Neural Computing and Applications (2020) 32:955–967
https://doi.org/10.1007/s00521-018-3758-9

DEEP LEARNING FOR MUSIC AND AUDIO

This time with feeling: learning expressive musical performance

Sageev Oore[1,2] · Ian Simon[3] · Sander Dieleman[4] · Douglas Eck[3] · Karen Simonyan[4]

Received: 23 December 2017 / Accepted: 28 September 2018 / Published online: 14 November 2018

Abstract
Music generation has generally been focused on either creating scores or interpreting them. We discuss differences between these two problems and propose that, in fact, it may be valuable to work in the space of direct *performance* generation: jointly predicting the notes *and also* their expressive timing and dynamics. We consider the significance and qualities of the dataset needed for this. Having identified both a problem domain and characteristics of an appropriate dataset, we show an LSTM-based recurrent network model that subjectively performs quite well on this task. Critically, we provide generated examples. We also include feedback from professional composers and musicians about some of these examples.

Keywords Music generation · Deep learning · Recurrent neural networks · Artificial intelligence

1 Preamble/request

Recognizing that "talking about music is like dancing about architecture",[1] we kindly ask the reader to listen to the linked audio in order to effectively understand the motivation, data, results, and conclusions of this paper. As this research is ultimately about producing music, we believe the actual results are most effectively perceived—indeed, only perceived—in the audio domain. This will provide necessary context for the verbal descriptions in the rest of the paper.

2 Introduction

In this work, we discuss training a machine-learning system to generate music. The first two keywords in the title are *time* and *feeling*: not coincidentally, our central thesis is that, given the current state of the art in music generation systems, it is effective to generate the expressive timing and dynamics information concurrently with the music. Here, we do this by directly generating improvised performances rather than creating or interpreting scores. We begin with an exposition of some relevant musical concepts.

2.1 Scores, performances, and musical abstraction

Music exists in the audio domain and is experienced through individuals' perceptual systems. Any "music" that is not in the audio domain (e.g., a text or binary file of any sort) is of course a representation of music: if it is not physically vibrating, it is not (yet) sound, and if it is not sound, it is certainly not music. The obvious implication is that for any representation, there are additional steps to transform that representation—whatever it might be—into sound. Those steps might be as local as the conversion from digital to analog waves, or as global as the human

Sageev Oore: Work done while author was at Google Brain.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00521-018-3758-9) contains supplementary material, which is available to authorized users.

✉ Sageev Oore
sageev@dal.ca

[1] Vector Institute, Toronto, Canada

[2] Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada

[3] Google Brain, 1600 Amphitheatre Rd, Mountain View, CA, USA

[4] DeepMind, 6 Pancras Square, London N1C 4AG, UK

[1] This quote has been attributed to a range of individuals from Laurie Anderson to Miles Davis, and numerous others.

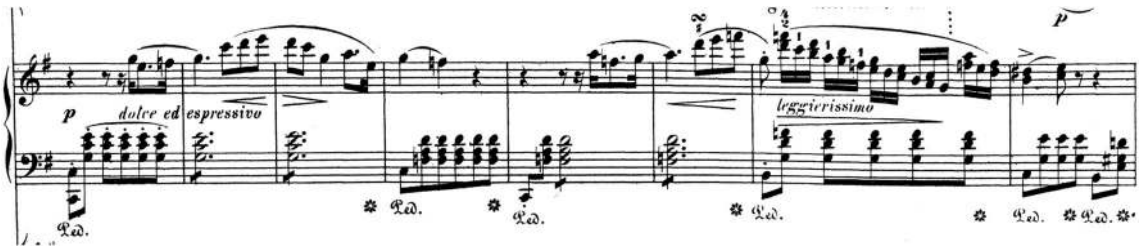

Fig. 1 Excerpt from the score of Chopin's Piano Concerto No. 1

performance of written score, for example. In generating music,[2] therefore, one must be aware of which of those steps is addressed directly by their generative system, which ones must be addressed in other ways, and, importantly, the impact of all of those choices on the listener's perception of the music, where it is ultimately experienced.

A defining characteristic of a representation, then, is what is omitted: what still needs to be added or done to it in order to create music from it, and the relation of that abstraction to our perceptual experience. With that consideration in mind, we now discuss some common symbolic representations.

2.1.1 Scores

Figure 1 is an example of a musical score [7]. It shows which notes to play and when to play them relative to each other. The timing in a score is aligned to an implicit and relative *metrical grid*. For example, quarter notes are the same duration as quarter note rests, twice the duration of eighth notes, and so on. Some scores additionally specify an absolute tempo, for example, in quarter notes per minute.

And yet, by the time the music is heard as audio, most of this timing information will have been intentionally *not followed exactly*! For example, in classical music from the 1800s onward, *rubato* developed: an expressive malleability of timing that overrides metrical accuracy (i.e., can deviate very far from the grid), and this device is both frequent and essential for making perceptual sense of certain pieces. Another example of a rhythmic construct that is not written in scores is *swing*, a defining quality of many African American music traditions.[3]

But tempo is not the only way in which the score is not followed exactly. *Dynamics* refers to how the music gets louder and quieter. While scores do give information about dynamics, in this respect, too, their effectiveness relies heavily on conventions that are not written into the score. For example, where the above score says "*p*," it means to play quietly, but that does tell us how quietly, nor will all the notes be equally quiet. When there is a *crescendo* marking indicating to get louder, in some cases the performer will at first get momentarily quieter, creating space from which to build. Furthermore, when playing polyphonic piano music, notes played at the same time will usually be played at different dynamic levels and articulated differently from one another in order to bring out some voices over others.

Phrasing includes a joint effect of both expressive timing and dynamics. For example, there is a natural correlation between the melody rising, getting louder, and speeding up. These are not rules, however; skilled performers may deliberately choose to counteract such patterns to great effect.

We can think of a score as a highly abstract representation of music. The effective use of scores, i.e., the assumption by a composer that a score will subsequently be rendered well as music, relies on the existence of conventions, traditions, and individual creativity. For example, Chopin wrote scores where the pianist's use of *rubato* is expected; indeed, the score requires it in order to make sense. Similarly, the melodies in jazz lead sheets were written with the understanding that they will be *swung* and probably embellished in various ways. There are numerous other instrument-specific aspects that scores do not explicitly represent, from the vibrato imbued by a string player to the tone of a horn player. Sometimes, the score won't make perceptual sense without these elements.

In short, the mapping from score to music is full of subtlety and complexity, all of which turns out to be very important in the perceptual impact that the music will have. To get a sense of the impact of these concepts, we recommend that the reader listen:

- first to a *direct rendering* of the above score here: https://clyp.it/jhdkghso, played according to the written grid and quantized to 16th notes. Then,
- listen to an *expressive performance* [33] of it here: https://clyp.it/x24hp1pq.

[2] In this text, we use the term "generation" to refer to computational generation, as opposed to human creation or performance.

[3] While explaining swing is outside the current scope, we do note that it is occasionally incorrectly described in terms of triplets.

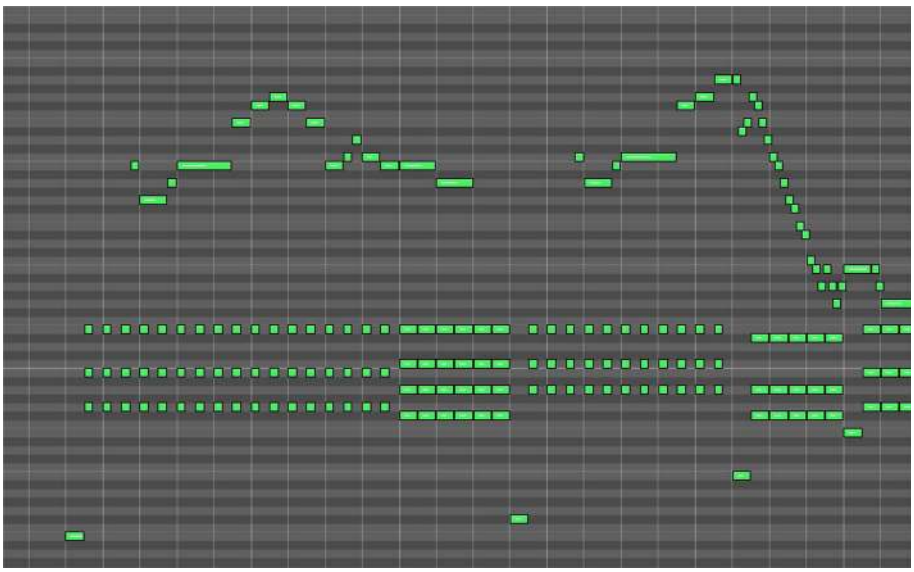

Fig. 2 Piano roll based on the score in Fig. 1. The horizontal axis represents time; the vertical axis represents pitch; each rectangle is a note; and the length of the rectangle corresponds to the duration of the note

2.1.2 MIDI

MIDI is a communication protocol for digital musical instruments: a symbolic representation, transmitted serially, that indicates NOTE_ON and NOTE_OFF events and allows for a high temporal sampling rate. The loudness of each note is encoded in a discrete quantity referred to as *velocity* (the name relates to how fast a piano key is pressed). While MIDI encodes note timing and duration, it does not encode qualities such as timbre; instead, MIDI events are used to trigger playback of audio samples.

MIDI can be visualized as a piano roll—a digital version of the old player piano rolls. Figure 2 is an example of a MIDI piano roll corresponding to the score shown in Fig. 1. Each row corresponds to one of the 128 possible MIDI pitches. Each column corresponds to a uniform time step. If note i is ON at time t and had been pressed with velocity v, then element $(t, i) = v$. So, at 125 Hz, 6 s of MIDI data would be represented on a grid of size $128 \times (6 \times 125)$. Actual MIDI sampling can be faster than this, so even at 125 Hz we are still subsampling from the finest available temporal grid.

We refer to a score that has been rendered directly into a MIDI file as a **MIDI Score**. That is, it is rendered with no dynamics and exactly according to the written metrical grid. As given earlier, https://clyp.it/jhdkghso is an example of this.

If, instead, a score has been performed, by a musician for example, and that performance has been encoded into a MIDI stream, we refer to that as a **MIDI Performance**. https://clyp.it/x24hp1pq is an example (also given previously) of a MIDI performance.

3 Factoring the music generation process: related work

Figure 3 shows one way of factoring the music generation process. The first stage shown in this figure is composition, which yields a score. The score is then performed. The performance is rendered as sound, and finally that sound is perceived. In the analog world, of course, performance and rendering the sound are the same on a physical instrument, but in the digital world, those steps are often separate. While other views of the process are possible, this one provides us a helpful context for considering much of the existing relevant work. Noting that sound generation and perception (the last two steps in Fig. 3) are outside our scope, in the rest of this section we focus primarily on composition and performance.

Perhaps it is precisely because music is so often perceived as a profoundly human endeavor that there has also been, in parallel, an ongoing fascination with automating its creation. This fascination long predates notions such as the Turing test (ostensibly for discriminating automation of the most human behavior) and has spawned a range of efforts: from attempts at the formalization of unambiguously strict rules of composition to incorporation of complete random chance into scores and performances. The use of rules exemplifies the algorithmic (and largely deterministic) approach to music generation, one that is interesting and outside the scope of the current work; for background on this, we refer the reader, for example, to the text by Nierhaus [31]. Our present work, on the other hand, lies in a part of the spectrum that incorporates probability and sampling.

Aleatory refers to music or art that involves elements of randomness, derived from the Latin *alea (alee)*, meaning

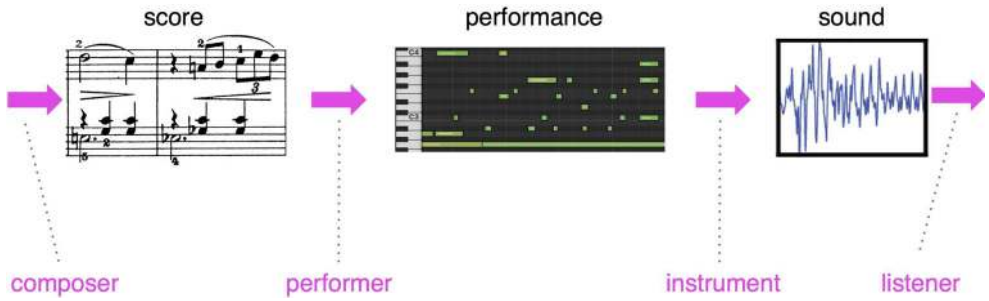

Fig. 3 Factoring music generation. We can see music as starting with the composition of a score; that score gets turned into a performance (shown as a MIDI piano roll); that MIDI roll, in turn, gets rendered into sound using a synthesizer; and finally the resulting audio gets perceived as music by a human listener

"die (dice)." Dice were used in the 1700s to create music in a game referred to as *Musikalisches Würfelspiel* [1, 16, 31]: the rolled numbers were used to select from pre-composed fragments of music. Some of these compositions were attributed to Mozart and Haydn, though this has not been authenticated.

Two centuries later, as the foundations of AI were being set, the notion of automatically understanding (and therefore generating) music was among the earliest applications to capture the imagination of researchers, with papers on computational approaches to perception, interpretation, and generation of music by Simon, Longuet-Higgins, and others [23–26, 36]. Since then, many interesting efforts were made [8, 15, 19, 30, 34, 40], and it is clear that in recent years both interest and progress in score generation have continued to advance, for example Lattner et al. [22], Boulanger-Lewandowski et al. [2], Bretan et al. [4], Herremans et al. [17], Roberts et al. [35], Sturm [37], to name only a few. Briot et al. [5] provide a survey of generative music models that involve machine learning. Herremans et al. [18] provide a comprehensive survey and satisfying taxonomy of music generation systems. McDonald [28] gives an overview highlighting some key examples of such work.

Corresponding to the second step in Fig. 3 is a body of work often referred to as EMP (Expressive Musical Performance) systems. For example, the work by Chacon and Grachten [6], inspired by the Linear Basis Models proposed by Grachten and Widmer [14], involves defining a set of hand-engineered features, some of which depend on having a score with dynamic expression marks, others on heuristics for musical analysis (e.g., a basis function indicating whether the note falls on the first beat of a measure of 4/4). Widmer and Goebl [44] and Kirke and Miranda [21] both present extensive and detailed surveys of work done in the field of computational EMPs. In the latter survey, the authors also provide a tabular comparison of 29 systems that they have reviewed. Out of those systems, two use neural networks (one of which also uses performance rules) and a few more use PCA, linear regression, KCCA, etc. Some of the other systems that involve some learning do so by learning rules in some way. For example, the KTH model [10] consists of a top-down approach for predicting performance characteristics from rules based on local musical context. Bresin [3] presents two variations of a neural network-based system for learning how to add dynamics and timing to MIDI piano performance.

Grachten and Krebs [13] use a variety of unsupervised learning techniques to learn features with which they then predict expressive dynamics. Building on that work, van Herwaarden et al. [43] use an interesting combination of an RBM-based architecture, a note-centered input representation, and multiple datasets to—again—predict expressive dynamics. In both of these cases, the dynamics predictions appear to depend on the micro-timing rather than being predicted jointly as in the present work.

Teramura et al. [38] observe that many previous performance rendering systems "often consist of many heuristic rules and tend to be complex. It makes [it] difficult to generate and select the useful rules, or perform the optimization of parameters in the rules." They thus present a method that uses Gaussian Processes to achieve this, where some parameters can be learned. In their ostensibly simpler system, "for each single note, three outputs and corresponding thirteen input features are defined, and three functions each of which returns one of three outputs and receive the thirteen input features, are independently learned." However, some of these features, too, depend on certain information; for example, they compute the differences between successive pitches, and this only works in compositions where the voice leading is absolutely clear; in the majority of classical piano repertoire, this is not the case. In Laminae, Okumura et al. [32] systematize a set of context-dependent models, building a decision tree which allows rendering a performance by combining contextual information.

Moulieras and Pachet [29] use a maximum entropy model to generate expressive music, but their focus is again

monophonic plus simple harmonic information. They also explicitly assume that "musical expression consists in local texture, rather than long-range correlations." While this is fairly reasonable at this point, and indeed it is hard to say how much long-range correlation is captured by our model, we wished to choose a model which, at least in principle, allowed the possibility of modeling long-range correlation: ultimately, we believe that these correlations are of fundamental importance. Malik and Ek [27] use a neural network to learn to predict the dynamic levels of individual notes while assuming quantized and steady timing.

4 Choosing Assumptions and a Problem Domain

4.1 Assumptions

While we have briefly surveyed systems including both compositional and EMP, we note that our focus is to generate new music, not to perform existing scores (an interesting problem in itself, for which there are some very effective systems as described above). In this context, computational models naturally make assumptions; let us review potential implications of some of these when generating music and identify some of the choices we make in our own model in these respects.

- *Metric Abstraction* Many compositional systems abstract rhythm in relation to an underlying grid, with metric-based units such as eighth notes and triplets. Often this is further restricted to step sizes at powers of two. Such abstraction is oblivious to many essential musical devices, including, for example, rubato and swing as described in Sect. 1.1.1.
  We choose a temporal representation based on absolute time intervals between events, rounded to 8 ms.
- *No Dynamics* Nearly every compositional system represents notes as ON or OFF. This binary representation ignores dynamics, which constitute an essential aspect of how music is perceived. Even if dynamic level were treated a global parameter applied equally to simultaneous notes, this would still defeat the ability of dynamics to differentiate between voices, or to compensate for a dense accompaniment (that is best played quietly) underneath a sparse melody.
  We allow each note to have its own dynamic level.
- *Monophony* Some systems only generate monophonic sequences. Admittedly, this is a natural starting point: the need to limit to monophonic output is in this sense entirely understandable. This can work very well for instruments such as voice and violin, where the performer also has sophisticated control beyond quantized pitch and the velocity of the note attack. The perceived quality of monophonic sequences may be inextricably tied to these other dimensions that are difficult to capture and usually absent from MIDI sequences.

  In our experience, the leap from monophonic to polyphonic generation is a significant one. A survey of the literature shows that most systems that admit polyphony still make assumptions about its nature—either that it is separable into chords, or that it is separable into voices, or that any microvariation in tempo applies to all voices at once (as opposed to allowing one voice to come in ahead of the beat), and so forth. Each of these assumptions is correct only sometimes. We settled on a representation that turned out to be simpler and more agnostic than this, in that it does not make any of these assumptions:
  We specify note events one at a time, but allow the system to predict an arbitrary number of simultaneous notes, should it be so inclined.

Generally speaking, in contrast to many of the methods discussed in Sect. 2, our approach makes no assumptions about the features other than the information that is known to exist in MIDI files: velocity, timing, and duration of each note. We do not require computing or knowing the time signature, we do not require knowing the voice leading, we do not require inferring the chord, and so on. While additional information could be both useful and interesting, given the current state of the art and available data, we are focused on showing how much can be done without defining any rules or heuristics at all; we simply try to model the distribution of the existing data. Listening to some of the examples, one hears that our system generates a variety of natural time feels, including 3/4, 4/4, and odd time signatures, and they never feel rhythmically heavy-handed.

4.2 Problem Domain: Simultaneously Composing and Performing

In Fig. 4, we show a few different possible entry points to the music generation process. For example, at one extreme, we can subsume all steps into a single mechanism so as to predict audio directly, as is done by WaveNet, with impressive results [42]. Another approach is to focus only on the instrument synthesis aspect [9], which is an interesting problem outside the scope of our present work. As described in Sect. 2, compositional systems generate scores that require performances, and EMP systems take scores and generate corresponding performances.

Here, we demonstrate that *jointly predicting composition and performance with expressive timing and dynamics*, as illustrated in Fig. 4d, is another effective domain for

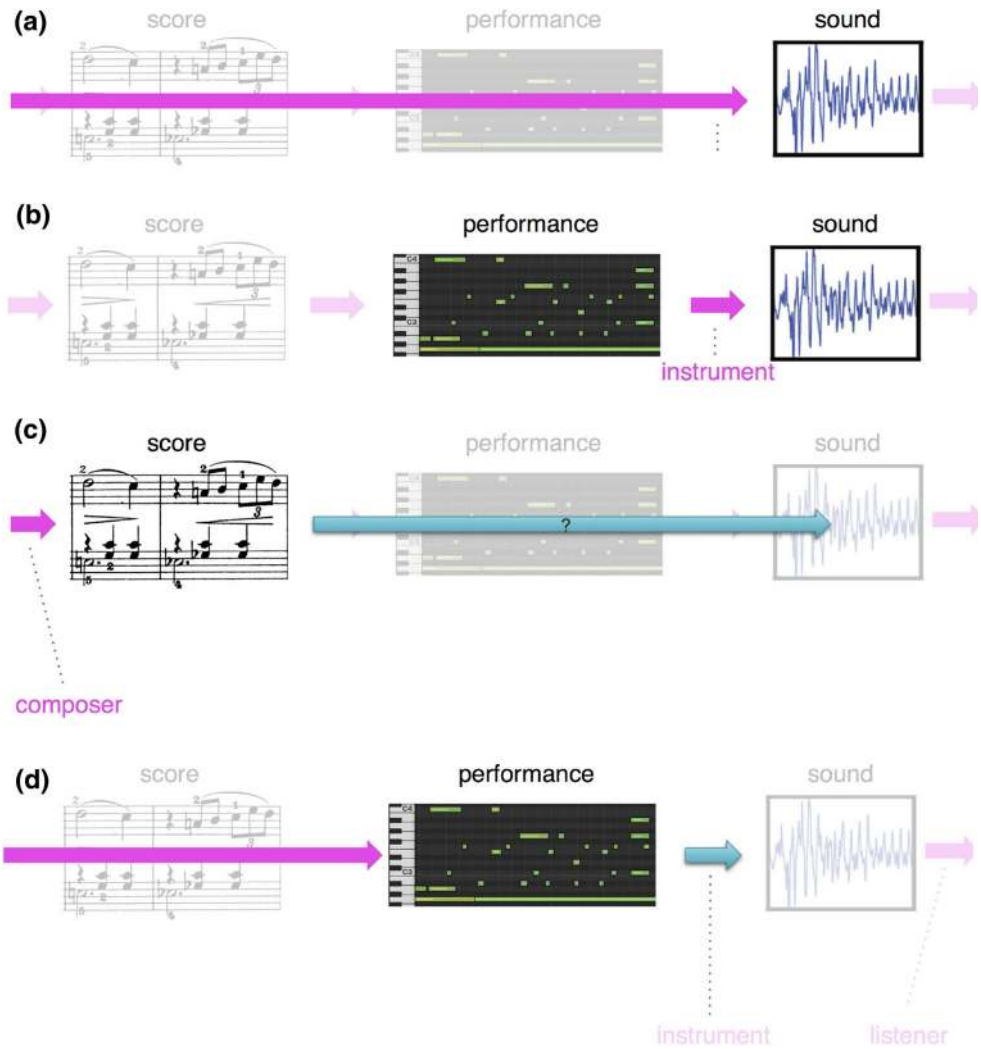

Fig. 4 Here are four different entry points into the generative process. The magenta arrows represent machine-learned generators. **a** One extreme, exemplified by WaveNet [42], is to jump directly into the generation of the audio, as shown on the top. **b** The next diagram represents learning the instrument synthesizer model (e.g., NSynth [9]). **c** The third diagram represents generating scores, i.e., learning to compose. In that case, some unspecified process is still needed in order to convert the score into audio, and therein lies one of the problems with score-based generation at the moment. **d** Finally, the bottom diagram represents bypassing the generation of scores, and directly generating performances, as we propose here. In this case, a synthetic instrument is needed in order to convert the performance into audio. For an instrument such as piano, doing this quite well is feasible

music generation given the current state of the art. Furthermore, it creates output that can be listened to without requiring additional steps beyond audio synthesis as provided by a piano sample library.

While the primary evidence for this will be found simply by listening to the results, we mention two related discussion points about the state of the art:

- *Music with very long-term, fully coherent structure is still elusive.* In "real" compositions, long-term structure spans the order of many minutes and is coherent on many levels. There is no current system that is able to learn such structure effectively. That is, if $\Delta t = 8$ ms, then even for just 2 minutes, $P(e_{i+15000}|e_i)$ should be different from $P(e_{i+15000})$. There is no current system that effectively achieves anywhere near this for symbolic MIDI representation.
- *Metrics for evaluating generated music are very limited.* Theis and others [39, 41] have given clear arguments about the limitations of metrics for evaluating the quality of generative models in the case of visual models, and their explanations extend naturally to the case of musical and audio models. In particular, they point out that ultimately, "models need to be evaluated directly with respect to the application(s) they were intended for." In the case of the generative music models that we are considering, this involves humans listening.

Taken together, what this means is that systems that generate musical scores face a significant evaluation dilemma. Since by definition any listening-based evaluation must operate in the audio space, either (a) the scores must be rendered directly and will lack expression entirely, or (b) a human or other system must perform the scores, in which case the quality of the generated score is hard to disentangle from the quality of the performance.[4] Furthermore, the lack of long-term structure compounds the difficulty of evaluation, because one of the primary qualities of a good

[4] For example, listening to the direct score and performance clips given above, it should be clear that other than perhaps very experienced musicians, it would be extremely difficult for a listener

score is precisely in its long-term structure. This implicitly bounds the potential significance of evaluating a short and context-free compositional fragment.

With these considerations in mind, we generate directly in the domain of musical performance. A side benefit of this is that informal evaluation becomes more potentially meaningful: musicians and non-musicians alike can listen to clips of generated performances while (1) not being put off by the lack of expressiveness and (2) not needing to disentangle the different elements that contributed to what they hear, since both the notes and how they are all played were all generated by the system.[5] We also note that our approach is consistent with many of the points and arguments recently made by Widmer [45]. As one example, where Widmer points out that "details of the performance contribute much to the character of the music, and how it affects listeners," a premise of our work is that it is valuable to incorporate performance into the generative musical model even at the compositional level.

5 Data

If we wish to predict expressive performance, we need to have the appropriate data. We use the International Piano-e-Competition dataset [20], which contains MIDI captures of roughly 1400 performances by skilled pianists. The pianists were playing a Disklavier, which is a real piano that also has internal sensors that record MIDI events corresponding to the performer's actions. The critical importance of good data is well known for machine learning in general, but here we note some particular aspects of this dataset that made it well suited for our task.

5.1 Homogeneous

The dataset was sufficiently homogeneous in a set of important ways. It might be easy to underestimate the importance of any of the following criteria, and so we list them all explicitly here with some discussion:

First, it was all classical music This helps the coherence of the output.

Second, it was all solo instrumental music If one includes data that is for two or more instruments, then it no longer makes sense to train a generative model that is expected to generate for a solo instrument; there will be many (if not most) passages where what one instrument is doing is entirely dependent on what the other instrument is doing. The text analogy would be hoping for a system to learn to write novels by training it on only one character's dialogue from movies and plays. There will occasionally be self-sufficient monologues, but generally speaking, well-written dialogue has already been distilled by the playwright and makes more sense when voices are not removed from it.

Third, that solo instrument was consistently piano Classical composers generally write in a way that is very specific to whichever instrument they are writing for. Each instrument has its own natural characteristics, and classical music scores (i.e., that which is captured in the MIDI representation) are very closely related to the timbre of that instrument (i.e., how those notes will be "rendered"). One exception to this is that Bach's music tends to sound quite good on any instrument; for example, it is OK to train a piano system on Bach vocal chorales.

Fourth, the piano performances were all done by humans The system did not have to contend with learning from a dataset where some of the examples were synthesized, some were "hand-synthesized" to appear like human performances, etc. Each of those classes has its own patterns of micro-timing and dynamics, and each may be well suited for a variety of music-related tasks, but if the goal is to have a system trained on performances, it is helpful that the training data are in fact performances and not proxies of performances.

Finally, all of those humans were experts If we wish the system to learn about human performance, that human performance must match the listener's concept of what "human performance" sounds like, which is usually performances by experts. The casual evaluator might find themselves slightly underwhelmed were they to listen to a system that has learned to play like a beginning pianist, even if the system has done so with remarkable fidelity to the dynamic and velocity patterns that occur in that situation.

5.2 Realizable

The fact that the solo instrument was piano had additional advantages. Synthesizing audio from MIDI can be a challenging problem for some instruments. For example, having velocities and note durations and timing of violin music would not immediately lead to good-sounding violin audio at all. The problems are even more evident if one considers synthesizing vocals from MIDI. Here, that the piano is a percussive instrument buys us an important benefit: synthesizing piano music from MIDI can sound quite realistic (compared to synthesizing instruments that allow

Footnote 4 continued
to hear the audio of the MIDI Score and intuitively understand that that same passage could sound as it does in the MIDI Performance.

[5] We emphasize that these observations do not apply to the development of tools for composers, where score fragment generation might be appropriate. Also, we reiterate that this discussion is made in relation to the current state of the art.

Fig. 5 Example of the representation used for PerformanceRNN. The progression illustrates how a MIDI sequence (e.g., shown as a MIDI roll consisting of a long note followed by a shorter note) is converted into a sequence of commands (on the right-hand side) in our event vocabulary. Note that an arbitrary number of events can in principle occur between two time shifts

continuous timbral control). Thus, when we generate data, we can properly realize it in audio space and therefore have a good point of comparison. Conversely, capturing the MIDI data of piano playing provides us with a sufficiently rich set of parameters that we can later learn enough in order to be able to render audio. Note that with violin or voice, for example, we would need to capture many more parameters than those typically available in the MIDI protocol in order to get a sufficiently meaningful set of parameters for expressive performance.

6 RNN model

We modeled the performance data with an LSTM-based Recurrent Neural Network. The model consisted of three layers of 512 cells each, although the network did not seem particularly sensitive to this hyperparameter. We used a temporally non-uniform representation of the data, as described next.

6.1 Representation: time shift

A MIDI excerpt is represented as a sequence of events from the following vocabulary of 413 different events:

- **128 NOTE-ON** events: one for each of the 128 MIDI pitches. Each one starts a new note.
- **128 NOTE-OFF** events: one for each of the 128 MIDI pitches. Each one releases a note.
- **125 TIME-SHIFT** events: each one moves the time step forward by increments of 8 ms up to 1 s.
- **32 VELOCITY** events: each one changes the velocity applied to all subsequent notes (until the next velocity event).

The neural network operates on a one-hot encoding over this event vocabulary. Thus, at each step, the input to the RNN is a single one-hot 413-dimensional vector. For the piano-e-competition dataset, a 15-s clip typically contains 600 such one-hot vectors, although this varies considerably (and roughly linearly with the number of notes in the clip).

While the minimal time step is a fixed absolute size (8 ms), the model can skip forward in time to the next note event. Thus, any time steps that contain rests or simply hold existing notes can be skipped with a single event. The largest possible single time shift in our case is 1 s but time shifts can be applied consecutively to allow effectively longer shifts. The combination of fine quantization and time-shift events helps maintain expressiveness in note timings while greatly reducing sequence length compared to an uncompressed representation.

This fine quantization is able to maintain expressiveness in note timings while not being as sparse as a grid-based representation. This sequence representation uses more events in sections with higher note density, which matches our intuition.

Figure 5 shows an example of a small excerpt of MIDI performance data converted to our representation. Figure 6 shows a diagram of the basic RNN architecture.

6.2 Generation

The model generates output auto-regressively: it samples stochastically from the softmax output distribution and uses beam search; the selected event is then fed back as an

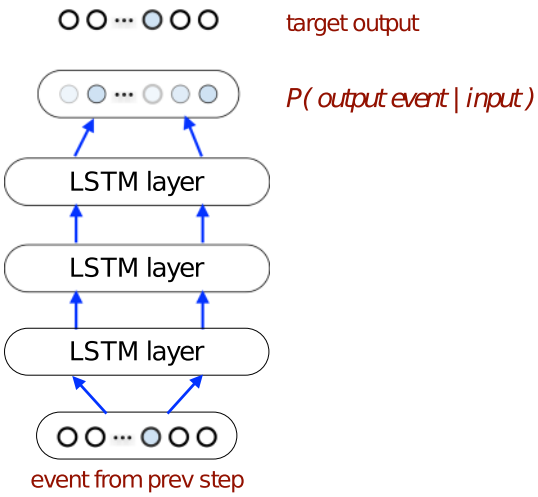

Fig. 6 Basic RNN architecture consists of three hidden layers of LSTMs, each layer with 512 cells. The input is a 413-dimensional one-hot vector, as is the target, and the model outputs a categorical distribution over the same dimensionality as well

input into the system. Teacher forcing is used during training. It is worth noting that, in this form, the model cannot, for example, follow a given score, but it can be given an initial priming sequence at the beginning of generation if one wishes, though we have typically started with an empty sequence. The output generation is somewhat analogous to musical free improvisation, where the model's next choice depends on the choices it has made up to that point.

6.3 Training and data augmentation

We train the models by first separating the data into 30-s clips, from which we then select shorter segments. We train using stochastic gradient descent with a mini-batch size of 64 and a learning rate of 0.001 and teacher forcing.

6.3.1 Augmentation

We augment the data in two different ways, for different runs:

Less augmentation:

- Each example is transposed up and down all intervals up to a major third, resulting in eight new examples plus the original.
- Each example is stretched in time uniformly by $\pm 2.5\%$ and $\pm 5\%$, resulting in four new examples plus the original.

More augmentation:

- Each example is transposed up and down all intervals up to five or six semitones to span a full octave, resulting in 11 new examples plus the original.
- Each example is stretched in time uniformly by up to $\pm 10\%$.

6.3.2 Quantization

In Sect. 3, we describe some forms of quantization that can be harmful to perceived musical quality. Our models also operate on quantized data; however, unlike much prior work, we aim for quantization levels that are below noticeable perceptual thresholds.

Timing — Friberg and Sundberg [11] found that the just noticeable difference (JND) when temporally displacing a single tone in a sequence was generally no finer than roughly 10 ms. Other studies have found that the JND for change in tempo is no finer than roughly 5%. We note that for a tempo of 120 bpm, each beat lasts for 500 ms, and therefore this corresponds to a change of roughly 25 ms. Given that at that tempo beats will frequently still be subdivided into two or triplets, that would correspond to a change of roughly 8 ms per subdivided unit. We therefore assume that using a sampling rate of 125 Hz (i.e., 1000/8) should generally be below the typical perceptual threshold.

Dynamics — Working with piano music, we have found that 32 different "steps" of velocity are sufficient. Note that there are about eight levels of common dynamic marking in classical music (from *ppp* to *fff*), so it may well be the case that we could do with fewer than 32 bins, but our objective was not to find the lower bound here.

6.3.3 Predicting pedal

In the RNN model, we experimented with predicting sustain pedal. We applied PEDAL_ON by directly extending the lengths of the notes: for any notes on during or after a PEDAL_ON signal, we delay their corresponding NOTE_OFF events until the next PEDAL_OFF signal. This made it a lot easier for the system to accurately predict a whole set of NOTE_OFF events all at once, as well as to predict the corresponding delay preceding this. Doing so may have also freed up resources to focus on better prediction of other events as well. Finally, as one might expect, including pedal made a significant subjective improvement in the quality of the resulting output.

7 Results

We begin with the most important indicator of performance: generated audio examples.

7.1 Examples

In these examples, our systems generated all MIDI events: timing and duration of notes as well as note velocities. We then used freely available piano samples to synthesize audio from the resulting MIDI file.

A small set of examples are available at https://clyp.it/user/3mdslat4. We strongly encourage the reader to listen. These examples are representative of the general output of the model. We comment on a few samples in particular, to give a sense of the kind of musical structure that we observe:

- RNN Sample 4: This starts off with a slower segment that goes through a very natural harmonic progression in G minor, pauses on the dominant chord, and then breaks into a faster section that starts with a G major chord, then passes through major chords related to G minor (Bb, etc). Harmonically, this shows structural coherence even while the tempo and feel shift. At around 12s, the "left hand" uses dynamics to bring out an inner voice in a very natural and appropriate way.
- RNN Sample 7: This excerpt begins very reminiscent of a Schubert Impromptu, although it is sufficiently different that it has clearly not memorized it. There is a small rubato at the very beginning of the phrase, especially on the first note, which is musically appropriate. The swells in the phrasing make musical sense, as do the slight pauses right before some of the isolated notes in the left hand (e.g., the E at 0:10s, the F♯ at around 12.5 s).
- RNN Sample 2: This excerpt begins in a classical style (e.g., Haydn or Mozart). Interestingly, the same way that one note (an F) is repeated in the right hand in the first few seconds, after a pause, the next phrase begins, and then at around 8 s, the left hand mirrors that articulation pattern with a set of descending repeated notes (A♭, G, F).

7.2 Log-likelihood

We begin by noting that objective evaluation of these kinds of generative is fundamentally very difficult, and measures such as log-likelihood can be quite misleading [39]. Nevertheless, we provide comparisons here over several different hyperparameter configurations for the RNN.

Table 1 Log-loss of RNN model variants trained on the Piano-e-competition performance dataset and evaluated on a held-out subset

Model	Log-loss	Description
RNN	.765	Baseline RNN trained on 15-s clips
RNN-NV	.619	Baseline without velocity
RNN-SUS	.663	Baseline with pedaled notes extended
RNN-AUG+	.755	Baseline with more data augmentation
RNN-AUG-	.784	Baseline with less data augmentation
RNN-30s	.750	Baseline trained on 30-s clips
RNN-SUS-30s	.664	Baseline + pedal + 30-s clips

Table 1 contains the per-time-step log-loss of several RNN model variants. The baseline model is trained on 15-s performance clips, ignoring sustain pedal and with the two forms of data augmentation described in Sect. 5.3.

Note that while RNN-NV has the best log-loss, this variant is inherently easier as the model does not need to predict velocities. In the RNN-SUS variant, sustain pedal is used to extend note durations until the pedal is lifted; this aids prediction as discussed in Sect. 5.3.3.

7.3 Feature-based Analysis

One way to evaluate the generated samples is to compare hand-selected statistical features with the same features computed over the real data. For example, Grachten and Widmer [14] compute the relationship between MIDI pitch number and MIDI velocity and show that for two different datasets with different performers, the best fit first three orders of polynomials are remarkably similar. Following this, we compute the same statistics over the Yamaha dataset and over our generated samples. In Fig. 7, we plot these results analogously to the plots given by Grachten and Widmer. Both of our plots are indeed extremely similar to theirs.

In another example of a carefully selected musical feature, Goebl [12] observes that when two or more notes are played nearly simultaneously in piano performance, the note corresponding to the melody is generally played slightly in advance of the other notes. While Goebl had repeated performances of the same piece, and therefore the melody/accompaniment split could be computed a single time and applied to many performances, it was impractical to measure this directly in our case, because each sample is an entirely different set of notes, so that would involve determining the melody notes by hand over thousands of notes of generated examples. However, if we crudely assume that, most of the time, the melody occurs in the upper register, then if we look for nearly simultaneous groups of notes (i.e., occurring within a 30 ms window)

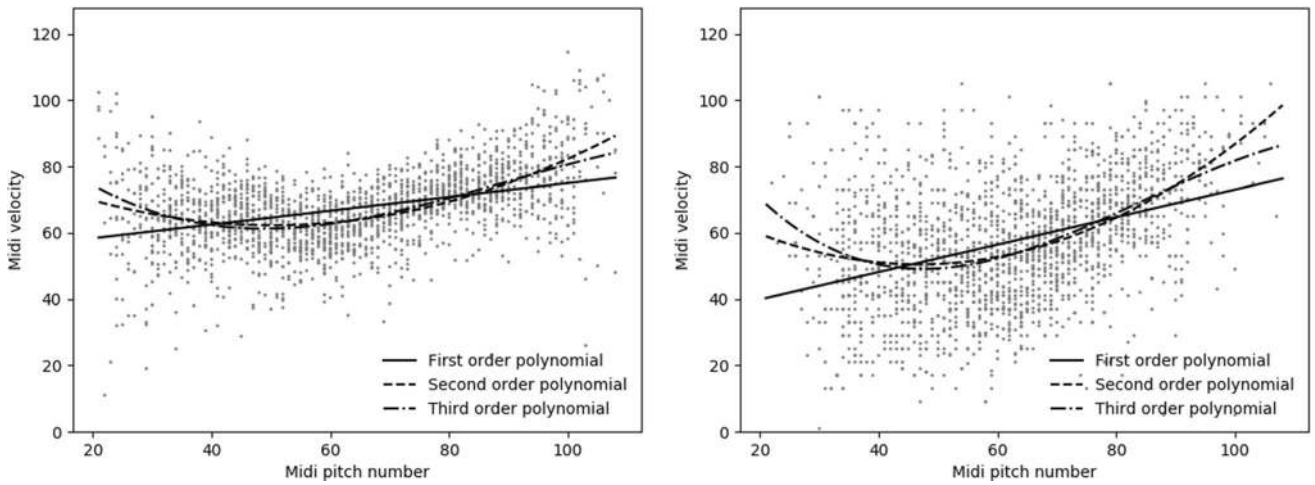

Fig. 7 Left: Pitch–velocity relationship for the real dataset. Right: Pitch–velocity relationship for a set of generated examples. Each data point is a pair *(pitch, average velocity)* for one MIDI file/excerpt, where mean velocity is taken over the nonzero velocities. For clarity, only approximately $\frac{1}{10}$ of the real data is shown in the scatterplot, but all of it is used to calculate the interpolation. Roughly 1000 data points were computed analogously from a generated set of samples from the model

that span both above and below C4, then we would expect to see more drops in pitch within such windows than rises. When we measured this, then, both for the real data and for the generated samples, we found that this was the case. That is, when groups of notes were played nearly simultaneously, and some of those notes were above C4 and others were below C4, then it was indeed the case that the upper notes were usually played before the lower ones.

Thus, we see that both our real and generated samples display characteristics that match other datasets for which the same, musically informed features have been computed. Yet, we do not know how these samples sound. We thus solicited feedback from professional musicians.

7.4 Informal Feedback From Professional Composers and Musicians

We gave a small set of clips to professional musicians and composers for informal comments. We were not trying to do a Turing test, so we mentioned that the clips were generated by an automated system, and simply asked for any initial reactions/comments. Here is a small, representative subset of the comments we received (musical background in bold, some particularly interesting excerpts are italicized for later discussion):[6]

TV/Film composer:

Fantastic!!!! How many hours of learning [...] here? This [...] absolutely blows the stuff I've heard online out of the solar system. The melodic sense is still foggy, in my view, but it's staggering that it makes nice pauses with some arcing chord progressions quite nicely. I think that it's not far from actually coming up with a worthwhile melody. [...] How does it know what "inspirational emotion" to draw from? or is it mostly doing things "in the likeness of"?

Fascinating!!

Composer and Professional Musician

In terms of performance I'm quite impressed with the results. It sounds more expressive than any playback feature I've worked with when using composition software.

In terms of composition, I think there is more room for improvement. The main issue is lack of consistency in rhythmic structure and genre or style. For example, Sample 1 starts with a phrase in Mozarts style, then continues with a phrase in Waltons style perhaps, which then turns into Scott Joplin... Sample 2 uses the harmonic language of a late Mahler symphony, along with the rhythmic language of a free jazz improvisation (I couldn't make a time signature out of this clip). Sample 3 starts with a phrase that could be the opening of a Romantic composition, and then takes off with a rhythmic structure that resembles a Bach composition, while keeping the Romantic harmonic language. Sample 4 is the most consistent of all. It sounds like a composition in the style of one

[6] It is worth considering some of the qualitative remarks given below with respect to characteristics such as those listed in Table 1 in the work by Friberg et al. [10].

of the Romantic piano composers (such as Liszt perhaps) and remains in that style throughout the clip.

Music Professor:

[. . .] *I'd guess human because of a couple of "errors" in there, but maybe the AI has learned to throw some in! [. . .]*

Pianist, TV & Film Composer:

Sample 1: resembles music in the style of Robert Schumann's Kinderszenen or some early romantic salon music. I'm fond of the rest after the little initial chord and melody structure. The tempo slows down slightly before the rest which sounds really lively and realistic—almost a little rubato. Then, the distinct hard attack. Nice sense of dynamics. Also nice ritardando at the end of the snippet. Not liking the somewhat messy run but this almost seems as if someone had to study a little bit harder to get it right—*it seems wrong in a human way.*

Sample 2: reminds me of some kind of Chopin waltz, rhythm is somewhat unclear. The seemingly wrong harmony at the beginning seems to be a misinterpretation of grace notes. The trill is astonishing and feels light and airy.

Sample 3: Could be some piece by Franz Schubert. Nice loosely feeling opening structure which shifts convincingly into fierce sequence with quite static velocity. This really reminds me of Schubert because Johann Sebastian Bach shines through the harmonic structure as it would have with Schubert. Interesting effort to change the dynamic focus from the right to the left hand and back again.

This is really interesting!

Piano Teacher:

Sample 1: Sounded almost Bach-like for about the first bar, then turned somewhat rag-timey for the rest

Sample 2: Here we have a *very drunken Chopin*, messing around a bit with psychedelics

Does that help at all? Also, what do you mean by a regular piano sample library? *Did you play these clips as composed by the AI system?*

Overall, we note that the comments were quite consistent in terms of perceiving a human quality to the performance. Indeed, even though we made an effort to explain that all aspects of the MIDI file were generated by the computer, some people still wanted to double-check whether in fact these were human performances.

While acknowledging the human quality of the performances, many of the musicians also questioned the strength of the long-term compositional structure. Indeed, creating music with long-term structure (e.g., more than several seconds of structure) is still a very challenging problem.

Many musicians identified the "style" as the mix of classical composers of which the data indeed consisted.

8 Conclusion

We have considered various approaches to the question of generating music and propose that it is currently effective to generate in the space of MIDI performances. We describe the characteristics of an effective dataset for doing so and demonstrate a system that achieves this quite effectively.

Our resulting system creates audio that sounds, to our ears, like a pianist who knows very well *how* to play, but has not yet figured out exactly *what* they want to play, nor is quite able remember what they just *played*. Professional composers and musicians have provided feedback that is consistent with the notion that the system generates music which, on one hand, does not yet demonstrate long-term structure, but where the local structure, for example phrasing, dynamics, is very strong. Indeed, even though we did not frame the question as a Turing test, a number of the musicians assumed that (or asked whether) the samples were performed by a human.

Acknowledgements We gratefully acknowledge the members of the Magenta team at Google Research for numerous discussions. We thank the reviewers for helpful comments. We thank Nick Barreyre for assistance in creating some of the figures.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Boehmer K (1967) Zur Theorie der offenen Form in der neuen Musik. Edition Tonos, Darmstadt
2. Boulanger-Lewandowski N, Bengio Y, Vincent P (2012) Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In:

Proceedings of the 29th international conference on machine learning
3. Bresin R (1998) Artificial neural networks based models for automatic performance of musical scores. J New Music Res 27:239–270
4. Bretan M, Oore S, Engel J, Eck D, Heck L (2017) Deep music: towards musical dialogue. In: Proceedings of AAAI
5. Briot J-P, Hadjeres G, Pachet F (2017) Deep learning techniques for music generation—a survey. CoRR https://arxiv.org/abs/1709.01620v1
6. Cancino Chacn CE, Grachten M (2016) The basis mixer: a computational romantic pianist. In: Late-breaking demo session of the 17th international society for music information retrieval conference, New York, NY
7. Chopin Frédéric (1830) Piano Concerto No. 1 in E minor, Op. 11
8. Eck D, Schmidhuber J (2002) Finding temporal structure in music: blues improvisation with lstm recurrent networks. In: Proceedings of the 12th IEEE workshop on neural networks for signal processing, pp 747–756
9. Engel J, Resnick Cinjon, Roberts Adam, Dieleman Sander, Norouzi Mohammad, Eck Douglas, Simonyan Karen (2017) Neural audio synthesis of musical notes with wavenet autoencoders. In Proceedings of the 34th international conference on machine learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, pp 1068–1077
10. Friberg A, Bresin R, Sundberg J (2006) Overview of the kth rule system for musical performance. Adv Cognit Psychol 2(2–3):145–161
11. Friberg A, Sundberg J (1992) Perception of just noticeable time displacement of a tone presented in a metrical sequence at different tempos. In Kungl. tekniska hoegskolan, Speech Transmission Laboratory (ed) Quarterly progress and status report, vol 33. Royal Institute of Technology, Stockholm, pp 49–55
12. Goebl W (2001) Melody lead in piano performance: Expressive device or artifact? J Acoust Soc Am 110(1):563–572
13. Grachten M, Krebs F (2014) An assessment of learned score features for modeling expressive dynamics in music. IEEE Trans Multimed 16(5):1211–1218
14. Grachten M, Widmer G (2012) Linear basis models for prediction and analysis of musical expression. J New Music Res 41(4):311–322
15. Griffith N, Todd PM (eds) (1999) Musical networks: parallel distributed perception and performance. MIT Press, Cambridge
16. Hedges S (1978) Dice music in the eighteenth century. Music Lett 59:180–187
17. Herremans D, Chew E (2017) Morpheus: automatic music generation with recurrent pattern constraints and tension. IEEE Trans Affect Comput. https://doi.org/10.1109/TAFFC.2017.2737984
18. Herremans D, Chuan C-H, Chew E (2017) A functional taxonomy of music generation systems. ACM Comput Surv 50(5):69:1–69:30
19. Hild H, Feulner J, Menzel W (1991) Harmonet: a neural net for harmonizing chorales in the style of j.s. bach. In Proceedings of the 4th international conference on neural information processing systems, NIPS'91, pp 267–274, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc
20. International Piano-e-Competition http://www.piano-e-competition.com/. Accessed 15 Feb 2018
21. Kirke A, Miranda ER (2013) An overview of computer systems for expressive music performance. In: Kirke A, Miranda E (eds) Guide to computing for expressive music performance. Springer, London
22. Lattner S, Grachten M, Widmer G (2018) Imposing higher-level structure in polyphonic music generation using convolutional restricted Boltzmann machines and constraints. J Creat Music Syst 3(1)
23. Lindblom B, Sundberg J (1969) Perception of just noticeable time displacement of a tone presented in a metrical sequence at different tempos. In: Quarterly progress and status report (ed) Kungl. tekniska hoegskolan, Speech Transmission Laboratory, vol 10. Royal Institute of Technology, Stockholm, pp 53–86
24. Longuet-Higgins HC (1976) The perception of melodies. Nature 263:646
25. Longuet-Higgins HC (1978) The perception of music. Interdiscip Sci Rev 3:148–156
26. Longuet-Higgins HC, Steedman MJ (1971) On interpreting bach. Mach Intell 6(221):241
27. Malik I, Ek CH (2017) Neural translation of musical style. CoRR https://arxiv.org/abs/1708.03535arXiv:1708.03535
28. McDonald K (2017) Neural nets for generating music. Medium https://medium.com/artists-and-machine-intelligence/neural-nets-for-generating-music-f46dffac21c0). Accessed 15 Nov 2017
29. Moulieras S, Pachet F (2016) Maximum entropy models for generation of expressive music. https://arxiv.org/abs/1610.03606
30. Mozer MC (1994) Neural network music composition by prediction. Connect Sci 6(2):247–280
31. Nierhaus G (2009) Algorithmic composition: paradigms of automated music generation. Springer, Vienna
32. Okumura K, Sako S, Kitamura T (2014) Laminae: a stochastic modeling-based autonomous performance rendering system that elucidates performer characteristics. In: International Computer Music Conference (ICMC), Athens, Greece
33. Oore S (2017) Recording of Chopin Piano Concerto No. 1 in E Minor, Op. 11, 1st movement, (unreleased)
34. Pachet F (2003) The continuator: musical interaction with style. J New Music Res 32(3):333–341
35. Roberts A, Engel J, Hawthorne C, Simon I, Waite E, Oore S, Jaques N, Resnick C, Eck D (2016) Interactive musical improvisation with magenta. In: Demonstration track in neural information processing systems (NIPS)
36. Simon HA, Sumner RK (1968) Pattern in music. In: Kleinmuntz B (ed) Formal representation of human judgement. Wiley, New York
37. Sturm B, Santos JF, Ben-Tal O, Korshunova I (2016) Music transcription modelling and composition using deep learning. In: Proceedings of 1st conference on computer simulation of musical creativity, Huddersfield, UK
38. Taniguchi Y, Makimoto S, Teramura K, Okuma H, Maeda S (2008) Gaussian process regression for rendering music performance. In: Proceedings of the 10th international conference on music perception and cognition (ICMPC 10), Japan
39. Theis L, van den Oord A, Bethge M (2016) A note on the evaluation of generative models. In: ICLR
40. Todd PM, Loy G (eds) (1991) Music and connectionism. MIT Press, Cambridge
41. van den Oord A, Dambre J (2015) Locally-connected transformations for deep gmms. In: Proceedings of the 32nd international conference on machine learning, Lille, France
42. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior AW, Kavukcuoglu K (2016) Wavenet: a generative model for raw audio. CoRR, https://arxiv.org/abs/abs/1609.03499
43. van Herwaarden S, Grachten M, de Haas WB (2014) Predicting expressive dynamics using neural networks. In Proceedings of the 15th conference of the international society for music information retrieval, pp 47–52
44. Widmer G, Goebl W (2004) Computational models of expressive music performance: the state of the art. J New Music Res 33(3):203–216
45. Widmer G (2017) Getting closer to the essence of music: the Con Espressione manifesto. ACM Trans Intell Syst Technol (TIST) 8(2):19:1–19:13