McAuley/Dong ERSP 2023 Proposal

Wanning Lu, Hoang Phan, Diego Gomez

Fall 2023

1 Research Context and Problem Statement

Much of the music that is heard today is still created and performed by humans. This is because music performance rendering, which is the task of synthesizing audio from sheet music, still creates results that sound robotic and unnatural. Features such as timing and dynamics vary between each performer, and it is hard to determine what factors go into the natural quality of human performance. This poses a challenge to modern-day composers, whose only option for gauging the quality of their pieces is by physically playing them on their respective instruments. In addition to composers, music performance rendering can also be an effective tool for creators who want to incorporate music into their work, such as game developers and movie producers. It can add to the depth of their work without requiring substantial musical ability.

One of the biggest obstacles in creating a model that can translate from sheet music to an expressive performance is the limited number of datasets. This is especially a prominent issue for the rendering of any instrument besides piano. Previous models, such as VirtuosoNet [6] and ScorePerformer [1], were able to effectively render piano performances due to the availability of datasets such as MAESTRO [5] and ASAP [4]. These datasets contain a combined total of over 300 hours of aligned MIDI and performance data, where MIDI is a type of computer language that allows for sheet music to be represented digitally.

Because there is a lack of datasets, previous models focusing on violin music rendering have attempted to forego the use of aligned sheet and performance data altogether or use other datasets that contain other instruments. One such model is Vivi [8], which uses a physical model of a violin to replicate the natural quality produced by an actual violin. However, since the model doesn't rely on any dataset, it requires the user to train the model with several hours of self-produced playing, making it difficult for any inexperienced violin player to use the model. Other models have focused instead on generating finger positionings [2] and bow movements [7], but these features are not easily synthesized into audio, necessitating the use of an actual human performer.

Meanwhile, PerformanceNet [10] is a model that was trained on MusicNet, a dataset consisting of 34 hours of aligned MIDI and performance data for a

variety of instruments, such as piano, flute, and violin. However, since the dataset contains mostly piano data, it only has around 3 hours of solo violin performances. Additionally, the model makes use of a convolutional neural network, which is less efficient and robust than a transformer [9], which we will use for our model.

To address the lack of high-quality and easy-to-use violin rendering models, we propose a model that will be trained on the Bach dataset [3] and use a transformer. It will produce expressive performance features for the corresponding sheet music features. Since the Bach dataset is currently the most extensive dataset for solo violin performances, we expect that our model will produce even more natural-sounding results than its predecessors. This will also expand the scope of music rendering, which was previously limited to piano.

2 Proposed solution

In this project, we will focus on *timing*, an important aspect of music performance rendering. Typically, human performances differ in the timing of each note from that indicated on sheet music. Timing of notes can usually be divided into two components: onset and duration. Onset refers to the time when the note begins to play, and duration refers to how long it will be played. These features can also be interpreted as the start and end of a note. Additionally, the timing of a note also depends on its pitch and velocity. The pitch of a note determines how high or low it is, while the velocity determines "how hard we should play the note", translating roughly to the loudness of the note. For our project, we will train a neural network model to predict the expressive start and end times of each note given their onset, duration, and pitch in sheet music. We will not consider velocity as an input since it is not always available on sheet music, but it is something we want to look into to improve the model's accuracy.

We will use the Bach violin dataset [3] to train our model, which contains around 6.5 hours of violin audio paired with sheet music. The dataset has already been preprocessed, with sheet music features that include the onset, offset (corresponding to duration), and pitch represented numerically in a three-valued tuple, or list, for each note. Additionally, the preprocessed data also includes the real timing, which is the start and end time of each note in a real performance, represented in a two-valued tuple for each note. For our input, we will use the sheet music features from 90% of the dataset, which we will iteratively feed to the model. After each iteration, we will compute error values from a loss function to tune our model. Overall, Figure 1 below gives a high-level representation of how we are training the model.

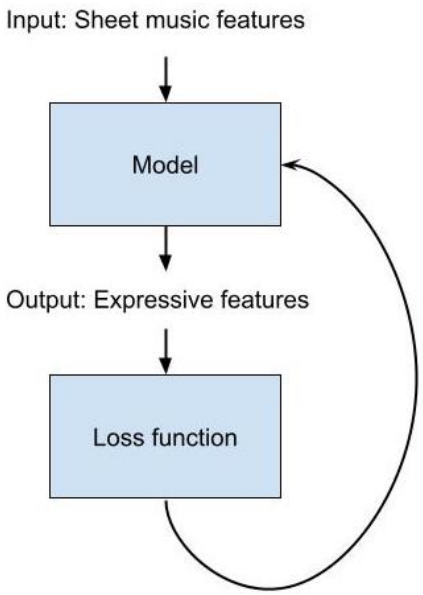

Figure 1: The model is trained by comparing its output with the actual performance features, which is then passed back into the model to modify its algorithm.

For the model, we will use a specific type of neural network architecture called a *Transformer* [9]. A transformer is highly effective for processing sequential data and is well-suited for handling long sequences. Sequential data refers to information in which one point depends on another. Sheet music is an example of sequential data, where one note follows another in a 'sequence'. What differentiates a transformer from previous models, such as a recurrent neural network (RNN), is its "self-attention" layer. It allows for the model to weigh the importance of every element in the input, adjusting their influence on the output. Figure 2 below depicts an example of what the self-attention layer does.

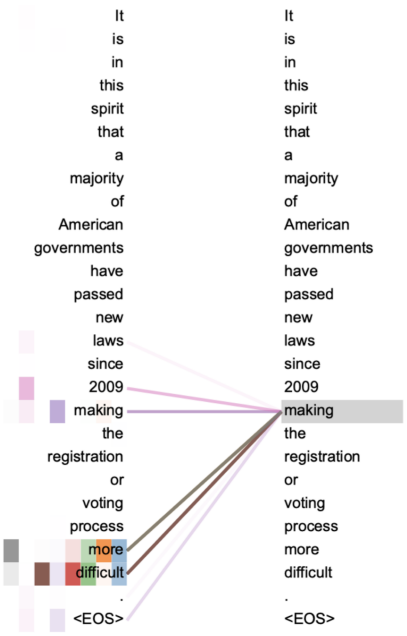

Figure 2: This is the result after an iteration through the self-attention layer, which determines the importance of elements in an input sequence. Words with more colors indicate their dependence on the word 'making'.

This is a vast improvement to an RNN, which tends to "forget" previous entries from the input. As seen in Figure 3, while an RNN takes the cumulative output and feeds it to the next input, a transformer feeds the output of *every* iteration run to the next one, allowing the model to "understand" the importance of each element to accurately predict the next one.

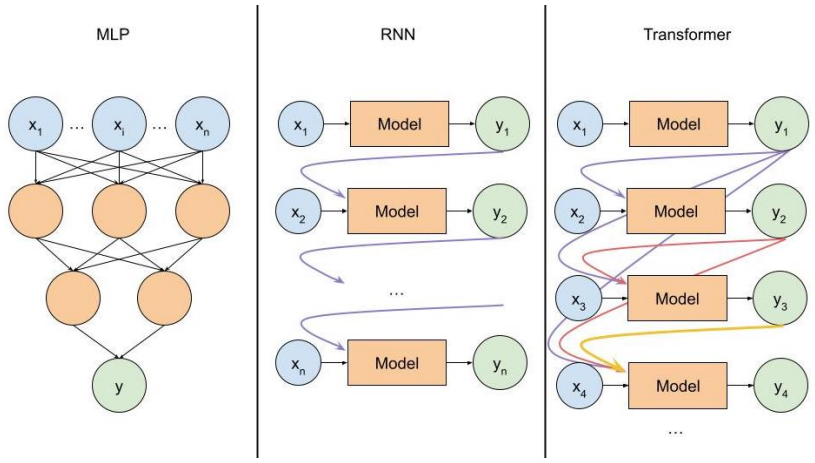

Figure 3: The different neural network architectures, increasing in complexity and performance from left to right.

Since the transformer architecture is complex, we will first start with simpler models by building a multilayer perception (MLP) network, extend to an RNN, and finally implement a transformer.

To continuously tune our model to produce more accurate results, we will calculate the error, which is the sum of the distances between the generated and real features, as shown in Equation (1). In this equation, n represents the number of notes in the piece. This will serve as our loss function. Ultimately, our goal is to minimize this value such that the generated timing of the performance will be very similar to that of an actual human performance.

$$\text{error} = \sum_{i=1}^{n} \sqrt{(\text{start}_{\text{generated}} - \text{start}_{\text{real}})^2 + (\text{end}_{\text{generated}} - \text{end}_{\text{real}})^2} \quad (1)$$

Once we finish tuning, the model will be able to accurately produce a list of two-valued tuples containing the expressive start and end times. We can then synthesize these features along with the predetermined pitch and velocity to generate playable MIDI files. We will achieve this by using Muspy, a Python library for synthesizing MIDI files into audio.

Additionally, in the future, we can further improve our model by predicting dynamics, encoded as the velocity, in addition to timing. The Bach dataset also provides information on the velocity of each note for each piece and performer, so we can reuse the dataset to re-train our model with the addition of dynamics or attempt to gather a better dataset to train our model.

3 Evaluation and Implementation Plan

Since our model will be trained and tuned on 90% of the Bach violin dataset, we will use the remaining 10% for the evaluation of the model. By doing this, we are testing how effective the model is against unseen data. We will have two methods to evaluate the model. Our first method will be to survey music experts and non-experts to receive qualitative feedback on the naturalness of our generated results. Our goal is to survey a minimum of 20 students and professors at UC San Diego. They will listen to a generated performance from our model, an actual performance of the same piece, and the raw MIDI directly translated from the sheet music. Afterward, each participant will be asked to identify which pieces they believed were generated by a computer and rank all three based on their perceived naturalness.

Our second method of evaluation will be to compare the generated start and end timings separately with the actual timings by using the loss function used to train our model. If the computed error values are close to 0, then we will know that our model produces "natural" results similar to that of an actual performance.

A limitation to our model is that it may encounter the issue of overfitting due to limitations in the dataset, which includes only Bach as the sole composer. The model may then struggle to generate expressive features accurately for other composers. To refine our model further, we plan to gather additional data on various composers, including Paganini, Kreisler, and others, using techniques similar to those employed in the Bach dataset.

Timeline

Fall Quarter 2023

Weeks 7-8

- Make an appointment with the Writing Hub
- Polish proposal draft and review with advisors, Javahir, and Lisa

Weeks 9-10

- Final adjustments to the proposal
- Work on proposal presentation

Winter Quarter 2024

Weeks 1-2

- Get adjusted to working with the Bach dataset by building a multilayer perceptron (MLP) network to generate expressive features

Weeks 3-4

- Build an RNN model to replace the MLP network

Weeks 5-6

- Build a transformer model to replace the RNN model

Weeks 7-10

- Conduct a literature search for models with better architecture to see if we can replicate
- Collect more data on sheet music and performance MIDI from other composers
- Potentially exploring piano rendering performances, depending on progress

Spring Quarter 2024

Weeks 1-2

- Evaluate the model by computing the error value and surveying
- Find other models to compare our model against
- Start planning for poster presentation and research paper

Weeks 3-7

- Continue working on paper and poster
  - Make an appointment with Writing Hub
  - Revise with mentors and Herman

Weeks 8-10

- Finalize paper and poster
- Practice presenting

4 Revisions

- Modify the Research Context and Problem Statement to make it more clear, changing the scope from generative performance to music performance rendering to cause less confusion
- Added graphic comparing MLP/RNN/Transformer

- Added graphic illustrating the training process
- Provided a more in-depth view of the models (especially how a transformer is better than an RNN), as well as information on the validation step and input/outputs
- Included surveying people as an additional form of evaluation
- Updated captions to the figures and integrated them to the text
- Clarified that we are using tuples to represent each note and not a matrix
- Updated the evaluation plan to make it more clear and address limitation
- Revised introduction to make motivation and context more clear
- Included more previous work on violin rendering, such as Vivi, PerformanceNet
- Revised last paragraph of introduction to clarify our contribution
- Re-organized parts of the solution section for better flow
- Fixed grammar and flow of sentences in some parts
- Added missing information for sources

References

[1] Ilya Borovik and Vladimir Viro. Scoreperformer: Expressive piano performance rendering with fine-grained control. In *Ismir 2023 Hybrid Conference*, 2023.

[2] Vincent KM Cheung, Hsuan-Kai Kao, and Li Su. Semi-supervised violin fingering generation using variational autoencoders. In *ISMIR*, pages 113–120, 2021.

[3] Hao-Wen Dong, Cong Zhou, Taylor Berg-Kirkpatrick, and Julian McAuley. Deep performer: Score-to-audio music performance synthesis. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[4] Francesco Foscarin, Andrew McLeod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. ASAP: a dataset of aligned scores and performances for piano transcription. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 534–541, 2020.

[5] Curtis Hawthorne, Andriy 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 *ICLR*, 2018.

[6] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. Virtuosonet: A hierarchical rnn-based system for modeling expressive piano performance. In *20th International Society for Music Information Retrieval Conference*, pages 908–915, 2019.

[7] Jun-Wei Liu, Hung-Yi Lin, Yu-Fen Huang, Hsuan-Kai Kao, and Li Su. Body movement generation for expressive violin performance applying neural networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3787–3791. IEEE, 2020.

[8] Graham Percival, Nicholas Bailey, and George Tzanetakis. Physical modelling and supervised training of a virtual string quartet. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, page 103–112, New York, NY, USA, 2013. Association for Computing Machinery.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 2017.

[10] Bryan Wang and Yi-Hsuan Yang. Performancenet: Score-to-audio music generation with multi-band convolutional residual network. *Proceedings of*

the AAAI Conference on Artificial Intelligence, 33(01):1174–1181, Jul. 2019.