemblance to Slakh [8] and AAM [9], datasets which comprise audio synthesized from multi-instrument MIDI tracks. However, our approach differs primarily in that we specifically leverage virtual instrument software with string-level note control, and design our synthesis pipeline accordingly. We also synthesize each track with multiple virtual guitars and playing styles, support a subset of playing techniques specified in DadaGP, and incorporate humanization effects such as varying vibrato levels. SynthTab contains roughly 13,113 hours of audio spanning 20,715 tracks and 23 timbral profiles.

In order to investigate the utility of SynthTab, we conduct cross-dataset experiments using three existing GTT datasets featuring real guitar recordings. Our experiments shows that a baseline model trained individually on any of these datasets tends to exhibit poor generalization with respect to the others. We demonstrate that by pre-training on SynthTab, such overfitting can be mitigated, leading to improvements in both same-dataset and cross-dataset scenarios. As the first large-scale guitar audio dataset with tablature annotations, SynthTab paves the way for training larger and more complex machine learning models for GTT and adjacent tasks. Additionally, our proposed synthesis pipeline[2] can enable the research community to synthesize custom or even larger and more diverse guitar audio-tablature datasets using arbitrary symbolic tablature.

2. RELATED WORK

Recently, there has been increasing attention on the task of GTT, largely driven by the development of guitar audio-tablature datasets. The **IDMT-SMT-Guitar** [4] dataset is an early example featuring recordings of various electric guitars, pickup settings, playing styles, and playing techniques. It is split into subsets, three of which include string-level note annotations and contain isolated notes and chords, twelve short licks, and five short pieces, respectively. **GuitarSet** [5] consists of 360 improvised short recordings of experienced guitarists

*Authors contributed equally. This work is partially supported by National Science Foundation (NSF) grants No. 1846184 and No. 2222129, and synergistic activities funded by NSF grant DGE-1922591.

[1] As evidenced by the large community surrounding websites such as `https://www.ultimate-guitar.com`.

[2] All code and data is made available at `www.synthtab.dev`.

playing a single acoustic guitar, along with corresponding tablature annotations, acquired through the use of a hexaphonic pickup. Annotations are obtained by de-bleeding and performing pitch tracking on the isolated audio from each separate string. The **EGDB** [6] dataset extends the methodology of GuitarSet to collect 240 clean direct input (DI) audio signals from an experienced guitarist playing a single electric guitar using a hexaphonic pickup. Various pieces of guitar tablature are precisely played with the assistance of a click track, and an alignment procedure based on onset detection is leveraged to obtain high-resolution note annotations. Additional audio is created by feeding the DI signals into several virtual amplifiers.

While these datasets have been immense contributions to the research community, they are relatively limited, with each amounting to only a few hours of audio. Their size pales in comparison to the datasets available for other AMT tasks, such as MAESTRO [10] for piano transcription or E-GMD [11] for drum transcription, which each contain hundreds of hours of data. They also contain little variation in guitar timbre and therefore inharmonicity, a property known to be crucial for differentiating between strings [12]. Recent work on GTT has largely utilized only GuitarSet [13, 14, 15], following the cross-validation paradigm. As such, it is unclear whether these approaches can successfully generalize to out-of-domain data.

In parallel, there has also been work on guitar tablature language modeling [16, 17], enabled by recent success in sequence modeling [18] for music and the proliferation of large collections of symbolic tablature. The **DadaGP** [7] dataset is one such collection containing symbolic tablature for 26,181 popular songs in the multi-track GuitarPro[3] format. DadaGP offers diverse and high quality guitar tablature at scale, and has created new opportunities for regularizing GTT models [19] and synthesizing realistic data.

3. AUDIO SYNTHESIS PIPELINE

In this section, we outline the major steps carried out in order to synthesize symbolic tablature from DadaGP [7] and to create SynthTab.

3.1. Sourcing Tracks from DadaGP

Given the time and memory requirements associated with rendering audio, and for additional practical considerations, only a subset of the tablature within the multi-track GuitarPro files of DadaGP is selected for synthesis. We synthesize tracks corresponding to acoustic and electric guitars (MIDI instrument numbers 25-31) with six strings and no tempo changes, following a simple procedure to avoid synthesizing duplicate tracks with different GuitarPro versions.

3.2. Converting GuitarPro to JAMS

Track data is parsed from the respective GuitarPro files using the `PyGuitarPro` package [20]. In order to more conveniently represent the data for synthesis and as ground-truth, information relevant to timing, notes, and a subset of playing techniques is extracted and stored by string using the JAMS [21] format, as in GuitarSet [5].

3.3. Converting JAMS to MIDI

JAMS annotations for each track are split by string and converted to MIDI data. Virtual instrument software with string-level note control can then be driven with the MIDI to produce audio for each string. Pitch modulation techniques, including *bends* and *vibrato*, are encoded via standard MIDI control channels. Random perturbations are made to the pitch ceiling of notes with vibrato, and a small amount of vibrato is applied randomly to the remaining notes to simulate human playing. Other supported playing techniques, including *hammer-ons*, *pull-offs*, *slides*, *palm-muting*, and *harmonics*, are encoded by region via keyswitches, dedicated MIDI notes recognized by virtual instruments with pitch outside of their playing range.

	P	*F*	*Th*	*B*	*N*	*B/N*	*B/Mi*	*Mi/N*
L	×	×						
T	×	×						
M	×	×						
SJ			×					
SH	×	×						
TC				×	×			
VC				×	×			
LP				×		×		
PF				×	×	×		
SC				×	×		×	×
E					×			

Table 1. Timbral variations in SynthTab, organized by guitar plugin (rows) and playing style (columns). See Section 3.5 for more details.

3.4. Rendering & Mixing Audio

Rendering is automated using the DawDreamer package [22]. With little effort, DawDreamer can be used to load MIDI into standalone VST plugins, render the corresponding audio, and store the entirety of the signal in random-access memory. String-level MIDI is rendered individually to ensure proper string usage. Before mixing the audio signals, fundamental frequency annotations surrounding each note are extracted using the YIN [23] algorithm. Mixtures are subsequently created by averaging the string-level audio signals.

3.5. SynthTab

Using the methodology described above, we acquire ground-truth and synthesize symbolic tablature using the Ample Sound guitar plugin suite[4]. Acoustic guitar tracks are synthesized using the *L*, *T*, *M*, and *SJ* acoustic guitar plugins. Electric guitar tracks are synthesized using the *SH*, *LP*, *TC*, *VC*, *PF*, *SC*, and *E* electric guitar plugins. All synthesized audio for electric guitar corresponds to DI. Each track is synthesized multiple times with varying styles for each guitar, including *pick* (P), *finger* (F), and *thumb* (Th) style playing, and *bridge* (B), *neck* (N), and *middle* (Mi) pickup settings. In total, there are 7 timbral variations for acoustic guitar and 16 timbral variations for electric guitar. Tables 1 and 2 summarize the timbral variation and distribution of tracks in SynthTab. Further statistics related to DadaGP tracks are reported in [7]. All audio is rendered at 24-bit with a 44,100 Hz sampling rate and encoded as FLAC files. Rendering took roughly a week with a 24-core Mac Studio M2 Ultra.

4. EXPERIMENTS

Since the cues that help differentiate between strings are highly dependent on instrument-specific properties, cross-dataset evaluation is especially important for GTT. We conduct experiments where a baseline model is trained and evaluated on real guitar recordings

[3] See `https://www.guitar-pro.com` for more information.

[4] Available at `https://amplesound.net/en/index.asp`.

Midi Guitar	# Tracks	Styles	Total Hours
Acoustic Nylon (25)	5501	7	1620.40
Acoustic Steel (26)	5149		1890.95
Electric Jazz (27)	1572	16	1305.73
Electric Clean (28)	2989		2793.04
Electric Muted (29)	504		467.47
Overdriven (30)	1556		1534.21
Distortion (31)	3444		3501.09

Table 2. SynthTab track distribution by MIDI instrument.

from the IDMT-SMT-Guitar [4] (abbreviated IDMT), GuitarSet [5], and EGDB [6] datasets, which each contain unique timbral properties (see Section 2 for more details). To our knowledge, this is the first attempt to perform cross-dataset benchmarking for GTT. For our experiments, we only utilize the twelve-lick subset of IDMT, due to limited content within the other subsets, and the clean DI signals from EGDB, to maintain consistency with the audio rendered for SynthTab. We create training, validation, and testing splits randomly by track for IDMT and GuitarSet, following 8:1:1 and 10:1:1 ratios, respectively. For EGDB we adopt the pre-defined 8:1:1 split.

In order to explore the capacity of SynthTab to progress research on GTT, we then utilize a larger version of the model and perform the same set of experiments with and without pre-training on SynthTab. Due to potential conflicts between tablature distributions, we only train on data originating from tracks corresponding to *Acoustic Nylon*, *Acoustic Steel*, *Electric Jazz*, and *Electric Clean*. We also hold out 10% of tracks and the *SJ-Th* and *E-N* timbral profiles for validating the model during the pre-training phase.

4.1. Baseline Model

We adopt TabCNN [13], one of the earliest DNNs proposed for GTT, as the baseline model in order to carry out all experiments. TabCNN is a lightweight convolutional neural network (CNN) which produces string-level fret class estimates for fixed-length windows of the Constant-Q Transform (CQT) [24] of an audio signal. Classes consist of silence, open string, and the first 19 frets, and are predicted separately for each string using $\mathrm{softmax}$ activation. Since there are six strings, the output of TabCNN is a 126-dimensional six-hot vector. For the second set of experiments with and without pre-training, we add more complexity to the model by quadrupling the number of filters in each convolution layer. This version of the model is referred to as TabCNNx4. Although more recent GTT models have been proposed, no models have yet been subject to cross-dataset evaluation, and TabCNN is comparatively simple and yields solid performance.

4.2. Evaluation Metrics

All models are evaluated using the GTT metrics proposed in [13], which measure frame-level tablature and multi-pitch estimation proficiency. However, we only report F_1-score % (F_1) for a more compact presentation of results. Fret class predictions must be made on the correct string for tablature estimation, whereas only the nominal pitch of predictions are considered for multi-pitch estimation. Tablature F_1 signifies the ability of a model to correctly estimate pitch activity and differentiate between strings, which is essential for GTT. As such, it is used as the validation criterion to select the best model for each experiment. Both F_1-scores are averaged across all tracks within the validation or evaluation set for final scores.

4.3. Training Details

All training and fine-tuning is done with AdaDelta optimizer using an initial learning rate of 1.0 and batch size 32 on a single NVIDIA RTX 4090. When fine-tuning pre-trained models, the initial learning rate and batch size are reduced to 0.1 and 8, respectively. Prior to training, all audio is downsampled to 22,050 Hz. CQT features with 192 bins, 24 bins per octave, and a hop size of 512 is computed for each track during the feature extraction stage. Batches are created randomly from the respective training datasets, and each track is sampled only once per epoch with a sequence length of 500 frames.

5. RESULTS

5.1. Initial Cross-Dataset Benchmarking

The initial cross-dataset benchmarking results for TabCNN are presented in Table 3. As a sanity check, we note that the performance of TabCNN trained and tested on GuitarSet is comparable[5] to what was originally reported in [13]. However, it is clear that in each case the model has trouble generalizing to unseen data. Performance for the unmatched datasets is substantially weaker than that of the matched dataset for all three experiments. This issue is more prominent for tablature estimation, and is likely caused by the domain mismatch across GuitarSet, IDMT, and EGDB, which each feature different guitars and recording conditions. These experiments indicate that timbral features learned by a GTT model for an individual dataset exhibit low transferability under this experimental setup.

Tablature F_1 (%)		**Test**		
		GuitarSet	**IDMT**	**EGDB**
Train	**GuitarSet**	**80.1**	57.9	55.2
	IDMT	15.3	**63.1**	28.8
	EGDB	37.1	23.5	**70.5**
Multi-Pitch F_1 (%)		**Test**		
		GuitarSet	**IDMT**	**EGDB**
Train	**GuitarSet**	**84.5**	72.5	67.2
	IDMT	27.7	**66.8**	43.4
	EGDB	48.2	45.1	**74.0**

Table 3. Transcription results for TabCNN when trained on individual datasets and evaluated on the testing subset of each dataset.

F_1 (%)	**Val.**	**Test**		
	SynthTab	**GuitarSet**	**IDMT**	**EGDB**
Tablature	64.2	43.1	13.8	57.0
Multi-Pitch	77.4	70.2	74.2	74.4

Table 4. SynthTab validation results and individual dataset testing results for TabCNNx4 pre-trained on SynthTab with no fine-tuning.

5.2. Investigating Pre-Training on SynthTab

The results for TabCNNx4 pre-trained on SynthTab, without any further fine-tuning, are presented in Table 4. In general, the pre-trained model appears to follow the same trends as described in Section 5.1. There are several possible explanations for this phenomenon, including once again domain mismatch between SynthTab and the evaluation datasets, training data that is too broadly distributed with no

[5]Note that the original paper conducted six-fold cross-validation.

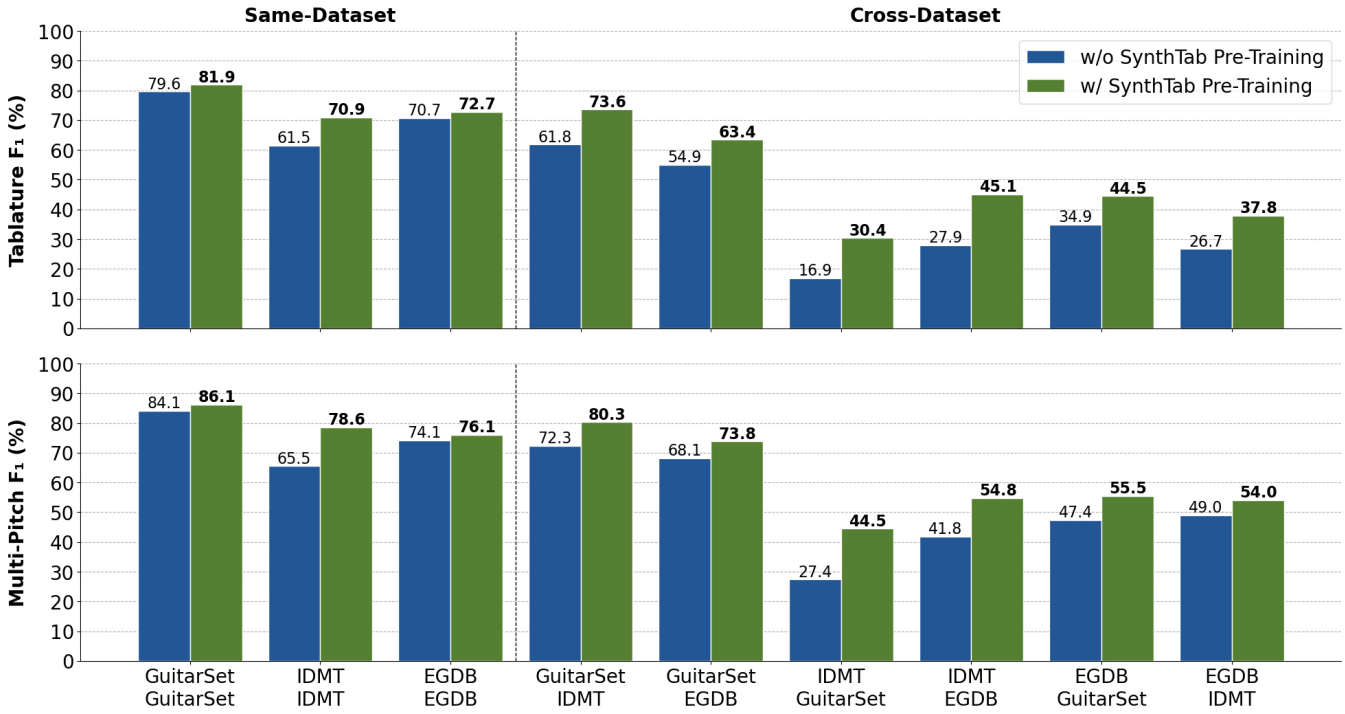

Fig. 1: Transcription results for TabCNNx4 with and without pre-training on SynthTab when trained or fine-tuned on individual datasets and evaluated on the testing subset of each dataset. Training or fine-tuning datasets are listed above testing datasets for each experiment scenario.

specificity (*e.g.* acoustic-only, electric-only, or data from a specific guitar model), or use of a GTT model with insufficient complexity. We observe that the tablature estimation performance of the pre-trained model improves slightly with respect to the unmatched experiments on GuitarSet and EGDB, though this could be due to the larger model. Multi-pitch estimation performance improves for all unmatched experiments, indicating that SynthTab can provide a solid foundation for training more general transcription models.

The cross-dataset results for TabCNNx4 with randomly initialized weights versus the model pre-trained on SynthTab are plotted in Figure 1. Without pre-training, TabCNNx4 achieves similar performance to TabCNN under all training and testing scenarios for both tablature estimation and multi-pitch estimation. In all same-dataset and cross-dataset experiments, pre-training on SynthTab yields improved performance, which in some cases is quite substantial. The relative improvement in most cross-dataset experiments is more pronounced for tablature estimation, meaning it cannot be explained wholly by more robust multi-pitch estimation. In several cross-dataset experiments, fine-tuning lowered performance on the testing dataset relative to the pre-trained model without fine-tuning. However this is not too surprising, given the domain mismatch between the individual datasets and the small amount of fine-tuning data available in some cases. Interestingly, for the pre-trained model fine-tuned on GuitarSet, the performance on IDMT actually exceeds that of the corresponding same-dataset experiments.

5.3. Discussion

Our results show that SynthTab has the potential to benefit GTT models by helping them learn more general timbral features and by improving their underlying multi-pitch estimation robustness. Models pre-trained on SynthTab and fine-tuned on subsequent data unanimously outperform the models that are only trained on the individual datasets. This observation validates the utility of the proposed synthesis methodology and the use of SynthTab for GTT. However, the results also suggest that there are significant challenges in training GTT models to generalize to completely unseen guitar data.

Future work will consist of further investigating the issue of generalization and further exploring the utility of SynthTab for GTT and related tasks. We plan to expand and improve the quality of dataset, with particular focus on better humanization strategies, varying the virtual recording environment and post-processing techniques for synthesis, and the incorporation of data augmentation for more effective training. Finally, with the realization of SynthTab, we will proceed with novel and more advanced model development focused on leveraging the scale and diversity of the dataset.

6. CONCLUSION

In this work, we presented SynthTab, the first large-scale synthesized guitar audio dataset with string-accurate tablature annotations. Utilizing the large collection of symbolic tablature available in DadaGP and commercial virtual instrument software, our proposed synthesis pipeline produces high-quality, string-accurate audio with varied timbre. We have shown through cross-dataset experimentation that training a baseline model on existing guitar audio datasets leads to poor generalization due to domain mismatch and the homogeneity of such datasets. We also demonstrate that pre-training on SynthTab and fine-tuning on individual datasets can lead to more robustness and improved generalization capacity. SynthTab creates new opportunities for advancing guitar transcription research, and future work will consist of further expansion, improvement, and exploration.

7. REFERENCES

[1] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert, "Automatic music transcription: An overview," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2019.

[2] Yu-Te Wu, Berlin Chen, and Li Su, "Multi-instrument automatic music transcription with self-attention-based instance segmentation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, vol. 28, pp. 2796–2809, 2020.

[3] Rachel M Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Pablo Bello, "MedleyDB: A multitrack dataset for annotation-intensive MIR research.," in *Proceedings of ISMIR*, 2014.

[4] Christian Kehling, Jakob Abeßer, Christian Dittmar, and Gerald Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score and instrument-related parameters," in *Proceedings of DAFx*, 2014.

[5] Qingyang Xi, Rachel M. Bittner, Johan Pauwels, Xuzhou Ye, and Juan P. Bello, "GuitarSet: A dataset for guitar transcription," in *Proceedings of ISMIR*, 2018.

[6] Yu-Hua Chen, Wen-Yi Hsiao, Tsu-Kuang Hsieh, Jyh-Shing Roger Jang, and Yi-Hsuan Yang, "Towards automatic transcription of polyphonic electric guitar music: A new dataset and a multi-loss transformer model," in *Proceedings of ICASSP*, 2022.

[7] Pedro Sarmento, Adarsh Kumar, CJ Carr, Zack Zukowski, Mathieu Barthet, and Yi-Hsuan Yang, "DadaGP: A dataset of tokenized GuitarPro songs for sequence models," in *Proceedings of ISMIR*, 2021.

[8] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux, "Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019.

[9] Fabian Ostermann, Igor Vatolkin, and Martin Ebeling, "AAM: A dataset of artificial audio multitracks for diverse music information retrieval tasks," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2023, no. 1, pp. 13, 2023.

[10] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *Proceedings of ICLR*, 2019.

[11] Lee Callender, Curtis Hawthorne, and Jesse Engel, "Improving perceptual quality of drum transcription with the expanded groove MIDI dataset," *arXiv preprint arXiv:2004.00188*, 2020.

[12] Jacob Møller Hjerrild and Mads Græsbøll Christensen, "Estimation of guitar string, fret and plucking position using parametric pitch estimation," in *Proceedings of ICASSP*, 2019.

[13] Andrew Wiggins and Youngmoo Kim, "Guitar tablature estimation with a convolutional neural network," in *Proceedings of ISMIR*, 2019.

[14] Sehun Kim, Tomoki Hayashi, and Tomoki Toda, "Note-level automatic guitar transcription using attention mechanism," in *Proceedings of EUSIPCO*, 2022.

[15] Frank Cwitkowitz, Toni Hirvonen, and Anssi Klapuri, "FretNet: Continuous-valued pitch contour streaming for polyphonic guitar tablature transcription," in *Proceedings of ICASSP*, 2023.

[16] Yu-Hua Chen, Yu-Hsiang Huang, Wen-Yi Hsiao, and Yi-Hsuan Yang, "Automatic composition of guitar tabs by transformers and groove modeling," in *Proceedings of ISMIR*, 2020.

[17] Pedro Sarmento, Adarsh Kumar, Yu-Hua Chen, CJ Carr, Zack Zukowski, and Mathieu Barthet, "GTR-CTRL: Instrument and genre conditioning for guitar-focused music generation with transformers," in *Proceedings of EvoMUSART*, 2023.

[18] Yu-Siang Huang and Yi-Hsuan Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proceedings of ACM MM*, 2020.

[19] Frank Cwitkowitz, Jonathan Driedger, and Zhiyao Duan, "A data-driven methodology for considering feasibility and pairwise likelihood in deep learning based guitar tablature transcription systems," in *Proceedings of SMC*, 2022.

[20] Sviatoslav Abakumov, "PyGuitarPro," [Online], available at: `https://github.com/Perlence/PyGuitarPro`.

[21] Eric J. Humphrey, Justin Salamon, Oriol Nieto, Jon Forsyth, Rachel M. Bittner, and Juan Pablo Bello, "JAMS: A JSON annotated music specification for reproducible MIR research," in *Proceedings of ISMIR*, 2014.

[22] David Braun, "DawDreamer: Bridging the gap between digital audio workstations and python interfaces," in *Late Breaking Demo of ISMIR*, 2021.

[23] Alain De Cheveigné and Hideki Kawahara, "YIN, a fundamental frequency estimator for speech and music," *Journal of the Acoustical Society of America (JASA)*, vol. 111, no. 4, pp. 1917–1930, 2002.

[24] Judith C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical Society of America (JASA)*, vol. 89, no. 1, pp. 425–434, 1991.