Deep Linear Autoregressive Model for Interpretable Prediction of Expressive Tempo

Akira Maezawa
Yamaha Corporation
akira.maezawa@music.yamaha.com

ABSTRACT

Anticipating a human musician's tempo for a given piece of music using a predictable model is important for interactive music applications, but existing studies base such an anticipation based on hand-crafted features. Based on recent trends in using deep learning for music performance rendering, we present an online method for multi-step prediction of the tempo curve, given the past history of tempo curves and the music score that the user is playing. We present a linear autoregressive model whose parameters are determined by a deep convolutional neural network whose input is the music score and the history of tempo curve; such an architecture allows the machine to acquire a music performance idioms based on musical contexts, while being able to predict the timing based on the user's playing. Evaluations show that our model is capable of improving the tempo estimate over a commonly-used baseline for tempo prediction by 18%.

1. INTRODUCTION

When multiple musicians play in a music ensemble for the first time, each player responds to one another by listening to each other and anticipating each others' timing. Realizing this kind of on-the-fly online timing prediction for machines is important for interactive computer systems such as automatic accompaniment systems, since the system needs to respond in real-time in spite of the delays in computer systems and/or mechanical actuators, which can be on the order of hundreds of milliseconds [1].

In this kind of problem setting, key requirements are (1) awareness to the common musical idioms associated with a particular music score, (2) awareness to how the human performer has executed the playing, and (3) interpretability of the system behavior. Awareness to the music score is important because the music score and expressive parameters are highly correlated [2]. For example, musicians often slow down before the end of the song. Awareness to the actual performance by the human musician is also important because, as much as the music score provides strong cues on musical idioms, it is the performer who ultimately chooses to abide by or defy it. Interpretability is important because it allows musicians to anticipate how the system will respond to their playing.

Recently, it was shown in the analysis of duet interaction [3] that using hand-crafted features from the music score and the performance helps in timing prediction, as opposed to using the performance features alone. However, it uses simple hand-crafted features from the music score, potentially limiting the kind of information captured from the music score. To bypass design of handcrafted features, in a different problem of music performance rendering, deep learning has shown promise for acquiring features that are relevant for the prediction of note strengths [4].

Inspired by these works, this paper presents a tempo prediction method that takes into account both the music performance context and the surrounding music score context, and learns the feature representation in a data-driven manner [1] . This is achieved by training a linear autoregressive (AR) model of the tempo, whose coefficients are generated from a deep neural network (DNN) that takes both the performance history and musical context as the inputs.

Our contributions are as follows:

1. We propose deep linear AR model, a linear AR model whose coefficients are modeled with a DNN.
2. We apply the deep linear AR model for human tempo prediction, allowing online tempo prediction that is both music performance-aware and music context-aware.
3. We evaluate our model, comparing it with other commonly-used baseline methods for human tempo prediction in interactive music systems and show that DNN-based feature extraction surpasses hand-crafted features. Furthermore, through application of performance rendering, we shed light on the kind of musical context the system learns to acquire through the DNN.

Audio examples of the inferences made by our method is available at https://sites.google.com/view/deep-linear-ar-for-tempo/.

2. RELATED WORK

2.1 Automatic accompaniment

Predicting the human player's tempo is a critical component in automatic accompaniment systems [5–8]. To tune

[1] In this paper, we use the word "tempo" interchangably with the beat duration.

the prediction to a particular performer that plays a particular piece, such a system often learns a model of tempo curve from multiple rehearsals. Timing prediction through rehearsals, however, is agnostic to musical contexts, so it is not possible to predict the expressive timing on a piece that has never been rehearsed before. While it is possible to use tempo markings written in the music score [9], but it is often cumbersome to prepare such an annotation. This paper is concerned with enabling the machine to anticipate expressive timing on a piece that has not been played before, or to respond to spontaneous musical ideas for pieces that have been rehearsed.

2.2 Music performance rendering

Music performance rendering method generates a human-like tempo curve, given a previously unseen music score [10, 11]. It is critical in this task to extract features from the music score that are relevant to music performance, a reign in which deep learning has shown promise, particularly for predicting note strengths [4] and timings [12]. Unfortunately, predicting and responding to live human performance is outside the scope of the problem definition. This paper is concerned with using an external tempo curve played by a human musician to predict the tempo curve, using a model that is amenable to online inference.

2.3 Duet interaction

Duet interaction [3], the task of predicting the machine response given a human playing in a human-machine ensemble, exploits the music score to improve the quality of timing and dynamics prediction with a few number of rehearsals. A limitation is that the method requires hand-crafted features from the music score. This paper is concerned with using the idea of duet interaction for human timing prediction.

2.4 Deep non-linear AR models

Recently, deep neural networks have been applied to sequence prediction tasks, where non-linear AR model [13] has shown success. However, its behavior is often difficult to predict ahead of time. In real-time systems like automatic accompaniment, it is desirable for the system to exhibit a known dynamics ahead of time, using models like linear autoregressive models [5] for which stability and sensitivity is easy to analyze. This paper is concerned with generating a mathematical model based on linear autoregressive process, so that the behavior of the system for a given piece of music can be anticipated beforehand, while enjoying the high-level feature design that deep learning offers.

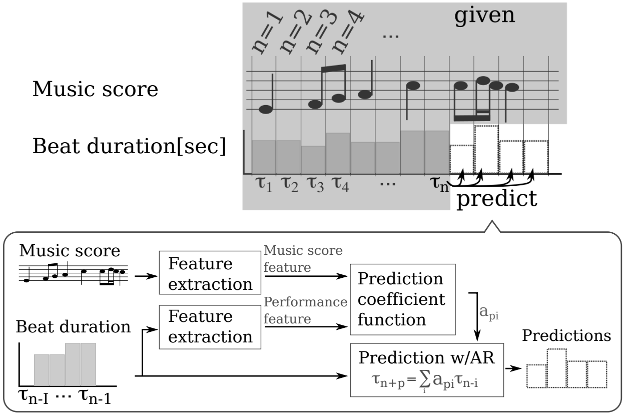


Figure 1: Overview of our method. Our method, given the music score and the history of beat duration, predicts the successive durations. It uses a DNN that generates AR model coefficients to predict the beat duration from past beat durations.

3. OUR METHOD

The goal of our method is to predict the tempo curve of a musician who plays a new piece of music score, as if a group of musicians are anticipating each other's timing for a piece that they play for the first time. We require *multi-step* predictions: at a given time instance when the user is playing some position n in the music score, we predict the tempo at n plus p eighth notes, ranging from $p = 1$ to $p = P$. This way, it is possible to predict the future playing position for various interactive systems with different latencies.

As shown in Figure 1, the music score is assumed to be segmented at an eighth-note level, where the nth segment is associated with a segment duration τ_n. When the player has just finished playing the nth segment, our goal is to predict the future segment durations, τ_{n+1} to τ_{n+P}, given the music score and the segment durations played by the player up to now, $\{\tau_{n'}\}_{n' \leq n}$.

Our method predicts the timing with an AR model of order I. The AR coefficients are determined by two inputs. First, since it is autoregressive, it uses the segment duration history of the current performance, $\{\tau_{n'}\}_{n' \leq n}$. Second, since the music score and the tempo are highly correlated, it uses the music score information around the current segment n, which we denote by S_n. It contains (1) the notes written in the score, *i.e.*, the pitches, the start times and the durations, and (2) metric information, *i.e.*, the meter and the relative position inside the measure.

3.1 Deep linear AR model for timing prediction

We formulate the timing prediction as a multi-step prediction problem. Suppose that the performer has played just up to segment n. We assume that the expected segment duration τ_{n+p} ($p > 0$) depends only on the performance history $\tau_{n' \leq n}$ and the music score S_n. Furthermore, we assume that the residual follows a zero-mean Laplacian noise with scale λ. We assume Laplacian noise because it is tolerant to outliers of the IOI. Then, based on the assumptions described later in Section 3.1.1, we can formulate timing prediction as a maximum likelihood estimation of the following probabilistic model:

$$\tau_{n+p} | \Theta, S_n, \boldsymbol{\tau}_n \sim \mathcal{L}\Big(\sum_{i=0}^{I-1} a_{p,i}(S_n, \boldsymbol{\tau}_n; \Theta) \tau_{n-i}, \lambda \Big), \quad (1)$$

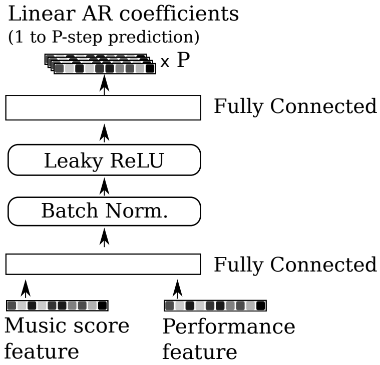


Figure 2: The prediction coefficient function. It is a NN that, given the music score feature and the performance feature, generates the AR model parameters of order I, for up to P-step prediction.

where $\mathcal{L}(\mu, \lambda)$ is a Laplace distribution with the position parameter μ and the scale parameter λ, Θ denotes a set of arbitrary model parameters, and $\boldsymbol{\tau}_n = [\tau_{n-I+1} \cdots \tau_n]$. Our goal is to design the non-linear function $a_{p,i}(S_n, \boldsymbol{\tau}_n; \Theta)$, which we call the **prediction coefficient function**, and to learn its model parameters Θ.

We represent the prediction coefficient function $a_{p,i}$ as a neural network composed of two fully-connected layers as shown in Figure 2. The number of neurons for the hidden layers is 300 and the number of output neurons is $P \times I$. It uses leaky rectified linear units (ReLU) for the activation function and each layer is batch-normalized [14].

The inputs to the network are low-dimensional feature representations of the music score S_n and the performance $\boldsymbol{\tau}_n$. We denote these features respectively by u_n and v_n and call them the **music score feature** and **performance feature** respectively.

3.1.1 Derivation of the deep linear AR model

We assume that at segment n, only the previous I coefficients contribute to the estimate of $\tau_{n'>n}$. Then, we model τ_{n+p} as the following non-linear AR process:

$$\tau_{n+p} = f_p(S_n, \{\tau_{n'}\}_{n' \leq n}; \Theta) + \epsilon_{n,p}; \epsilon_{n,p} \sim \mathcal{L}(0, \lambda). \quad (2)$$

To make the model's behavior more predictable, which is beneficial for real-time interactive music applications, f_p is approximated by a first-order Taylor expansion with respect to $\boldsymbol{\tau}$ to yield the following:

$$\begin{aligned}\tau_{n+p} \approx \boldsymbol{\tau}_n^T (\nabla_{\boldsymbol{\tau}} f_p(S_n, \boldsymbol{\tau}; \Theta)|_{\boldsymbol{\tau}=\mathbf{0}} \\ + H(S_n, \boldsymbol{\tau}_n; \Theta)) + \epsilon_{n,p}, \quad (3)\end{aligned}$$

where H is the higher-order term left-divided by $\boldsymbol{\tau}_n^T$, and we assumed that the constant term is zero[2]. Thus, we arrive at Equation 1, where $a_{p,i} = (\nabla_{\boldsymbol{\tau}} f_p(S_n, \boldsymbol{\tau}; \Theta)|_{\boldsymbol{\tau}=\mathbf{0}} + H(S_n, \boldsymbol{\tau}_n; \Theta))_i$.

3.1.2 Relationship with linear AR and deep non-linear AR models

Our model is a compromise between a linear AR model used in automatic accompaniment systems [5] and a deep non-linear AR model used in areas like speech generation [13]. It is a linear AR process, whose model parameters are governed by a non-linear function $a(\cdot)$.

Our modeling approximation is inspired by the success of shortcut connections in deep learning [15, 16]: our model can be thought of as having a multiplicative shortcut connection from the input $\boldsymbol{\tau}_n$ to the output, so that the output gradient is able to fully exploit the input.

3.2 DNN for feature extraction

For computing the music score feature u_n, we extract the following attributes from the music score S_n:

1. $\phi_n^{(1)} \in \{0,1\}^{12}$: Denotes the downbeat phase; it is a one-hot vector that indicates, at segment n, the number of segments that have elapsed since the last downbeat.
2. $\phi_n^{(2)} \in \{0,1\}^{12}$: Denotes the meter; it is a one-hot representation of the meter at the current measure, expressed as the number of segments inside a measure, with the longest meter of 12/8.
3. $\phi_n^{(3)} \in \{0,1\}^{128\times 20}$: Denotes the notated notes; it is a binary piano-roll representation of the music score between segment $n-2$ and $n+2$; the piano-roll is quantized at 32nd-note level, and the pitch is represented as MIDI note number between 0 and 127.

Given these data, we extract the music score feature using a DNN shown in Figure 3. Namely, $\phi^{(1)}$ and $\phi^{(2)}$ are concatenated and passed through a fully-connected layer to obtain an intermediate feature $\phi^{(m)}$. $\phi^{(3)}$ is passed through three convolutional layers followed by a fully-connected layer to obtain another intermediate feature $\phi^{(p)}$. We use leaky ReLU for activation, followed by batch-normalization and max-pooling. The convolutional layers and max-pooling layers are designed so that the network becomes (1) sensitive to particular harmonic progressions or note patterns, (2) sensitive to position in the score, and (3) relatively invariant to transposition. Specifically, for the first layer, we attempt to capture interval relationship by using kernel size of twelve semitones by two 32nd notes. Furthermore, to achieve invariance on transposition while remaining sensitive to the temporal positions, max-pooling is done only on the pitch axis, spanning four semitones. To obtain the music score feature u_n, we concatenate these intermediate features from the current measure and W neighboring segments, $\{\phi_{n'}^{(m)}, \phi_{n'}^{(p)}\}_{n'=n-W}^{n+W}$. By evaluating from $n-W$ up to $n+W$, we incorporate both prior and upcoming contexts, both of which are relevant for musical expression [17].

For the performance feature v_n, we use $\boldsymbol{\tau}_n$, normalizing it to have zero mean and unit variance. Thus, at the expense of ignoring the dependency of average tempo on tempo expression [18], it expresses the local trend of the

[2] Incorporating the constant term yielded in poor results in preliminary experiments.

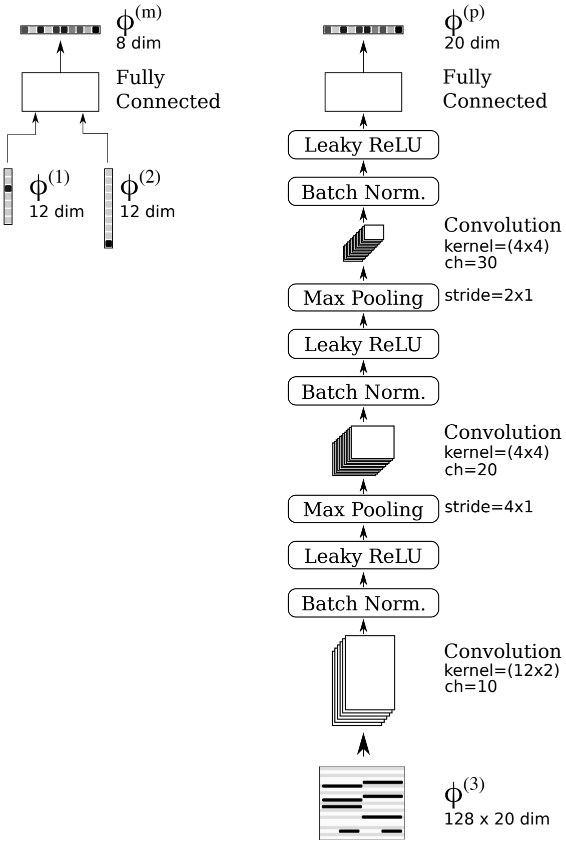


Figure 3: DNN for music score feature extraction.

tempo change, while being invariant under the change of the average tempo at segment n.

3.3 Training

In our method, we train the parameters related to DNN for music score feature extraction, and the parameters related to the prediction coefficient function $a_{p.i}(u_n, v_n; \Theta)$. We train the parameters as to maximize the likelihood of the ground-truth segment duration, which amounts to minimizing the l_1 loss with respect all the parameters, accumulated over all the songs in the training data. Specifically, for each song in the training data, the loss is given as follows, where N is the number of segments in the song, $\hat{\tau}_n$ is the ground-truth segment duration, and $\hat{S}$ is the music score:

$$\sum_{n=I}^{N}\sum_{p=1}^{P}|\hat{\tau}_{n+p} - \sum_{i=0}^{I-1} a_{p,i}(u_n(\hat{S}), v_n(\hat{\boldsymbol{\tau}}_n))\hat{\tau}_{n-i}|. \quad (4)$$

3.3.1 Data augmentation strategies

Since the segment duration is expected to be invariant under transposition, we augment the data by randomly shifting the piano-roll $\phi^{(3)}$ by -5 to +5 semitones. Furthermore, we simulate the motor noise of a human musician, inspired by models of sensorimotor synchronization [19]; we add to τ_n a correlated Gaussian noise e_n, given as $e_n = \epsilon_n - \epsilon_{n-1}$ where ϵ_n is a white noise with a standard deviation of 10 ms. Time-stretching, a common data-augmentation strategy for audio [20], was not used, because our model is invariant under change of the average tempo in a piece.

4. EVALUATION

To evaluate our method, we conducted three experiments. First, we evaluated the effectiveness of incorporating the music score and the performance history for tempo prediction. Second, we conducted an ablation study for assessing the effect of using a deep convolutional neural network for music score feature extraction. Third, we qualitatively analyzed the typical predictions made by our model, by applying our model for tempo curve generation.

In the subsequence experiments, we let $P = 8$, $I = 24$, and $W = 24$. To train the model, we used ADAM [21] for seven epochs with a batch-size of 128, with the same hyper-parameters used in [21]. We directly minimized the loss function, with no pre-training.

4.1 Dataset

We evaluated our method on 52 virtuoso solo pieces played by different people, mostly pieces from the Romantic era such as Chopin, Liszt, Schubert, and virtuosic Beethoven piano sonatas [3] . The pieces were chosen because they often contain extreme tempo fluctuations, owing to the high freedom allowed in playing.

First, for each piece, a digital music score was prepared as a standard MIDI file. Second, performance data for each piece was obtained from Yamaha e-Piano competition, which contains performances by different performers on a Yamaha Disklavier player piano to record the MIDI performance data (up to sixteen interpretations per piece). We obtained a total of 250 MIDI performance data. Finally, for each MIDI performance data, the ground-truth segment durations and the music score were obtained by aligning it to the corresponding music score MIDI data. The alignment was obtained by using a symbolic alignment method, followed by a manual inspection by a trained musician.

Of the 52 pieces, we used 47 pieces for training and 5 for testing, using ten-fold cross validation (about 712,000 training samples).

4.2 Evaluation of the prediction method

In this experiment, we evaluated the effectiveness of using the performance feature and the music score feature. To this end, we have evaluated the prediction error of multi-step prediction. Specifically, for each prediction step p, we computed the mean absolute error of the predicted duration between the current segment n and segment $n + p$, and the actual duration. We have evaluated the prediction errors using the following methods:

1. Condition **MA**: τ_{n+p} is predicted as a moving average of τ_{n-I} to τ_{n-1}, similar to [22].

[3] A full list of the repertoire is available at the web page mentioned in the introduction.

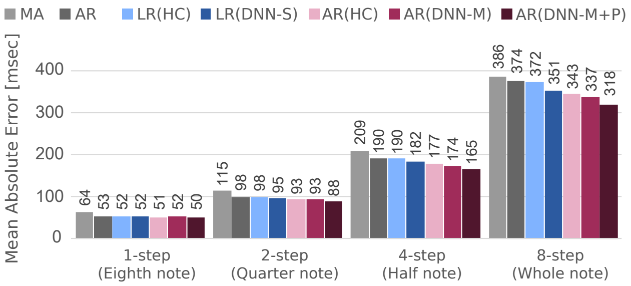


Figure 4: Mean absolute error of multi-step prediction with different baselines. Differences are statistically significant for all pairs of the baseline and the proposed method (Kruskal-Wallis Test applied pairwise, $p < 0.01$).

2. Condition **AR**: τ_{n+p} is predicted with a single AR(I) model, similar to [5].

3. Condition **LR(HC)** (HC=Hand-Crafted): τ_{n+p} is predicted with linear regression, using as the input variables the timing history and hand-crafted features used in duet interaction. Specifically, the hand-crafted features consists of the segment durations, the beat phase and the coefficients to the quadratic regression of the highest and lowest notes. This condition amounts to a simplified model of duet interaction [3], where the player's performance is used to predict his own timing. To make a fair comparison on music score feature extraction, we omit features that are not obtainable from the music score and the beat duration history, such as the note strengths.

4. Condition **LR(DNN-M)**: Same as **LR(HC)**, except instead of hand-crafted features, the DNN-based music score feature is used.

5. Condition **AR(HC)**: Same as **LR(HC)**, except instead of using linear regression on hand-crafted features, autoregressive coefficients are directly estimated from the hand-crafted features using the DNN described in Figure 2.

6. Condition **AR(DNN-M)**: The proposed method without the performance feature. It uses a DNN-based score feature extraction, and a DNN-based autoregressive coefficient extraction.

7. Condition **AR(DNN-M+P)**: The proposed method. It uses both performance and a DNN-based score features, and a DNN-based autoregressive coefficient extraction.

4.2.1 *Results and discussion*

The results are shown in Figure 4. First, the proposed method consistently outperforms the baseline methods, by up to 18% when compared with MA for 8-step prediction.

Second, prediction with an AR model outperforms a LR model, for both hand-crafted and linear features. That is, AR(HC) outperforms LR(HC) and AR(DNN-M) outperforms LR(DNN-M). This is surprising because linear regression is, in our context, auto-regression with an additional bias term explained by the features. This shows that auto-regressive models without a bias term is beneficial for expressive tempo prediction.

Third, feature extraction with DNN surpasses hand-crafted features, for both linear regression and auto-regressive models. That is, LR(DNN-M) outperforms LR(HC) and AR(DNN-M) outperforms AR(HC). This shows that directly training feature extraction from a symbolic music information is beneficial for expressive tempo prediction.

Finally, the incorporation of the performance history $\boldsymbol{\tau}$ improves the prediction. That is, AR(DNN-M+P) outperforms AR(DNN-M). The effect is more prominent when making a prediction with a long forecast like a half note ahead (4-step) or a whole note ahead (8-step).

4.2.2 *Distributions of the prediction errors*

The distribution of 8-step prediction error is shown in Figure 5. For sake of clarity, we only show distributions that highlight properties of different features or prediction models.

It first shows that moving average (MA), the simplest method of all, is unbiased but suffers from the worst outlier. This is reasonable because it is good at tracking steady tempo, but has no capability to anticipate the next tempo during tempo changes.

Second, AR model tends to make negative errors, *i.e.*, it tends to anticipate that the next note will slow down. Such a tendency arises because musicians tend to slow down more often in a given song than they speed up. This kind of asymmetric tempo change encourages the AR model to anticipate every note to be slowing down when using squared error for training.

Finally, the proposed method enjoys increased robustness against outliers. Therefore the primary benefit of incorporating performance feature is the capability to prevent a large mistake.

4.3 Evaluation of music score feature extraction

We have conducted an ablation study to assess which components are effective for computing the music score feature. To this end, we have compared the multi-step prediction errors when using different kinds of music score feature extractor as follows:

1. Condition **FC**: The music score feature is obtained using one fully-connected layer applied to the piano-roll. Combined with the two fully-connected layers in the prediction coefficient function, this model is a perceptron with three hidden layers, making it similar in essence to the architecture used for generation of expressive dynamics from the music score [4].

2. Condition **Conv1**: The music score feature is obtained by one convolutional layer followed by max-pooling and fully-connected layer.

3. Condition **Conv2**: Same as Conv1, except we use two convolutional layers.

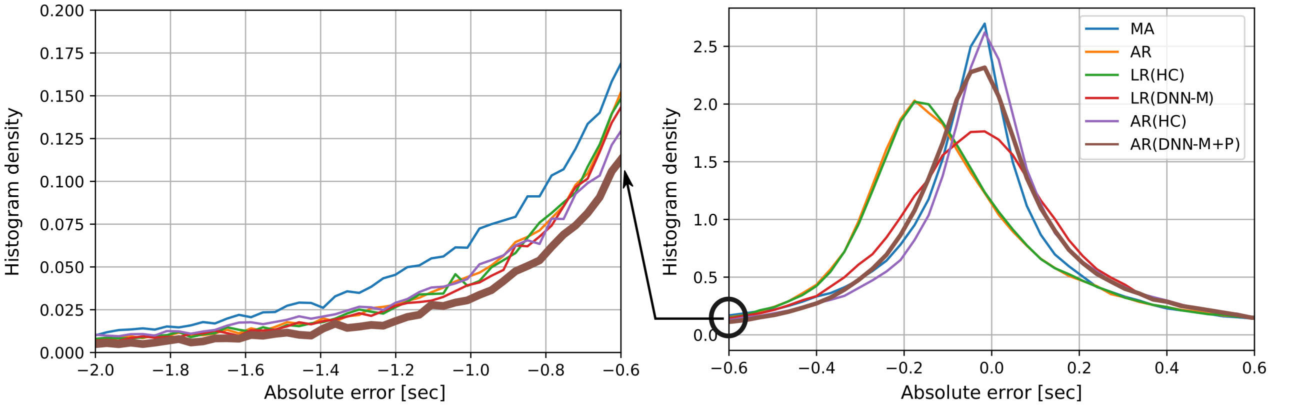


Figure 5: Histograms of the prediction errors, centered about the origin (right), and zoomed in for tails (left).

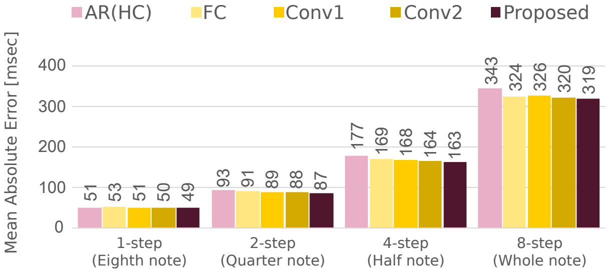


Figure 6: Multi-step prediction error with different ways of computing the music score feature. Differences of absolute errors are significant, except for condition FC and Conv1 of 2-step prediction and FC and Conv2 of 8-step prediction (Kruskal-Wallis Test, $p < 0.01$).

4.3.1 Results and discussion

The results are shown in Figure 6. It can be seen that even with simple network like FC, it helps to automate feature extraction, as can be seen by comparing with AR(HC). Furthermore, addition of more convolutional layers improves the accuracy.

4.4 Analysis through performance rendering

To qualitatively analyze the kind of prediction made by the model, we analyzed the tempo curve generated by the model when it predicts the tempo based on its own predictions. That is, instead of predicting the tempo curve based on a human performance, we drove the AR model with its 1-step prediction, and apply the following low-pass filter to let the prediction stay about some average m:

$$\tau_n = (1-\alpha)m + \alpha \sum_{i=0}^{I-1} a_{1,i}(S_n, \boldsymbol{\tau}_n; \Theta)\tau_{n-i}. \quad (5)$$

Here, $\alpha \in [0,1]$ is a parameter that controls how much τ_n reverts to m. m was set to the the mean tempo for each piece, and α was set to 0.5.

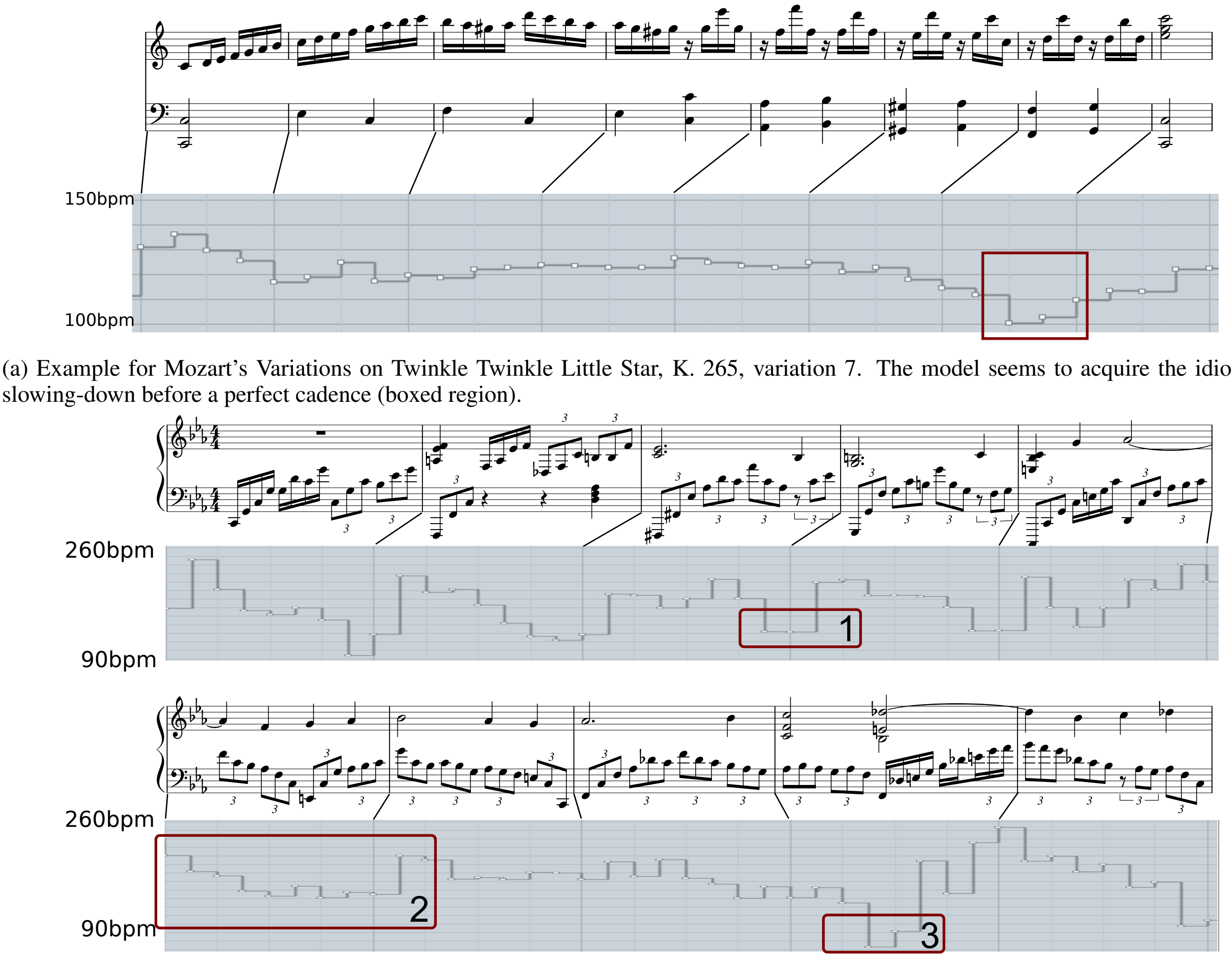
4.4.1 Results and discussion

In Figure 7, we present two examples from songs not contained in the training dataset, one demonstrating a performance idiom pertaining to harmony and another specific to piano playing. We invite the readers to listen to the examples at the web page mentioned in the introduction.

First, the method seems to capture performance idioms related to cadences. To demonstrate, Figure 7a shows an excerpt from a simple piece, Mozart's Variations on Twinkle Twinkle Little Star. The generated tempo slows down in the bounded rectangle, which is a perfect cadence. It shows that the method is capable of capturing a common idiom of slowing down before a cadence. This behavior is quite consistent and also seen in other variations as well.

Second, the method seems to also capture a typical idiom pertaining to the left-hand technique. Figure 7b shows an excerpt from a technically and harmonically more complex piece, Rachmaninoff's Piano Concerto No. 2. We observe, qualitatively, a few idioms related to piano playing. First, the beginning of the bar tends to start slowly, forming an arc-like tempo curve (label "1" in Figure 7b). This kind of playing is consistent with a typical piano playing [2,23]. Second, this kind of behavior is not hard-coded, but rather depends on the surrounding musical context; for example, the tempo does not slow down in a non-cadence progression (label "2," a progression from D dim7 to C7/E, which will resolve to Fm). Furthermore, the most prominent drop in the tempo occurs at a climactic segment inside the phrase (label "3"). These behavior seem concordant with how this particular piece is played.

The generated tempo curve depends on harmonic changes or accompaniment patterns, suggesting that the music score feature extraction was able to learn relevant relationship between music score and tempo changes. We believe that the kind of predictions made by our method captures the essence of music context and performance for making sensible timing predictions in interactive music systems. These results show the expressiveness of our model, despite the fact it uses far fewer information from the score than those typicaly used in music performance rendering [11], but they also qualitatively address some limitations of our method. First, it does not take into account the genre. The generated tempo curves are mostly in the style of late Romantic pieces which tend to exaggerate the tempo, but sometimes such exaggerated tempo curves are stylistically inappropriate for earlier music like the Mozart example. Second, the system is agnostic to the larger structural con-



(a) Example for Mozart's Variations on Twinkle Twinkle Little Star, K. 265, variation 7. The model seems to acquire the idiomatic slowing-down before a perfect cadence (boxed region).

(b) Example for Rachmaninoff's Piano Concerto No. 2, Mvt. 1, measure 52-61. The model seems to acquire the idiomatic arc-like tempo curve (denoted 1), but the behavior is dependent on surrounding context (denoted 2). The most dramatic drop of tempo occurs at Fm → E dim7/F progression (denoted 3).

Figure 7: Examples of the generated tempo curves.

text. For example, the Mozart example shows the A_1 section to a variation whose structure is ternary, *i.e.*, A_1BA_2. The system consistently slows down the last cadence, but it is generally appropriate to only slow down the A_2 section. Third, the method is agnostic to (1) additional cues in the music score, such as the phrase, the expression and the tempo markings, and (2) performance cues like the articulation and the dynamics.

4.4.2 *On the ease of stability analysis*

It is easy to analyze and modify the behavior of our model since we model the prediction as a linear AR model, whose properties are well-understood. We believe that such a capability to analyze and correct the system's behavior is beneficial for real-time interactive music applications such as automatic music accompaniment, since it provides an interpretable form of performance guarantee for human musicians.

To demonstrate, we have trained our model and estimated the AR coefficients at one point for a given music score S_n and past performance history τ_{n-i}, for 1-step tempo prediction. Figure 8 shows the poles and the frequency response of the inferred AR process. It can be seen, for example, that the system is unstable because the poles are outside the unit circle, resonating to oscillations at a normalized frequency of 0.15, or about a dotted eighth note. If a stable behavior is desired, then it is possible to correct the AR coefficients such that the maximum magnitude response is bounded.

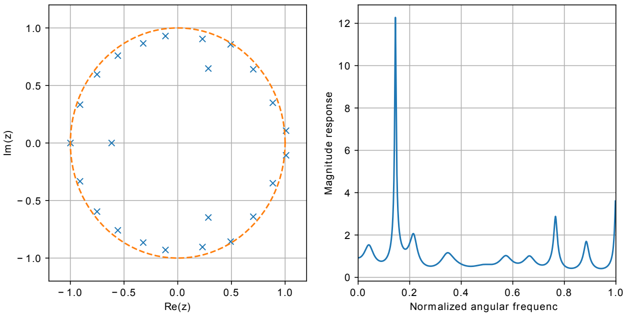


Figure 8: Pole-zero diagram (left) and the frequency response (right) of the autoregressive model inferred using our method.

5. CONCLUSION

This paper proposed an online method for predicting a human performer's expressive timing, based on the music score and the performance history. The method is both music score-aware and performance-aware, and is capable of extracting useful features from the score that are relevant to timing prediction. We have shown on a difficult dataset of expresssive virtuoso piano playing that (1) incorporating both contextual information from the performance and the music score contributes to accurate timing prediction, (2) a deep architecture, especially convolutional architecture, is useful for extracting relevant features from the music score, and (3) the model seems to acquire common idioms in piano playing, according to the generated tempo curves.

Future work includes (1) integrating the model with interactive music systems, (2) predicting more aspects of human music performance like the dynamics, (3) incorporating of more elements of the music score like the dynamics, phrasing and expressive marking, and (4) incorporation of additional performance cues such as the dynamics and body gestures for prediction.

6. REFERENCES

[1] A. Maezawa and K. Yamamoto, "MuEns: A multimodal human-machine music ensemble for live concert performance," in *Proc. CHI*, 2017, pp. 4290–4301.

[2] C. Palmer, "Music performance," *Annual review of psychology*, vol. 48, no. 1, pp. 115–138, 1997.

[3] G. Xia, Y. Wang, R. B. Dannenberg, and G. Gordon, "Spectral learning for expressive interactive ensemble music performance," in *Proc. ISMIR*, 2015, pp. 816–822.

[4] S. Van Herwaarden, M. Grachten, and W. B. De Haas, "Predicting expressive dynamics in piano performances using neural networks," in *Proc. ISMIR*, 2014, pp. 45–52.

[5] S. Wada, Y. Horiuchi, and S. Kuroiwa, "Temo prediction model for accompaniment system," in *Proc. ICMC*, 2014, pp. 1298–1303.

[6] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proc. ICMC*, 1984, pp. 193–198.

[7] A. Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music," in *Proc. ICMC*, 2008, pp. 33–40.

[8] C. Raphael, "A Bayesian network for real-time musical accompaniment," in *Proc. NIPS*, 2001, pp. 1433–1439.

[9] F. Krebs and M. Grachten, "Combining score and filter based models to predict tempo fluctuations in expressive music performances," in *Proc. SMC*, Copenhagen, Denmark, 2012.

[10] K. Okumura, S. Sako, and T. Kitamura, "Laminae: A stochastic modeling-based autonomous performance rendering system that elucidates performer characteristics," in *Proc. ICMC*, 2014.

[11] G. Widmer, S. Flossmann, and M. Grachten, "YQX plays Chopin," *AI Magazine*, vol. 30, no. 3, p. 35, 2009. [Online]. Available: https://www.aaai.org/ojs/index.php/aimagazine/article/view/2249

[12] M. Grachten and C. Cancino Chacon, *Temporal Dependencies in the Expressive Timing of Classical Piano Performances*, 04 2017, pp. 360–369.

[13] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *Arxiv*, 2016. [Online]. Available: https://arxiv.org/abs/1609.03499

[14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift." in *Proc. ICML*, 2015, pp. 448–456. [Online]. Available: http://dblp.uni-trier.de/db/conf/icml/icml2015.html#IoffeS15

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[17] E. Istok, A. Friberg, M. Huotilainen, and M. Tervaniemi, "Expressive timing facilitates the neural processing of phrase boundaries in music: Evidence from event-related potentials," *PLOS ONE*, vol. 8, no. 1, pp. 1–11, 01 2013. [Online]. Available: https://doi.org/10.1371/journal.pone.0055150

[18] H. Honing, "Is expressive timing relational invariant under tempo transformation?" *Psychology of Music*, vol. 35, no. 2, pp. 276–285, 2007. [Online]. Available: https://doi.org/10.1177/0305735607070380

[19] B. H. Repp, "Sensorimotor synchronization: a review of the tapping literature," *Psychonomic bulletin & review*, vol. 12, no. 6, pp. 969–992, 2005.

[20] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," in *Proc. ISMIR*, 2015.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[22] R. Yamamoto, S. Sako, and T. Kitamura, "Robust online algorithm for real-time audio-to-score alignment based on a delayed decision and anticipation framework," in *Proc. ICASSP*, May 2013, pp. 191–195.

[23] D. Stowell and E. Chew, "Bayesian MAP estimation of piecewise arcs in tempo time-series," in *Proc. CMMR*, 2012.