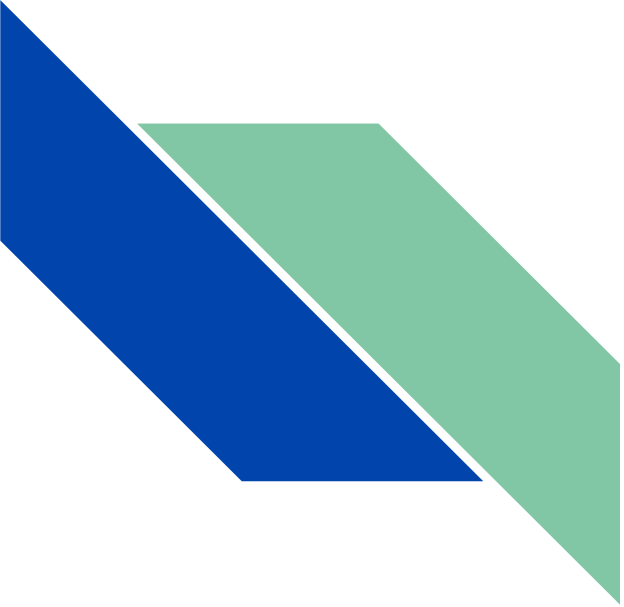


Music Generation: An Introduction

Frank Cwitkowitz and Mojtaba Heydari

Agenda

1. Task Overview
2. Fundamental Concepts
3. Common Practices
4. Recurrent Architectures
5. Adding Reinforcement Learning
6. Generative Adversarial Training
7. Variational Auto-Encoding

Music

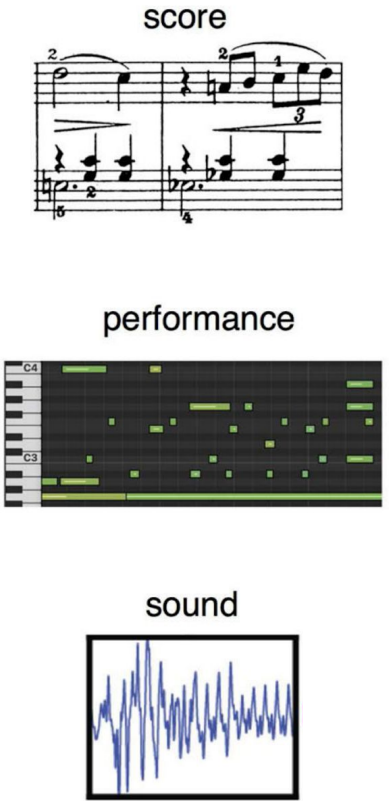
- Sound organized **across time**
- Defining attributes:
  - Melody, Rhythm, Expression,
  - Texture, Timbre, Harmony, etc.
- Generally contains **patterns and repetitions**
- Follows a theoretical framework which gives us **rules**

Music Generation

- Applications
  - Endless loop
  - Ideas
  - User control
  - Accompaniment
- Challenges
  - Creativity
  - Long-term structure
  - Music theory

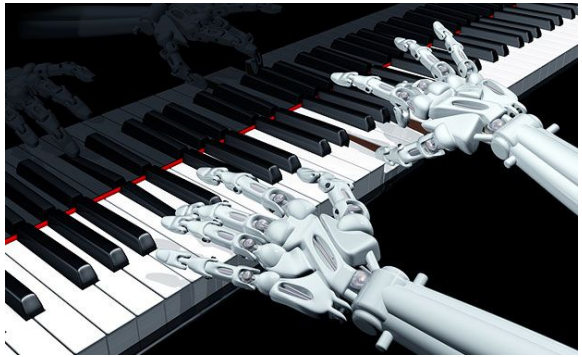
Music Modeling

- Music stored as digital audio
- Symbolic representations of music
  - Sheet music
  - Musical Instrument Digital Interface (MIDI)
  - Others (*e.g.* text)
- Reduction of music to states
- Discretization of time
- Can transpose to common key



Music Generation Approaches

- Rule-based methods
- Machine learning approaches
  - Recurrent Neural Networks (RNNs)
  - Reinforcement Learning (RL)
  - Convolutional Neural Networks (CNNs)
  - Variational AutoEncoder (VAE)
  - Generative adversarial Network (GAN)



Training Data

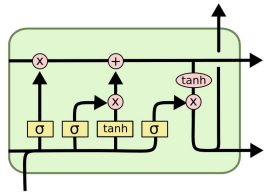
- Mainly collections of MIDI
  - Encoded automatically using digital instruments
  - Can capture expressiveness
  - Quantized to convert to piano-roll
- Other symbolic forms such as text
- Important for the sequence modeling task
  - Used to learn sampling distribution

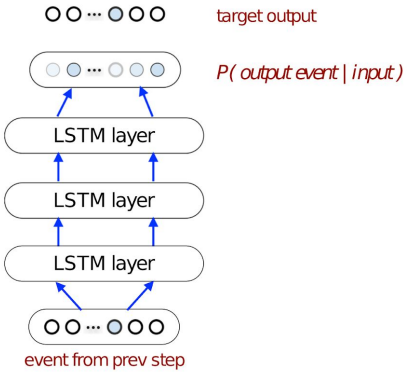
Evaluation

- Turing test (subjective)
- Musical analysis of outputs (subjective)
- Objective metrics

Recurrent Neural Networks

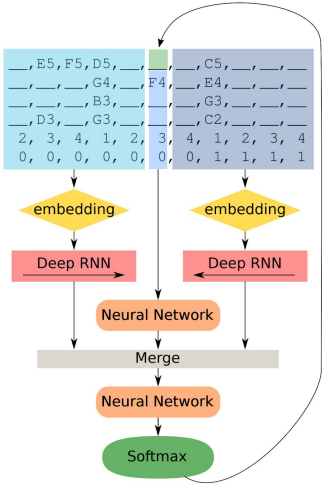
- Train on a dataset to learn note probabilities from data
- Generating new music
  - Seed network with a few initial states
  - Sample output distribution to get next state
  - Feed in output as subsequent input
- Simple RNNs are not enough
  - Gradient problems make learning long-term dependencies challenging
  - LSTMs allow for better flow of information





Recurrent Neural Networks

- **Music transcription modelling and composition using deep learning**
  - 3-layer (512 cell) LSTM network for text token modeling
- **DeepBach: a Steerable Model for Bach Chorales Generation**
  - Fix one voice and generate rest with Gibbs sampling
- **This time with feeling: learning expressive musical performance**
  - Include time shift and velocity setting in events



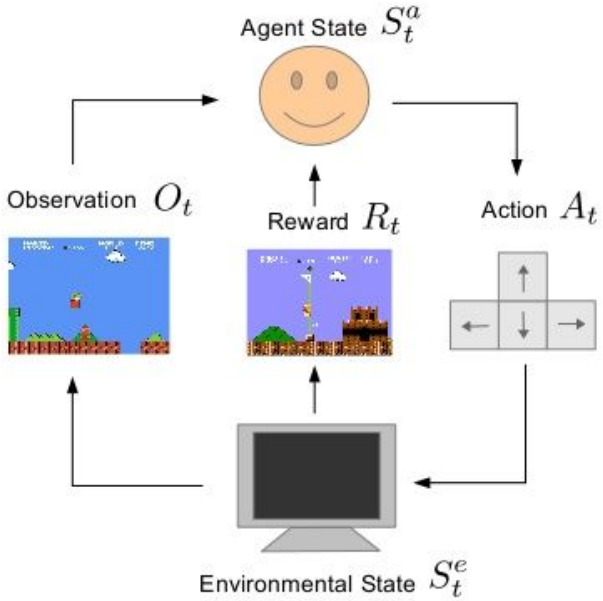
<s> M:4/4 K:Cmix |: g c (3 c c c b 2 a b | g c (3 c c c d B B a | g c (3 c c c b 2 a b |1 c' a b g f 2 b a :| |2 c' a b g f 2 e f |: g c' c' b g a b a | g c' c' b g f d f | g c' c' b g a b g | c' a b g f b a b :| <\s>

Recurrent Neural Networks

- Potential Issues
  - Overly repetitive
  - Excessive use of the same note
  - Lack of consistent global structure
  - Dataset dependent

Reinforcement Learning

- Teach an agent to maximize reward in an environment
  - Rules are unknown
  - No supervisor, only a reward signal
  - Feedback is delayed
  - Agent's actions affect the subsequent data it receives
- Value of action summed with that of future optimal actions (Q-learning)

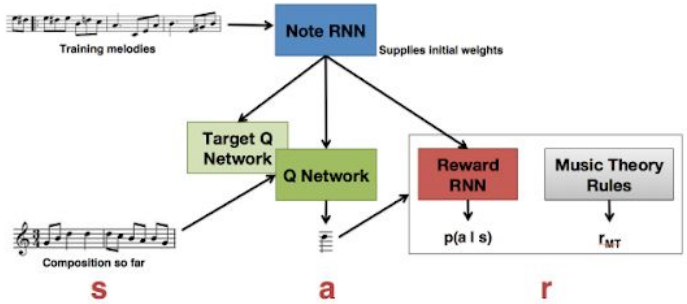


$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

- Current Q-table value we are updating
- Learning rate
- Reward
- Discount
- Estimated reward from our next action

Applying RL to MG

- Penalize or reward various decisions
  - Stay in same key
  - Begin/end on same note
  - Low self-correlation at small scales
  - Small steps and large intervals

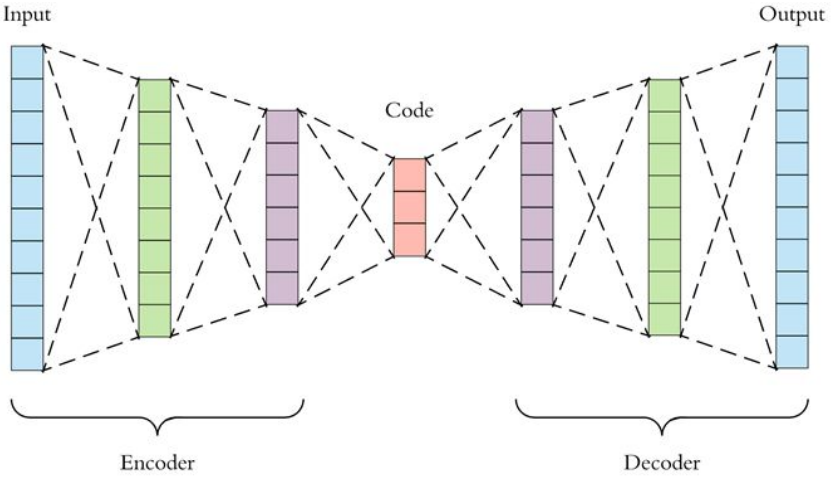


- **Generating Music by Fine-Tuning Recurrent Neural Networks with Reinforcement Learning**
- **Polyphonic Music Composition with LSTM Neural Networks and Reinforcement Learning**

AutoEncoder

- Useful for decreasing dimensionality (keep in mind that each code is a feature!)
- Could be lossy

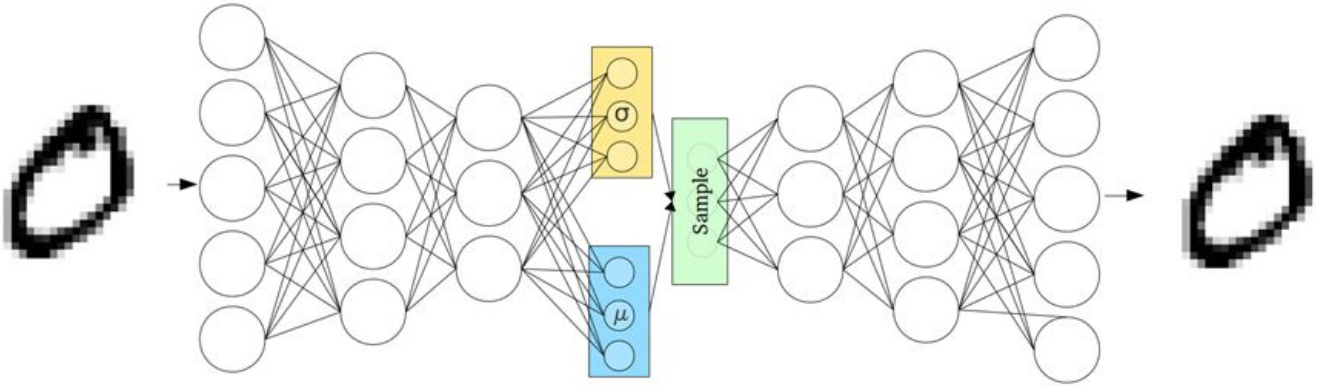
Original



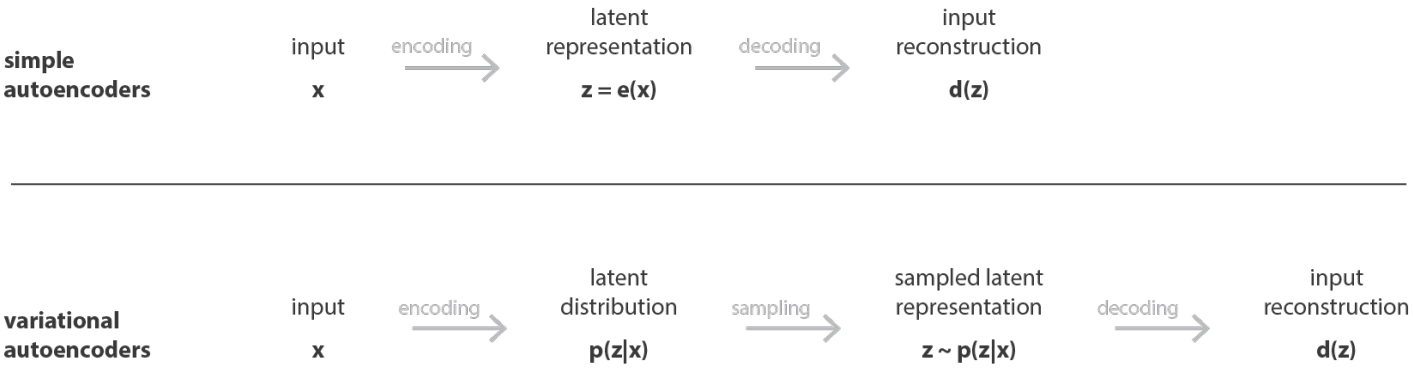
Generated

Variational AutoEncoder

- Regularised versions of autoencoders
- Making the generative process possible



AE vs. VAE


simple autoencoders
input
x
encoding
latent representation
z = e(x)
decoding
input reconstruction
d(z)
variational autoencoders
input
x
encoding
latent distribution
p(z|x)
sampling
sampled latent representation
z ~ p(z|x)
decoding
input reconstruction
d(z)

VAE Music Generation Example

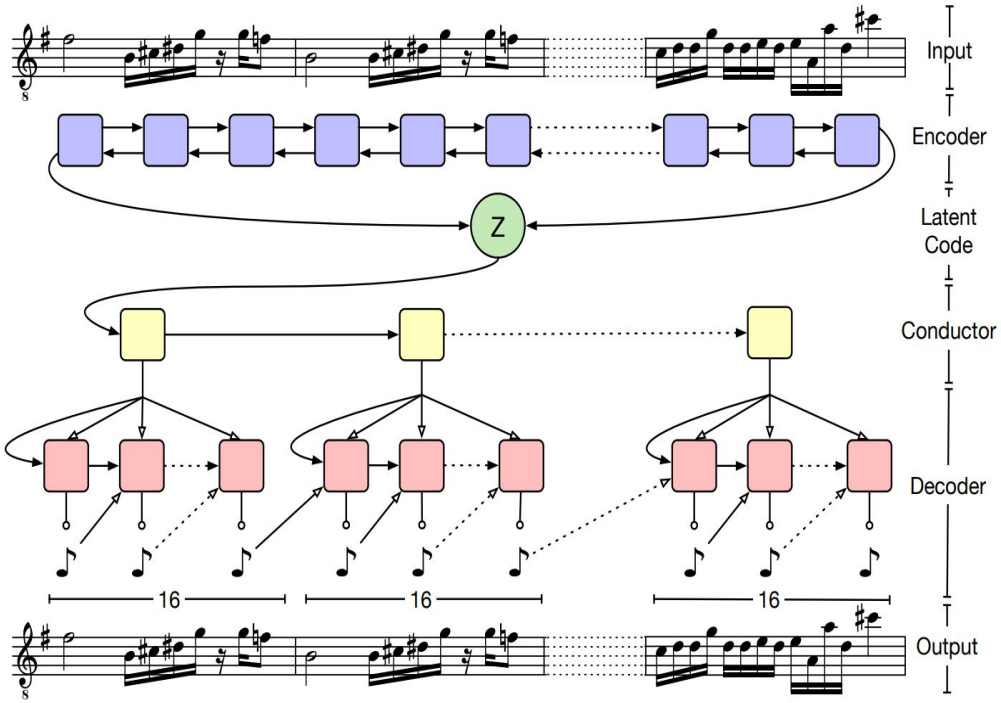
A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music

Latent distribution parameters:

$$\mu = W_{h\mu} h_T + b_\mu$$
$$\sigma = \log\left(\exp(W_{h\sigma} h_T + b_\sigma) + 1\right)$$

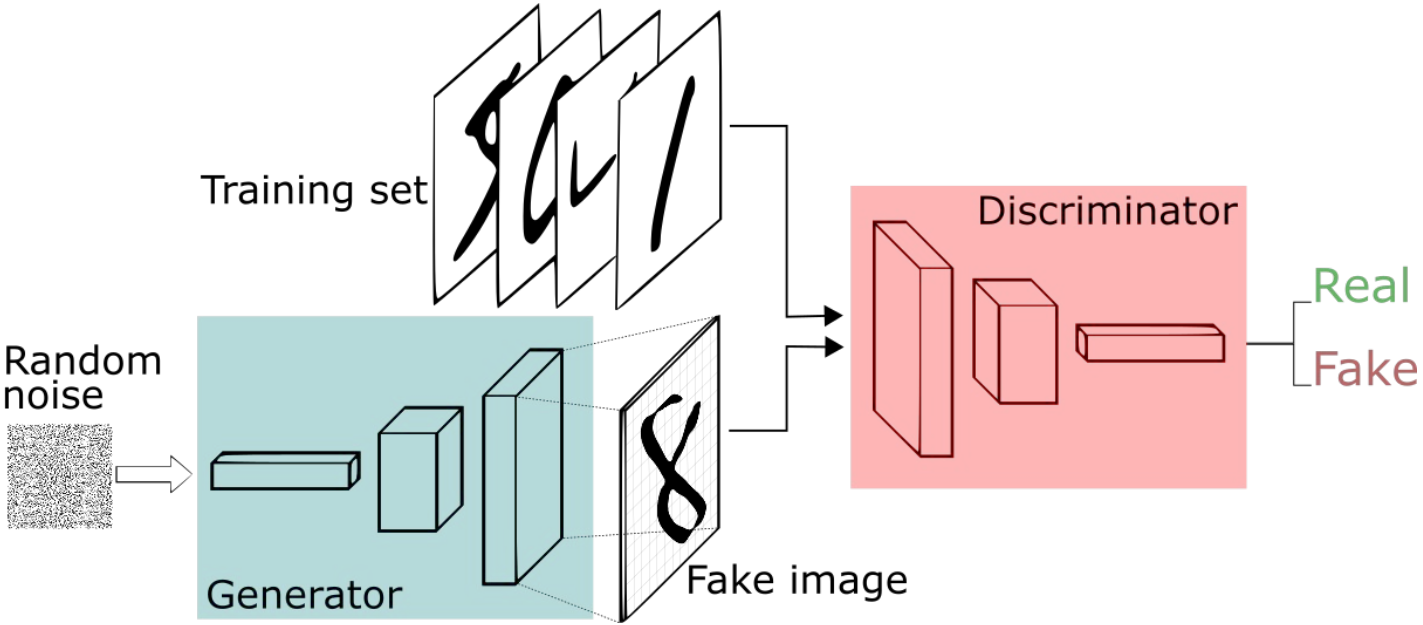
Interpolation equation

$$c_\alpha = \alpha z_1 + (1 - \alpha) z_2$$



Demos:https://storage.googleapis.com/magentadata/papers/musicvae/index.html

Generative Adversarial Networks

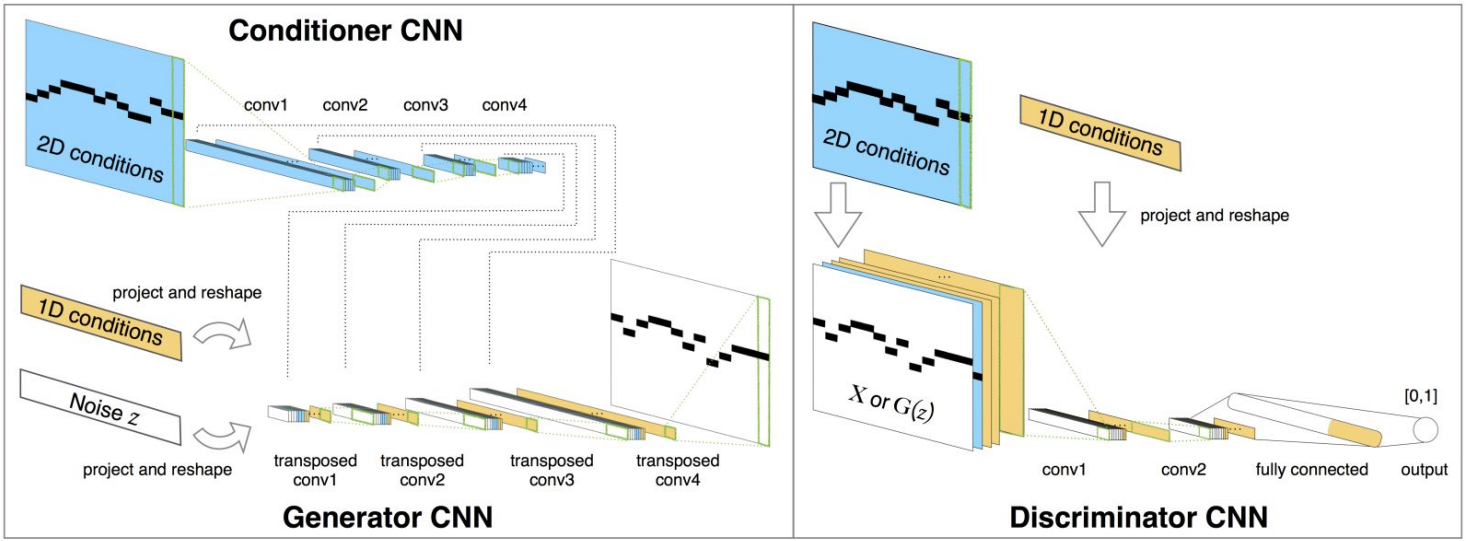


GANs for Images

Photo via Art and Artificial Intelligence Laboratory, Rutgers University

GAN Music Generation Example

MIDINET: A Convolutional Generative Adversarial Network For Symbolic-Domain Music Generation



Questions...

Appendix

Music transcription modelling and composition using deep learning

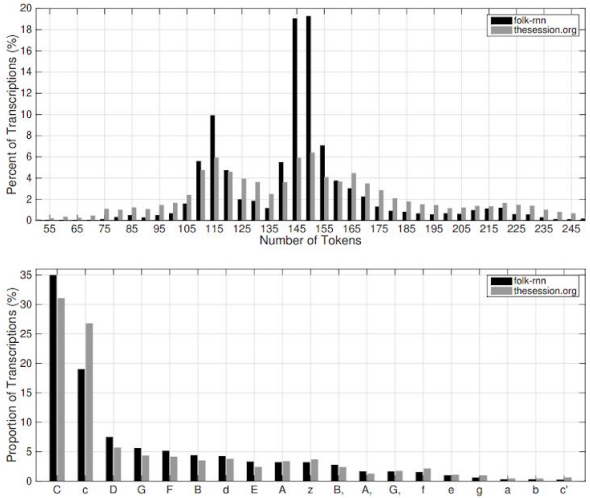


Fig. 1. Top: Distribution of the number of tokens in a transcription for the 6,101 transcriptions created by our *folk-rnn* system, compared with those in its (transposed) training dataset. Bottom: Proportion of transcriptions that conclude on a given pitch.

Appendix

DeepBach: a Steerable Model for Bach Chorales Generation

Algorithm 1 Pseudo-Gibbs sampling

Input: Chorale length L, metadata $\mathcal{M}$ containing lists of length L, probability distributions (p_1, p_2, p_3, p_4), maximum number of iterations M
Create four lists $\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3, \mathcal{V}_4)$ of length L
{The lists are initialized with random notes drawn from the ranges of the corresponding voices (sampled uniformly or from the marginal distributions of the notes)}
for m from 1 to M **do**
    Choose voice i uniformly between 1 and 4
    Choose time t uniformly between 1 and L
    Re-sample $\mathcal{V}_i^t$ from $p_i(\mathcal{V}_i^t | \mathcal{V}_{\setminus i,t}, \mathcal{M}, \theta_i)$
end for
Output: $\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3, \mathcal{V}_4)$

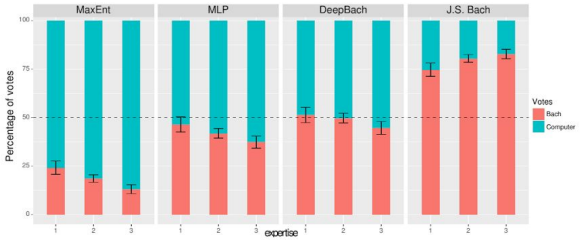


Figure 5. Results of the "Bach or Computer" experiment. The figure shows the distribution of the votes between "Computer" (blue bars) and "Bach" (red bars) for each model and each level of expertise of the voters (from 1 to 3), see Sect. 3.2 for details.

Figure 6. Results of the "Bach or Computer" experiment. The figure shows the percentage of votes for Bach for each of the 100 extracts for each model. For each model, a specific order for the x-axis is chosen so that the percentage of Bach votes is an increasing function of the x variable, see Sect. 3.2 for details.

Appendix

This time with feeling: learning expressive musical performance

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

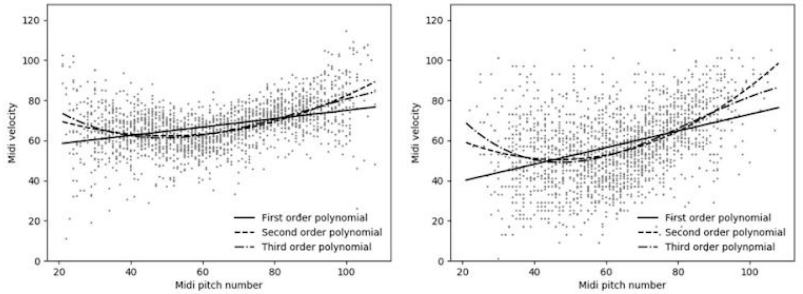


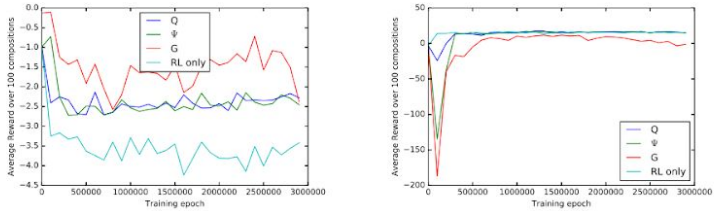
Fig. 7 Left: Pitch–velocity relationship for the real dataset. Right: Pitch–velocity relationship for a set of generated examples. Each data point is a pair *(pitch, average velocity)* for one MIDI file/excerpt, where mean velocity is taken over the nonzero velocities. For clarity, only approximately $\frac{1}{10}$ 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

Appendix

Generating Music by Fine-Tuning Recurrent Neural Networks with Reinforcement Learning

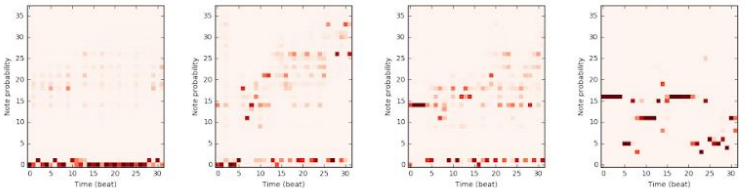
Metric	Note RNN	Q	Ψ	G
Notes not in key	0.09%	1.00%	0.60%	28.7%
Mean autocorrelation - lag 1	-.16	**-.11**	**-.10**	.55
Mean autocorrelation - lag 2	.14	**.03**	**-.01**	.31
Mean autocorrelation - lag 3	-.13	**.03**	**.01**	17
Notes excessively repeated	63.3%	**0.0%**	**0.02%**	**0.03%**
Compositions starting with tonic	0.86%	**28.8%**	**28.7%**	0.0%
Leaps resolved	77.2%	**91.1%**	**90.0%**	52.2%
Compositions with unique max note	64.7%	56.4%	59.4%	37.1%
Compositions with unique min note	49.4%	51.9%	**58.3%**	**56.5%**
Notes in motif	5.85%	**75.7%**	**73.8%**	**69.3%**
Notes in repeated motif	0.007%	**0.11%**	**0.09%**	0.01%

Table 1: Statistics of music theory rule adherence based on 100,000 randomly initialized compositions generated by each model. The top half of the table contains metrics that should decrease, while the bottom half contains metrics that should increase. Bolded entries represent significant improvements over the *Note RNN* baseline.



(a) *Note RNN* reward: $\log p(a|s)$ (b) Music theory reward

Figure 2: Average reward obtained by sampling 100 compositions every 100,000 training epochs. The three models are compared to a model trained using only the music theory rewards r_{MT}.



(a) *Note RNN* (b) *Q* (c) Ψ (d) *G*

Figure 3: Compositions generated by each model. The probability placed on playing each note is shown on the vertical axis, with red indicating higher probability.

Appendix

Polyphonic Music Composition with LSTM Neural Networks and Reinforcement Learning

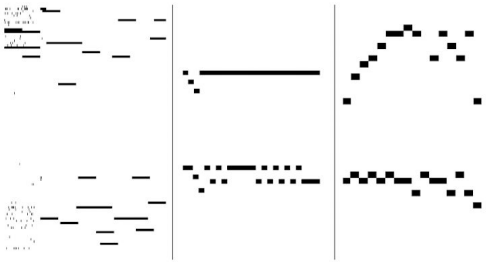


Figure 2: **Compositions from the network at 4 Epochs(left), 20 Epochs(center) and 50 epochs(right). Y-Axes are not to equal scales.**

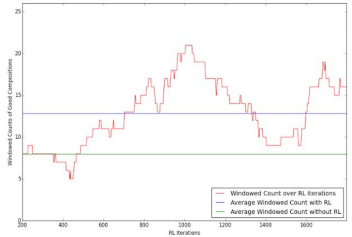


Figure 3: **Moving Windowed Count of Good Compositions against RL Iterations**

	Average over 200 Compositions	
Attribute	*With RL*	*Without RL*
Dyads, Triads and Seventh Chords	**0.24**	0.12
Pitch Entropy	**0.82**	0.71
Very short/long duration incidence	0.05	0.08
Repeated Identical Note-sets	**0.06**	0.20
Aggregated Rest Duration	**0.08**	0.14
Rest Count	0.07	0.11
Cross-Correlation peak	**0.13**	0.2

Table 1: **Improvements in compositional attributes when using Reinforcement Learning**

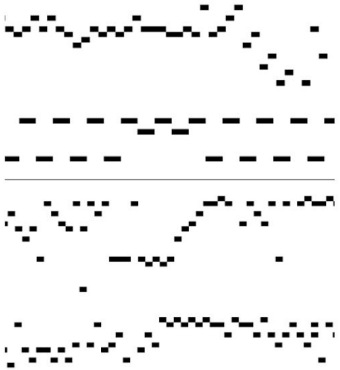


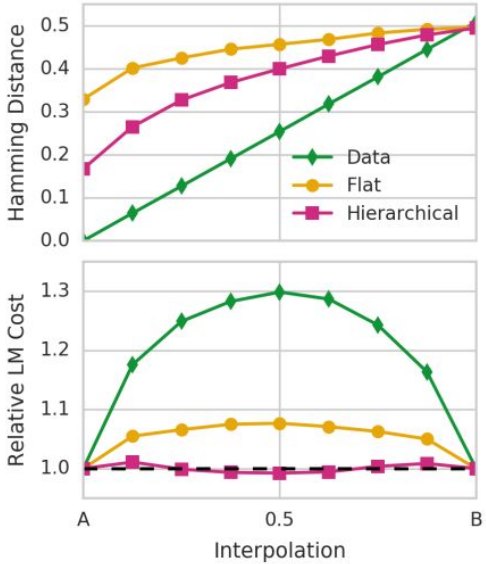
Figure 4: **Two example Compositions dominated by Für Elise(top) and Espana Op. 165 Prelude(bottom)**

Appendix

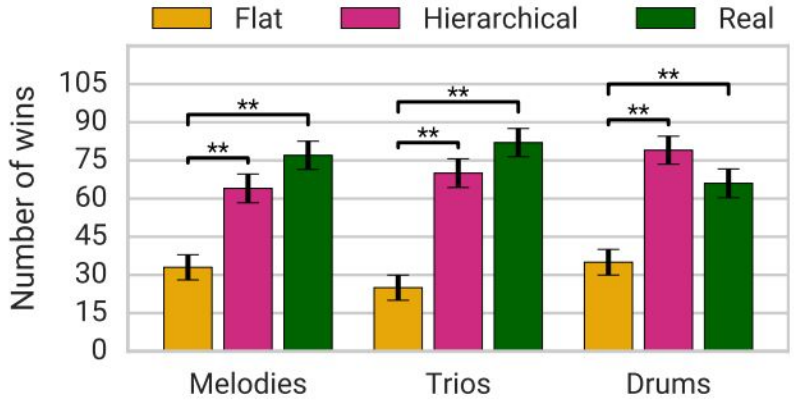
VAE algorithm:
1-Two-layer bidirectional LSTM (Encoder)
2-Concatenate the outputs and pass them through 2 FC layers latent distribution parameters μ and σ :
3-Passing z through a FC layer and obtain C series
4-Passing C's through a FC to obtain initial states of each subsequences
5-Concatenating previous generated note with initial state to generate the new one

Results of "A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music"**:**

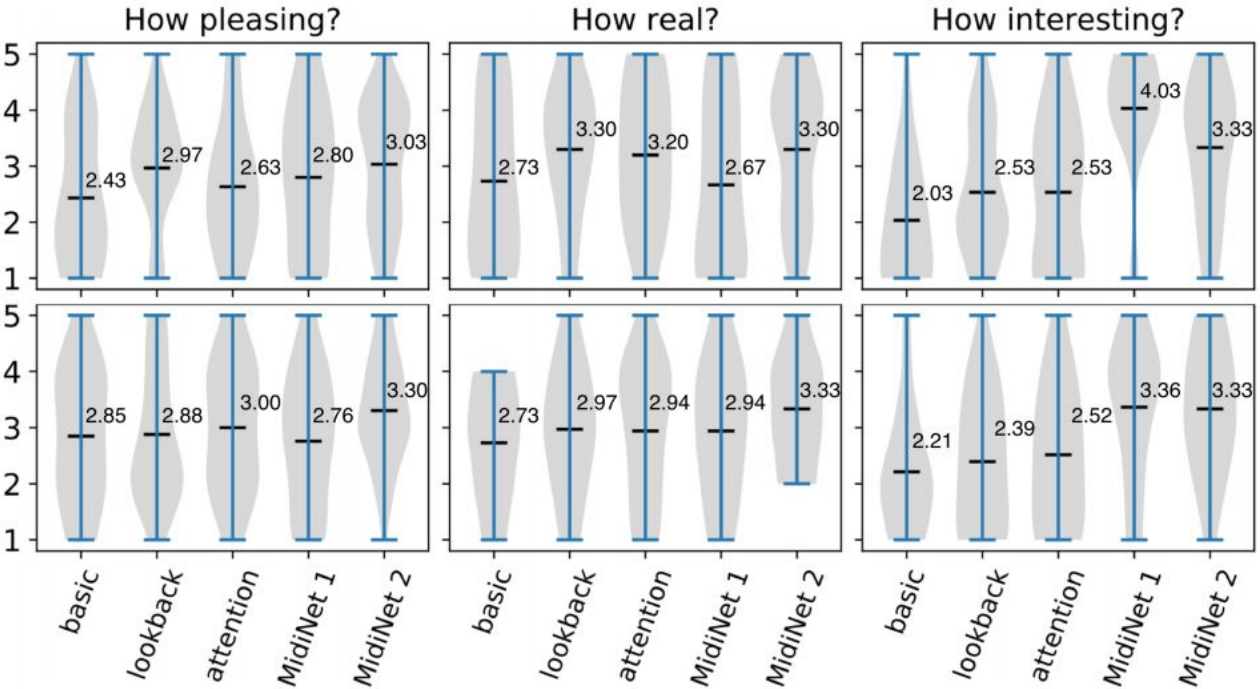
Interpolation Results:



Model	Teacher-Forcing		Sampling	
	Flat	Hierarchical	Flat	Hierarchical
2-bar Drum	0.979	-	0.917	-
2-bar Melody	0.986	-	0.951	-
16-bar Melody	0.883	**0.919**	0.620	**0.812**
16-bar Drum	0.884	**0.928**	0.549	**0.879**
Trio (Melody)	0.796	**0.848**	0.579	**0.753**
Trio (Bass)	0.829	**0.880**	0.565	**0.773**
Trio (Drums)	0.903	**0.912**	0.641	**0.863**



Results of MIDINET: A CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORK FOR SYMBOLIC-DOMAIN MUSIC GENERATION



Different types of GAN structure