# **Piano Genie**

Chris Donahue\* UC San Diego **Ian Simon** Google AI Sander Dieleman DeepMind

## Abstract

We present Piano Genie, an intelligent controller which augments musical creativity by allowing non-musicians to improvise on the piano. With Piano Genie, a user performs on a simple interface with eight buttons, and their performance is decoded into the space of plausible piano music in real time. To learn a suitable mapping procedure for this problem, we train recurrent neural network autoencoders with discrete bottlenecks: an encoder learns an appropriate sequence of buttons corresponding to a piano piece, and a decoder learns to map this sequence back to the original piece. At improvisation time, we substitute a user's input for the encoder output, and play the decoder's prediction each time the user presses a button. To improve the interpretability of Piano Genie's performance mechanics, we impose musically-salient constraints over the encoder's outputs.

## 1 Introduction

While most people have an innate sense of and appreciation for music, comparatively few are able to participate meaningfully in its creation. A non-musician could endeavor to achieve proficiency on an instrument, but the time and financial requirements may be prohibitive. Alternatively, a non-musician could operate a system which automatically generates complete songs at the push of a button, but this would remove any sense of ownership over the result. We seek to sidestep these obstacles by designing an intelligent interface which takes high-level specifications provided by a human and maps them to plausible musical performances.

The practice of "air guitar" offers hope that non-musicians can provide such specifications [7]; performers strum fictitious strings with rhythmical coherence and even move their hands up and down an imaginary fretboard in correspondence with *melodic contours*, i.e. rising and falling movement in the melody. This suggests a pair of attributes which may function as a communication protocol between non-musicians and generative music systems: 1) rhythm, and 2) melodic contours. In addition to air guitar, games such as *Guitar Hero* [8] also make use of these. However, both experiences only allow for the imitation of experts and provide no mechanism for music *creation*.

In this work, we present *Piano Genie*, an intelligent controller allowing non-musicians to improvise on the piano while retaining ownership over the result. In our web demo, a participant improvises on eight buttons, and their input is translated into a piano performance by a neural network running in the browser in real-time.<sup>2</sup> Piano Genie has similar performance mechanics to those of a real piano: pressing a button will trigger a note that sounds until the button is released. Multiple buttons can be pressed simultaneously to achieve polyphony. The mapping between buttons and pitch is non-deterministic, but the performer can control the overall form by pressing higher buttons to play higher notes and lower buttons to player lower notes.

Because we lack examples of people performing on 8-button "pianos", we adopt an unsupervised strategy for learning the mappings. Specifically, we use the *autoencoder* setup, where an *encoder* learns to map 88-key piano sequences to 8-button sequences, and a *decoder* learns to map the button

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

<sup>\*</sup>Research performed while interning at Google AI.

<sup>&</sup>lt;sup>2</sup>Web Demo: goo.gl/magenta/pianogenie, Videos: goo.gl/Bex4xn, Code: goo.gl/eKvSVP



Figure 1: Piano Genie consists of a discrete sequential autoencoder. A bidirectional RNN encodes monophonic piano sequences (88-dimensional) into smaller discrete latent variables (shown here as 4-dimensional). The unidirectional decoder is trained to map the latents back to piano sequences. During inference, the encoder's discrete embeddings are replaced by a human improvising on buttons.

sequences back to piano music (Figure 1). At improvisation time, we replace the encoder's output with a user's button presses, evaluating the decoder in real time.

**Related work** There is extensive prior work [11, 3, 5, 6] on *supervised* learning of mappings from different control modalities to musical gestures. These approaches require users to provide a training set of control gestures and associated labels. There has been less work on *unsupervised* approaches, where gestures are automatically extracted from arbitrary performances. Scurto and Fiebrink [13] describe an approach to a "grab-and-play" paradigm, where gestures are extracted from a performance on an arbitrary control surface, and mapped to inputs for another. Our approach differs in that the controller is fixed and integrated into our training methodology, and we require no example performances on the controller. Like Piano Genie, some systems use user-provided contours to *compose* music [12, 10], though these systems do not allow for real-time improvisation.

## 2 Piano Genie

We wish to learn a mapping from sequences  $y \in [0, 8)^n$ , i.e. amateur performances of n presses on eight buttons, to sequences  $x \in [0, 88)^n$ , i.e. professional performances on an 88-key piano. To preserve a one-to-one mapping between buttons pressed and notes played, we assume that both y and x are monophonic sequences. Given that we lack examples of y, we propose using the autoencoder framework on examples x. Specifically, we learn a deterministic mapping  $enc(x) : [0, 88)^n \mapsto [0, 8)^n$ , and a stochastic inverse mapping  $P_{dec}(x|enc(x))$ .

We use LSTM recurrent neural networks (RNNs) [9] for both the encoder and the decoder. For each input piano note, the encoder outputs a real-valued scalar, forming a sequence  $\operatorname{enc}_s(x) \in \mathbb{R}^n$ . To discretize this into  $\operatorname{enc}(x)$  we quantize it to k = 8 buckets equally spaced between -1 and 1, and use the straight-through estimator [2] to bypass this non-differentiable operation in the backwards pass. More details of this model can be found in the full version of our paper [4].

We train this system to minimize the negative log likelihood of the decoder distribution  $P_{\text{dec}}(\boldsymbol{x}|\text{enc}(\boldsymbol{x}))$ . To discourage the encoder from producing values outside of [-1,1], we additionally minimize a margin loss  $L_{\text{margin}} = \sum max(|\text{enc}_s(\boldsymbol{x})| - 1, 0)^2$ . These two terms may result in a learned representation that is not aligned with human intuition; i.e. a lower button will not necessarily correspond to a lower note. To suggest this, we also regularize the representation by  $L_{\text{contour}} = \sum max(1 - \Delta x \Delta \text{enc}_s(\boldsymbol{x}), 0)^2$ . Intuively speaking, this contour regularization term penalizes the model when the "shape" of the button sequence does not match that of the music.

We train Piano Genie on the the Piano-e-Competition data [1], which contains around 1400 performances by skilled pianists. We flatten each polyphonic performance into a single sequence of notes ordered by start time, breaking ties by listing the notes of a chord in ascending pitch order. To keep the latency low at inference time, we use relatively small RNNs consisting of two layers with 128 units each. Because the encodings learned by our contour-regularized model are reflective of human intuition, we can substitute a human's decisions at improvisation time (see goo.gl/Bex4xn).

#### Acknowledgements

We would like to thank Adam Roberts, Anna Huang, Ashish Vaswani, Ben Poole, Colin Raffel, Curtis Hawthorne, Doug Eck, Jesse Engel, Jon Gillick, Yingtao Tian and others at Google AI for helpful discussions and feedback throughout this work. We would also like to thank Julian McAuley, Stephen Merity, and Tejaswinee Kelkar for helpful conversations.

#### References

- [1] Piano-e-competition. http://www.piano-e-competition.com/, 2018.
- [2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv:1308.3432*, 2013.
- [3] Frédéric Bevilacqua, Rémy Müller, and Norbert Schnell. MnM: a Max/MSP mapping toolbox. In NIME, 2005.
- [4] Chris Donahue, Ian Simon, and Sander Dieleman. Piano genie. arXiv:1810.05246, 2018.
- [5] Rebecca Fiebrink, Dan Trueman, and Perry R Cook. A meta-instrument for interactive, on-thefly machine learning. In *NIME*, 2009.
- [6] Nicholas Gillian and Benjamin Knapp. A machine learning toolbox for musician computer interaction. In *NIME*, 2011.
- [7] Rolf Inge Godøy, Egil Haga, and Alexander Refsum Jensenius. Playing "air instruments": mimicry of sound-producing gestures by novices and experts. In *International Gesture Workshop*, 2005.
- [8] Harmonix. Guitar Hero. Game, 2005.
- [9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.
- [10] Tetsuro Kitahara, Sergio I Giraldo, and Rafael Ramírez. Jamsketch: a drawing-based real-time evolutionary improvisation support system. In *NIME*, 2017.
- [11] Michael Lee, Adrian Freed, and David Wessel. Neural networks for simultaneous classification and parameter estimation in musical instrument control. In *Adaptive and Learning Systems*, 1992.
- [12] Udit Roy, Tejaswinee Kelkar, and Bipin Indurkhya. Trap: An interactive system to generate valid raga phrases from sound-tracings. In *NIME*, 2014.
- [13] Hugo Scurto and Rebecca Fiebrink. Grab-and-play mapping: Creative machine learning approaches for musical inclusion and exploration. In *ICMC*, 2016.