AUTOMATIC SINGING TRANSCRIPTION BASED ON ENCODER-DECODER RECURRENT NEURAL NETWORKS WITH A WEAKLY-SUPERVISED ATTENTION MECHANISM

Ryo Nishikimi[1] *Eita Nakamura*[1] *Satoru Fukayama*[2] *Masataka Goto*[2] *Kazuyoshi Yoshii*[1]

[1]Graduate School of Informatics, Kyoto University, Japan
[2]National Institute of Advanced Industrial Science and Technology (AIST), Japan

ABSTRACT

This paper describes neural singing transcription that estimates a sequence of musical notes directly from the audio signal of singing voice in an end-to-end manner without time-aligned training data. A conventional approach to singing transcription is to perform vocal F0 estimation followed by musical note estimation. The performance of this approach, however, is severely limited because the F0 estimation errors propagate to the note estimation step and rich acoustic information cannot be used. In addition, it is difficult and time-consuming to split continuous signals of singing voices into segments corresponding to musical notes for making precise time-aligned transcriptions. To solve these problems, we use an encoder-decoder model with an attention mechanism that can automatically learn an input-output alignment and mapping, even from non-aligned training data. The main challenge of our study is to estimate temporal categories (note values) in addition to instantaneous categories (pitches). We thus propose a novel loss function for the attention weights of time-aligned notes for semi-supervised alignment training. By gradually reducing the weight of the loss function, a better input-output alignment can be learned much more quickly. We showed that our method performed well for isolated singing voice in popular music.

Index Terms— Automatic singing transcription, end-to-end learning, sequence-to-sequence learning, encoder-decoder recurrent neural networks, attention mechanism

1. INTRODUCTION

Automatic singing transcription (AST) refers to estimating a sequence of musical notes of a sung melody from music audio signals, and forms a basis for music information retrieval (MIR) because the melody is the most prominent part of popular music that influences the impression of a song. The estimated musical notes can be used for singing voice generation [1], musical grammar analysis, query-by-humming, and active music listening [2].

Many studies have been conducted on AST. A naive approach to AST is to sequentially use a singing voice separation method that extracts singing voice from music audio signals [3–5] and an F0 estimation method that estimates F0 trajectories from singing voice [5–11]. This approach requires an additional step to estimate the semitone-level pitches and note values of musical notes by quantizing F0 trajectories in the time and frequency domains. Most studies, however, have focused on only pitch quantization for estimating piano rolls [12–14], while note-value quantization (*a.k.a.* rhythm transcription) has been investigated independently [15–17]. Some studies have tried to jointly estimate the pitches and note values of musical notes from F0 trajectories using a musical language model that represents the pitches, rhythms, and scales of musical notes [18, 19]. The major problem of the aforementioned methods are that the errors occurred in the F0 estimation adversely affect the note estimation.

This work was supported in part by JST ACCEL No. JPMJAC1602, JSPS KAKENHI No. 16H01744, No. 16J05486, Kayamori Foundation, and the Kyoto University Foundation.

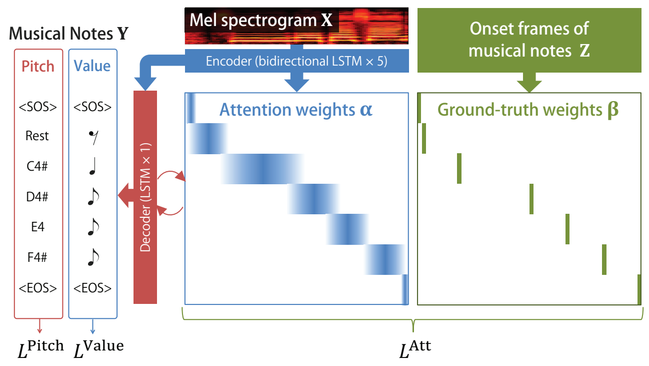

Fig. 1. Our encoder-decoder model with an attention mechanism for end-to-end AST. This model is trained by minimizing the weighted sum of loss functions for ground-truth pitches and note values, as well as alignment information (onset times) if available.

In this paper, we propose an end-to-end approach for AST. AST is similar to automatic speech recognition (ASR) in that audio spectra are converted into a sequence of meaningful symbols (musical notes or characters). Inspired by the success of sequence-to-sequence or end-to-end learning in ASR [20–22] and machine translation [23, 24], we focus on an encoder-decoder model with an attention mechanism for directly estimating a sequence of musical notes from a sequence of vocal spectra. The encoder-decoder model is a deep neural network (DNN) that can convert an input sequence into an output sequence of different lengths. The attention mechanism is an efficient method for dealing with long sequences by setting attention weights (weak alignment) between output and input sequences.

A key difference between AST and ASR is that temporal categories (note values) must be estimated in addition to instantaneous categories (pitches). To deal with such fine temporal structure as rhythms, it is crucial to train attention weights precisely. Although a simple solution is to apply supervised learning for the attention weights, instead of the common unsupervised learning, there is a problem of the limitation of available training data for AST.

To solve this, we propose a novel framework based on semi-supervised learning for the attention weights by using a limited amount of partially inaccurate time-aligned data as guiding information (Fig. 1). Specifically, we propose a novel loss function that evaluates the input-output alignment estimated by the attention mechanism. If time-aligned data are available, we calculate the cross-entropy loss between the estimated normalized attention weights and the corresponding ground-truth weights for each note. We find that when the model is trained successfully in an unsupervised manner, the attention weights of each note tend to concentrate around its onset frame. As the ground-truth weights for each note, therefore, we choose a one-hot vector that is peaked at the onset frame of the note. Although the onset time annotations are often unreliable, the model can learn correct alignment by gradually reducing the weight of the attention loss function. The main contribution of this study is an easy-to-implement general technique for effectively avoiding bad local optima and accelerating convergence in the framework of semi-supervised end-to-end learning when both aligned and non-aligned data are available.

2. ATTENTION-BASED ENCODER-DECODER MODEL

This section reviews the standard encoder-decoder model with an attention mechanism for sequence-to-sequence learning [21].

2.1. Encoder

The encoder transforms a sequence of feature vectors (input data) $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T] \in \mathbb{R}^{F \times T}$ into a sequence of latent representations $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_T] \in \mathbb{R}^{E \times T}$, where T, F, and E indicate the length of the input sequence, the dimension of the feature vectors, and the dimension of the latent vectors, respectively. The encoder is usually formulated as a recurrent neural network (RNN). In this study, we use a multi-layer bidirectional long short-term memory unit (LSTM).

2.2. Decoder with Attention Mechanism

The decoder predicts a sequence of symbols $\mathbf{Y} = [y_1, \ldots, y_N]$ from the latent vectors $\mathbf{H}$, where N indicates the number of symbols predicted by the decoder. $y_n \in \{1, \ldots, I\}$ indicates the n-th predicted symbol, where I indicates the vocabulary size of the decoder. The vocabulary includes two special symbols: $\langle \texttt{sos} \rangle$ and $\langle \texttt{eos} \rangle$. The attention-based decoder consists of a unidirectional RNN and recursively calculates the following steps:

$$\boldsymbol{\alpha}_n = \mathrm{Attend}(\mathbf{s}_{n-1}, \boldsymbol{\alpha}_{n-1}, \mathbf{H}), \tag{1}$$

$$\mathbf{g}_n = \sum_{t=1}^{T} \alpha_{nt} \mathbf{h}_t, \tag{2}$$

$$y_n = \mathrm{Generate}(\mathbf{s}_{n-1}, \mathbf{g}_n), \tag{3}$$

$$\mathbf{s}_n = \mathrm{Recurrency}(\mathbf{s}_{n-1}, \mathbf{g}_n, y_n), \tag{4}$$

where $\mathbf{s}_n \in \mathbb{R}^D$ indicates the n-th hidden state of the decoder, and Attend, Generate, and Recurrency are functions that perform operations on vectors and matrices.

Eq. (2) represents the attention mechanism. $\boldsymbol{\alpha}_n \in \mathbb{R}^T$ is a vector of normalized weights representing the degrees of relevance between the input sequence $\mathbf{X}$ and a note y_n. Each element of $\boldsymbol{\alpha}_n$ is given by

$$\alpha_{nt} = \frac{\exp(e_{nt})}{\sum_{t'=1}^{T} \exp(e_{nt'})}, \tag{5}$$

$$e_{nt} = \mathrm{Score}(\mathbf{s}_{n-1}, \mathbf{h}_t, \boldsymbol{\alpha}_{n-1}), \tag{6}$$

where Score is a function that calculates a raw weight. In this paper, we use a convolutional function [21] given by

$$\mathbf{f}_n = \mathbf{F} * \boldsymbol{\alpha}_{n-1}, \tag{7}$$

$$e_{nt} = \mathbf{w}^\top \tanh\left(\mathbf{W}\mathbf{s}_{n-1} + \mathbf{V}\mathbf{h}_t + \mathbf{U}\mathbf{f}_{nt} + \mathbf{b}^{\mathrm{Att}}\right), \tag{8}$$

where $\mathbf{F} \in \mathbb{R}^{C \times F}$ is a convolutional filter, $\mathbf{f}_n \in \mathbb{R}^{T \times C}$ is the result of the convolution, and C and F indicate the number of channels and the size of the filter. $\mathbf{w} \in \mathbb{R}^A$ indicates a weight vector, $\mathbf{W} \in \mathbb{R}^{A \times D}$, $\mathbf{V} \in \mathbb{R}^{A \times E}$, and $\mathbf{U} \in \mathbb{R}^{A \times C}$ represent weight matrices, and $\mathbf{b}^{\mathrm{Att}} \in \mathbb{R}^A$ represents a bias vector. Here, A is the number of rows of $\mathbf{W}$, $\mathbf{V}$, and $\mathbf{U}$, as well as the number of elements of $\mathbf{b}^{\mathrm{Att}}$.

Eq. (3) represents the generation of y_n from the previous hidden state $\mathbf{s}_{n-1}$ and the weighted sum $\mathbf{g}_n$ as folows:

$$\boldsymbol{\pi} = \mathrm{Softmax}\left(\mathbf{P}\mathbf{s}_{n-1} + \mathbf{Q}\mathbf{g}_n + \mathbf{b}^{\mathrm{Gen}}\right), \tag{9}$$

$$y_n = \underset{y_n \in \{1,\ldots,K\}}{\mathrm{argmax}} \left(\pi_{y_n}\right), \tag{10}$$

where $\mathbf{P} \in \mathbb{R}^{I \times D}$, $\mathbf{Q} \in \mathbb{R}^{I \times E}$ represent weight matrices, and $\mathbf{b}^{\mathrm{Gen}} \in \mathbb{R}^I$ is a bias vector.

Eq. (4) represents the calculation of the next state $\mathbf{s}_n$. Note that the ground-truth symbol is used as y_n in the training phase, whereas in the inference phase, y_n is predicted by the decoder at the previous step and the symbol prediction stops when the output sequence reaches a specified maximum length or when $\langle \texttt{eos} \rangle$ is generated.

3. PROPOSED METHOD

This section explains the proposed method of AST with a modified attention-based encoder-decoder model (Fig. 1).

3.1. Problem Specification

Our goal is to train an encoder-decoder model that takes as input a mel-scale spectrogram $\mathbf{X} = \mathbf{x}_{1:T} \in \mathbb{R}^{F \times T}$ of an isolated solo singing voice and outputs a sequence of musical notes $\mathbf{Y} = y_{1:N} = (p_n, v_n)_{1:N}$ by using the onset frames of those notes $\mathbf{Z} = z_{1:N}$ if available, where T, F, and N indicate the number of time frames, that of frequency bins, and that of musical notes, respectively. $\mathbf{x}_t \in \mathbb{R}^F$ indicates the mel-scale spectrum at frame t. Each note y_n is represented as a pair of a semitone-level pitch $p_n \in \{1, \ldots, K\}$ and a 16th-note-level duration $v_n \in \{1, \ldots, L\}$. K and L indicate the size of the pitch vocabulary (including the rest) and that of the note-value vocabulary, respectively. $z_n \in \{1, \ldots, T\}$ indicates the onset frame of the musical note y_n in the spectrogram $\mathbf{X}$.

3.2. Pitch and Note Value Decoder

In AST, it is necessary to output two symbols: a pitch and a note value. One possibility is to regard the pair of a pitch and a note value as one symbol. Such a model, however, is difficult to train from a limited amount of training data because the joint vocabulary size is $I = K \times L$. To reduce the vocabulary size to $I = K + L$, we extend the decoder to separately output both symbols at the same time as follows:

$$\boldsymbol{\phi} = \mathrm{Softmax}\big(\hat{\mathbf{P}}\mathbf{s}_{n-1} + \hat{\mathbf{Q}}\mathbf{g}_n + \hat{\mathbf{b}}\big), \tag{11}$$

$$p_n = \underset{p_n \in \{1,\ldots,K\}}{\mathrm{argmax}} \left(\phi_{p_n}\right), \tag{12}$$

$$\boldsymbol{\psi} = \mathrm{Softmax}\big(\bar{\mathbf{P}}\mathbf{s}_{n-1} + \bar{\mathbf{Q}}\mathbf{g}_n + \bar{\mathbf{b}}\big), \tag{13}$$

$$v_n = \underset{v_n \in \{1,\ldots,L\}}{\mathrm{argmax}} \left(\psi_{v_n}\right), \tag{14}$$

where $\hat{\mathbf{P}} \in \mathbb{R}^{K \times D}$, $\hat{\mathbf{Q}} \in \mathbb{R}^{K \times E}$, $\bar{\mathbf{P}} \in \mathbb{R}^{L \times D}$, and $\bar{\mathbf{Q}} \in \mathbb{R}^{L \times E}$ indicate weight matrices, and $\hat{\mathbf{b}} \in \mathbb{R}^K$ and $\bar{\mathbf{b}} \in \mathbb{R}^L$ indicate bias parameters. To treat a pitch and a note value as separate symbols, we use two different $\langle \texttt{sos} \rangle$ and $\langle \texttt{eos} \rangle$ symbols for the pitch and note-value prediction, In short, the pitch vocabulary includes $\langle \texttt{sos\_p} \rangle$ and $\langle \texttt{eos\_p} \rangle$, and the note-value vocabulary includes $\langle \texttt{sos\_v} \rangle$ and $\langle \texttt{eos\_v} \rangle$.

3.3. Loss Function for Attention Weights

We define a loss $\mathcal{L}^{\text{Att}}$ for attention weights $\boldsymbol{\alpha}=[\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N]^\top \in \mathbb{R}^{N\times T}$ to guide them to ideal values. To calculate $\mathcal{L}^{\text{Att}}$, the guiding ground-truth weights $\boldsymbol{\beta}$ for $\boldsymbol{\alpha}$ are introduced by using the optional data $\mathbf{Z}$. We find that when the model is successfully trained without calculating $\mathcal{L}^{\text{Att}}$, the attention weights of each note tend to concentrate around its onset frame. A one-hot vector is thus considered to be a good choice for $\boldsymbol{\beta}$ as follows:

$$\beta_{nt} = \begin{cases} 1 & (z_n = t) \\ \varepsilon & (\text{otherwise}) \end{cases}, \tag{15}$$

where ε is a small positive number. After each row of $\boldsymbol{\beta}$ is normalized, $\mathcal{L}^{\text{Att}}$ is given by the cross entropy as follows:

$$\mathcal{L}^{\text{Att}} = -\frac{\lambda}{N}\sum_{n=1}^{N}\sum_{t=1}^{T}\beta_{nt}\log\alpha_{nt}, \tag{16}$$

where λ is a hyperparameter to scale loss $\mathcal{L}^{\text{Att}}$, which is gradually decreased during training to learn optimal input-output alignment that is considered to be different from hard alignment $\boldsymbol{\beta}$.

3.4. Training and Inference Algorithms

In the training phase, the pitch and note value are separately converted to one-hot vectors, and then input to the decoder. The loss $\mathcal{L}^{\text{Pitch}}$ for the output pitches and the loss $\mathcal{L}^{\text{Value}}$ for the output note values are given by their cross entropies. The sum of $\mathcal{L}^{\text{Pitch}}$, $\mathcal{L}^{\text{Value}}$, and $\mathcal{L}^{\text{Att}}$ is defined as the total loss function to be minimized. In the inference phase, the pitch and note value obtained by Eqs. (12) and (14) at the previous time step are converted into one-hot vectors and used for predicting the current symbol. This process stops when the output sequence reaches a specified maximum length or when $\langle\texttt{eos\_p}\rangle$ or $\langle\texttt{eos\_v}\rangle$ is generated.

4. EVALUATION

This section describes experiments conducted to evaluate the performance of the proposed model for AST.

4.1. Experimental Data

To evaluate our model, we used 54 popular songs with reliable annotations from the RWC Music Database [25]. We split the audio signals of isolated singing voice and the corresponding time-aligned musical scores [26] into all possible segments ranging from 1 measure to 8 measures with an overlap of 1 measure. When a note crossed a bar line, it was included in the precedent measure. Rests in a measure were concatenated into a single rest. Musical notes longer than a whole note were discarded. We used 44, 5, and 5 songs as training, validation, and test data, respectively.

All songs were sampled at 44.1 kHz, and we used an STFT with a Hann window of 2048 points and a shifting interval of 441 points (10 ms) for calculating magnitude spectrograms, which we normalized to make the maximum value 1. Since a tempo determines note values, which would be difficult to correctly estimate for the proposed method without any mechanism to estimate tempos, all songs were modified to a BPM of 150 using a phase vocoder for the training, validation, and test data. We standardized the spectrograms for each frequency bin, and then calculated the mel-scale spectrograms with 229 channels.

4.2. Configurations

The vocabulary of pitches consisted of a rest, $\langle\texttt{sos\_p}\rangle$, $\langle\texttt{eos\_p}\rangle$, and 40 semitone-level pitches from E2 to G5 ($K = 43$). The vocabulary of note values consisted of $\langle\texttt{sos\_v}\rangle$, $\langle\texttt{eos\_v}\rangle$, and 16 values which were integer multiples of a 16th note up to a whole note ($L = 18$). We discarded any data containing out-of-vocabulary notes, and it was also assumed that an input sung melody could be represented as a monophonic sequence of musical notes.

Table 1. Word error rates on the test data.

Configuration of λ	NER [%]	PER [%]	VER [%]
$\lambda = 1$	48.8	31.6	34.8
$\lambda = 0$	119.8	90.0	96.8
λ is gradually reduced	43.0	24.7	29.6

We applied frame stacking [27] with a stack size of 5 and a skip size of 1 to the mel-scale spectrograms, and added zero frames at the both ends to align with $\langle\texttt{sos}\rangle$s and $\langle\texttt{eos}\rangle$s. The encoder consisted of five-layers of bidirectional LSTMs with 300×2 cells. We set the dropout rate to 0.2 for each layer. The decoder consisted of one-layer LSTM with 200 cells. The number of channels and the filter size were $C = 10$ and $F = 100$. We used a padding size and a stride of convolution in the attention mechanism of 50 and 1, respectively. Adam [28] with a standard setting was used to optimize the proposed model. The hyperparameter A was 200. To avoid overfitting, a weight decay (L2 regularization) with a controllable hyperparameter of 10^{-5} was used. To prevent weight parameters from diverging, gradient clipping with a threshold of 5.0 was also used. All weight parameters of fully-connected layers were initialized with random values drawn from the uniform distribution $\mathcal{U}(-0.1, 0.1)$. The filter of the one dimensional CNN in the attention mechanism and the weight parameters of the encoder and decoder were initialized by He's method [29]. All bias parameters were initialized with zeros. The small positive value ε was set to 10^{-4}. The batch size and the number of epochs were 20 and 15. Pytorch v0.4.1 [30] was used for implementation.

To verify the effectiveness of $\mathcal{L}^{\text{Att}}$, we compared the following three configurations of the hyperparameter λ:

- λ is fixed to 1.
- λ is fixed to 0.
- λ is initialized to 1 and reduced by 10^{-4} at each iteration.

We also examined the effectiveness of the semi-supervised learning when only a part of $\mathbf{Z}$ is available. We randomly selected data from $\mathbf{Z}$ at a rate of 5%, 10%, 25%, 50%, and 75%, and only the selected data were used to calculate the guiding ground-truth weights $\boldsymbol{\beta}$ and the loss function $\mathcal{L}^{\text{Att}}$ with a gradually reduced λ. For evaluation, we used the model which minimized the average of validation losses per an epoch during training.

4.3. Evaluation Metrics

Performance was measured using word error rate (WER) defined as

$$\text{WER} = \frac{N_S + N_D + N_I}{N} \times 100\ [\%], \tag{17}$$

where the numerator represents the Levenshtein distance between ground-truth and estimated note sequences. N_S, N_D, and N_I indicate the minimum number of substitutions, deletions, and insertions required to change the estimated sequence into the ground-truth. N indicates the number of ground-truth musical notes. We used three variants of WER: note error rate (NER) for evaluating both pitches and note values, pitch error rate (PER) for evaluating only pitches, and value error rate (VER) for evaluating only note values.

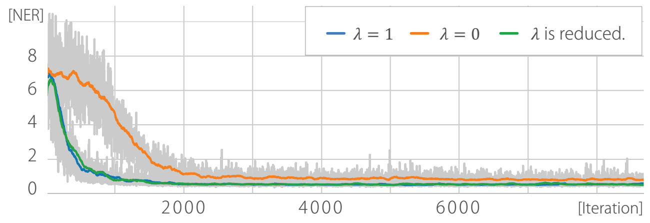

Fig. 2. NERs calculated on the validation data during the training. Grey lines indicate NERs of each iteration, and colored lines indicate the average values of the NERs for the past 100 iterations.

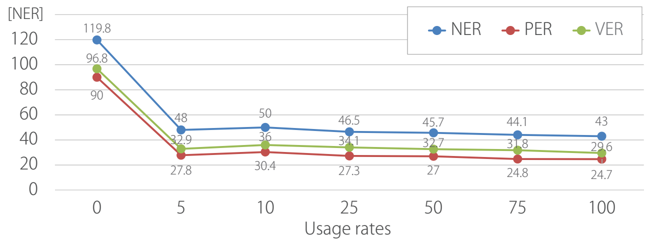

Fig. 3. WERs with different usage rates of training data $\mathbf{Z}$.

4.4. Experimental Results

Experimental results are shown in Table 1. The models obtained using the proposed loss function clearly outperformed those obtained with $\lambda = 0$. Fig. 2 shows NERs calculated on the validation data during the training of each proposed model. It indicates that the proposed loss function was effective in accelerating the convergence of NERs and finding a better solution with lower validation loss. An interesting fact is that better WERs were obtained by gradually reducing the value of λ than by fixing λ to 1. Since the time-aligned onset annotations and guiding ground-truth weights $\boldsymbol{\beta}$ were not always accurate, gradually reducing λ enables the proposed model to automatically find better time alignment.

Examples of attention weights and musical notes estimated by the proposed model are illustrated in Fig. 4. The yellow score estimated with $\lambda = 0$ included many 8th notes that were not in the ground-truth score, and vague attention weights were learned. On the other hand, the blue and green scores estimated using the proposed loss function were nearly correct, and peaked attention weights were learned. Given that musical note estimation failed with softly-learned attention weights, it seems to be reasonable to use the peaked ground-truth weights. Although a rest that was not in the ground-truth score was estimated near the center of the bottom score, the input spectrogram certainly had an unvoiced interval near the 300th frame. The annotated data regards the silence as a continuation of the musical note, whereas the proposed method estimated a rest for the silent section of the spectrogram. Offset detection is still an open problem in music transcription, and we will consider offsets further in future work.

Experimental results with different usage rates of $\mathbf{Z}$ are shown in Fig. 3. Note that the cases of using 0% and 100% of $\mathbf{Z}$ in Fig. 3 correspond to the second and third rows of Table 1, respectively. As shown in Fig. 3, the WERs were almost monotonically reduced as the usage rate of $\mathbf{Z}$ was increased. Especially, the WERs were drastically reduced even if only 5% of $\mathbf{Z}$ was used. This indicates that the loss function $\mathcal{L}^{\text{Att}}$ is very effective even when a small amount of supervised data is available.

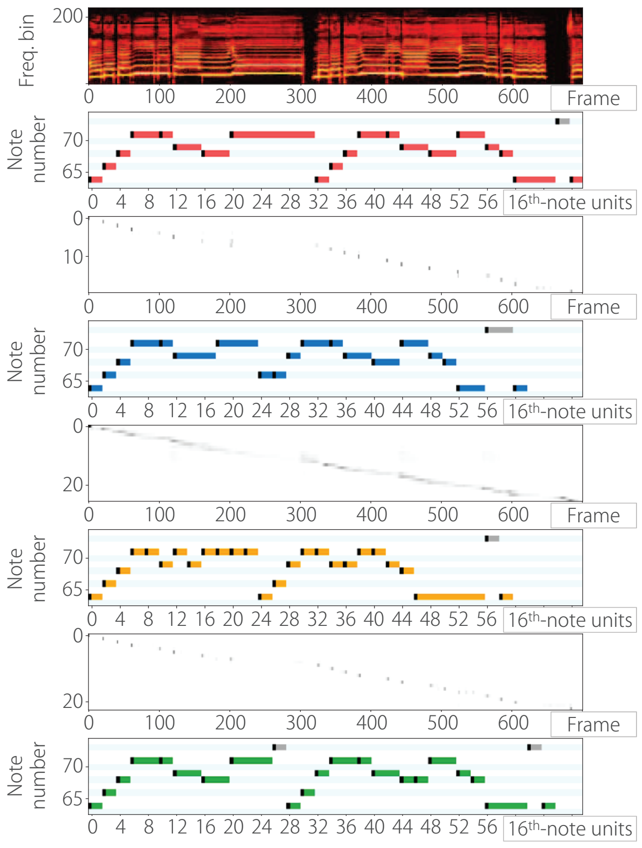

Fig. 4. Examples of attention weights and musical notes estimated by the proposed method. Red, blue, yellow, and green horizontal lines indicate musical notes, grey lines indicate rests, and black squares indicate the onset positions of the musical notes. The top two figures are the input spectrogram and the ground-truth musical notes. The subsequent figures are attention weights and musical notes for $\lambda = 1$, $\lambda = 0$, and the gradual reduction of λ from top to bottom.

5. CONCLUSION

This paper has presented the method for estimating the musical notes of a sung melody with an attention-based encoder-decoder model. We extended the standard model to simultaneously predict a pitch and a note value at each step. We also proposed a new loss function for attention weights and a semi-supervised training method for better performance and faster convergence. The experimental results showed that the proposed encoder-decoder model has great potential for end-to-end AST, and that the performance of AST was improved by using the attention loss function.

One of the most important research directions for future work is to integrate the proposed model with an end-to-end RNN for estimating time-varying tempo and beats/downbeats from audio signals. Although tempo changes often occur in popular songs, the proposed model assumes that the tempo of a song is constant. Such integration should improve the robustness of the note value estimation. We would also like to extend the proposed method to directly deal with polyphonic music signals without singing voice separation. This would enable us to leverage a huge amount of music recordings with non-aligned melody transcriptions.

6. REFERENCES

[1] M. Blaauw and J. Bonada, “A neural parametric singing synthesizer modeling timbre and expression from natural songs,” *Applied Sciences*, vol. 7, no. 12, 2017.

[2] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T Nakano, “Songle: A web service for active music listening improved by user contributions,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2011, pp. 311–316.

[3] Y. Li and D. Wang, “Separation of singing voice from music accompaniment for monaural recordings,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1475–1487, 2007.

[4] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson, “Singing-voice separation from monaural recordings using robust principal component analysis,” in *IEEE ICASSP*, 2012, pp. 57–60.

[5] Y. Ikemiya, K. Yoshii, and K. Itoyama, “Singing voice analysis and editing based on mutually dependent F0 estimation and source separation,” in *IEEE ICASSP*, 2015, pp. 574–578.

[6] D. J. Hermes, “Measurement of pitch by subharmonic summation,” *The Journal of the Acoustical Society of America*, vol. 83, no. 1, pp. 257–264, 1988.

[7] M. Goto, “A real-time music-scene-description system: Predominant-f0 estimation for detecting melody and bass lines in real-world audio signals,” *Speech Communication (ISCA Journal)*, vol. 43, no. 4, pp. 311–329, 2004.

[8] J. Salamon and E. Gómez, “Melody extraction from polyphonic music signals using pitch contour characteristics,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[9] J.-L. Durrieu, G. Richard, B. David, and C. Févotte, “Source/filter model for unsupervised main melody extraction from polyphonic audio signals,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 564–575, 2010.

[10] M. Mauch and S. Dixon, “pYIN: A fundamental frequency estimator using probabilistic threshold distributions,” in *IEEE ICASSP*, 2014, pp. 659–663.

[11] R. Schramm, A. Mcleod, M. Steedman, and E. Benetos, “Multi-Pitch Detection and Voice Assignment for a Cappella Recordings of Multiple Singers,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 552–559.

[12] E. Molina, L. J. Tardón, A. M. Barbancho, and I. Barbancho, “Sipth: Singing transcription based on hysteresis defined on the pitch-time curve,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 23, no. 2, pp. 252–263, 2015.

[13] L. Yang, A. Maezawa, J. B. L. Smith, and E. Chew, “Probabilistic transcription of sung melody using a pitch dynamic model,” in *IEEE ICASSP*, 2017, pp. 301–305.

[14] N. Kroher and E. Gómez, “Automatic transcription of flamenco singing from polyphonic music recordings,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 5, pp. 901–913, 2016.

[15] H. Takeda, N. Saito, T. Otsuki, M. Nakai, H. Shimodaira, and S. Sagayama, “Hidden markov model for automatic transcription of MIDI signals,” in *IEEE Workshop on Multimedia Signal Processing (MMSP)*, 2002, pp. 428–431.

[16] C. Raphael, “A hybrid graphical model for rhythmic parsing,” *Artificial Intelligence*, vol. 137, no. 1-2, pp. 217–238, 2002.

[17] M. Hamanaka, M. Goto, H. Asoh, and N. Otsu, “A learning-based quantization: Unsupervised estimation of the model parameters,” in *International Conference on Multimodal Interfaces (ICMI)*, 2003, pp. 369–372.

[18] R. Nishikimi, E. Nakamura, M. Goto, K. Itoyama, and K. Yoshii, “Scale- and rhythm-aware musical note estimation for vocal F0 trajectories based on a semi-tatum-synchronous hierarchical hidden semi-markov model,” in *International Society for Music Information Retrieval Conference, (ISMIR)*, 2017, pp. 376–382.

[19] E. Nakamura, R. Nishikimi, S. Dixon, and K. Yoshii, “Probabilistic sequential patterns for singing transcription,” in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2018.

[20] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *IEEE ICASSP*, 2016, pp. 4960–4964.

[21] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 577–585.

[22] R. Prabhavalkar, T. N. Sainath, B. Li, K. Rao, and N. Jaitly, “An analysis of ”attention” in sequence-to-sequence models,” in *INTERSPEECH*, 2017, pp. 3702–3706.

[23] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015, pp. 1412–1421.

[24] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *International Conference for Learning Representations (ICLR)*, 2015, pp. 1–15.

[25] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, “RWC Music Database: Popular, classical and jazz music databases,” in *International Conference on Music Information Retrieval (ISMIR)*, 2002, pp. 287–288.

[26] M. Goto, “AIST annotation for the RWC music database.,” in *International Conference on Music Information Retrieval (ISMIR)*, 2006, pp. 359–360.

[27] H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition,” in *INTERSPEECH*, 2015, pp. 1468–1472.

[28] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *arXiv preprint arXiv:1412.6980*, 2014, pp. 1–15.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034.

[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.