# Hierarchical Variational Autoencoders for Music

**Adam Roberts***
Google Brain
adarob@google.com

**Jesse Engel**
Google Brain
jesseengel@google.com

**Douglas Eck**
Google Brain
deck@google.com

## Abstract

In this work we develop recurrent variational autoencoders (VAEs) trained to reproduce short musical sequences and demonstrate their use as a creative device both via random sampling and data interpolation. Furthermore, by using a novel hierarchical decoder, we show that we are able to model long sequences with musical structure for both individual instruments and a three-piece band (lead, bass, and drums). Finally, we demonstrate the effectiveness of scheduled sampling in significantly improving our reconstruction accuracy.

## 1 Introduction

A common approach for generating text, sequences of musical notes, or pixels in an image involves sampling values autoregressively from a recurrent neural network. However, recent works have applied variational autoencoders [Kingma and Welling, 2013] with recurrent decoders to the generation of full sentences [Bowman et al., 2016] and sketches [Ha and Eck, 2017]. By optimizing these models both for their ability to reconstruct full inputs and to constrain their intermediate embeddings, $z$, to a tractable distribution, $p(z)$ (typically $\mathcal{N}_k(0, I)$), they enable explorations of the latent space and interpolations between points, while also producing realistic, novel samples. These rich latent spaces can be used to provide artists with significant control over their creations, giving them access to an intuitive palette to work from.

In this paper, we apply similar variational autoencoding techniques to the generation of musical sequences and demonstrate our ability to model sequences of various scales: 2-bar “loops", 32-bar lead melodies, and 16-bar “trios" (lead, drums, bass). One challenge with training recurrent autoencoders is that teacher forcing reduces the decoder’s dependence on the latent code, decreasing reconstruction accuracy at inference time. We show that applying scheduled sampling [Bengio et al., 2015] during training helps resolve this issue.

## 2 Results

### 2.1 Loops

In order to enable the creation of interpolations between different musical loops, we separately model 2-bar sequences for melodies and drum beats. Melody loops are represented as a sequence of 32 categorical variables taking one of 130 discrete states per sixteenth note: 128 note-on pitches, a hold state, and a rest state. Drums are represented as 32 categorical variables taking one of 512 discrete states per sixteenth note, representing each of the possible combinations between 9 types of hits: kick, snare, closed hi-hat, open hi-hat, low tom, mid tom, high tom, crash cymbal, and ride cymbal.

For each, we train a variational autoencoder with a $\mathcal{N}_{256}(0, I)$ prior having a bidirectional LSTM encoder and 3-layer LSTM decoder, achieving near-perfect reconstruction with low KL divergence (Table 1 Rows 1 and 2). We found that using scheduled sampling during training significantly improved reconstruction accuracy and sample quality even though it decreases teacher-forced next-step prediction accuracy. Full architectural and training details can be found in the Appendix.

Table 1: Accuracies for teacher-forced next-step predictions (“Next") and reconstructions (“Recon"). Each baseline is a decoder-only autoregressive model. Scheduled sampling increases reconstruction accuracy and dependence on the latent code, as illustrated by the additional divergence from the prior.

| | Baseline | | VAE | | | VAE w/ Scheduled Sampling | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | | Accuracy | | KL | Accuracy | | KL |
| Model | Next | Recon | Next | Recon | (nats) | Next | Recon | (nats) |
| 2-bar Drum | 0.722 | 0.395 | **0.979** | 0.917 | 70.7 | 0.972 | **0.963** | 73.5 |
| 2-bar Melody | 0.723 | 0.619 | 0.986 | 0.951 | 72.9 | **0.989** | **0.987** | 76.2 |
| 32-bar Melody | 0.736 | 0.579 | **0.824** | 0.674 | 173.2 | 0.724 | **0.722** | 343.8 |
| 16-bar Trio (Lead) | 0.735 | 0.423 | **0.818** | 0.653 | 177.4 | 0.763 | **0.761** | 811.6 |
| 16-bar Trio (Bass) | 0.781 | 0.414 | **0.882** | 0.692 | "" | 0.824 | **0.823** | "" |
| 16-bar Trio (Drums) | 0.810 | 0.221 | **0.926** | 0.808 | "" | 0.871 | **0.868** | "" |

Supplemental Figures 2 and 3 show the results of interpolating between pairs of melody loops and drums beats, respectively. Audio for these interpolations can be heard at `https://goo.gl/twGuP2`. We find these interpolations to be very smooth and musical, even when bridging highly distinct endpoints.

### 2.2 Lead Melodies

Having proven our ability to reconstruct short loops, we move our attention to lengthier melodic sequences that demonstrate aspects of long-term structure. The simple LSTM decoder we used to model loops does not scale well to 32-bar sequences, likely due to the limited context of the LSTM and the difficulty in parsing note-level detail directly from a latent code, $z$, representing 512 steps.

With the knowledge that we can decode near-perfect reconstructions of 2-bar sequences from a $z$ with an LSTM, we introduce a novel recurrent hierarchical decoder. In the first level, an LSTM is initialized with $z$ and then sequentially outputs 16 embeddings. These embeddings are used to initialize a lower-level LSTM that autoregressively produces 512 sixteenth notes (32 bars). After every 32 outputs (2 bars), the state of the LSTM is reset using the next embedding from the first-level model. Furthermore, these embeddings are concatenated with the output at each step to provide the next input.

With this architecture, we are able to successfully reconstruct and randomly sample 32-bar melodies with aspects of long-term structure (Table 1 Row 3, Supplemental Figure 4 and `https://goo.gl/twGuP2`).

### 2.3 Trios

Finally, we consider the problem of modelling the relationship between instruments in a composition. We make a small modification to our hierarchical decoder, splitting the second level into 3 separate LSTMs, one each to decode lead, bass, and drum parts. We use the same 130-category distribution for the bass and lead as for melody above and concatenate the three instruments at each step as the input to our encoder.

Once again, we are able to reconstruct (albeit with lower accuracy; Table 1, Rows 4-6) and randomly sample musically realistic, novel sequences from the model (Supplemental Figure 5, `https://goo.gl/twGuP2`).

## 3 Discussion

Having a rich, intuitive latent space of musical sequences will enable numerous innovative interactions for musical creativity. Future work will involve identifying and building proper interfaces to this space. Furthermore, recent work has demonstrated a notable aesthetic effect in musical sequences produced by modelling detailed aspects of performances such as polyphony, micro-timing, and dynamics [Simon and Oore, 2017]. It should be possible to extend this work to model these features, given the proper datasets.

**Acknowledgments**

The authors wish to thank Colin Raffel and David Ha for their helpful feedback and discussions.

## References

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013. URL `http://arxiv.org/abs/1312.6114`.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, 2016. URL `http://arxiv.org/abs/1511.06349`.

David Ha and Douglas Eck. A neural representation of sketch drawings. *arXiv preprint*, 2017. URL `http://arxiv.org/abs/1704.03477`.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam M. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. URL `http://arxiv.org/abs/1506.03099`.

Ian Simon and Sageev Oore. Performance rnn: Generating music with expressive timing and dynamics. magenta.tensorflow.org, 2017. URL `https://magenta.tensorflow.org/performance-rnn`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. URL `http://arxiv.org/abs/1412.6980`.

## Appendix

### Code Availability

TensorFlow implementations of all models will soon be made available at `https://github.com/tensorflow/magenta`.

### Architecture Details

### 2-bar Melody and Drum Architectures

The encoder used for 2-bar drum and melody models is made up of a single-layer bidirectional LSTM, with 2048 units per cell. The final output in each direction is concatenated and passed through a linear layer to produce 512 outputs. Half of the outputs are used as the $\mu$ and the other half are used as a $\sigma_{\text{pre}}$, where $\mu$ and $\sigma = \exp(2\sigma_{\text{pre}})$ parameterize a 512-dimension multivariate Gaussian distribution with a diagonal covariance matrix for $z$.

In the decoder, the $z$ is passed through a linear layer to initialize the state of a 3-layer LSTM with 2048 units per layer. This LSTM autoregressively produces individual sixteenth note events, passing its output through a linear layer and softmax to create a distribution over the 130/512 melody/drum classes. The categorical distribution is used to compute a cross-entropy loss during training or samples at inference time. In addition to generating the initial state at the start of each bar, the $z$ is concatenated with the previous output as the input at each time step.

### 32-bar Melody Architecture

This architecture is diagrammed in Figure 1.

The encoder is made up of a single-layer bidirectional LSTM, with 2048 units per cell. The final output in each direction is concatenated and passed through a linear layer to produce 1024 outputs. Half of the outputs are used as the $\mu$ and the other half are used as a $\sigma_{\text{pre}}$, where $\mu$ and $\sigma = \exp(2\sigma_{\text{pre}})$

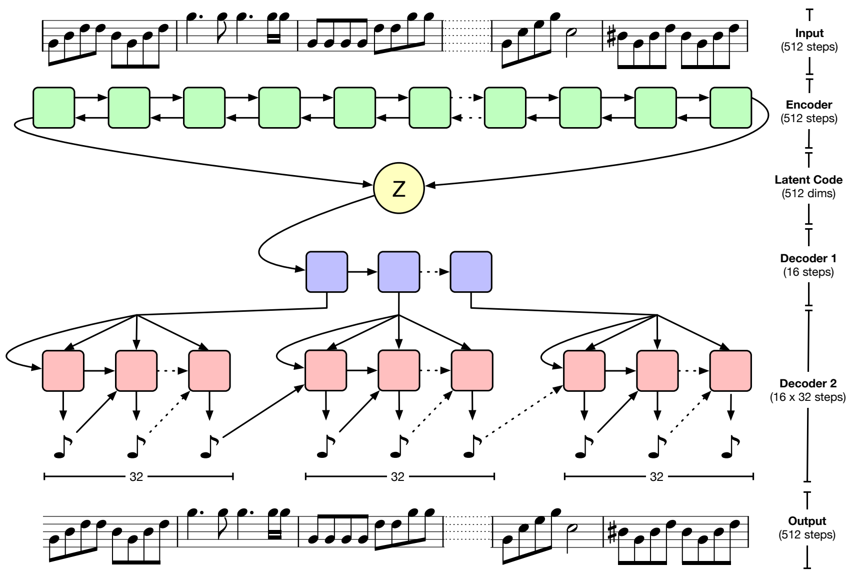

Figure 1: Architecture for recurrent hierarchical melody VAE. LSTM cells shown in the same color share weights and linear layers between levels are omitted.

parameterize a 512-dimension multivariate Gaussian distribution with a diagonal covariance matrix for $z$.

We use a recurrent hierarchical decoder to model long melodies. First, the $z$ goes through a linear layer to initialize the state of a 2-layer LSTM with 1024 units per layer, which outputs 16 embeddings of size 512 each, one for each pair of bars. Each of these embeddings are passed through a linear layer to produce 16 initial states for another 2-layer LSTM with 1024 units per layer. This lower-level LSTM autoregressively produces individual sixteenth note events, passing its output through a linear layer and softmax to create a distribution over the 130 classes. The categorical distribution is used to compute a cross-entropy loss during training or samples at inference time. In addition to generating the initial state at the start of each bar, the embedding for the current bar is concatenated with the previous output as the input at each time step.

## 16-bar Trio Architecture

The encoder is the same as for the 32-bar melody architecture, except the one-hot encodings of the lead, bass, and drum labels are concatenated as the inputs.

We use a similar hierarchical decoder as in the case of the 32-bar melody architecture, except there are 3 independent bar-level LSTMs as the second level. Each of these LSTMs uses the same sequence of first-level embeddings to initialize/reset their states at each bar and to concatenate with previous outputs as the input at each time step.

## Dataset

The datasets were built by first scraping the web for publicly-available MIDI files, resulting in $\sim 1.5$ million unique files. We removed those that were identified as having a non-$\frac{4}{4}$ time signature and used the encoded tempo to determine bar boundaries, quantizing to 16 notes per bar (sixteenth notes).

For the 2-bar drum loops, we used a 2-bar sliding window (with a stride of 1 bar) to extract all unique 2-bar drum sequences (channel 10) with at most a single bar of consecutive rests, resulting in 3.8 million examples.

For 2-bar (32-bar) melodies, we used a 2-bar (32-bar) sliding window (with a stride of 1 bar) to extract all unique non-drum sequences with at most a single bar of consecutive rests, resulting in 28.0

million (1.6 million) unique examples. During training we augmented the data by shifting all pitches in each example by a randomly sampled amount in the interval [-5, 6].

For the trio data, we used a 16-bar sliding window (with a stride of 1 bar) to extract all unique sequences containing an instrument with a program number in the piano, chromatic percussion, organ, or guitar interval, [0, 31], one in the bass interval, [32, 39], and one that is a drum (channel 10), with at most a single bar of consecutive rests in any instrument. If there were multiple instruments in any of the three categories, we took the cross product to consider all possible combinations. This resulted in 9.4 million examples.

## Training Details

All models are trained with the Adam optimizer [Kingma and Ba, 2015], with a learning rate annealed from 1e-3 to 1e-5 with exponential decay rate 0.9999, using a batch size of 512. We used scheduled sampling [Bengio et al., 2015] with an inverse-sigmoid schedule and a rate of 1000.

The 2-bar melody and drum models were given a tolerance of 48 free bits ($\sim$ 33.3 nats) and had their KL cost weight, $\beta$, annealed from 0.0 to 0.2 with exponential decay rate 0.99999 over 100k steps.

The 32-bar melody model was given a tolerance 256 free bits ($\sim$ 177.4 nats) with $\beta = 1$. It was trained to 150k steps.

The 16-bar trio model was given a tolerance 256 free bits ($\sim$ 177.4 nats) and had its $\beta$ annealed from 0.0 to 0.2 with exponential decay rate 0.99999 over 50k steps.

## Sampling Details

Random, unconditioned samples were made by first drawing a random point $z \sim \mathcal{N}_k(0, I)$, where $k = 256$ for the 2-bar models and $k = 512$ for the 16- and 32-bar models. $z$ was then passed through the decoder of a given model and the outputs sampled from the categorical distributions produced by the decoder at each step, using a softmax temperature of 0.5.

Interpolations were made by first encoding selected examples from the evaluation set into $z_a$ and $z_b$. Intermediate outputs outputs were then sampled from the model decoder using points in the latent space computed at equally-spaced intervals between $z_a$ and $z_b$ via Slerp (spherical linear interpolation).

## Supplemental Figures

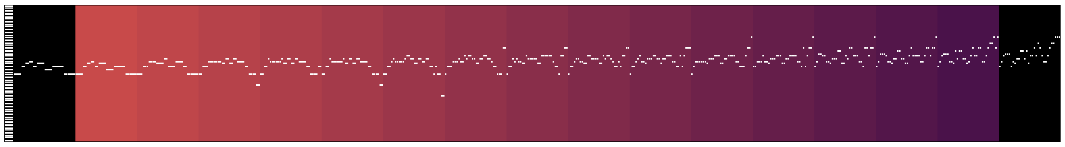

Figure 2: Melody interpolation between evaluation examples (shown in black) over 15, 2-bar steps. Synthesized audio can be heard at `https://goo.gl/twGuP2`.

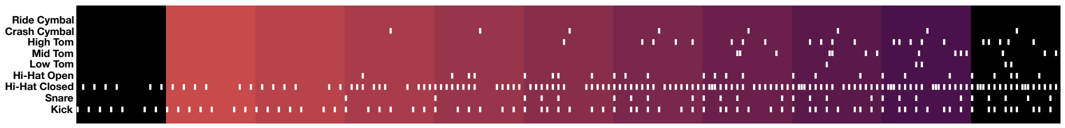

Figure 3: Drum interpolation between evaluation examples (shown in black) over 9, 2-bar steps. Synthesized audio can be heard at `https://goo.gl/twGuP2`.

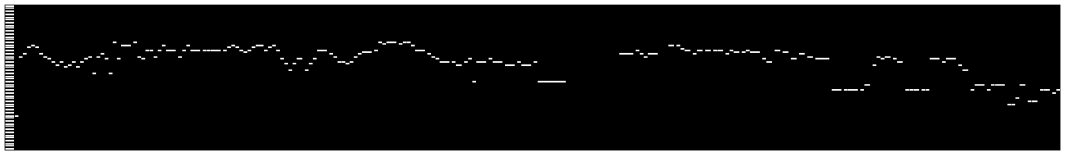

Figure 4: Prior sample from 32-bar melody model. Synthesized audio can be heard at `https://goo.gl/twGuP2`.

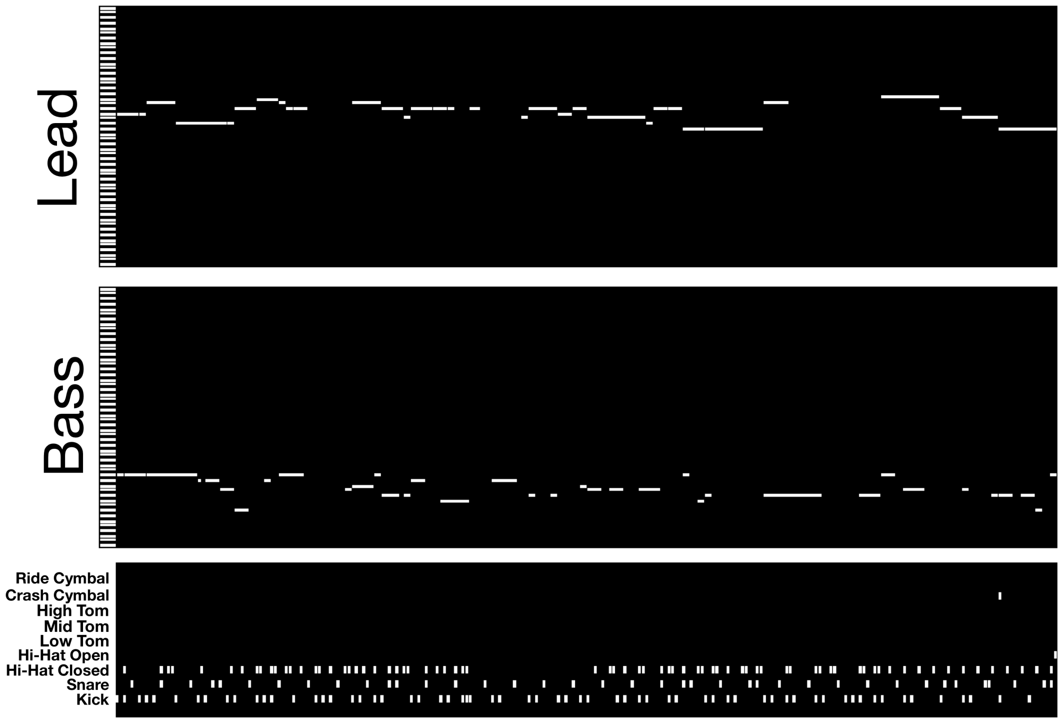

Figure 5: Sample from 16-bar “trio" model with lead, bass, and drum parts. Synthesized audio can be heard at `https://goo.gl/twGuP2`.