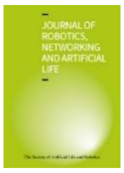
Journal of Robotics, Networking and Artificial Life
Vol. 11(3); March (2026), pp. 190-195
ISSN (Online): 2352-6386; ISSN (Print): 2405-9021
https://alife-robotics.org/jrnal.html



Research Article

A Research on an Editing Support System for Automatic Piano Performance to Improve Dynamics Prediction Accuracy

Taiyo Goto[1], Yoshiki Hori[1], Eiji Hayashi[1]
[1] *Kyushu Institute of Technology, 680-4 Kawazu, Iizuka-shi, Fukuoka, 820-8502, Japan*
Email: gotou.taiyou711@mail.kyutech.jp, haya@mse.kyutech.ac.jp

ARTICLE INFO

Article History

Received 19 November 2024
Accepted 03 December 2025

Keywords

Deep Learning
Automatic Piano
Dropout
Skip connections

ABSTRACT

Automated piano player systems enable highly precise keystrokes and pedal operations. However, a direct translation of score data into performance often results in a mechanical and inexpressive sound, failing to capture the nuanced dynamics of a human pianist's interpretation. This discrepancy arises because pianists uniquely determine various performance parameters, such as note loudness (Velo), the time between notes (Step), and individual note durations (Gate). While deep learning has been utilized in prior research to predict these parameters, the accuracy of Velo prediction, in particular, has remained a significant challenge. To address this limitation, this paper proposes a novel deep learning system specifically designed to enhance the accuracy of Velo prediction by integrating two distinct neural networks. Furthermore, experiments conducted with our system demonstrate improved prediction accuracy compared to previous studies.

1. Introduction

Previously, our laboratory developed an automatic piano player system[1] equipped with 90 dedicated actuators, with one assigned to each of the 88 piano keys as well as to the damper and soft pedals. The controller synchronizes key and pedal motions with millisecond-level precision. To reproduce challenging passages such as pianissimo notes and rapid key repetitions, our laboratory modeled the piano action using a proprietary dynamic model, computed an optimal drive waveform for every key, and stored them in a dedicated database. During playback, the system retrieves the appropriate waveform in real time, enabling stable reproduction of delicate soft tones and ultra-fast repeated keystrokes that even experienced pianists find difficult.

This system can exhibit human-like expressiveness when supplied with high-quality control data. In practice, it requires MIDI (Musical Instrument Digital Interface) events in which each note is annotated with realistic timing and loudness—represented by its MIDI velocity (Velo) value. Feeding it raw score information alone yields performances that sound mechanical and unnatural. Computational modeling of expressive music performance has been studied for decades to bridge this gap between score and performance[2]. In recent years, deep-learning-based inference frameworks have been proposed to generate expressive performances. For instance, Oore et al. [3]demonstrated that recurrent neural networks (RNNs) can learn to reproduce realistic dynamics and timing from human performances. Building on these advancements, prior studies [4], [5] in our laboratory have made initial attempts at modeling human-like performance by focusing on reproducing the performance style of a single pianist. However, accurate prediction of note Velo remains an open problem. To address this limitation, we propose a novel deep-learning architecture that combines two complementary neural subnetworks to enhance the prediction of dynamic expression. We selected this approach because it enables us to feed an expanded set of features into the second network. This paper describes the design of the proposed model which combines two networks and compares its inference results for Velo prediction with those of existing approaches.

2. Data used

The training corpus comprised piano performances encoded in the MIDI standard. Each MIDI file contains parameters of performance information that capture the pianist’s expressive nuances, as well as score information derived from the written notation using the music notation software MuseScore[6]. Because raw MIDI messages are not directly suitable for machine-learning models, we parsed the files to extract the necessary performance information and score information, and converted them into a structured numerical feature set appropriate for supervised learning.

2.1. *Performance information*

We define as performance information those aspects of the performance data that differ across pianists and represent them with four parameters. Table 1 summarizes these parameters.

Table 1 Performance information

Parameters	Detail	Unit
Velo	Loudness	-
Step	Time between notes	ms
Gate	Note duration	ms
Time	Note onset time	ms

・Velo

The parameter inferred in this study. Velo represents the loudness of a note. Although loudness is generally proportional to the keystroke force and key-strike velocity, the actual sound-pressure level (SPL) produced by different keys at maximum and minimum velocity may differ slightly. In the MIDI specification, loudness is quantized into 128 levels ranging from 0 to 127. However, when converting these values into SPL, inconsistencies arise in the actual acoustic output during performance.

To address this, we define the maximum SPL (corresponding to MIDI velocity 127) as the level generated when all keys are struck at a velocity of 1.0m/s and adopt the highest SPL observed among them. For the minimum SPL, we assume a stable value of 62 dB SPL, which can be produced reliably even with soft strikes, and assign it to velocity 1. The difference between the maximum and minimum SPL values, approximately 40 dB SPL is then equally divided into 127 steps.

The step size of the SPL corresponding to each increment in the MIDI velocity scale is given by Eq. (1):

$$\Delta_{SPL} = \frac{SPL_{max} - 62dB\ SPL}{127} \tag{1}$$

・Step

Step represents the time interval between the onsets of consecutive notes.

・Gate

Gate represents the sounding duration of a note. We define it as the elapsed time between the moment the tone is produced and the instant when the damper returns to the string during an actual performance. The end point is referenced to the key position at which the damper first contacts the string and stops its vibration; this position is determined by slowly releasing the key from its fully depressed state.

・Time

Time represents the onset time of a note, relative to the start of the performance.

2.2. *Score information*

We define score information as the aspects of the performance data that are fixed irrespective of the performer, such as note identities and written dynamics, and represent it with five parameters. Table 2 summarizes these parameters.

Table 2 Score information

Parameters	Detail	Unit
Tstep	Duration between notes on the score	ms
Tgate	Note duration on the score	ms
Key	Note pitch	-
Bar	Bar ID	-
Dyn	Dynamics symbols	-

・Tstep

Tstep parameter represents the notated time interval between the onsets of consecutive notes.

・Tgate

Tgate parameter represents the notated duration of a note.

・Key

Key specifies the musical pitch of a note, where higher values represent higher pitches. The standard MIDI protocol supports a wide pitch range from 0 to 127, exceeding the gamut of a conventional 88-key piano. However, as the goal of this research is to generate automated performances for an 88-key grand piano, we have constrained the Key parameter to the corresponding MIDI note numbers 21 to 108, encompassing all 88 keys.

・Bar

Bar indicates the measured number within the score. A measure is a segment of time defined by a given number of beats, which serves as a fundamental unit for organizing musical rhythm. In accordance with standard notation practice, measures are numbered sequentially, starting from 1 for the first measure.

・Dyn

Dyn parameter represents musical dynamics, which are symbols or markings in a musical score that indicate the relative loudness or softness of a note or phrase.

To make the Dyn parameter usable in machine-learning models, we converted each dynamic marking into a numerical value. The mapping was based on the default Velo settings assigned to dynamic symbols in the music-notation software MuseScore[6]. Notes without an explicit dynamic marking are assumed to sound midway between mp and mf; therefore, their Dyn value are set to 72. In addition, ff and sf were assigned the same value, because these two markings are performed at an equivalent Velo. Details of the Dyn used are shown in Table 3.

Table 3 Dynamics Symbols

Dynamics Symbol	Italian term	Meaning in English	Velo
pp	*Pianissimo*	Very soft	33
p	*Piano*	Soft	49
mp	*Mezzo piano*	Moderately soft	64
mf	*Mezzo forte*	Moderately loud	80
f	*Forte*	Loud	96
ff	*Fortissimo*	Very loud	112
sf	*Sforzato*	Strong accent on the note	112

3. Inference System

Fig.1 illustrates the inference system developed in this study. The architecture comprises two subnetworks, Network 1 and Network 2, each containing three hidden layers using Rectified Linear Unit (ReLU) [7] activation functions. Music features are first fed into Network 1; its Velo predictions are then concatenated with the original inputs and passed to Network 2. The second subnetwork produces the final Velo. To mitigate overfitting, skip connections[8] and dropout[9] were applied to every hidden layer.

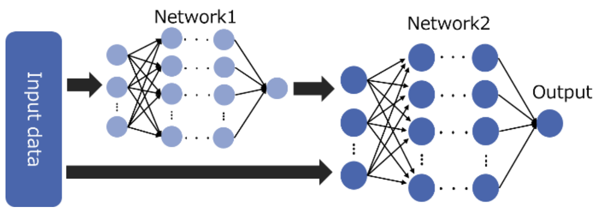


Fig.1 Network layout diagram

3.1. *Skip connections*

A skip connection provides an alternative pathway that bypasses intermediate layers: the output of an earlier layer is added directly, element by element, to the output of a deeper layer and then propagated forward[8]. This shortcut facilitates back-propagation of gradients, mitigating vanishing-gradient problems. As a result, deeper networks can be trained more easily, and information loss in intermediate layers is reduced, leading to better predictive performance. The same mechanism stabilizes training, accelerates convergence, and enables larger models to achieve stronger generalization. For these reasons, skip connections play a pivotal role in boosting overall model accuracy. An example is shown in Fig.2.

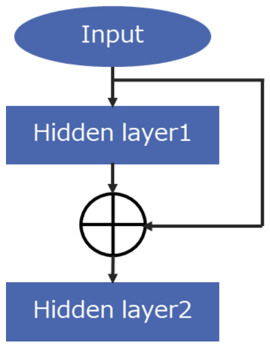


Fig.2 Skip connections

3.2. *Dropout*

Dropout[9] is a widely used regularization technique designed to curb overfitting in deep neural networks by injecting stochasticity into the learning process. During each forward pass in training, every hidden unit is independently masked out with a fixed probability p. Consequently, the subset of active neurons varies from one mini-batch to the next. As illustrated in Fig.3 , the grey circles indicate units that are temporarily deactivated, whereas the blue circles remain active and propagate information.

At test time, all units are restored, but their outgoing weights are scaled by the factor $1-p$. This scaling approximates the effect of averaging the predictions of the many sub-networks sampled during training, yielding an efficient ensemble without the computational burden of evaluating each model separately. Dropout plays a key role in the present architecture by stabilizing optimization and boosting overall velocity-prediction accuracy.

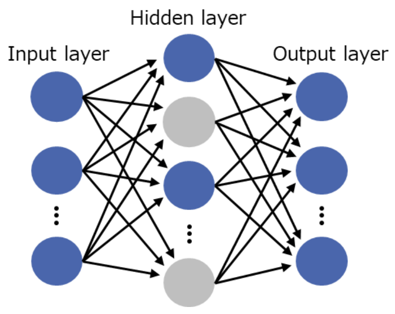


Fig.3 Dropout

4. Experiments

4.1. *Experimental details*

This study uses performance data from two pianists: Vladimir Davidovich Ashkenazy and Sviatoslav Richter. We prepared eight pieces for each pianist and conducted separate experiments for each group without mixing the datasets. In each experiment, one piece was set aside as test data, and the remaining seven were used to train the model. The Ashkenazy repertoire was selected based on prior studies[4] to enable direct comparison. For both groups, we extracted five score-level features—Key, Bar, Dyn, Tgate, and Tstep—and the performance-level feature Velo. The model parameters were optimized using the Adam algorithm[10]. After training, the model predicted the Velo for the test piece based solely on its score data. For the Ashkenazy group, Prelude Op. 28-7 served as the test piece. For the Richter group, Wohltemperierte Klavier I-23 BWV868 Prelude was used as the test piece. The pieces used for the Ashkenazy and Richter datasets are listed in Table 4 and Table 5, respectively.

Table 4 List of pieces used in the experiment1

Pieces ID	Pieces performed by Ashkenazy
1	Etude Op.10-3
2	Etude Op.10-4
3	Nocturne Op.9-2
4	Prelude Op.28-1
5	Prelude Op.28-4
6	Prelude Op.28-15
7	Prelude Op.28-20
8	Prelude Op.28-7

Table 5 List of pieces used in the experiment2

Pieces ID	Pieces performed by Richter
1	Wohltemperierte Klavier I-1 BWV846Prelude
2	Wohltemperierte Klavier I-7 BWV852 Prelude
3	Wohltemperierte Klavier I-7 BWV852 Fuga
4	Wohltemperierte Klavier I-13 BWV858 Prelude
5	Wohltemperierte Klavier I-13 BWV858 Fuga
6	Wohltemperierte Klavier I-23 BWV868 Fuga
7	Wohltemperierte Klavier II-2 BWV871 Fuga
8	Wohltemperierte Klavier I-23 BWV868 Prelude

4.2. *Data partitioning for cross-validation*

In this study, we used eight performances each by pianists Vladimir Davidovich Ashkenazy and Sviatoslav Richter, and determined the model's hyperparameters by applying *K*-fold cross-validation [11]. In K-fold cross-validation, the dataset is divided into *K* mutually exclusive subsets; at each iteration one subset is held out as validation data while the remaining *K-1* subsets are used for training, so that every subset is used as validation exactly once.

In our experiment, the validation process was conducted separately for each pianist. For the Ashkenazy dataset, Prelude Op. 28-7 was kept as a fixed test piece. Similarly, for the Richter dataset, Wohltemperierte Klavier I-23 BWV868 Prelude was fixed as the test piece. In both cases, the other seven pieces were repeatedly partitioned into five for training and two for validation, generating 21 distinct training–validation splits. Evaluating the model on every split eliminates dependence on any particular division of the data and thus provides a more reliable estimate of model performance. The 21 data patterns are shown in Table 6.

Table 6 Pattern of data

Pattern ID	Train pieces number	Evaluate pieces number	Test pieces number
1	1,2,3,4,5	6,7	8
2	1,2,3,4,6	5,7	8
⋮			
17	2,3,4,5,7	1,6	8
18	2,3,4,6,7	1,5	8
19	2,3,5,6,7	1,4	8
20	2,4,5,6,7	1,3	8
21	3,4,5,6,7	1,2	8

4.3. *Experimental Results*

As defined in Section 2.1, Velo denotes MIDI velocity and serves as a proxy for sound loudness. We plotted the Velo predicted by the inference system alongside the ground-truth Velo extracted from the performance data. Fig.4 shows the results for the Ashkenazy dataset, while Fig.6 shows the results for the Richter dataset. In these figures, the orange line, *Predict Velo*, represents the predicted values, whereas the blue line, *Tune Velo*, depicts the original performance values. For reference, Fig.5 and Fig.7 present the Velo generated by Network 1 alone for Ashkenazy and Richter, respectively. A closer agreement in shape and level between *Predict Velo* and *Tune Velo* indicates better performance, as quantified by the error metrics reported below.

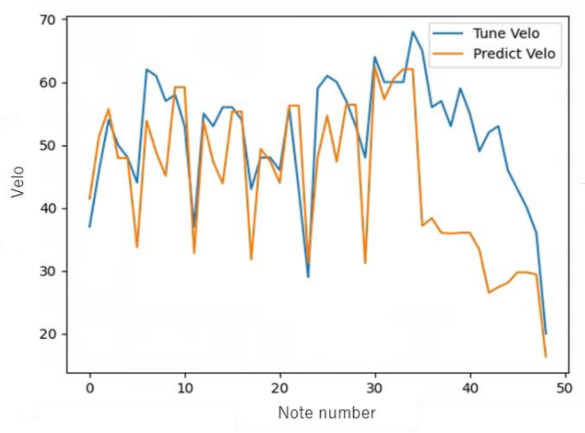


Fig.4 Inference results in the two-stage system_ Ashkenazy

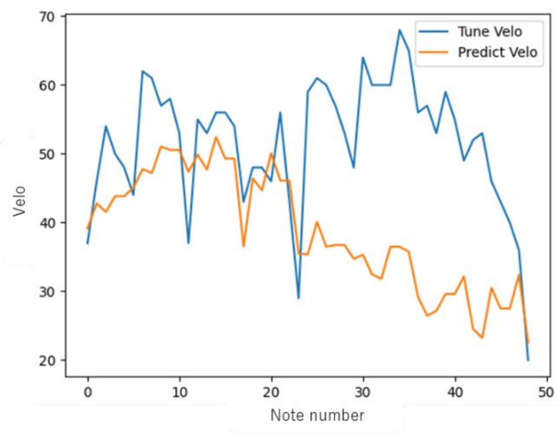


Fig.5 Inference results in the first stage system_ Ashkenazy

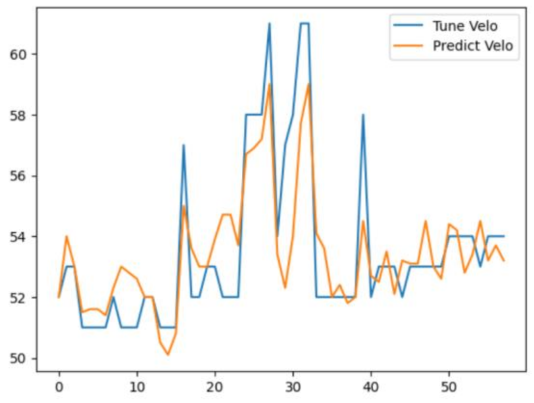


Fig.6 Inference results in the two-stage system_ Richter

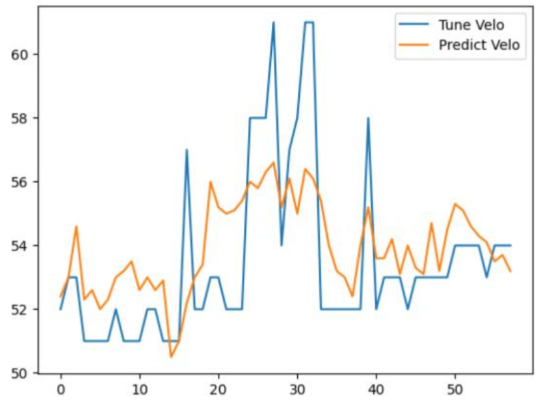


Fig.7 Inference results in the first stage system_ Richter

5. Consideration

Fig.8 presents the inference results reported in a previous study[4], which also utilized the Ashkenazy dataset. In that study, the model employed an inference system with two hidden layers at the mid-level processing stage.

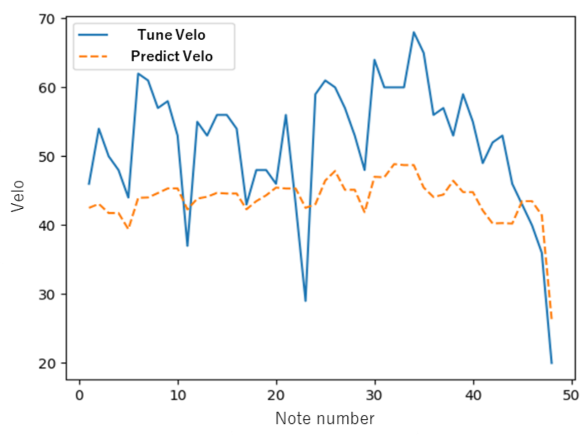


Fig.8 Inference results in previous research

The two-network system illustrated in Fig.4 reproduces the ground-truth Velo successfully up to roughly note number 35 in the score and outperforms the single-network approach reported in the previous study (Fig.8) within this initial segment. This improvement appears to stem from the intermediate features extracted by Network 1: when these features are passed to Network 2, they amplify subtle dynamic variations that remain scarcely discernible in Network 1's stand-alone output (Fig.5). Beyond note number 35, however, this correspondence diminishes. The waveform predicted by Network 1 shows little resemblance to the reference, and although Network 2 remains closer, it still fails to capture the abrupt drop in loudness near the end of the piece—an error that may arise from hyper-parameter settings that do not accommodate the large final-section fluctuations adequately.

Regarding the Richter dataset, the output from Network 1 shown in Fig.7 exhibits dynamic variations that are either too subtle or, conversely, excessive in certain sections. However, it can be confirmed that the graph output from Network 2 (Fig.6) has a shape much closer to the Tune Velo values compared to Network 1. Nevertheless, the increase in Velo around note numbers 25–30 was not sufficiently captured, which we consider a remaining challenge.

6. Conclusion

This study sets out to improve the prediction accuracy of note-level loudness, Velo, by employing an architecture that combines two complementary neural networks. Experimental results show that the two-network system produces Velo that follow the ground-truth performance data more closely than those obtained in earlier single-network studies. As a next step, we will extend the inference framework so that it can cope with large dynamic fluctuations and abrupt changes in articulation. We plan to capture notated dynamic marks such as crescendo and decrescendo. These cues are not present in raw MIDI files, so we will extract them from score images using optical music recognition. The detected symbols will be time-aligned with their corresponding note events, converted into numerical descriptors, and incorporated into the model as additional input features alongside the existing score- and performance-level parameters. After retraining the network with these enriched features, we will reassess its performance on the same eight-piece dataset and conduct ablation studies to measure the individual contributions of each newly introduced symbol. We will also evaluate the approach on recordings by other pianists to assess cross-performer generalization and robustness. Furthermore, to improve the modeling of long-term musical dependencies, we plan to explore attention-based architectures such as the Music Transformer[12]. Ultimately, we aim to generalize this methodology to larger corpora that span diverse musical styles, thereby moving toward a comprehensive system capable of faithfully reproducing the expressive intent embedded in written music.

References

1. E. Hayashi, M. Yamane, H. Mori, Development of a moving coil actuator for an automatic piano, Int. J. Japan Soc. Prec. Eng. 28 (1994), 164–169.
2. Gerhard Widmer, Werner Goebl, Computational models of expressive music performance: The state of the art, Journal of New Music Research, 2004, pp. 203–216.
3. Donghoon Jeong, Taegyun Kwon, Youngmoo Kim, Keunwoo Lee, Juhan Nam, VirtuosoNet: A hierarchical RNN-based system for modeling expressive piano performance, Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR) 2019, pp. 908–915.
4. Yoshiki Hori, Eiji Hayashi, A Research on a System Using Deep Learning for Inferring Piano Performance, Journal of Robotics, Networking and Artificial Life, Vol-10-3, pp.282-285, 2023-2024.
5. Takaaki Ueno, Sakmongkon Chumkamon, Eiji Hayashi, The research about editing system of performance information for player piano. -Develop inference methods using machine learning -, Proceedings of International Conference on Artificial Life and Robotics 2023, pp. 333-336.
6. MuseScore Team, MuseScore: Free music notation software, Available at: https://musescore.org/ja.
7. V. Nair, G. E. Hinton, Rectified Linear Units Improve Restricted Boltzmann Machines, Proceedings of the 27th International Conference on Machine Learning (ICML), pp. 807-814, 2010.
8. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep residual learning for image recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016, pp. 770–778.
9. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Learning Research, 2014, pp. 1929–1958.
10. Diederik P. Kingma, Jimmy Ba, Adam: A method for stochastic optimization, Proceedings of the 3rd International Conference on Learning Representations (ICLR) 2015.
11. R. Kohavi, A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection, Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI), Vol. 2, pp. 1137-1143, 1995.
12. C. Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, D. Eck, Music Transformer: Generating Music with Long-Term Structure, Proceedings of the International Conference on Learning Representations (ICLR), 2019.

Authors Introduction

Mr. Taiyo Goto

He received bachelor degree in Engineering in 2024 from mechanical system engineering, Kyushu Institute of Technology in Japan. He is currently a Master student at Kyushu Institute of Technology and conducts research at Hayashi Laboratory.

Mr. Yoshiki Hori

He received bachelor degree in Engineering in 2023 from mechanical system engineering, Kyushu Institute of Technology in Japan. He is currently a Master student at Kyushu Institute of Technology and conducts research at Hayashi Laboratory.

Prof. Eiji Hayashi

Prof. Eiji Hayashi is a professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received the Ph.D. (Dr. Eng.) degree from Waseda University in 1996. His research interests include Intelligent mechanics, Mechanical systems and Perceptual information processing. He is a member of The Institute of Electrical and Electronics Engineers (IEEE) and The Japan Society of Mechanical Engineers (JSME).