COUNTERPOINT BY CONVOLUTION

Cheng-Zhi Anna Huang[*] **& Tim Cooijmans**[†]
Department of Computer Science and Operations Research
University of Montreal
Montreal, MC H3T 1N8, Canada
{chengzhiannahuang,cooijmans.tim}@gmail.com

Adam Roberts
Magenta, Google Brain
Mountain View, CA 94043, U.S.
adarob@google.com

Aaron Courville
Department of Computer Science & Operations Research
University of Montreal
Montreal, MC H3T 1N8, Canada
aaron.courville@umontreal.ca

Douglas Eck
Magenta, Google Brain
Mountain View, CA 94043, U.S.
deck@google.com

ABSTRACT

Machine learning models of music typically break down the task of composition into a chronological process, composing a piece of music in a single pass from beginning to end. Human composers on the other hand write music in a nonlinear fashion, scribbling motives here and there, often revisiting choices previously made. We introduce COCONET, a deep convolutional inpainting model trained to reconstruct partial scores. Our model is an instance of ConvNADE Uria et al. (2016), however we wrap the NADE sampling procedure in a blocked Gibbs process to improve sample quality. We show the versatility of our method on three generative tasks: conditioned rewriting, partial score completion, and unconditioned polyphonic music generation.

1 INTRODUCTION

Machine learning can be used to create compelling art. This was shown recently by DeepDream (Mordvintsev et al., 2015), an optimization process that created psychedelic transformations of images that bent our visual imagination. A similar idea underlies a variety of style transfer algorithms (Gatys et al., 2015), which impose textures and colors from one image onto another. More recently, the multistyle pastiche generator (Dumoulin et al., 2016) exposes adjustable knobs that allow users of the system fine-grained control over style transfers. Neural doodle (Champandard, 2016) further closes the feedback loop between user and algorithm.

We wish to bring similar artistic tools to the domain of music. Whereas previous work in music has relied mainly on sequence models such as Hidden Markov Models (HMMs, Baum & Petrie (1966)) and Recurrent Neural Networks (RNNs, Rumelhart et al. (1988)), we instead employ convolutional neural networks due to their locality and invariance properties. Moreover, convolutional neural networks have shown to be extremely versatile once trained, as shown by a variety of creative abuses in the literature. (Mordvintsev et al., 2015; Gatys et al., 2015; Almahairi et al., 2015; Lamb et al., 2016)

[*]This work was carried out during the author's Summer internship at Google Brain Magenta and continued at her current Fall internship at MILA.

[†]This work was carried out during the author's Summer internship at Google Brain Magenta and continued during his PhD studies at MILA.

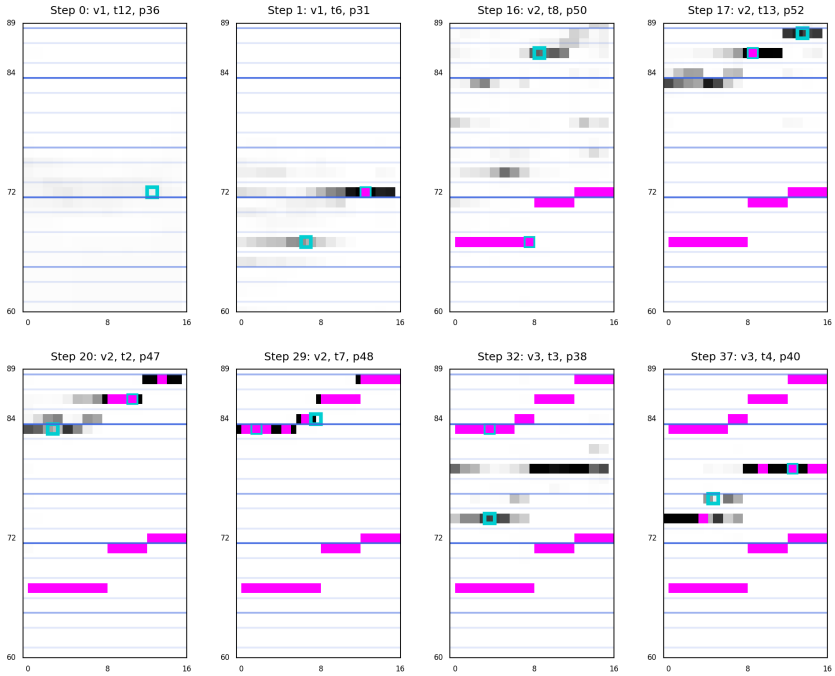

Figure 1: Unconditioned musical composition with Coconet

Section 2 discusses other work in the domain of music. In Section 3 we highlight NADE, the framework that encompasses our approach. Section 4 introduces the musical equivalent of image inpainting, the task we train our model to solve. The details of our convolutional model are laid out in Section 5, which includes discussion of the sampling procedure. Results of quantitative and qualitative evaluations are reported in Section 6, and Section **??** concludes.

2 Related Work

Sequence models such as HMMs and RNNs are a natural choice for modeling music. However, one of the challenges in adapting such models to music is that music generally consists of multiple interdependent streams of events. This can be most clearly seen in the notion of counterpoint, which refers to the relationships between the movement of individual instruments in a musical work. Compare this to typical sequence domains such as speech and language, which involve modeling a single stream of events: a single speaker or a single stream of words.

Successful application of sequence models to music hence requires serializing or otherwise re-representing the music to fit the sequence paradigm. For instance, Liang (2016) serialize four-part Bach chorales by interleaving the parts, while Allan & Williams (2005) construct a chord vocabulary. Boulanger-Lewandowski (2014) adopt a piano roll representation, which is a binary matrix $\mathbf{X}$ such that x_{ij} is hot if some instrument is playing pitch i at time j. To model the joint probability distribution of the multi-hot pitch vector $\mathbf{x}_j$, they employ a Restricted Boltzmann Machine (RBM (Smolensky, 1986; Hinton et al., 2006)) or Neural Autoregressive Distribution Estimator (Uria et al., 2016) at each time step.

Moreover, the behavior of human composers does not fit the chronological mold assumed by previous authors. A human composer might start his work with a coarse chord progression and iteratively refine it, revisiting choices previously made. Sampling according to $x_t \sim p(x_t|x_{<t})$, as is common, cannot account for the kinds of timeless dependencies that composers employ. Not only does this

limit the compositional abilities of chronological models, it also limits their usefulness as compositional aids.

Hadjeres et al. (2016) sidestep the choice of causal factorization and instead employs an undirected Markov model to learn the relationships between neighboring notes in a score. Sampling involves Markov Chain Monte Carlo (MCMC) using the model as a Metropolis-Hastings (MH) objective. The model permits constraints on the state space to support tasks such as melody harmonization. However, the Markov assumption is severely limiting.

3 NADE

Instead, we turn to the Neural Autoregressive Distribution Estimator (NADE, Uria et al. (2016)) and its extensions. A NADE models a multivariate distribution $p(\mathbf{x})$ through a factorization

$$p_\theta(\mathbf{x}) = \prod_d p_\theta(\mathbf{x}_{o_d} \mid \mathbf{x}_{o_{<d}}) \tag{1}$$

where o is a permutation, and the parameters θ are shared among the conditionals. NADE can be trained for all orderings o simultaneously using the orderless NADE (Uria et al., 2016) training procedure. This procedure relies on the observation that, thanks to parameter sharing, computing $p_\theta(\mathbf{x}_{o_c} \mid \mathbf{x}_{o_{<d}})$ for all $c \geq d$ is no more expensive than computing it only for $c = d$. Hence for a given o and d we can simultaneously obtain partial losses for all orderings that agree with o up to d.

A NADE thus trained offers random access to conditional distributions $p_\theta(\mathbf{x}_i|\mathbf{x}_J)$ based on any set of contextual variables $\mathbf{x}_J$ that might already be known. Our approach to modeling music can be seen as an instance of ConvNADE (Uria et al., 2016), in which p_θ consists of a convolutional neural network.

4 Musical Composition as Inpainting

We consider the musical equivalent of inpainting, a versatile setting that generalizes popular tasks such as melody harmonization, partial score completion and composition from scratch. Inpainting (Bertalmio et al., 2000) is the task of restoring damaged or missing parts of an image. In machine learning, image inpainting has found popularity as an unsupervised learning task, where a model is trained to reconstruct an image after it has been corrupted by a random process (Vincent et al., 2008; Pathak et al., 2016).

Inpainting readily carries over to music when we view it as a stack of piano rolls represented by the binary three-tensor $\mathbf{x} \in \{0, 1\}^{I \times T \times P}$. Here I denotes the number of instruments, T the number of time steps, P the number of pitches, and $\mathbf{x}_{i,t,p} = 1$ iff the ith instrument plays pitch p at time t. We will assume each instrument plays exactly one pitch at a time, that is, $\sum_p \mathbf{x}_{i,t,p} = 1$ for all i, t.

For the present work we will restrict ourselves to the study of four-part Bach chorales as used in prior work (Allan & Williams, 2005; Boulanger-Lewandowski, 2014). Hence we assume $I = 4$ throughout. We discretize pitch according to equal temperament, but constrain ourselves to only the range that appears in our training data (MTS **?** pitches 36 through 88). Time is discretized at the level of 16th notes for similar reasons.

Given a training example $\mathbf{x} \sim p(\mathbf{x})$, we randomly choose a mask $\mathbf{m} \in \{0, 1\}^{I \times T}$ where $\mathbf{m}_{i,t} = 0$ indicates that $\mathbf{x}_{i,t}$ is to be corrupted. We obtain the corrupted example by a broadcasted elementwise multiplication

$$\tilde{\mathbf{x}}_{i,t,p} = \mathbf{x}_{i,t,p}\mathbf{m}_{i,t} \tag{2}$$

and ask the model to reconstruct $\mathbf{x}$ given $\tilde{\mathbf{x}}$ and $\mathbf{m}$. The loss function is given by

$$\mathcal{L}(\mathbf{x}; \mathbf{m}, \theta) = -\sum_{i,t}(1 - \mathbf{m}_{i,t}) \log p_\theta(\mathbf{x}_{i,t} \mid \mathbf{x}_\mathbf{m}, \mathbf{m}) \tag{3}$$

$$= -\sum_{i,t,p}(1 - \mathbf{m}_{i,t})\mathbf{x}_{i,t,p} \log p_\theta(\mathbf{x}_{i,t,p} \mid \mathbf{x}_\mathbf{m}, \mathbf{m}) \tag{4}$$

where p_θ refers to the probability under the model parameterized by θ and $\mathbf{x}_\mathbf{m}$ denotes the variables $\mathbf{x}_{i,t,p}$ with $\mathbf{m}_{i,t} = 1$. We wish to minimize the expected loss

$$\mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \mathbb{E}_{\mathbf{m} \sim p(\mathbf{m})} \mathcal{L}(\mathbf{x}; \mathbf{m}, \theta). \tag{5}$$

This loss function is equivalent to that used by Uria et al. (2016), and we follow their approach in estimating the expectations by sampling. Per Equation 3, the loss for any one sample $(\mathbf{x}, \mathbf{m})$ consists of $M = \sum_{i,t} 1 - \mathbf{m}_{i,t}$ terms of the form $\log p_\theta(x_{i,t} \mid \mathbf{x}_\mathbf{m}, \mathbf{m})$. To ensure consistent estimation of the negative log-likelihood of the joint $p_\theta(\mathbf{x})$, the sample losses need to be rebalanced. Uria et al. (2016) address this issue by reweighting the losses according to the number of terms M:

$$\tilde{\mathcal{L}}(\mathbf{x}; \mathbf{m}, \theta) = \frac{IT}{M} \mathcal{L}(\mathbf{x}; \mathbf{m}, \theta). \tag{6}$$

We instead explore the use of a sampling procedure for $\mathbf{m}$ that satisfies

$$p(\mathbf{m}) \propto \frac{IT}{M}, \tag{7}$$

which proceeds by first sampling a mask size $0 < M \leq IT$ according to categorical probabilities $p(M) \propto \frac{IT}{M}$ and subsequently choosing uniformly randomly among the set of masks of size M. We suspect that sampling with appropriate probabilities in the first place is better than scaling contributions after the fact. However, it is plausible that neither strategy is particularly helpful due to effects of parameter sharing that are difficult to analyze, or even because some terms are simply harder to model than others. Therefore we additionally consider simply i.i.d sampling each element $\mathbf{m}_{i,t}$ according to a Bernoulli distribution.

For the task of inpainting, we might wish to increase the difficulty by emphasizing masks that corrupt large contiguous regions, as otherwise the model might learn only superficial local relationships. This is discussed in Pathak et al. (2016) for the case of image inpainting, where a model might learn only that pixels are similar to their neighbors. Similar low-level relationships hold in our case, as our piano roll representation is binary and very sparse. For instance, we might mask out only a single sixteenth step in the middle of a long-held note, in which case reconstructing the masked out step does not require any deep understanding of music.

5 Counterpoint by Convolution

We approach the task outlined above using a deep convolutional neural network (LeCun & Bengio, 1995). This choice is motivated by the locality of contrapuntal rules and their near-invariance to translation, both in time and in the frequency spectrum.

The input to the model consists of the piano rolls $\mathbf{x}$ concatenated with the (broadcasted) masks $\mathbf{m}$ along the first axis:

$$\mathbf{h}^0_{i,t,p} = \mathbf{x}_{i,t,p} \tag{8}$$

$$\mathbf{h}^0_{I+i,t,p} = \mathbf{m}_{i,t} \tag{9}$$

where the first dimension ranges over channels and the time and pitch dimensions are convolved over.

$$\mathbf{a}^l = \mathrm{BN}(\mathbf{W}^l * \mathbf{h}^{l-1}; \gamma^l, \beta^l) \tag{10}$$

$$\mathbf{h}^l = \mathrm{ReLU}(\mathbf{a}^l + \mathbf{h}^{l-2}) \qquad \text{for } 3 < l < L-1 \text{ and } l = 0 \bmod 2 \tag{11}$$

$$\mathbf{h}^L = \mathbf{a}^L \tag{12}$$

With the exception of the first and final layers, all of our convolutions preserve the size of the input. That is, we use "same" padding throughout and all activations $h^l, 1 < l < L$ have 128 channels. The network consists of 64 layers with 3×3 filters on each layer. After each convolution we apply batch normalization Ioffe & Szegedy (2015) (denoted by $\mathrm{BN}(\cdot)$) with statistics tied across time and pitch. After every second convolution, we introduce a skip connection from the hidden state two levels below to reap the benefits of residual learning He et al. (2015).

Finally, we obtain independent predictions for the pitch at each instrument/time pair:

$$\hat{p}_\theta(\mathbf{x}_{i,t,p} \mid \mathbf{x}_\mathbf{m}, \mathbf{m}) = \frac{\exp(h^L_{i,t,p})}{\sum_p \exp(h^L_{i,t,p})} \tag{13}$$

Based on the predictions, we compute the loss (Equation 3) and optimize with respect to the parameters $\theta = \mathbf{W}^1, \gamma^1, \beta^1, \ldots, \mathbf{W}^{L-1}, \gamma^{L-1}, \beta^{L-1}$ by stochastic gradient descent with step size determined by Adam (Kingma & Ba, 2014).

To sample from the model, we start with an empty (zero everywhere) piano roll $\mathbf{x}^0$ and mask $\mathbf{m}^0$ and populate them iteratively by the following process (Algorithm 1). We feed the piano roll $\mathbf{x}^s$ and mask $\mathbf{m}^s$ into the model to obtain a set of categorical distributions $p_\theta(\mathbf{x}_{i,t}|\mathbf{x}^s_{\mathbf{m}^s}, \mathbf{m}^s)$ for (i,t) such that $\mathbf{m}^s_{i,t} = 0$. As the $\mathbf{x}_{i,t}$ are not conditionally independent, we cannot simply sample from these distributions independently. However, if we sample from one of them, we can compute new conditional distributions for the others. Hence we randomly choose one $(i,t)^{s+1}$ to sample from, and let $\mathbf{x}^{s+1}_{i,t}$ equal the one-hot realization. Update the mask so that $\mathbf{m}^{s+1}_{i,t} = 1$ and repeat until the piano roll is populated. This procedure is easily generalized to tasks such as melody harmonization and partial score completion by starting with a nonempty piano roll.

Algorithm 1 Sampling from the model.

Given an initial piano roll $\mathbf{x}$ and mask $\mathbf{m}$
while $\sum_{i,t} \mathbf{m}_{i,t} < IT$ **do**
  Choose $(i,t) \sim \mathrm{Uniform}((i,t) : \mathbf{m}_{i,t} = 0)$
  Choose $p \sim p_\theta(\mathbf{x}_{i,t,p}|\mathbf{x}_\mathbf{m}, \mathbf{m})$
  $\mathbf{x}_{i,t,p} \leftarrow 1$
  $\mathbf{m}_{i,t} \leftarrow 1$
end while

5.1 Gibbs sampling

We find that when starting with an empty piano roll, a single pass of sequential generation yields poor samples. Instead, we make multiple passes: each time, we mask out some part of the piano roll and then repopulate it. This is a form of blocked Gibbs sampling (Liu, 1994), where each block is itself sampled sequentially using Algorithm 1. The procedure is specified by Algorithm 2.

Blocked sampling is crucial for mixing, as the high temporal resolution of our representation causes strong correlations between consecutive notes. For instance, without blocked sampling, it would take many steps to snap out of a long-held note.

Algorithm 2 Sampling from the model by blocked Gibbs.

Given an initial piano roll $\mathbf{x}$ and mask $\mathbf{m}$
loop n times
  if not the first iteration **then**
    Choose $\mathbf{m} \sim p(\mathbf{m})$ ▷ Determine a block to resample
    $\mathbf{x}_{i,t} \leftarrow 0 \; \forall (i,t) : \mathbf{m}_{i,t} = 0$
  end if
  while $\sum_{i,t} \mathbf{m}_{i,t} < IT$ **do** ▷ Sample from the joint distribution of $\mathbf{x}_{\neg \mathbf{m}}$
    Choose $(i,t) \sim \text{Uniform}((i,t) : \mathbf{m}_{i,t} = 0)$
    Choose $p \sim p_\theta(\mathbf{x}_{i,t,p} | \mathbf{x}_\mathbf{m}, \mathbf{m})$
    $\mathbf{x}_{i,t,p} \leftarrow 1$
    $\mathbf{m}_{i,t} \leftarrow 1$
  end while
end loop

Table 1: Negative log-likelihood (NLL) in nats per note on the validation set for the BACH dataset.

Model	Note-level NLL
Bachbot (Liang, 2016)	0.477
Coconet, balanced by scaling	0.55 ± 1.31
Coconet, balanced by sampling	0.95 ± 1.51
Coconet, i.i.d Bernoulli(0.25)	0.63 ± 1.58
Coconet, i.i.d Bernoulli(0.50)	0.92 ± 2.24

6 EVALUATION

We evaluate our approach on the JSB dataset that is also used by previous authors (Allan & Williams, 2005; Boulanger-Lewandowski, 2014; Liang, 2016). The dataset consists of 382 four-part Bach chorales. We compare with Liang (2016); Boulanger-Lewandowski (2014) based on note-level likelihood. Note that our train/valid/test folds differ from those used by other authors.

However, evaluation of generative models is hard (Theis et al., 2015). The gold standard for evaluation is qualitative comparison by humans, and we therefore report results of a human evaluation study. Our model is able to achieve compelling results on three tasks: conditioned rewriting, partial score completion, and unconditioned polyphonic generation.

Conditioned rewriting refers to the case where the initial piano roll $\mathbf{x}^0$ is taken from the validation set, and the mask $\mathbf{m}$ is one everywhere, and the piece is rewritten using Algorithm 2. In partial score completion, we similarly initialize the initial piano roll with an existing piece, but mask out part of it and repopulate it in a single pass of Algorithm 1. Finally, unconditioned generation starts with an empty piano roll and mask, and populates using Algorithm 2.

6.1 EVALUATING LOG-LIKELIHOOD

To estimate the log-likelihood of the data ${}^1\mathbf{x}, {}^2\mathbf{x}, \ldots, {}^n\mathbf{x}$, we uniformly choose a random ordering ${}^j o$ for each ${}^j\mathbf{x}$, and compute the log-likelihood for each data point according to

$$\log \hat{p}_\theta({}^j\mathbf{x}) = \sum_d \log p_\theta(\mathbf{x}_{{}^j o_d} | \mathbf{x}_{{}^j o_{<d}, {}^j o_{<d}}) \tag{14}$$

We repeat this procedure k times and average across all point estimates. The numbers for our models in Table 1 were obtained with $k = 10$.

6.2 HUMAN EVALUATIONS

We carried out a listening test using Amazon's MTurk to evaluate how our models compare to Bach. The natural question to answer is if music generated by our model can be indistinguishable from music composed by Bach. As the musical background of participants on MTurk is quite varied, we opt for another question, "which musical fragment do you prefer?". The goal is to assess if our model is able to generate music that could pass as music.

The study design is as follows: we compare three of our models to Bach chorales. The two models BERNOULLI(0.5) and BALANCED BY SAMPLING explores the impact of different blankout procedures, while DENOISING is a variation to the blankout procedure where the pianoroll is perturbed instead. For each of our models, we generate four samples from a blank pianoroll. For the Bach set, We randomly crop four samples from the chorale validation set. Resulting in four sets of sounds four sounds. All of the samples are two measures long, lasting twelve seconds. For each MTurk hit, two sounds are presented. These sounds are selected by first randomly choosing two sets, and then randomly choosing one sample from each set. Participants are then asked to rate which one of the two samples they prefer on a Likert scale. The study result in 192 ratings, where each model was involved in 92 pairwise comparisons. Figure 2 reports the number of times in a pairwise comparison a model/Bach was more preferred.

We performed post-hoc pairwise comparisons using Wilcoxon Signed Rank test. Bach was preferred over our balanced by sampling model. The pairs that did not show a statistically significant difference include BERNOULLI(0.5) vs Bach, DENOISING vs Bach, and BERNOULLI(0.5) vs DENOISING. This means in this setting users did not prefer fragments from Bach chorales over our two models DENOISING and BERNOULLI(0.5), and also they did not have a preference between our DENOISING and BERNOULLI(0.5) model.

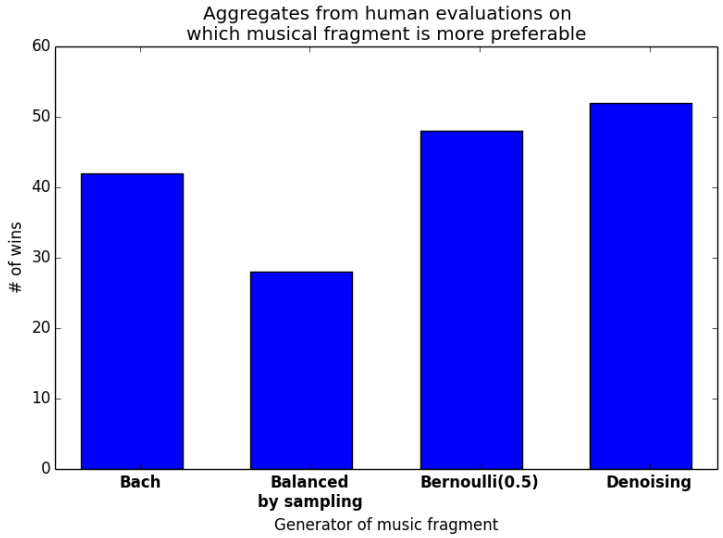

Figure 2: MTurk results from human evaluations on unconditioned generation.

7 CONCLUSION

We introduced a convolutional approach to modeling musical scores rooted in the NADE (Uria et al., 2016) framework and image inpainting Pathak et al. (2016). Participants in a user study preferred musical fragments generated by our model Coconet and those composed by Bach about equally often. We hope that this versatile model brings to the domain of music the kind of artistic exploration pioneered by Deep Dream (Mordvintsev et al., 2015).

REFERENCES

Moray Allan and Christopher KI Williams. Harmonising chorales by probabilistic inference. *Advances in neural information processing systems*, 17:25–32, 2005.

Amjad Almahairi, Nicolas Ballas, Tim Cooijmans, Yin Zheng, Hugo Larochelle, and Aaron Courville. Dynamic capacity networks. *arXiv preprint arXiv:1511.07838*, 2015.

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

Nicolas Boulanger-Lewandowski. Modeling high-dimensional audio sequences with recurrent neural networks. 2014.

Alex J. Champandard. Neural doodle, 2016. URL `https://github.com/alexjc/neural-doodle`.

Vincent Dumoulin, Johnathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

Gaëtan Hadjeres, Jason Sakellariou, and François Pachet. Style imitation and chord invention in polyphonic music with exponential families. *arXiv preprint arXiv:1609.05152*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Lamb, Vincent Dumoulin, and Aaron Courville. Discriminative regularization for generative models. *arXiv preprint arXiv:1602.03220*, 2016.

Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Feynman Liang. Bachbot: Automatic composition in style of bach chorales. *Masters thesis, University of Cambridge*, 2016.

Jun S Liu. The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 89(427):958–966, 1994.

Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks, 2015. URL `https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html`.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. *arXiv preprint arXiv:1604.07379*, 2016.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, DTIC Document, 1986.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *arXiv preprint arXiv:1605.02226*, 2016.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008.