VirtuosoNet: A HIERARCHICAL RNN-BASED SYSTEM FOR MODELING EXPRESSIVE PIANO PERFORMANCE

Dasaem Jeong[1] **Taegyun Kwon**[1] **Yoojin Kim**[1]
Kyogu Lee[2] **Juhan Nam**[1]
[1] Graduate School of Culture Technology, KAIST , Korea
[2] Graduate School of Convergence Science and Technology, Seoul National University, Korea
{jdasam, ilcobo2, luciaicul}@kaist.ac.kr, kglee@snu.ac.kr, juhannam@kaist.ac.kr

ABSTRACT

In this paper, we present our application of deep neural network to modeling piano performance, which imitates the expressive control of tempo, dynamics, articulations and pedaling from pianists. Our model consists of recurrent neural networks with hierarchical attention and conditional variational autoencoder. The model takes a sequence of note-level score features extracted from MusicXML as input and predicts piano performance features of the corresponding notes. To render musical expressions consistently over long-term sections, we first predict tempo and dynamics in measure-level and, based on the result, refine them in note-level. The evaluation through listening test shows that our model achieves a more human-like expressiveness compared to previous models. We also share the dataset we used for the experiment.

1. INTRODUCTION

Music performance is one of the most essential activities in music. Good performance requires not only translating notes in the score into physical actions with precise timing and right pitch on an instrument but also delivering emotions and messages through subtle controls of tempo, dynamics, articulations and other expressive elements.

There have been research interests in modeling expressive performance using a computational method. A recent review paper comprehensively summarized the history [4]. While some of previous work exploited computational modeling as a tool for understanding how humans perform [3], or listen to music [10], others focused on automatically generating expressive performances. The previous methods include rule-based approaches [2, 8], or probabilistic models [17, 29], and an artificial neural network [5, 11]. The instrument is mainly limited to piano because it is relatively easy to quantify the performances.

Recent approaches have attempted to apply deep learning to modeling expressive piano performance, such as rendering note velocity and deviation of note onset with vanilla recurrent neural network (RNN) [20], or predicting note velocity with a long short-term memory (LSTM) RNN [22]. Others introduced DNN models for generating polyphonic music with expressive timing and dynamics [13, 24]. While these models can generate performance MIDI notes, they are more like music composition models rather than expressive performance models that take music scores as input. Besides piano performance, a recent work presented DNN-based system for modeling expressive drum performance [9].

One of the bottlenecks in the DNN-based approach is the lack of dataset [4]. Since the task is rendering expressive performances from score inputs, the dataset should consist of music scores and their corresponding performances by human musicians. Furthermore, the pair of score and performance should be aligned in note-level to effectively train the model. Also, ideally, the list of music score and performance should cover various composers and performance styles.

In this paper, we present a hierarchical RNN-based model for expressive piano performance along with a dataset that we organized. The model takes MusicXML as input and generates performance MIDI with expressive tempo, dynamics, articulation and pedaling. The model consists of RNN with hierarchical attention network and conditional variational autoencoder (CVAE). In particular, the model predicts the performance features using a multi-scale approach; it first predicts tempo and dynamics in measure-level and, based on the result, fine-tunes them in note-level. A listening test with professional pianists shows that our model achieves a more human-like expressiveness compared to previous models.

2. DATASET

2.1 Performance and Score Data

As aforementioned, we need a dataset of human performances with their corresponding music scores to train a neural network model. A list of expressive performance datasets are summarized in [4]. Among others, Yamaha

Signature MIDI collection [1], which are recorded during Yamaha e-Competitions with computer-controlled pianos, is the largest public dataset that provides a substantial amount of expressive performance MIDI of professional pianists. Some of the pianists performed the same piece more than once in different rounds of the competition in different years. The Yamaha collection has been employed in automatic performance generation [24] and automatic music transcription and audio synthesis [12] as well.

While the Yamaha collection provides high-quality piano performance data in MIDI, it does not contain the corresponding music scores of the pieces. Thus, we collected the score files from another source. Specifically, we downloaded them from *MuseScore*, a community-based web platform of music score [2]. The scores were transcribed voluntarily by the community users and can be exported in MusicXML format. We also included our own transcriptions of scores to the dataset. While MIDI is suitable for representing performance, MusicXML aims to represent the Western music notation in its entirety. Therefore, MusicXML can contain various types of musical symbols such as rest, slur, beam, barline, key and time signature, articulation, ornament markings and so on, which are excluded in MIDI format.

2.2 Data Matching and Refinement

Since we collected the performance and score data from different sources, we had to match and refine them. In particular, transcription styles in the crowdsourced MusicXML files are not consistent. For example, some of the transcribers add extra expressions such as dynamics markings or tempo change to make the score sounds more expressive. They usually set them to "invisible objects" to make the transcribed score appear as the reference score. We deleted such extra markings added by transcribers. Also, we manually checked whether the performances followed the repetitions in the scores. If a performance skipped the repetition, we omitted the repetition from the score so that the performance and the score can be aligned.

To train a model with note-level score features and performance features, each note in the score should be matched to that in the performance. We employed a score-to-performance alignment algorithm proposed by Nakamura et al. [23]. The algorithm automatically handles asynchronously performed notes as well as missing and extra notes in the performance, and returns a list of note-to-note matches. Although the algorithm showed high accuracy in our test, a small amount of alignment errors can be critical in extracting performance features such as tempo or onset deviation. Since the dataset is too large to make manual corrections, we filtered out some erroneous matches based on simple rules and excluded them in training the performance model. For example, if a matched performance note is too close or even earlier than the previous note in the score, we regarded it as an alignment error. Also, if multiple notes have the same onset time in the score (e.g., chord notes) but one is too far from other notes in performance, we regarded it as an alignment error as well.

We found that this additional refinement made severe improvement on the training result, especially on onset deviation, or micro-timing, of individual notes. The standard deviation of onset deviation decreases from 7.369 to 0.053 after the refinement, where the unit is quarter-notes. Without the refinement, the prediction of onset deviation became too noisy that one could not perceive correct rhythm.

As a result, we collected music scores of 226 pieces by 16 composers in MusicXML and 1,052 piano performances in MIDI. After the matching and refinement, the score and performance data contain a total of 666,918 notes and 3,547,683 notes, respectively. Among the performance notes, 131,095 notes were failed to be aligned with score notes, and additional 114,914 notes were excluded by our refinement algorithm. The number of valid performance notes is ten times larger than the Magaloff corpus [7], which is the largest existing dataset for classical piano music [4].

3. SYSTEM ARCHITECTURE

3.1 Background

3.1.1 Input and Output Features

Designing input and output features is an important issue in performance modeling because it defines the characteristics of the computational task [4]. We followed the scheme we previously proposed in [15], which covers a wide range of score and performance features. The score features include pitch, duration, articulation marking, slur and beam status, tempo marking, dynamic markings, and so on. The performance features include absolute tempo, velocity, onset deviation, articulation and pedal usages. All the features are encoded in the note-level so that each note had the same dimension of score features and performance features.

3.1.2 Hierarchical Attention Network

Recent research has shown that a hierarchical approach can improve the performance of RNN model in modeling sequential data [6, 30]. It was also demonstrated that the hierarchical approach has advantages in generating symbolic music data [25]. In this paper, we employ a hierarchical attention network (HAN) to predict a sequence of performance features from a sequence of score features.

The HAN composes higher-level representations by summarizing lower-level representations in pre-defined hierarchical boundaries using a weighted sum. In our case, we set beat and measure as the hierarchical boundaries so that beat-level attention and measure-level attention summarize note-level and beat-level representations, respectively. Instead of directly implementing the HAN in [30], we combined it with the idea of multi-head attention [28] which splits the dimension into several heads and applies different weights of attention for each split.

Composing nodes through the attention layers can be described as follows. For each hierarchical boundary,

[1] http://www.yamahaden.com/midi-files
[2] https://musescore.com

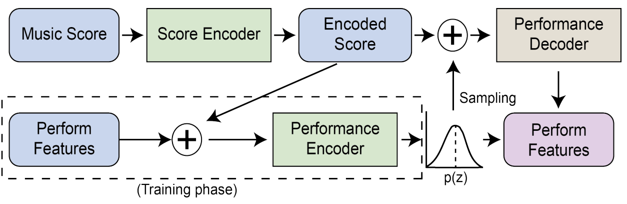

Figure 1. The overview of the proposed system.

which can be a beat or a measure in music score, the notes in the boundary can be indexed with $t \in [B_f, B_l]$, where B_f and B_l represent the index of the first and last notes in the selected boundary B. The lower-level hidden states $\mathbf{h}_t$ for t in the boundary B are summarized by context attention to compose a higher-level node $\mathbf{m}$. There are a total I number of attention heads indexed with i.

$$\begin{aligned}
\mathbf{u}_t &= \tanh(\mathbf{W}_a\mathbf{h}_t + \mathbf{b}_a)\\
\mathbf{u}_t^i &= \mathbf{u}_{t,i:(i+1)d}\\
\mathbf{h}_t^i &= \mathbf{h}_{t,i:(i+1)d}\\
\boldsymbol{\alpha}_v^i &= \frac{\exp(\mathbf{u}_t^{i\intercal}\mathbf{u}_c^i)}{\sum_t \exp(\mathbf{u}_t^{i\intercal}\mathbf{u}_c^i)} \qquad (1)\\
\mathbf{m}^i &= \sum_t \boldsymbol{\alpha}_t^i * \mathbf{h}_t^i\\
\mathbf{m} &= \text{Concat}(\mathbf{m}^0, ..., \mathbf{m}^I)
\end{aligned}$$

where $\mathbf{W}_a$ and $\mathbf{b}_a$ denote weight and bias parameters of attention, and $\mathbf{u}_c$ denotes a context vector representing query for importance, which are trainable parameters. The sequence of summarized nodes are fed into a new layer of LSTM.

3.1.3 Conditional VAE

A music score can be interpreted and performed in various styles, i.e. with a different tempo or phrasing. Therefore it is important to enable the performance modeling system to generate different types of performance. On the other hand, the variation of performance can be an obstacle for training the model, because it has to generate different outputs from the same input. To solve this problem we employed a conditional variational autoencoder (CVAE), which we proposed in our previous work [14].

VAE is a widely used generative models based on deep neural networks [19]. It is a type of autoencoder, which compresses input information into a lower dimensional latent vector and decodes the original information from the compressed latent vector. The main difference is that VAE constrains its latent vector to be sampled from a probability distribution. VAE consists of an encoder that models $q(z|x)$ and decoder to model $p(x|z)$. VAE also models the probability of latent vector $p(z)$, which usually has a normal distribution. The training loss of VAE can be define as follows:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{rec}} + \beta D_{KL}[(q(z|x)||p(z)] \qquad (2)$$

where $\mathcal{L}_{\text{rec}}$ is the reconstruction error from AE, D_{KL} is Kullback-Leibler divergence (KLD), and β is a weight for the KLD.

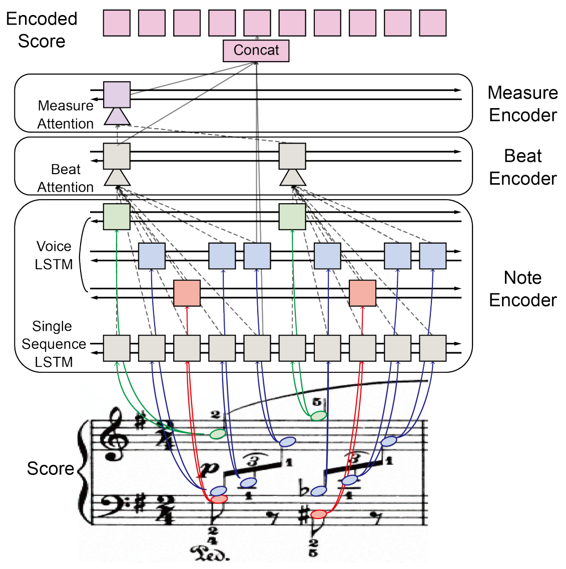

Figure 2. Diagram for Score Encoder with HAN and RNN

A conditional VAE (CVAE) provides an additional condition so that the output satisfies the given condition [27]. In our system, the condition is the learned score representation, and the target output are the performance features. The idea of employing CVAE for expressive performance modeling was first proposed in [21]. While the previous work encoded the latent vector in note-level, our idea is to encode the performance style in a longer-level, such as an entire piece.

3.2 Proposed System

Our proposed system consists of three parts: score encoder, performance encoder, and performance decoder as depicted in Figure 1.

The role of score encoder is to learn score representations C from an input sequence of notes. It consists of three hierarchical-levels: note, beat, and measure. Each level has a corresponding bidirectional LSTM unit with a different hidden size and number of layers. The note-level layer consists of two different LSTM units, one taking the input as a single sequence, and the other taking the input as voice-separated sequences. The "voice" means the voice index in MusicXML that represents an independent stream of music as depicted with different colors of notes in Figure 2. The hidden representations of the lower-level are summarized through the HAN to compose higher-level nodes. The output of the note-level LSTM is summarized to beat-level nodes and then they are fed into the beat-level LSTM. Similarly, we compose the measure-level LSTM. We concatenate the outputs of all the three layers in a note-level as depicted in Figure 2. The output of score encoder is a sequence with the same length as the input. Since we use multi-head attention instead of single-head attention, each attention head focuses on the different type of notes as illustrated in Figure 3.

We implemented the performance encoder using CVAE that models $q(z|C, y)$ to summarize the given performance y in score condition C to a probability distribution of the

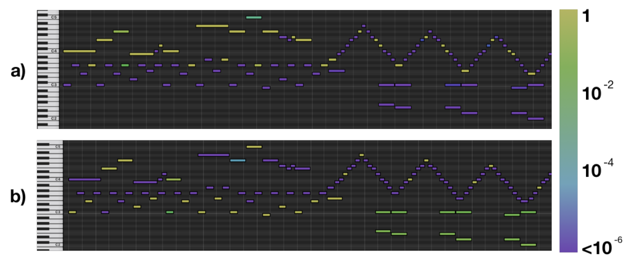

Figure 3. Visualization of attention weights from different attention heads. a) focuses more on the melody notes while b) focuses more on the bass or harmonic notes.

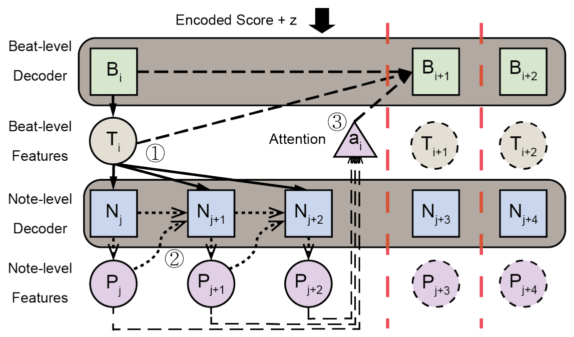

Figure 4. The figure shows how the beat-level decoder and the note-level decoder feed its results to the other. The dashed lines in red indicate the edge of beats.

latent vector z, which can be regarded as a performance style vector. C and y are concatenated and fed into a single dense layer that contracts the feature dimension. We use uni-directional note-level LSTM and measure-level HAN-LSTM to process the contracted input. The last output of the sequence from the measure-level LSTM is used to infer μ and σ of $q(z|C, y)$ by a dense layer.

During the actual performance generation from a given score, the performance encoding is bypassed, and the system randomly samples the style vector z from a normal distribution or exploits a pre-encoded z from other performances.

The performance decoder uses LSTMs to generate a sequence of performance features $\hat{y}$ for the given condition C and the style vector z. Since the tempo is always estimated in beat level, we have two different LSTM units, one in the beat-level and the other in the note-level. Both LSTMs are in auto-regressive, i.e., take their own output from the previous step as an input, and the outputs of the note-level decoder is fed into the beat-level decoder, and vice versa, as presented in Figure 4.

3.3 Measure-level Module

One of the main difficulties in expressive performance modeling is achieving long-term expression such as gradual change of tempo or contrast between loud and quiet sections. To solve this problem, we propose an optional measure-level module that predicts measure-level tempo and dynamics as presented in Figure 5. The main idea is to make our system predict overall progress of the performance in measure-level and then refine it in note-level. A similar idea achieved a successive result in image generation using GAN, which started training in a low resolution and progressively in higher resolutions [16].

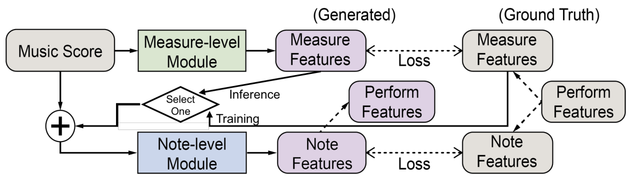

Figure 5. Diagram for Measure-level modules

To train the measure-level module, we have to define measure-level performance features. The measure-level tempo is defined by elapsed time to play the measure divided by the length of the measure in quarter-notes. We used average velocities of notes in the measure for a measure-level dynamics. The measure-level module has almost the same architecture with the note-level modules except that the output of the score encoder is the measure-level states instead of concatenated result of note, beat and measure hidden states. The performance encoder and decoder are also in measure level.

In this hierarchical approach, the note-level module takes not only the score data but also the output of the measure-level module as a concatenated input. It is possible to combine two modules as a single model or in a single training process, but we made two modules independent and trained them separately. Therefore, the note-level module is trained with ground-truth measure-level outputs.

4. EXPERIMENTS

4.1 Training

We split the dataset into training, validation, and test sets so that each set has a size of approximately 8:1:1 in the number of piece, performance, and notes, while considering the distribution of composers in each set. A single piece was included only in either of one of the splits. For the training set, we sliced the input sequences at the measure boundaries with the least size of 500 notes. When training the measure-level module, the sequence has at least 2000 notes or entire notes if the piece is short. The note is ordered by its appearance order and pitch. The features with continuous value was normalized to have zero mean and unit standard deviation.

We calculated the loss in mean square error (MSE) between each feature. The loss was calculated for each note and each output features, except the tempo, whose loss was calculated in beat level. During the training, the input sequences included all the notes that have non-matching performance notes, because missing notes in the input data can change the context of the other notes in the score. However, these notes were excluded in the loss calculation because we could not extract performance features for the notes. Since the articulation is largely affected by the sustain pedal, we reduced the weight for the articulation loss to 0.1 for notes with the sustain pedal pressed at the offset.

Model	Tempo	Vel	Dev	Artc	Pedal
Baseline	0.400	0.673	0.773	0.721	0.843
HAN-S	0.269	0.607	0.753	**0.688**	0.820
HAN-M	**0.220**	**0.532**	**0.747**	0.754	**0.810**

Table 1. Reconstruction loss of each model on the test set in MSE. Vel, Dev and Artc denote velocity, onset deviation and articulation, respectively.

We used the ADAM optimizer [18] with an initial learning rate of 0.0003 and dropout ratio of 0.1. To avoid that the system bypasses z during the decoding, we use the KLD weight annealing as proposed in [1], so that the KLD weight started from zero at the beginning of training, and increased to 0.02 gradually.

4.2 Model Configuration

The score encoder of our proposed system has three-layer dense network of size 128 with ReLU activation as an embedding layer, two-layer bidirectional(Bi)-LSTMs of size 128 for note-level and voice-level, two-layer Bi-LSTM of size 64 for beat-level, and one-layer Bi-LSTM of size 64 for measure-level. The performance encoder has two-layer unidirectional(Uni)-LSTM of size 16 for note-level and one-layer Uni-LSTM of size 16 for measure-level. The size of latent vector z in CVAE is 16. The performance decoder consists of one-layer Uni-LSTMs for beat-level and note-level both of size 64. The measure-level module has almost the same setting except that every hidden size of the network in the performance encoder is 8 including the latent vector z.

To compare our approach with HAN architecture and measure-level modules (HAN-M), we also trained two other models. The first model is a baseline model that uses only three-layers LSTM in note-level with hidden size of 256. The other model, which will be denoted as HAN-S, is a model that excludes the measure-level module. In HAN-S, the hidden size of beat-level layer in the score encoder and performance decoder was 128.

5. RESULTS

5.1 Reconstruction Error

Quantitative evaluation of modeling expressive performance is a not trivial issue. One of the frequently used quantitative evaluation method is calculating MSE of output features [4]. Comparing the predicted outputs with "target" performance can be arbitrary, because there can be various ways to perform the score. In our system, however, there is a performance encoder and a latent style vector z that, ideally, makes the output in a style of the target performance. Therefore, comparing output features with the target performances is more reasonable. Also, as a learning model, it is fair to check the test loss with the same criteria used for training.

Table 1 shows the reconstruction loss of each model on the test set which includes 21 pieces and 109 performances. The two HAN models achieved much less reconstruction error than the baseline model. This indicates that the hierarchical approach helps the model to generalize to unseen data. Between the two HAN models, HAN-M is slightly better than HAN-S. We have tested different parameter sizes for HAN-S so that HAN-S has a similar number of parameters with the sum of two modules in HAN-M, but the result was not much different.

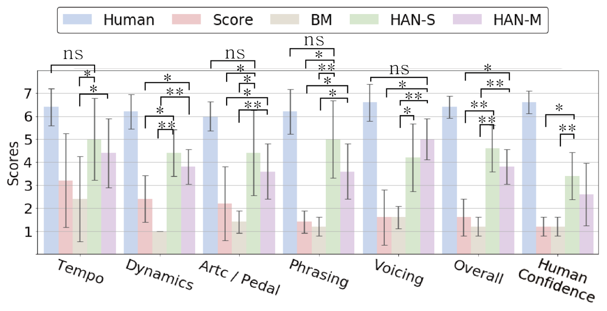

Figure 6. Average score of the listening test for Schubert Sonata in 7-point Likert scale. The t-test results between our models (HAN-S,M) and Human are marked with ns only if it is not significant. The results between our models and the others models are marked only if it is significant. "*" and "**" denote "p≤0.05" and "p≤0.01", respectively.

5.2 Listening Test

We also conducted a listening test to evaluate our model qualitatively. We asked five students, who are majoring piano at a college of music, to listen to the rendered performances and evaluate them with criteria presented in Figure 6 in 7-point Likert scale (1 - very bad, 7 - very good) with additional comments on the performance. We chose three pieces of different styles from our test set: the first movement from Beethoven's Piano Sonata No. 5 (cut before recapitulation), Chopin's Etude op. 10 No. 2 (entire piece), and the first movement from Schubert Piano Sonata D.664 (cut before development).

We prepared five different performances MIDI per piece: a human performance from Yamaha e-competition, a direct export from MusicXML score to MIDI by a notation program (MuseScore), each of rendered performances from HAN-S and HAN-M, and Basis Mixer (BM). The result exported from MuseScore had no tempo change but the velocities of notes were changed by a simple rule-based conversion of dynamic markings in the score. BM is the only publicly available model that does not require additional notation among previous expressive performance models [5]. It also achieved a highest score among other computational methods in previous research [26]. We included the BM model in the listening test and generated performances with the same MusicXML file we used for our model[3]. Since the recording and playback in audio systems can affect the quality of performance, we invited the participants to our studio and played the prepared MIDI files with a Yamaha Disklavier piano. Each performance was played once in a random order.

The result of evaluation on Schubert is presented in Figure 6 As expected, all participants gave highest scores in

[3] https://basismixer.cp.jku.at/static/app.html

every criteria for the human performance. Our proposed model, HAN-S and HAN-M, achieved higher scores in all seven criteria compared to other models. Two among five participants gave more than five out of seven points as the human confidence for the performance by HAN-S, and one gave five points to HAN-M. The t-test result showed that HAN-S and HAN-M showed statistically significant differences ($p \leq 0.05$) compared to the score and the BM model in overall ratings of the performance of Schubert.

The positive comments on our models were: *"the interpretation was interesting"* (Beethoven, HAN-M), *"felt that the flow of performance was humane"* (Beethoven, HAN-S), *"voicing was too good so that it felt like performed by multiple performers"* (Chopin, HAN-S), *"sounded like a performance by human with strong characteristics"* (Chopin, HAN-S), *"voicing was fine except some faults"* (Schubert, HAN-M), and *"sounded like machine-generated performance with fine pedaling"* (Schubert, HAN-S) .

There were also negative comments criticizing our models, such as *"used too much pedal"*, *"pedal points were unnatural"*, *"lack of color"*, *"too short articulation"*, *"some tempo or dynamic changes were unnatural"*, *"touch was too light"*, and *"it did not seem that the performer was listening to the performance"*. Although our models had predicted the pedal usage, the pedaling was often too deep and "dirty" or too shallow. The result showed that the note-level pedal embedding needs improvement.

The responses of our participants for performances by BM, which is a data-driven model based on RNN, were negative regardless of the piece. The comments from the participants said that *"although there was a clear intention to express phrasing, it was unnatural and sounded like a mechanical interpretation"* (Schubert), *"inaccurate and limping rhythm"* (Beethoven and Schubert),and *"the temporal gaps at measure boundary were unnatural"* (Chopin and Schubert). Unlike the Score MIDI, the performance by BM included clear change in tempo for phrasing. However, most of the participants gave almost the same level of negative response to its phrasing quality compared to the Score MIDI. This shows how difficult it is to model phrasing of the music.

The results were also largely differed by the characteristics of the piece. For example, Schubert's Sonata has a song-like melody with arpeggio accompaniments. Hence, it was important to model the natural phrasing, e.g., subtle change of tempo and velocity according to the melody. On the other hand, the fast chromatic scale in Chopin's Etude demands a stable tempo. Therefore, Score MIDI received six out of seven points for overall quality from three participants because the performance was in perfectly constant tempo with strict following of dynamic markings. The flexibility of tempo generated by our model was not favored by the participants in case of Chopin's Etude.

In summary, the results of listening test shows that our models have achieved more natural expressions compared to the other models, especially in a piece with song-like melodies. Modeling the pedal usage and a human-like stable tempo are issues to further investigate.

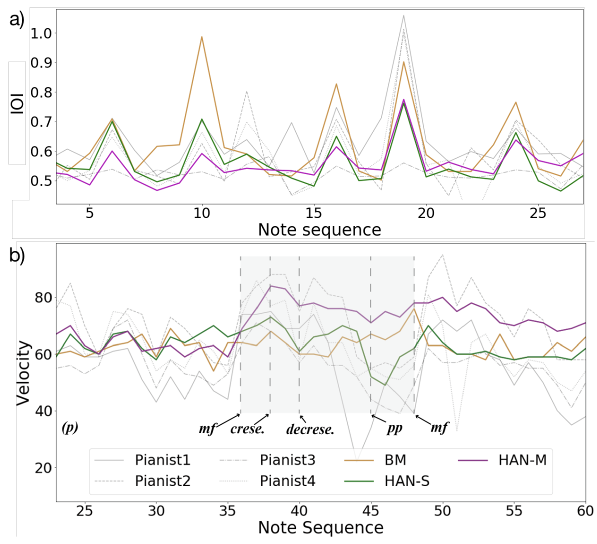

Figure 7. a) Local tempo changes and b) Dynamics change in different performances of Schubert's Piano Sonata

5.3 Case Study: Comparison in Tempo and Dynamics

The quality of phrasing can be also observed from examples. Figure 7-a) compares local tempo changes in difference performances of Schubert's Sonata. The local tempo is represented with inter-onset-interval (IOI) which is computed by dividing seconds into quarter-note. BM has an evident peak at around the 10th note, which was exaggerated than any other human pianists. In terms of Pearson correlation, HAN-S and HAN-M have a strong positive correlation with the pianists ($0.7<r<1.0$) while the BM model has a less positive correlation ($0.3<r<0.5$).

Figure 7-b) compares dynamics changes of melody notes in different performances of the same piece. The dynamics is represented with MIDI note velocity. Increasing and decreasing timings of HAN-M and HAN-S are generally similar to pianists. For example, *decrescendo* starts at note sequence 40 which follows *crescendo*, then *pp* starts at 45 and comes back to *mf* at 48. Both HAN-M and HAN-S show similar downward and upward curves with pianists while the BM model shows just slight upward curve. These trend can be proved by correlation coefficients among pianists and generated models. HAN-S and HAN-M have significant positive correlation with pianists ($0.3<r<0.7$) while BM has almost no correlation ($r<0.1$).

6. CONCLUSIONS

We introduced a hierarchical RNN-based system for modeling expressive piano performance and a dataset for training the model. Our listening test demonstrated that our model achieved more human-like musical expression compared to the previous model [5]. The source code and dataset are available in `https://github.com/jdasam/virtuosoNet`.

7. ACKNOWLEDGEMENT

This research was funded by Samsung Research Funding Center under the project number SRFC-IT1702-12.

8. REFERENCES

[1] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proc. of the 20th Conf. on Computational Natural Language Learning*, 2016.

[2] Sergio Canazza, Giovanni De Poli, and Antonio Rodà. Caro 2.0: an interactive system for expressive music rendering. *Advances in Human-Computer Interaction*, 2015:2, 2015.

[3] Carlos Eduardo Cancino-Chacón, Thassilo Gadermaier, Gerhard Widmer, and Maarten Grachten. An evaluation of linear and non-linear models of expressive dynamics in classical piano and symphonic music. *Machine Learning*, 106(6):887–909, Jun 2017.

[4] Carlos Eduardo Cancino-Chacón, Maarten Grachten, Werner Goebl, and Gerhard Widmer. Computational models of expressive music performance: A comprehensive and critical review. *Frontiers in Digital Humanities*, 5:25, 2018.

[5] Carlos Eduardo Cancino Chacón and Maarten Grachten. The basis mixer: a computational romantic pianist. In *Late-Breaking Demos of the 17th International Society for Music Information Retrieval Conf. (ISMIR)*, 2016.

[6] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2017.

[7] Sebastian Flossmann, Werner Goebl, Maarten Grachten, Bernhard Niedermayer, and Gerhard Widmer. The magaloff project: An interim report. *Journal of New Music Research*, 39(4):363–377, 2010.

[8] Anders Friberg, Roberto Bresin, and Johan Sundberg. Overview of the kth rule system for musical performance. *Advances in Cognitive Psychology*, 2(2-3):145–161, 2006.

[9] Jon Gillick, Adam Roberts, Jesse Engel, Douglas Eck, and David Bamman. Learning to groove with inverse sequence transformations. In *Proc. the 36th International Conf. on Machine Learning (ICML)*, volume 97, pages 2269–2279, 2019.

[10] Bruno Gingras, Marcus T Pearce, Meghan Goodchild, Roger T Dean, Geraint Wiggins, and Stephen McAdams. Linking melodic expectation to expressive performance timing and perceived musical tension. *Journal of Experimental Psychology: Human Perception and Performance*, 42(4):594, 2016.

[11] Sergio Giraldo and Rafael Ramirez. A machine learning approach to ornamentation modeling and synthesis in jazz guitar. *Journal of Mathematics and Music*, 10(2):107–126, 2016.

[12] Curtis Hawthorne, Andrew 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 *Proc. of the International Conf. on Learning Representations (ICLR)*, 2019.

[13] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2019.

[14] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *Proc. the 36th International Conf. on Machine Learning (ICML)*, volume 97, pages 3060–3070, 2019.

[15] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Score and performance features for rendering expressive music performances. In *Proc. of Music Encoding Conf.*, 2019.

[16] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conf. on Learning Representations (ICLR)*, 2018.

[17] Tae Hun Kim, Satoru Fukayama, Takuya Nishimoto, and Shigeki Sagayama. Statistical approach to automatic expressive rendition of polyphonic piano music. In Alexis Kirke and Eduardo R. Miranda, editors, *Guide to Computing for Expressive Music Performance*, pages 145–179. London, 2013.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[20] Stanislas Lauly. Modélisation de l'interprétation des pianistes & applications d'auto-encodeurs sur des modèles temporels. Master's thesis, University of Montréal, 2010.

[21] Akira Maezawa. Deep piano performance rendering with conditional VAE. In *Late-Breaking Demos of the 19th International Society for Music Information Retrieval Conf. (ISMIR)*, 2018.

[22] Iman Malik and Carl Henrik Ek. Neural translation of musical style. *CoRR*, abs/1708.03535, 2017.

[23] Eita Nakamura, Kazuyoshi Yoshii, and Haruhiro Katayose. Performance error detection and post-processing for fast and accurate symbolic music alignment. In *Proc. of 18th International Society for Music Information Retrieval Conf. (ISMIR)*, 2017.

[24] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: learning expressive musical performance. *Neural Computing and Applications*, Nov 2018.

[25] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *Proc. of the 35th International Conf. on Machine Learning (ICML)*, pages 4364–4373, 2018.

[26] Emery Schubert, Sergio Canazza, Giovanni De Poli, and Antonio Rodà. Algorithms can mimic human piano performance: the deep blues of music. *Journal of New Music Research*, 46(2):175–186, 2017.

[27] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3483–3491, 2015.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[29] Gerhard Widmer, Sebastian Flossmann, and Maarten Grachten. YQX plays chopin. *AI magazine*, 30(3):35, 2009.

[30] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proc. of the 2016 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.