VirtuosoNet: A Hierarchical Attention RNN for Generating Expressive Piano Performance from Music Score

Dasaem Jeong Taegyun Kwon Juhan Nam
Graduate School of Culture Technology
KAIST
Daejeon, Republic of Korea
{jdasam, ilcobo2, juhannam}@kaist.ac.kr

Abstract

Interpreting and performing of music score is a challenging task for computers. We propose a musically structured hierarchical attention network to generate expressive piano performance in MIDI format given symbolic music scores such as musicXML. The network takes a sequence of input features extracted from each note in the score and returns performance parameters for the note. The model can render various expressive elements in music performance, including tempo change, dynamics, micro-timing of individual notes, and pedal control.

1 Introduction

Expressive performance is one of the most sophisticated musical activities. It demands the understanding of not only explicit symbolic information such as note and rhythm in the music score but also more abstract concepts such as phrasing and performance practice. Human musicians are trained over years to play music accurately while delivering emotions. There are a variety of competitions in which musicians perform the same pieces with their own interpretations and be judged of their musicality by jury.

There has been much research that challenged the modeling of expressive performance by computers. Among others, piano has been the most targeted musical instrument. An early approach introduced by Widmer *et al.* is a Bayesian model called *YQX* [1]. The model takes notes of a score as input and estimates inter-onset-interval, velocity, and articulation of each note for expressive piano performances. Recent approaches have mainly used neural networks for the modeling. Malik and Ek proposed a stylistic performance model based on recurrent neural network (RNN) and Siamese network, but they focused on note dynamics only. Simon and Oore introduced *Performance RNN*, an auto-regressive model directly learned on performance MIDI notes that contain expressive timing and dynamics [2]. However, the model did not take the score as input and thus the result was close to automatic musical composition with expressiveness. More recently, Maezawa proposed a model based on conditional variational autoencoder and showed promising results [3]. In this paper, we present a generative RNN model with musically structured hierarchical attention layers. Unlike the previous research, our model is capable of estimating absolute tempo, local time deviation of individual notes, velocity of individual notes, and control of sustain and soft pedals.

2 Feature Representations

To feed score input to the network, we define a list of input features. The features are encoded into every individual note. The entire notes are arranged as an one-dimensional sequence according to their time positions and pitch orders. Note-level features include pitch, duration, and relative position of a note in the measure. The tempo and dynamic markings in musicXML score such as *allegro, ritardando, ff, più f, crescendo* are embedded as a single vector. After calculating the start and end position of each marking, the embedded vector is concatenated to the correspondent note features. To handle various kinds of musical expressiveness terms, we made heuristic rules to convert them based on musical knowledge.

Since the same score can be performed with various global tempo and dynamics, we added global conditioning features to normalize diversity. It helped our model to focus on modeling relative changes based on given tempo and dynamics. In our preliminary experiment, we found that the network was not trained satisfactorily without those conditions. During the generation, users can select a specific value for these features to decide the overall tempo and dynamics of the performance.

To model human performances, we used performance features such as tempo, velocity, micro onset deviation, duration, and pedal usage. Each performance feature of note matches one-to-one with the score feature of the corresponding note, resulting in perfectly aligned sequence pairs. We should note that even though we apply heuristic rules to encode musical marking, the whole process is fully automatic and generally applicable to most of the piano scores. Furthermore, since we treat every single note individually, we can express every detail of performance without almost any loss of information. The details and full list of features can be found in the appendix sections.

3 Learning Model

Given a pair of note sequence in score $\mathbf{x} = \{x_1, ..., x_N\}$ and corresponding performance sequence $\mathbf{y} = \{y_1, ..., y_N\}$, global condition vector $\mathbf{c}$ can be calculated. VirtuosoNet generate the performance in an auto-regressive manner by modeling conditional probability $p(y_t|y_1, ..., y_{t-1}, \mathbf{x}, \mathbf{c})$.

The model consists of two parts. The first part encodes a sequence of notes inputs. The second part makes a prediction of performance parameters for the corresponding notes, based on previous performance and encoded score notes.

The score encoding part is based on Hierarchical Attention Network (HAN) model proposed by Yang *et al* [4]. HAN composes a higher-level of layer via attention network. We implemented three hierarchical levels in note, beat, and measure leveraging the prior knowledge of musical structure. Along with the networks, we added another RNN module that takes voice-separated note sequences as an input. All RNN modules in this score encoding parts are bidirectional LSTM.

We used performance MIDI data from Yamaha Piano-e-Competition [5] and symbolic score files from MuseScore website in musicXML format [6]. The data set consists of 25 pieces, all composed by Chopin, and total 217 performances. 80% was used for training and the remaining for validation. We augmented the dataset by randomly transposing the key twice for each performance. We matched the performance MIDI to musicXML in note level using Nakamura's MIDI-to-MIDI alignment algorithm [7] after converting the musicXML file into a MIDI format using MuseScore.

During the training, the true previous token y_{t-1} was given as a random probability of 30 % instead of the model's previous estimation $\hat{y}_{t-1}$ as an input for the prediction layer, which is similar to the strategy proposed by Bengio *et al.* [8] but using fixed random probability throughout the training. The input was normalized by its mean and variance. We used mean square error as a loss function.

4 Results

We tested the proposed model with the pieces that are not included in the model[1]. The test results showed some of typical characteristics in expressive performance such as natural progress of dynamics, emphasizing melody notes even when the melody notes is played in bass part, and tempo modification for natural phrasing. Also, the model showed clear differences according to the tempo and dynamics markings in the score. Furthermore, the model could take user inputs such as the beginning tempo or overall velocity and consider them during its generation process. For the future research, we will work on qualitative and quantitative evaluation of the results, including a Turing test.

[1]The demo videos of the piano performance results are available on `https://www.youtube.com/playlist?list=PLkIVXCxCZ08rD1PXbrb0KNOSYVh5Pvg-c`.

Acknowledgement

We used a musicXML parser developed by Magenta [9] and modified it for our use. This research was supported by Samsung Research Funding and Incubation Center for Future Research.

References

[1] Widmer, G., S. Flossmann, M. Grachten. YQX plays chopin. *AI magazine*, 30(3):35, 2009.

[2] Simon, I., S. Oore. Performance RNN: Generating music with expressive timing and dynamics. `https://magenta.tensorflow.org/performance-rnn`, 2017.

[3] Maezawa, A. Deep piano performance rendering with conditional VAE. In *19th International Society for Music Information Retrieval Conference (ISMIR) Late Breaking and Demo Papers*. 2018.

[4] Yang, Z., D. Yang, C. Dyer, et al. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. 2016.

[5] Yamaha disklavier education network: Signature midi collection. `http://www.yamahaden.com/midi-files`. Accessed: 2018-10-28.

[6] MuseScore sheet music. `https://musescore.com/sheetmusic`. Accessed: 2018-10-28.

[7] Nakamura, E., K. Yoshii, H. Katayose. Performance error detection and post-processing for fast and accurate symbolic music alignment. In *18th International Society for Music Information Retrieval Conference (ISMIR)*. 2017.

[8] Bengio, S., O. Vinyals, N. Jaitly, et al. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179. 2015.

[9] Magenta MusicXML parser. `https://github.com/tensorflow/magenta/blob/master/magenta/music/musicxml_parser.py`. Accessed: 2018-10-28.

A Feature Detail

The list of input features are: **a)** pitch in octave (int) and pitch class as one-hot vector **b)** duration **c)** pitch interval to next note (semitone) **d)** duration ratio to next note (log) **e)** relative position in measure [0 - 1] **f)** relative position in entire piece [0 - 1] **g)** measure length **h)** duration of following rest **i)** not followed by note in the same voice(0 or 1) **j)** time signature **k)** duration after the corresponding dynamic marking **l)** tempo marking vector (3-D vector) **m)** dynamic marking vector (4-D vector) **n)** notation marking vector for trill, fermata, accented, strong accented, staccato, tenuto, arpeggiate (7-D binary vector) Every duration or length is measured in a quarter note.

These global conditioning features are added to input features along with the corresponding score features: **a)** embedded tempo vector of the beginning of the piece **b)** tempo of the first twenty beat **c)** mean value of embedded dynamics vector of notes with *ppp, pp, p, mp* **d)** mean value of embedded dynamics vector of notes with *mf, f, ff, fff* **e)** mean velocity of notes marked with *ppp, pp, p, mp* **f)** mean velocity of notes marked with *mf, f, ff, fff*.

The output features are: **a)** Tempo in quarter note per minute, calculated for every beat (log) **b)** MIDI velocity **c)** deviation of the note onset compared to 'in-tempo position' in quarter note **d)** articulation **e)** value of sustain pedal at note onset [0 - 127] **f)** value of sustain pedal at note offset [0 - 127] **g)** smallest sustain pedal value between the note's onset and the offset [0 - 127] **h)** relative time position of the moment g) is detected (0 - 1) **i)** smallest sustain pedal value between the note offset and its closest next note onset **j)** elapsed time between the note offset and the moment i) is detected **k)** value of soft pedal at note onset.

We used additional features for modeling a trill note. The trill features are **a)** number of trill notes **b)** relative duration ratio of the first trill note **c)** relative duration ratio of the last trill note **d)** whether the trill starts with higher note (alternative trill).

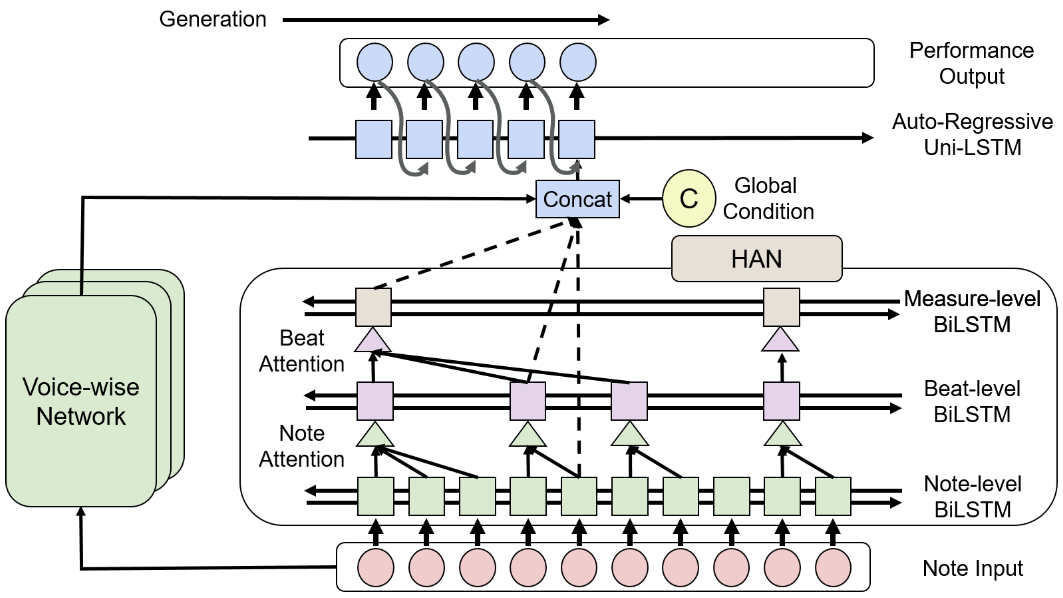

Figure 1: Network configuration of Hierarchical Attention Network (HAN)

B Network Detail

An illustration of the proposed model is shown in Figure 1. The Network takes temporally ordered sequence of score notes as input. For each note, the bidirectional LSTM units step forward along with it. On each beat, the outputs of the note-level LSTM within the beat are summarized by attention mechanism. Next layer, the beat-level LSTM takes outputs of the note-level attention. Similarly, the outputs of the beat-level layer are also fed into measure-level layer via the beat-level attention. To generate a performance sequence, performance features are generated note by note, with auto-regressive connection in the final LSTM layer. For each step, the outputs of cells in the corresponding note, beat and measures are concatenated with the output of voice-wise network, and offers score information. Global conditioning features are also incorporated in here.

In voice-wise network, notes are separated by its voice in MusicXML. Unlike HAN, it contains only note layers, and each layer only take notes in specific voice. Every layer shared its weight. By separating voice, notes in each sequence could have consecutive melody notes, which might helpful for RNN to model musical relation.

The note level LSTM is in 4 layers with size of 64 hidden units. The beat level LSTM consists of 2 layers with size of 32. The measure level LSTM is a single layer with 16 hidden units. The voice-wise network is double-layer LSTM with 64 hidden units. The final unidirectional LSTM is a single layer with 24 hidden units.

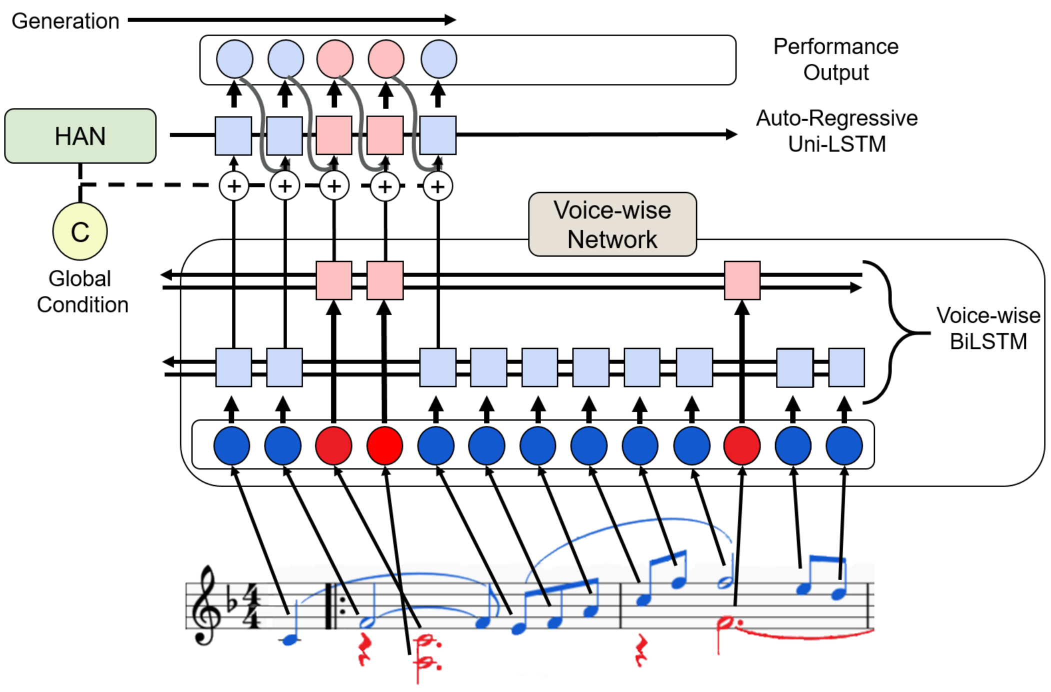

Figure 2: Network configuration of voice-wise network.