SEQUENCE-TO-SEQUENCE NETWORK TRAINING METHODS FOR AUTOMATIC GUITAR TRANSCRIPTION WITH TOKENIZED OUTPUTS

Sehun Kim
Nagoya University
kim.sehun@g.sp.m.is.nagoya-u.ac.jp

Kazuya Takeda
Nagoya University
kazuya.takeda@nagoya-u.jp

Tomoki Toda
Nagoya University
tomoki@icts.nagoya-u.ac.jp

ABSTRACT

We propose multiple methods for effectively training a sequence-to-sequence automatic guitar transcription model that uses tokenized music representation as an output. Our proposed method mainly consists of 1) a hybrid CTC-Attention model for sequence-to-sequence automatic guitar transcription that uses tokenized music representation, and 2) two data augmentation methods for training the model. Our proposed model is a generic encoder-decoder Transformer model but adopts multi-task learning with CTC from the encoder to speed up learning alignments between the output tokens and acoustic features. Our proposed data augmentation methods scale up the amount of training data by 1) creating bar overlap when splitting an excerpt to be used for network input, and 2) by utilizing MIDI-only data to synthetically create audio-MIDI pair data. We confirmed that 1) the proposed data augmentation methods were highly effective for training generic Transformer models that generate tokenized outputs, 2) our proposed hybrid CTC-Attention model outperforms conventional methods that transcribe guitar performance with tokens, and 3) the addition of multi-task learning with CTC in our proposed model is especially effective when there is an insufficient amount of training data.

1. INTRODUCTION

Automatic guitar transcription is a challenging task that has gained significant attention in the field of music information retrieval due to its potential applications in music analysis, performance evaluation, and transcription of music compositions. Despite recent advancements in the field, there are still several challenges that need to be addressed. One of the major challenges in automatic guitar transcription is the difficulty in extracting relevant features from the audio signal. The variations in timbre, pitch, and playing style make it difficult to distinguish individual notes accurately [1, 2].

Multiple methods exist for representing musical notation suitable for employment in a DNN framework. Two of the popular methods are pianoroll and tokenized musical representation. Figure 1 shows a visualization of the differences between pianoroll and tokenized music representations. Note that each can be converted from one to another. Pianoroll is a visual representation of music that uses a grid-like structure to display the timing, pitch, and duration of notes in a piece. Tokenized music representation is a type of symbolic music representation that breaks down a music signal into small, discrete tokens, which can be analyzed and processed as a sequence of tokens to extract relevant musical features such as pitch, duration, and timing. Recent studies have shown that using tokenized music representation over pianoroll can better learn the temporal dependency between different musical events [3]. Using tokenized music representation along with sequence-to-sequence models is particularly effective and have the potential to improve the performance of automatic music transcription model [1,4]. However, these models also face several challenges such as the lack of training data [1].

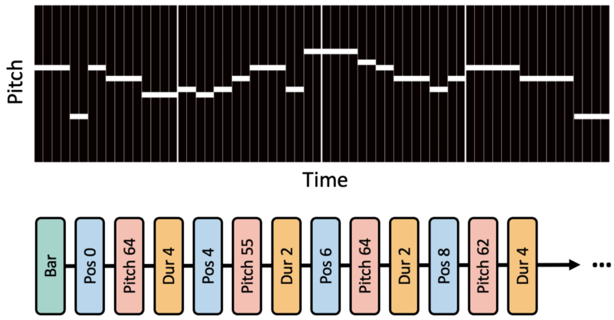

Figure 1. Examples of a pianoroll (upper) and tokenized music representation (lower).

The field of automatic speech recognition (ASR) has provided inspiration for improving automatic music transcription models, as both face similar challenges, including the need to extract relevant features and handle complex temporal and frequency relationships [5]. In the field of ASR, various techniques and models, such as connectionist temporal classification (CTC) [6], Transformer [7] models, data augmentation, transfer learning, and multi-task learning, have shown promising results [8, 9] and can potentially aid the performance of an automatic guitar transcription system.

CTC and attention are two popular techniques used in sequence-to-sequence models for various tasks. CTC is an efficient method for training models with an unknown alignment between input and output sequences. It can handle variable-length input and output sequences. However, it does not explicitly model dependencies between input and output sequences [10]. Attention, on the other hand, allows the model to selectively focus on different parts of the input sequence, improving the accuracy of the model on tasks that require complex dependencies. However, attention mechanism is too flexible in the sense that it allows extremely non-sequential alignments, making it relatively difficult to train [8].

In an attempt to improve the performance of an ASR system, researchers have also explored hybrid models that combine CTC and attention mechanisms [8]. The authors reported that the addition of CTC solves the misalignment issues and improves robustness and achieves fast convergence.

Although models such as Conformer-Transformer have been successful in ASR tasks [11], these were not used in guitar transcription mainly due to a lack of training data available. To resolve this issue, we propose data augmentation techniques and a hybrid CTC-Attention model suited for a guitar transcription system [1] that utilizes tokenized music representation and show the effectiveness of the proposed methods. Our contributions are summarized as follows.

- We propose two data augmentation methods for training a sequence-to-sequence model that utilizes tokenized music representation.
- We propose a hybrid CTC-Attention model for automatic guitar transcription.
- We conduct experimental evaluations to confirm the effectiveness of our proposed methods and prove that both the data augmentation techniques and the proposed model enhance guitar transcription performance.

2. RELATED WORK

2.1 Automatic guitar transcription

There have been many successful automatic guitar transcription systems [1, 12–15]. Some of them are based on audio signal processing to estimate the tablature score from a guitar sound signal [12]. Also, approaches that employ probabilistic models have been proposed in some automatic guitar transcription tasks. In [14, 15], a two-step method was employed, where the first step involves determining the pitch of each played note, and the second step involves computing the optimal finger positioning by combining the estimated pitch with physical limitations of feasible fingerings. Since this approach processes information in a sequential manner, information cannot flow from downstream components to upstream ones, making it difficult to be jointly optimized [16].

Most of the recent state-of-the-art systems were mainly based on end-to-end deep neural network (DNN) models since end-to-end DNN models have the advantage of the ability to jointly optimize the whole model, showing better results compared to multi-step approaches [13]. Wiggins et al. proposed a convolutional neural network (CNN)-based model architecture [13] that estimates the frame-wise fingering position of a guitar performance. In our previous work [17], a self-attention mechanism was introduced along with CNN to better capture long-term relations and estimate the fingering position in both frame-level and note-level. We proved the effectiveness of the self-attention mechanism used in the guitar transcription model. However, since the proposed systems in [13] and [17] do not detect onset, the output can not be interpreted into reproducible forms such as music score or MIDI.

2.2 Automatic music transcription using tokens

Recently, the use of generic encoder-decoder Transformer architecture has shown its potential in automatic music transcription tasks. Howthorne et al. proposed a generic Transformer architecture for an automatic piano transcription [4]. The proposed method takes mel-spectrogram of audio and autoregressively generates a token sequence. The tokenization method used in this work is similar to how MIDI file stores its note sequences. The vocabulary consists of `note`, `velocity`, and `time` tokens, with the addition of an end-of-sentence (`EOS`) token for ending the sequence. In this tokenization method, the timing of each note is represented with absolute time location within the segment, quantized into 10 ms bins. This kind of tokenization method which represents the time of a note in location as opposed to the time shift from the previous note was reported to work better [18].

Chen et al. proposed a multi-objective generic Transformer model that not only predicts token sequences but also frame-level onsets, offsets, and pitch activation [1]. In the original paper of [1], the authors report that although the proposed model is a generic Transformer model that generates a sequence of tokens, introducing multi-task learning with frame-level labels, i.e., the pianoroll representation, lowers the performance of the token-wise prediction from the decoder, but improves frame-level estimation performance compared to frame-level guitar transcription model proposed in [19]. The authors also mentioned that the lower performance of the model without multi-task learning (that only predicts token sequence) could be attributed to the insufficient size of the training set to learn a dependable language model for the decoder.

3. PROPOSED METHOD

3.1 Hybrid CTC-Attention model for tokenized guitar transcription

3.1.1 *Tokenization*

As for the tokenization method, we use a slightly modified version of revamped MIDI-derived events (REMI) [20].

[1] **Source code available**: https://github.com/KimSehun725/seq2seqGuitarTranscription

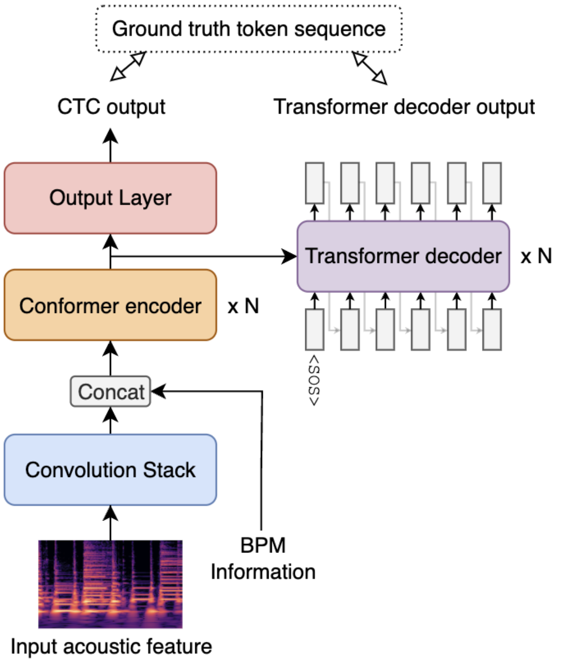

Figure 2. Overview of our proposed model architecture.

Since our main interest is to accurately transcribe a performance with pitch and timing information, we excluded velocity tokens and chord tokens from the original REMI. Excluding these tokens makes the total sequence shorter, making it easier to pass the restriction of CTC (token sequence length must be shorter than input sequence length), and easier to train. The tokenization method we used in our proposed approach includes the following vocabulary:

Blank [1 category] Used to represent the blank token when using CTC. This token gets dropped when decoding the final prediction when applying CTC.

Position [16 categories] Indicates the location in a bar quantized into 16th note. This token is placed to indicate the start position of a note.

Pitch [45 categories] Each class represents the pitch ranging from E1 to C5. This token is placed after the Position token to indicate the pitch of a note.

Duration [16 categories] Represents the duration ranging from the 16th note to a bar in 16th note increments. This token is placed after the Pitch token to indicate the duration of a note.

Bar [1 category] Token used to denote the start of a bar.

SOS, EOS [2 categories] Used to represent the start and end of a token sequence. These tokens are used for training and inferencing with the Transformer decoder.

3.1.2 Encoder

Figure 2 shows the overview of the proposed model architecture. We will refer to the left side of the figure as encoder and the right side as decoder from here on out.

The structure of the encoder of our proposed model is largely inspired by our previous automatic guitar transcription model proposed in [17], with some modifications for generating a token sequence and conditioning with BPM information. The encoder structure can be divided into three main parts: a convolution stack, a Conformer encoder, and an output layer for generating a token sequence to which CTC can be applied later.

The convolution stack has 2D convolution, max pooling, dropout layers, and a linear layer. Input features go through two convolution blocks with 2D convolution, batch normalization, and an activation function. Latent features are then subsampled by max pooling and refined by another convolution block and max pooling layer. Lastly, a linear layer is added to reduce dimension. Three dropout layers prevent overfitting after max pooling and the final linear layer.

The Conformer encoder closely follows the Conformer block architecture proposed in [21]. The Conformer encoder mainly consists of self-attention modules, convolution modules, and feed-forward modules. For the input to the Conformer encoder, first, we concatenate the output from the convolution stack and the given BPM information of the input segment. Then, the concatenated feature goes through a linear transformation layer, which is omitted from Figure 2 for simplicity. We concatenate BPM information to the output of the convolution stack because the problem formulation of our method is estimating a sequence of tokens based on both acoustic features and BPM information.

Finally, the output layer is a simple linear transformation layer with softmax function at the end for generating CTC token outputs. Unlike the model that predicts frame-level activation probability (pianoroll) from the encoder [1], the encoder of our proposed model generates the probability of token sequence, which we can directly calculate the loss between the output and the ground truth token sequence by calculating CTC loss [6].

During inference, we first apply argmax to the encoder outputs, then repeating tokens get merged. Then, the blank token gets removed to obtain the final output.

3.1.3 Decoder

The structure of the decoder in our proposed model is roughly the same as the ones used in ASR tasks or other symbolic music transcription tasks such as in [1, 4]. The decoder consists of multiple Transformer decoder blocks stacked in serial. The Transformer decoder block consists of a masked self-attention module, a cross-attention module, and a feed-forward module.

The decoder is trained and validated with a teacher forcing scheme where the token is predicted one step ahead without self-attention looking ahead by masking the self-attention with a diagonal mask in a non-autoregressive manner. During inference, we only give a start-of-sentence (SOS) token at first, and autoregressively generate the following tokens by selecting the most probable tokens at each timestep. The generation stops when the decoder generates an end-of-sentence (EOS) token.

3.1.4 Multi-task learning

Our proposed method is a multi-objective model with both the CTC output from the encoder and the output from the decoder. Therefore, we define a custom loss function for backpropagation.

First, we define the CTC loss $\mathcal{L}_{\mathrm{CTC}}$ as

$$\mathcal{L}_{\mathrm{CTC}} = -\sum_{\mathbf{y}\in\mathcal{B}} \log p(\mathbf{y}|\mathbf{x}), \tag{1}$$

where $\mathcal{B}$ is the set of all possible output sequences including blank symbols, $\mathbf{x}$ is the input sequence, and $p(\mathbf{y}|\mathbf{x})$ is the probability of generating the output sequence $\mathbf{y}$ from the encoder given the input sequence $\mathbf{x}$. The CTC loss is calculated by summing the negative log probabilities of all possible output sequences $\mathbf{y}$ that can be generated from the ground truth sequence. The loss encourages the model to learn to generate the correct output sequence while accounting for possible alignment errors between the input and output sequences.

Next, we define the cross-entropy loss as

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\mathbf{y}_{ij}\log\left(\hat{\mathbf{y}}_{ij}\right), \tag{2}$$

where N is the number of samples, C is the number of classes, $\mathbf{y}_{ij}$ is a binary indicator of whether sample i belongs to class j, and $\hat{\mathbf{y}}_{ij}$ is the predicted probability from the decoder of sample i belonging to class j.

The total loss function of our system $\mathcal{L}_{\mathrm{total}}$ can be expressed as

$$\mathcal{L}_{\mathrm{total}} = \alpha\mathcal{L}_{\mathrm{CTC}} + \mathcal{L}_{\mathrm{CE}}, \tag{3}$$

where α is the hyperparameter for controlling the weight of $\mathcal{L}_{\mathrm{CTC}}$.

3.2 Data augmentation

When using a tokenization method whose time resolution is in units of musical lengths, training a model to generate a sequence of tokens requires a large amount of data to properly model the language structure of the tokenization method and the concept of musical length. However, unlike musical instruments such as the piano, which have a significant amount of publicly available training data, the guitar lacks sufficient data for training. The goal of the proposed data augmentation methods is to scale up the amount of data used in the training process.

3.2.1 Bar overlap

In the field of automatic music transcription, splitting a musical excerpt into multiple segments is common, especially with the models that use the attention mechanism since the attention mechanism has a space complexity of $O(n^2)$ with respect to sequence length n. There have been many attempts to train a network by cutting musical pieces into exact lengths in seconds. However, there have been only a few attempts to handle music pieces by cutting them into the same musical length, e.g., 4 bars [17]. When cutting a musical piece into segments whose length is in units of bars, the most naive way of cutting it would be to cut without overlap so that the timing of the start of a segment is the end of the previous one. This results in obtaining $l_{excerpt}/l_{segment}$ segments, assuming that $l_{excerpt}$ is devidable by $l_{segment}$, where $l_{excerpt}$ denotes the bar length of a musical excerpt, and $l_{segment}$ denotes the bar length of a segment. This method was done in our previous work [17].

We propose a method that creates more segments when cutting musical excerpts, by overlapping segments, i.e. sliding a window with a hop length shorter than the window length. This results in obtaining $l_{excerpt}/l_{overlap} - (l_{segment} - 1)$ where $l_{overlap}$ denotes the bar length of the overlap.

3.2.2 Synthetic audio-MIDI pair

With the goal of training a model to properly learn how to generate the token sequence by reliably training the language model for the decoder with a large amount of data, we propose a method that can synthetically create an audio-MIDI pair dataset from MIDI-only data. We generate synthetic audio data by utilizing an oscillator such as a sinusoidal oscillator or a square wave oscillator. This results in obtaining an audio-MIDI pair with its audio being perfectly aligned with the matching MIDI, yet with unnatural timbre. With the synthetically generated audio-MIDI pair dataset, we pretrain the model before training with real-world data. This results in the decoder being trained as a reliable language model for tokenization method, and the model having preliminary knowledge regarding extracting pitch and timing information.

It is possible to use the method to produce an endless amount of data theoretically, by applying it to either a publicly available MIDI dataset or automatically generated MIDI from an automatic symbolic music generation model such as [20,22,23] since the method generates audio-MIDI pair data solely from MIDI.

3.3 Implementation details

Regarding the network settings for the encoder and decoder of our proposed model, we set the number of attention heads and the number of sequential Conformer blocks to 8 and 6 respectively. For the Transformer decoder, the number of attention heads and the number of sequential blocks are set to 4 and 4 respectively. The dimension of both the Conformer encoder and the Transformer decoder is set to 128. We implement the Conformer encoder using the ESPNet2 framework [11]. We use the leaky ReLU activation function [24] throughout the encoder, except for the activation functions in the Conformer encoder and the final output layer.

For the implementation of tokenization, we use MidiTok [25], but slightly modify the original implementation as mentioned in Section 3.1.1. As for the learning rate, we use a cyclic learning rate scheme [26]. The base learning rate is set to 1e-5, the max learning rate is set to 0.001, the number of training iterations in the increasing half of a cy-

	Encoder output		Decoder output	
Method	**F1**	**TER**	**F1**	**TER**
No data augmentation	0.363±0.159	0.469±0.185	0.526±0.154	0.713±0.219
Bar overlap (BO)	0.555±0.125	0.388±0.090	0.630±0.196	0.497±0.176
Pretrain (PT)	0.512±0.043	0.365±0.029	0.699±0.017	0.441±0.011
Proposed (BO+PT)	**0.666**±0.047	**0.307**±0.025	**0.804**±0.015	**0.336**±0.021

Table 1. Estimation metrics for evaluating the proposed data augmentation methods. For all metrics, we report the mean and standard deviation over the entire dataset. All the experiments were done with the proposed model.

	Encoder output		Decoder output	
Model	**F1**	**TER**	**F1**	**TER**
Baseline [1]	**0.767**±0.026	-	0.603±0.026	0.589±0.017
Attention only	-	-	0.784±0.014	0.345±0.021
Proposed	0.666±0.047	0.307±0.025	**0.804**±0.015	**0.336**±0.021

Table 2. Estimation metrics for our proposed model, compared with a baseline model and a model without multi-task learning with CTC. For all metrics, we report the mean and standard deviation over the entire dataset. All the experiments were done by applying both of the proposed data augmentation methods. Note that the baseline model does not have TER from the encoder output because the encoder output is pianoroll-like frame-level annotation, not tokens.

	Decoder output	
Model	**F1**	**TER**
Attention only + BO	0.114±0.046	1.520±0.378
Proposed + BO	**0.630**±0.196	**0.497**±0.176

Table 3. Estimation metrics for simulating the situation where only a small amount of training data is available by not pretraining with synthetic audio-MIDI pair data. We only show the results of the output from the decoder for simplicity.

cle is set to 4 epochs, and the same for the decreasing half. We made the peak learning rate decreases by a rate of 0.9 after each cycle. For the optimizer, we employ Rectified Adam (RAdam) [27]. Finally, we set the weight α which is used in the loss function, to 0.2.

4. EXPERIMENTAL EVALUATION

4.1 Experimental conditions

As for the MIDI data used in the pretraining phase, we used data from Classical Guitar MIDI Archives [28]. We filtered out the data that did not have the properties that we want such as having a time signature other than 4/4 beat or changing tempo over time. As a result, we obtained a total of 1232 minutes of data. We used square wave oscillator to make the synthetic audio, and split the dataset for training/validation/testing with a ratio of 0.9:0.05:0.05.

For the data used in the finetuning phase, we used the GuitarSet [29]. GuitarSet is a dataset for guitar transcription research containing 360 audio recordings, totaling approximately 3 hours. Since the dataset was recorded by six players, we left the recordings of one player for testing and used the rest of the data for training and validation. We rotated the test player to evaluate the methods with a six-fold cross-validation method. For both the pretraining data and finetuning data, we cut the tracks into segments with 4 bars, with 1 bar hop length.

Regarding the input of the network, first, we resampled the audio to 22050 Hz, then we converted the audio to a constant-Q transform (CQT) [30] with a hop length of 256 points, 24 bins per octave spanning over 8 octaves, resulting in a total of 192 frequency bins.

We evaluated the proposed methods in three points: 1) the effectiveness of data augmentation methods, 2) the performance of the model, 3) the effectiveness of introducing CTC in the proposed model when there is only a small amount of training data available. For the experiment measuring the effectiveness of the data augmentation techniques, we compared the metrics of 1) using GuitarSet only and not using bar overlap, 2) only applying bar overlap, 3) pretraining the model with synthetic audio-MIDI pair dataset without bar overlapping, and 4) using both bar overlapping and pretraining.

For the model comparison, we compared our proposed model with 1) the model proposed in [1] which refer to as baseline hereafter, 2) our proposed model without the use of CTC ($\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}}$) which we refer to as attention only model from here on out, and 3) our proposed model.

Regarding the network settings of the baseline model [1], we followed the settings described in the original paper, except we input a 4-bar-long acoustic feature to the network and use the same tokenization method as our proposed method.

4.2 Evaluation metrics

Since the output of our proposed model is a sequence of tokens that can be converted into pianoroll, we used different evaluation metrics for pianoroll domain and token domain.

In the pianoroll domain, we used precision, recall, and F1 score. If the output is a sequence of tokens, we decode the token sequence to pianoroll to calculate precision, recall, and F1 score. In the token domain, we used token er-

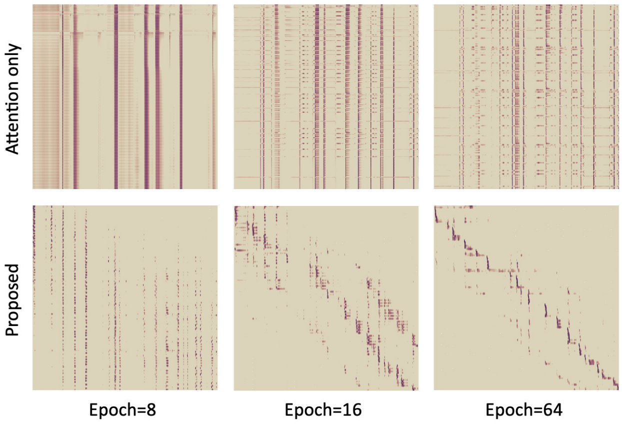

Figure 3. Comparison of the speed in learning alignments between acoustic features (horizontal axis) and tokens (vertical axis). The training was done using only the GuitarSet.

ror rate (TER), which is calculated similarly to word error rate (WER) which is a widely used metric in the research field of ASR and natural language processing (NLP). The only difference is that every element is a token index instead of a word in WER. The TER can be calculated as

$$\mathrm{TER} = \frac{S+D+I}{S+D+C}, \tag{4}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and C is the number of correct tokens.

The reason why we introduce TER as an evaluation metric is that we want to compensate for the mismatch between the metrics used in pianoroll domain and human perception. For instance, if we only evaluate the system in the pianoroll domain, the outputs with the notes that are shifted for a consistent amount of time will have very low scores. However in the same case, if we use TER as an evaluation metric, only the `position` tokens will lower the metric, not penalizing for the shifted notes as much.

4.3 Results

The result of the experiment comparing the effectiveness of the data augmentation methods is shown in Table 1. The result shows that both data augmentation methods are effective in raising the estimation performance of the proposed model. By comparing the effectiveness of bar overlapping and pretraining, pretraining shows slightly better results in the output from the decoder but worse in the output of the encoder.

The result of the experiment comparing the performance of the models mentioned in Section 4.1 is shown in Table 2. The result shows that our proposed model outperformed the baseline model and the proposed model without utilizing CTC (attention only) in both F1 score and TER. Upon comparing the estimation results of the attention-only model with the proposed model, it is evident that the latter produced superior results, thereby indicating that the inclusion of CTC considerably improves transcription performance.

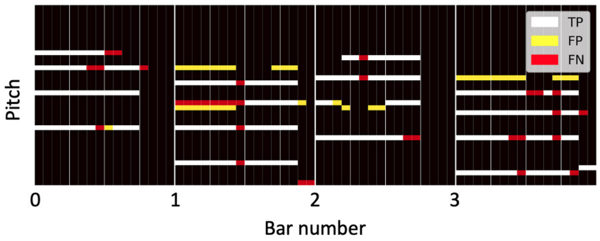

Figure 4. A sample of transcription result from our proposed model. TP, FP, and FN denote true positive, false positive, and false negative respectively.

The result of the experiment simulating a situation with a small amount of training data available is shown in Table 3. We simulated this situation by only using the GuitarSet for training. The result shows that the model with attention only performed very poorly when there is only a small amount of data. However, our proposed model which utilizes multitask learning with CTC outputs from the encoder performs better compared to attention only model. This indicates that employing multi-task learning with CTC is highly effective when there is an insufficient amount of data. Figure 3 shows the attention alignments between acoustic features and tokens. We observed that despite being trained for 64 epochs, the attention only model failed to acquire a reasonable alignment, whereas the suggested model achieved to acquire the desired alignments early on in the training process. The performance difference between the attention only model and the proposed model is likely to be attributed to the difference in the difficulty of learning the correct alignments.

While we did not include the outcomes of using solely synthetic audio-MIDI pair data for both training and testing in any of the tables, it is worth stating that during the pretraining phase of the experiment utilizing the proposed model and data augmentation methods, the output of the decoder with the test data yielded an F1 score of 0.959 and TER of 0.029. This indicates that the amount of data used in the pretraining was sufficient enough to train a reliable language model for the decoder.

5. CONCLUSION

In this paper, we proposed two data augmentation methods for training sequence-to-sequence networks that used tokenized music representation as output, and a hybrid CTC-Attention model for automatic guitar transcription. We confirmed that 1) both of the data augmentation methods are highly effective in training the sequence-to-sequence models when there is an insufficient amount of data, 2) our proposed hybrid CTC-Attention model outperforms conventional methods that transcribe guitar performance with tokens, and 3) the addition of multi-task learning with CTC in our proposed model is especially effective when there is an insufficient amount of training data.

6. ACKNOWLEDGMENT

This work was partly supported by JST CREST Grant Number JPMJCR19A3 and JSPS KAKENHI Grant Number 21H04892, Japan.

7. REFERENCES

[1] Y.-H. Chen, W.-Y. Hsiao, T.-K. Hsieh, J.-S. R. Jang, and Y.-H. Yang, "Towards automatic transcription of polyphonic electric guitar music: A new dataset and a multi-loss transformer model," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 786–790.

[2] X. Fiss and A. Kwasinski, "Automatic real-time electric guitar audio transcription," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 373–376.

[3] C. Zhang, Y. Ren, K. Zhang, and S. Yan, "Sdmuse: Stochastic differential music editing and generation via hybrid representation," *arXiv preprint arXiv:2211.00222*, 2022.

[4] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-sequence piano transcription with transformers," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[5] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 927–939, 2016.

[6] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks," in *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 01 2006, pp. 369–376.

[7] 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 (NIPS)*, vol. 30, pp. 5998–6008, 2017.

[8] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[9] H. Kheddar, Y. Himeur, S. Al-Maadeed, A. Amira, and F. Bensaali, "Deep transfer learning for automatic speech recognition: Towards better generalization," *arXiv preprint arXiv:2304.14535*, 2023.

[10] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[11] P. Guo, F. Boyer, X. Chang, T. Hayashi, Y. Higuchi, H. Inaguma, N. Kamo, C. Li, D. Garcia-Romero, J. Shi, J. Shi, S. Watanabe, K. Wei, W. Zhang, and Y. Zhang, "Recent developments on espnet toolkit boosted by conformer," in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5874–5878.

[12] C. Kehling, J. Abeßer, C. Dittmar, and G. Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score and instrument-related parameters," in *Proc. 17th International Conference on Digital Audio Effects (DAFx-14)*, 2014.

[13] A. Wiggins and Y. E. Kim, "Guitar tablature estimation with a convolutional neural network," in *Proc. 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 284–291.

[14] K. Yazawa, K. Itoyama, and H. G. Okuno, "Automatic transcription of guitar tablature from audio signals in accordance with player's proficiency," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 3122–3126.

[15] G. Hori, H. Kameoka, and S. Sagayama, "Input-output HMM applied to automatic arrangement for guitars," *Journal of Information Processing*, vol. 21, no. 2, pp. 264–271, 2013.

[16] G. W. Lee and H. K. Kim, "Two-step joint optimization with auxiliary loss function for noise-robust speech recognition," *Sensors*, vol. 22, no. 14, 2022.

[17] S. Kim, T. Hayashi, and T. Toda, "Note-level automatic guitar transcription using attention mechanism," in *2022 30th European Signal Processing Conference (EUSIPCO)*, 2022, pp. 229–233.

[18] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This time with feeling: Learning expressive musical performance," *Neural Computing and Applications*, vol. 32, pp. 955–967, 2020.

[19] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and Frames: Dual-Objective Piano Transcription," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*. Paris, France: ISMIR, Sep. 2018, pp. 50–57.

[20] 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.

[21] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

[22] G. Mittal, J. Engel, C. Hawthorne, and I. Simon, “Symbolic music generation with diffusion models,” in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[23] A. Muhamed, L. Li, X. Shi, S. Yaddanapudi, W. Chi, D. Jackson, R. Suresh, Z. C. Lipton, and A. J. Smola, “Symbolic music generation with transformer-gans,” in *35th AAAI Conference on Artificial Intelligence, AAAI*, 2021.

[24] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical evaluation of rectified activations in convolutional network,” *arXiv preprint arXiv:1505.00853*, 2015.

[25] N. Fradet, J.-P. Briot, F. Chhel, A. El Fallah Seghrouchni, and N. Gutowski, “MidiTok: A python package for MIDI file tokenization,” in *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[26] L. N. Smith, “Cyclical learning rates for training neural networks,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 464–472.

[27] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, “On the variance of the adaptive learning rate and beyond,” in *International Conference on Learning Representations (ICLR)*, 2020.

[28] “Classical guitar midi archives,” accessed on April 1, 2023. [Online]. Available: https://www.classicalguitarmidi.com/

[29] Q. Xi, R. Bittner, J. Pauwels, X. Ye, and J. P. Bello, “Guitarset: A dataset for guitar transcription,” in *Proc. 19th International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, Sep. 2018, pp. 453–460.

[30] J. Youngberg and S. Boll, “Constant-Q signal analysis and synthesis,” in *1978 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, 1978, pp. 375–378.