Vivace: Web Application for Real-Time feedback on Piano Performance

Jeremy Lee
UC Berkeley
jrmylee@berkeley.edu

Carmine Cella
UC Berkeley
carmine.cella@berkeley.edu

Hélène-Camille Crayencour
Centre National de la Recherche Scientifique
helene.camille.crayencour@gmail.com

Abstract

Software that provide feedback on music performance are being increasingly used by music students to support their daily practice. The dominant piano performance evaluation models are based on metrics such as playing correct notes and global tempo. However, these metrics are often limited in capturing expressive aspects of performance, which can be seen through slight variations a performer chooses to employ in their playing. In order to capture some of these variations, we propose a web application that serves to compare the user's playing to that of professional recordings of the same piece. The users playing is plotted in real time on a tempo-volume graph, alongside performances by great pianists such as Rubinstein and Horowitz. We then give this application to music students and from the feedback we receive, determine in what kind of live performance feedback will be useful to pianists.

1 Introduction

Technologies for audio processing, specifically in music, have become increasingly important, underscored by the launch of projects such as Google Magenta, which is a project aimed "to advance the state of the art in machine intelligence for music and art generation"[24]. An example of this is wave2midi2wave, which enables training of models that transcribe and compose new music using a notes as an intermediate representation of audio [19]. On the commercial end, Apps such as Simply Piano and FlowKey aim to use machines in order to give feedback to performers by processing audio streamed from the microphone, analyzing the audio, and providing an evaluation to the performer.

In this research, we develop a web application that compares a piano performance streamed from the microphone to performances from professional recordings of the same piece. The main problem that we aimed to solve with this system is how to provide feedback to a performer on expressiveness in their performance, as opposed to more elementary performative aspects like playing the correct keys. We chose to model expressiveness from the frame of tempo and volume varying across time.

Our proposed application, Vivace, displays on plots tempo and volume, with tempo on the x-axis and volume on the y-axis, and does so in 1 second intervals. The performer is able to select from a list of pieces Vivace supports, and when the user clicks start, the performance's tempo and volume is plotted alongside performances from 3-5 professional performers. These other performances are aggregated from Youtube, and the tempo and volume analysis is pre-computed and stored on a NoSQL database on the web server. Vivace uses a combination of signal processing techniques in order to provide near real-time feedback to the performer.

In addition to the development of this application, another contribution of this research is a user study that provides new insight on the conceptual, technical and design challenges that researchers and engineers face when developing applications to help students learning an instrument.

3rd Conference on AI Music Creativity (AIMC 2022)

2 Motivations and Background

2.1 Music Performance and Instrumental Music Learning

In music performance, the performer has the role of conveying the composers's ideas to the listener [1]. In this system of communication the composer's ideas are conveyed to the listener through the acoustic signal produced by the performer [3]. Music performance involves the performers' conceptual interpretation of the musical composition [2]. The acquisition of performance skills for a performer is a complex and long process that involves several aspects. Among them, expressiveness is a key aspect of performance, and the issue of teaching and learning expressiveness in instrumental music learning is addressed by a growing body of research [23].

The learning process generally involves aural modeling [8]. Students may learn from examples performed by their teacher. Research studies have also highlighted that it is common that performers listen to others' interpretations of a piece of music to help them understanding expressive intentions from the score or to provide guidance on their own conceptual interpretation of the musical composition [17]. Besides aural modeling and analysis of recordings, some studies have shown that imitative learning, that consists in imitating expressive timing of interpretations by others, can be used as a tool to help students developing their own musical insights and raise their awareness of music expression [4, 10, 9].

The recent advances in machine learning and computer science have given music technology an increasingly important role in educational activities [18]. It has fostered the development of tools for facilitating students learning, supported by recent developments in the Music Information Retrieval (MIR) research. For instance *Harmonic Touch*, a web platform proposes exercises to assess tonal harmony awareness. Some software also provide students with tools to accompany them in their music making, such as *Tonara*[1] for interactive score-following, or *Chordify*[2] that provides automated melody accompaniment. In this context, the development of tools for automatic music performance assessment [20, 22] and learning by visual feedback [14, 21, 15] is becoming an important area of research in the MIR field.

As noted in a recent review on performance evaluation technologies for the support of musical instrument learning [21], understanding the musical learning process and conceptualizing the various phases of music learning remains an open challenge. The main challenge in developing applications that can help musical students are in effectively developing computational approaches and technologies that provide meaningful feedback to the users. Moreover, there seems to be two specific challenges in developing such technologies. The first is that of finding useful metrics and best visualizing those metrics in an interface that music students can use seamlessly in their practice sessions. The second challenge is developing optimal computational approaches that can compute these metrics efficiently, so that feedback can be given to the user in real time.

2.2 Performance Assessment Technologies for Piano Learning

Tempo and dynamics are key features among the various aspects involved in expression in performance [7, 11]. In particular, in the context of modeling expressive piano performance [12], it has been shown that they are important parameters for capturing some aspect of the pianists' playing style.

There are a few applications that are considered state-of-the-art in performance evaluation for the piano. The most popular apps are Flowkey[26] and JoyTunes[25], which both track key presses and matches it to the selected piece. However, neither of these applications gives feedback on dynamics of the player, like tempo markings or dynamic markings, which are crucial to how people perceive and enjoy music [6].

To the best of our knowledge, there is no application that provides automatic feedback on expressiveness of piano performances in real time. Applications like Simply Piano [25] do not provide a evaluation of features like tempo and volume, which are crucial to measure expressiveness in musical performances.

[1] `https://www.tonara`

[2] `https://chordify.net`

Window Length (seconds)	MSE
1	9868
2	17541.0
3	22716.0
4	32188.0
5	34610.0

Figure 1: Non-Overlapping Window Tempo and mean squared error (MSE) of the tempo function

3 Methodology

Vivace is a web-based interactive interface for visualizing music analysis data, specifically for modeling tempo and volume across time for the student's performance along with other reference performances. It was developed to help performers gauge how they are performing relative to professional recordings of their piece.

The application, at startup, precomputes tempo and volume segments from a database of mp3 files of relevant performances. The application computes tempo and volume from each audio signal in an overlapping fashion, where we take 3 second chunks of audio and compute tempo and volume, and do this every 1 second. This allows Vivace to have more granular analysis, with less variation in the outputs due to higher tempo/volume domain resolution.

The segmentation parameters (window size of 3 seconds and a hop length of 1 second) have been determined by minimizing mean squared error of the tempo function as described below.

3.1 Tempo Function

The tempo function that is used is based on the tempo function used in the Librosa Python library [13]. The idea from this method of computing tempo is that a cyclic tempogram is formulated, which is a time-tempo representation for an audio signal. This tempogram can be seen as the tempo analogue to the spectrogram, which is a time-frequency representation of audio signals. The tempoogram computations are abstracted away by the Librosa library, and we simply apply the tempo function to reference performance signals as well as to the raw user performance from the microphone. The issue that we encountered was determining the window sizes for the audio signal that would lead to the most accurate prediction of tempo, since the tempo function requires a signal of reasonable length.

Since our tempo and volume functions are designed to be used on chunks of audio signal, we needed a way to best segment our signals to both maximize responsiveness(frequency of outputs to the user) and accuracy, since there is a trade off between the two. To illustrate this, having a large chunk or window size would give us a more accurate tempo approximation over a longer period of time, since we have more information about the performance. However, this comes at expensive of requiring more data, and thus slower tempo output to the user. The goal is to have a continuous flow of tempo and volume readings in the interface.

In order to determine the best method of segmenting the audio signals to run these functions on, we proceeded as follows. First, we generate randomly generate audio signals of 30 seconds each. Each audio clip contains segments of onsets that vary in tempo, with the length of each segment being randomized, as well as the tempo being randomized. The audio clip, along with a CSV containing labels of the audio clips (with columns: interval length, tempo(BPM)) are generated. From there, we run validation on our tempo function, varying our methods of inputting our signals into the tempo functions. We consider three methods of segmenting the signal, which are described as follows.

The first method is the simplest, with our tempo function being run on non-overlapping windows. We the vary window sizes from $n = 1$ to $n = 6$ and plot the results. The results are shown in Figure 1.

The second method involves using a median filter. Tempo is calculated on windows of length 1 second, and every k seconds the median of the previous tempos are outputted. For instance, with a window size of 1 second and k value of 3, our function would output a tempo value every 3 seconds. The results are shown in Figure 2.

Window Length (seconds)	MSE
1	25940.0
2	30933.0
3	30965.0
4	28598.0
5	28891.0

Figure 2: Median Filtered Window Tempo and mean squared error (MSE) of the tempo function

Window Length (seconds)	MSE
1	9868.0
2	9312.0
3	14445.0
4	16285.0
5	17454.0

Figure 3: Overlapping Window Tempo and Mean Squared Error(MSE) of the tempo function

Finally, the last method involves using overlapping windows of varying window sizes. Tempo is calculated on windows of length w, and with a hop size of 1 second. The results of this function are shown in Figure 3, with this method with window length of 2 seconds and hop size of 1 second leading to the best performance.

We ended up selecting the function that had the lowest overall mean squared error across the entire data set. On average, the overlapping window function performed the best, with window size of 2 seconds performing the best within that method. From a user standpoint, this method also happens to result in the most user friendly interface, since we manage to get a relatively large amount of information (2 seconds of audio input), while providing a tempo and volume in short intervals of 1 second. The downside to this method is that the inputs at the beginning and end have only 1 second of information and are thus zero-padded to 2 seconds.

3.2 Volume Function

An important property of dynamics is that of volume. Sound power expresses how much energy per unit time is emitted, through the air, by a sound source. Sound intensity, on the other hand, is used to denote the sound power per unit area, and is used in reference to a constant value, such as the threshold of hearing [16]. This application uses decibels to measure of the sound intensity, and is formulated as follows: $dB(I) := 10 * \log(\frac{I}{I_0})$, where I represents the sound intensity and I_0 represents the reference intensity. The reference intensity is set to be 10^{-12}, which is the minimum sound intensity of a pure tone a human can hear [16].

3.3 Interface

The Vivace interface is built in React.js, using Victory.js for charting performance data, doing so with a modular design so that features in the future can be easily implemented. The crux of the data processing happens through a Web Audio and Socket.io pipeline. The users web browser gains access to the microphone (if granted by the user), and from there the React application listens to a Web Audio event that fires at a sample rate determined by the browser. This event sends packets of audio which is emitted through a Socket.io event, for the backend to collect and do data processing on.

The Vivace backend is built in Python, and is served as a Flask server. The server exposes several Socket.io events that are used to collect and emit data for the frontend. The first event is one that is sent by the frontend interface which sends relevant metadata such as browser sampling rate and the piece that user selected. Another event that is defined is one that accepts an audio packet, and the frontend sends this audio at every $1/T$ seconds, where T is the browsers sampling rate. Finally, we have an event that is fired by the browser every 1 second, which asks the server to compute tempo and volume for the audio that has been collected by the server. At this point, two seconds of audio have been collected, except for the at the beginning where we do not have enough information and

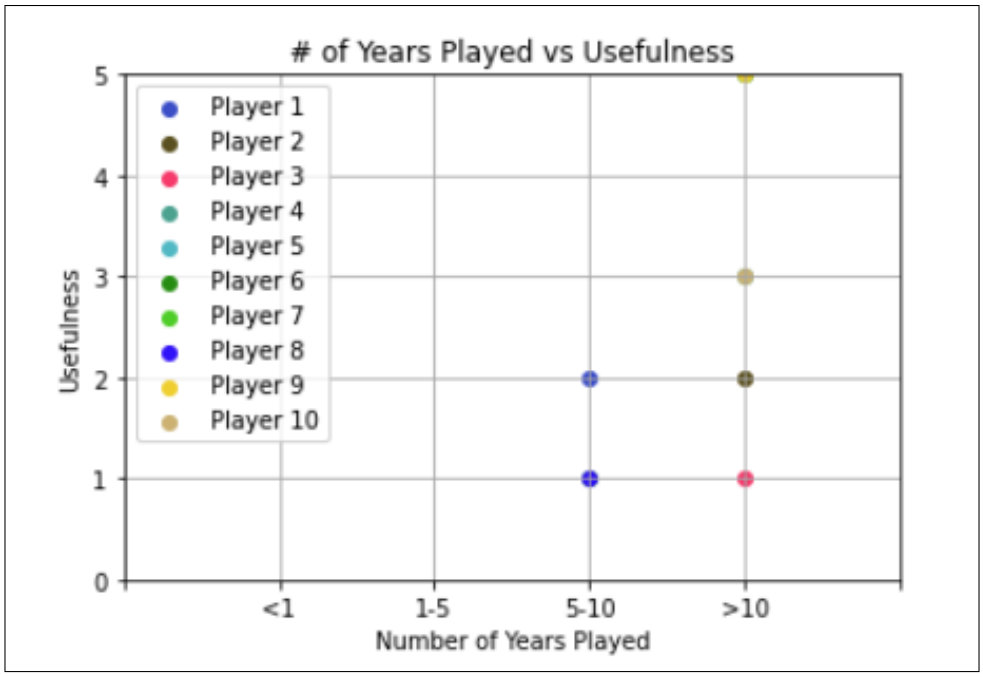

Figure 4: Number of Years Played vs Usefulness of Use

the signal is zero padded to 2 seconds. Tempo and volume is calculated using this signal, and emitted to the browser. After this, the server cuts the first 1 second of the signal out, so that once another 1 second of audio is received, this processed is repeated on an overlapped window of length 2 seconds.

4 Experiments

We deployed Vivace to the cloud and distributed the application to music students and pianists to use(10 users total). We asked each player to choose one piece from a list of beginner to intermediate level classical pieces. We asked each player to play a piece from beginning to end, multiple times if needed, and give feedback via a survey.

On a high level, most players that used the application were advanced level players, with 70 percent of the users saying they have played more than 10 years. The remaining users reported having played piano for 5-10 years. 60 percent of the pianists reported being 15-20 years old, and the remaining as older than 20 years old. We asked each user how accurate they thought the application was in measuring tempo and volume, and 44 percent of the users rated it a 3 out of 5, and 33 percent rated it 4 out of 5. There was also very little overlap in the pieces that each user played, with only three pieces being chosen by more than 1 user.

The types of questions that we asked varied, starting with specific questions about the application and its usefulness, and ending with questions about the users personal experiences in music and general feedback they had for us. Specifically, we wanted to correlated skill level or years played with metrics like how useful this application this was, and how easily they were able to focus on the application while playing.

The questions were designed in order to gain feedback on a few different things. First, we wanted to know what kind of users this would most benefit. We also wanted to know if the method of giving feedback, a graph that updates in real-time, would give valuable information to the user, as well as if the information was easily digestible to the user while they are practicing.

5 Discussion

In Figure 4, it is evident that advanced players (more than 10 years of playing) found this application more useful than intermediate players. The detailed answers of the participants give some explanations. The advanced pianists in the survey responded that they did not expect this application to be useful to beginner or intermediate pianists because of how large-scale the feedback on their expressivity was. They added that beginner and intermediate pianists typically need more detailed direction in order to improve their playing. Advanced users are able to change the style of their playing depending on the application feedback on how close they are to a specific performer, but generally beginner and intermediate players are not able to make those kinds of judgement calls, either due to technical limitations or lack of musical maturity.

The advanced pianists also explained that the application, while accurate, was hard to keep track of while playing, which suggests the need to create an interface that reveals more information at cursory glances of the screen. The need for a more informative at-a-glance interface may also suggest the need for information that better interprets the data. Specifically, there have been research done on the effect that aspects of performance such as tempo, pedalling, and timing have on the perceived mood of the music [5]. Integrating the effect that metrics such as tempo and volume have on the listener's perception of a performance into an application can serve to give more digestible and easily interpreted feedback for performers, especially those early in their studies.

6 Conclusion

In this paper, we propose a web application that gives real-time feedback on piano performance. The application plots the users tempo and volume against the tempo and volumes of professional pianists playing the same piece. In providing a set of metrics that a piano player can use in real-time to compare to other performers, the users could adjust their performance with feedback from the application.

The experiments that we conducted were aimed to find out which users this application should target, and if the interface was designed in a way that would be useful to performers, specifically with respect to giving feedback on the expressive aspects of performance. We found, from the intermediate to advanced musicians we surveyed, that advanced players were more likely to find Vivace more useful than beginner and intermediate pianists. This was because advanced players were more likely to be able to make adjustments to their playing based on how they were playing relative to other pianists, likely due to more technical skill and musical maturity. However, often times even the advanced pianists found the interface too difficult to keep track of while performing, suggesting the need for a interface that provides more at-a-glance information to the user. Furthermore, users suggested more visualizations, such as tempo and volume vs time, that would provide useful metrics for them. Feedback from intermediate to advanced musicians on Vivace underscores the technical and design challenges that engineers face when developing applications to help students learning an instrument. We plan on using what we learned from writing this paper in order to create a more useful and easy to use application for students in a variety skill levels, and to hopefully make learning a musical instrument more accessible than ever.

References

[1] Igor Stravinsky. *Poetics of music in the form of six lessons*. Vol. 66. Harvard University Press, 1947.

[2] Edward T Cone. *Musical form and musical performance*. Ed. by New York: Norton. 1968.

[3] Roger A. Kendall and Edward C. Carterette. "The Communication of Musical Expression". In: *Music Perception: An Interdisciplinary Journal* 8.2 (1990), pp. 129–163.

[4] Eric F Clarke. "Imitating and evaluating real and transformed musical performances". In: *Music Perception* 10.3 (1993), pp. 317–341.

[5] Alf Gabrielsson and Patrik N. Juslin. "Emotional Expression in Music Performance: Between the Performers Intention and the Listeners Experience". In: *Psychology of Music* 24.1 (1996), pp. 68–91. DOI: 10.1177/0305735696241007.

[6] Stuart B. Kamenetsky, David S. Hill, and Sandra E. Trehub. "Effect of Tempo and Dynamics on the Perception of Emotion in Music". In: *Psychology of Music* 25.2 (1997), pp. 149–160. DOI: 10.1177/0305735697252005.

[7] Caroline Palmer. "Music performance". In: *Annual review of psychology* 48.1 (1997), pp. 115–138.

[8] Robert H Woody. "The relationship between musicians' expectations and their perception of expressive features in an aural model". In: *Research Studies in Music Education* 18.1 (2002), pp. 57–65.

[9] K. Riley and E. Coons. "Improving pianists' rhythmic performance in score reading through imitation and feedback". In: *Journal of Technology in Music Learning* 3 (2005), pp. 2–14.

[10] Lisboa Tânia et al. "Mastery through imitation: A preliminary study". In: *Musicae Scientiae* 9.1 (2005), pp. 75–110.

[11] Renee Timmers. "Predicting the similarity between expressive performances of music from measurements of tempo and dynamics". In: *The Journal of the Acoustical Society of America* 117.1 (2005), pp. 391–399.

[12] Asmir Tobudic and Gerhard Widmer. "Learning to Play Like the Great Pianists." In: *IJCAI*. 2005, pp. 871–876.

[13] Peter Grosche, Meinard Muller, and Frank Kurth. "Cyclic tempogram—A mid-level tempo representation for musicsignals". In: *2010 IEEE International Conference on Acoustics, Speech and Signal Processing* (2010). DOI: 10.1109/icassp.2010.5495219.

[14] Alex Brandmeyer et al. "Learning expressive percussion performance under different visual feedback conditions". In: *Psychological Research* 75.2 (2011), pp. 107–121.

[15] Renee Timmers and Makiko Sadakata. "Training expressive performance by means of visual feedback: existing and potential applications of performance measurement techniques". In: *Expressiveness in music performance* (2014), pp. 304–330.

[16] MEINARD MLLER. *FUNDAMENTALS OF MUSIC PROCESSING: audio, analysis, algorithms, applications*. SPRINGER, 2016.

[17] Georgia Volioti and Aaron Williamon. "Recordings as learning and practising resources for performance: Exploring attitudes and behaviours of music students and professionals". In: *Musicae Scientiae* 21.4 (2017), pp. 499–523.

[18] Federico Avanzini et al. "A web platform to foster and assess tonal harmony awareness". In: *International Conference on Computer Supported Education*. Springer. 2019, pp. 398–417.

[19] Curtis Hawthorne et al. *Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset*. 2019. arXiv: 1810.12247 [cs.SD].

[20] Alexander Lerch et al. "Music performance analysis: A survey". In: *arXiv preprint arXiv:1907.00178* (2019).

[21] Vsevolod Eremenko et al. "Performance assessment technologies for the support of musical instrument learning". In: (2020).

[22] Alexander Lerch et al. "An Interdisciplinary Review of Music Performance Analysis". In: *TISMIR* (2021).

[23] Henrique Meissner. "Theoretical Framework for Facilitating Young Musicians' Learning of Expressive Performance". In: *Frontiers in psychology* 11 (Jan. 2021), pp. 584171–584171.

[24] URL: https://magenta.tensorflow.org/.

[25] URL: https://www.joytunes.com/.

[26] *Learn How to Play Piano Online - Piano Learning App*. URL: https://www.flowkey.com/en.