Handwritten Chinese Character Generation via Conditional Neural Generative Models

Weirui Kong (95892162) **Bicheng Xu (94555166)**
Department of Computer Science
University of British Columbia
Vancouver, Canada V6T 1Z4
{weiruik, bichengx}@cs.ubc.ca

Abstract

Neural generative models are currently active research directions in the area of machine learning. In this work, we use three different neural generative models to generate handwritten Chinese characters conditioned on their GBK encodings[1]. Specifically, the three models we propose are a conditional convolutional generative adversarial network (cCGAN) model, a conditional convolutional variational autoencoder (cCVAE) model and a hybrid model combining both cCVAE and cCGAN.

1 Introduction

Chinese characters have been used continually for over three millennia by more than a quarter of the world's population [1]. While writing Chinese characters by hand is relatively easy and obvious for humans, it is a bit challenging for machines to generate handwritten Chinese characters directly.

Neural generative models, particularly generative adversarial network (GAN) [2] and variational autoencoder (VAE) [3], have attracted many research attentions. Strong progress has been made on these directions. Lots of works show that GAN and VAE can generate perfectly looking MNIST handwritten digits[2] [4, 5, 6], and even good-looking natural images [7, 8]. However, there is little previous work focusing on directly generating nice-looking handwritten Chinese characters conditioned on their GBK encodings via these methods.

Generating handwritten Chinese characters is more difficult than generating handwritten digits, because Chinese characters are much more complicated and structured than digits. Generating Chinese characters is also different from generating realistic-looking natural images based on texts, since we can not get the semantics of Chinese characters from their labels, i.e., their GBK encodings. Therefore, it is worthwhile to see the performance of the neural generative models generating handwritten Chinese characters conditioned on their labels.

In this work, we propose three different neural generative models to generate nice-looking handwritten Chinese characters conditioned on their labels. The three models are one GAN-based model, one VAE-based model and one hybrid model consisting of VAE and GAN.

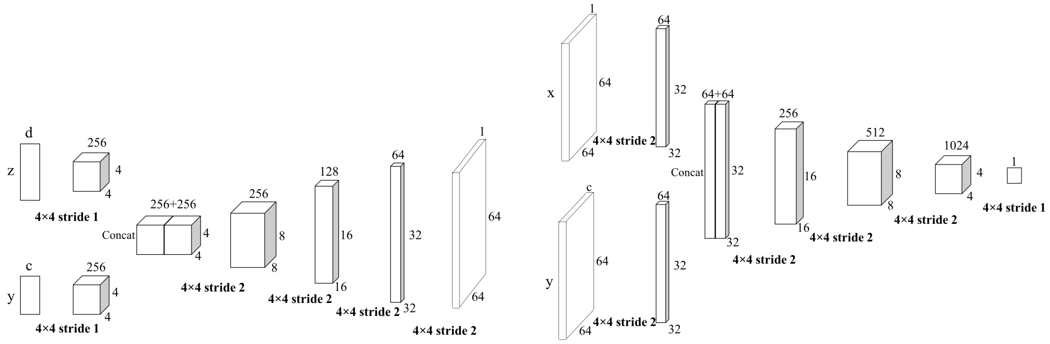
2 Related Work

Generative adversarial network (GAN). The idea of GAN is first introduced in [2], and [9] proposes a conditional version of GAN. The idea of using the condition in [9] is to combine the latent vector

[1] GBK is a type of encoding for simplified Chinese characters.

[2] http://yann.lecun.com/exdb/mnist/

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) Generator.

(b) Discriminator.

Figure 1: Our cCGAN model.

or the data sample with the condition into a large tensor, and then fit the tensor into the traditional GAN framework [2]. Both generator and discriminator utilize the condition. We share the same idea of conditioning as [9]. [10] proposes a conditional GAN based algorithm to remove rain streaks from a single image. [11] builds a conditional GAN model to generate images of outdoor scenes from attributes and semantic layouts. Their model has two conditions, while we only have one condition in our case, i.e., the label of the character to be generated.

Variational autoencoder (VAE). VAE is first defined in [3]. [4] proposes a recurrent neural network based VAE model for image generation, and [12] introduces a VAE method for text generation which incorporates convolution and deconvolution operations. However, neither the image generation model nor the text generation method has any conditions. Walker et al. [13] proposes a conditional VAE algorithm to predict the dense trajectory of pixels in a scene. In our case, we use a conditional VAE model to generate Chinese characters, that is, images.

VAE-GAN. There also exist works that combine both VAE and GAN. In [14], the authors build a voice conversion system from non-parallel speech corpora using a combined VAE-GAN model. [15] presents a VAE-GAN based method to measure similarities in data space. With this method, the authors show that they can effectively generate good-looking human faces. Wu et al. [16] build a 3D-VAE-GAN model to generate 3D objects. Compared to these works, our task is relatively easy. We want to see the effectiveness of this type of model in Chinese character generation.

3 Approach

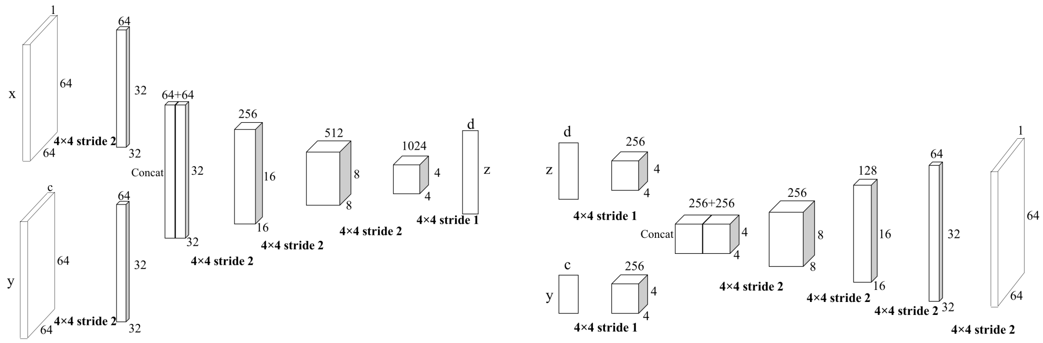
We apply three neural generative models to the task of handwritten Chinese character generation conditioned on the characters' labels. Our conditional convolutional GAN (cCGAN) model is adapted from [17]. We build our conditional convolutional VAE (cCVAE) model by turning the discriminator and generator of cCGAN into the encoder and decoder. We further combine the encoder of cCVAE with cCGAN as our third model. In this section, we first review the conditional GAN (Section 3.1) and the conditional VAE (Section 3.2) frameworks, and then introduce how to add convolution operations into these frameworks (Section 3.3). Finally, we discuss how to combine them together (Section 3.4).

3.1 Conditional generative adversarial network

GAN approach [2] consists of a generator network (G) and a discriminator network (D). G is trained to synthesize data samples resembling the training data distribution from a latent vector, and D is trained to distinguish whether or not a sample belongs to the training data. This can be viewed as a minmax two-player game. Formally, we can formulate the GAN training as optimizing

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))], \tag{1}$$

where x is a sample from the training data distribution p_{data}, and z is a random vector sampled from a known noise distribution p_z.



(a) Encoder.

(b) Decoder.

Figure 2: Our cCVAE model.

The conditional generative adversarial network [9] is an extension of GAN in which both D and G receive an additional vector of condition y as input. The conditional GAN objective is given by

$$\min_G \max_D \mathbb{E}_{x,y \sim p_{data}}[\log D(x,y)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z,y),y))]. \qquad (2)$$

In our task, x is a handwritten Chinese character, and y is the one hot representation of the character's label. z is drawn from a standard normal distribution.

3.2 Conditional variational autoencoder

VAE method [3] includes an encoder and a decoder. The encoder $q_\theta(z|x)$ embeds a data sample x into a latent representation z, and the decoder $p_\phi(x|z)$ reconstructs the data sample back based on the latent vector z. After the VAE model is trained, to generate data samples, we use the decoder and sample z from a standard normal distribution. There are two terms in the loss function when training VAE, a reconstruction loss which penalizes the dissimilarity between the real data sample and the generated one, and a KL divergence loss which penalizes the distance between the distribution $q_\theta(z|x)$ produced by the encoder and a standard normal distribution $p(z)$. Mathematically, the loss function can be written as

$$L(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x)}[\log p_\phi(x|z)] + KL(q_\theta(z|x)||p(z)). \qquad (3)$$

We often choose $q_\theta(z|x)$ to be a Gaussian because we have a closed-form solution for the KL divergence between two Gaussians. Given condition y, the loss function for a conditional VAE is given by

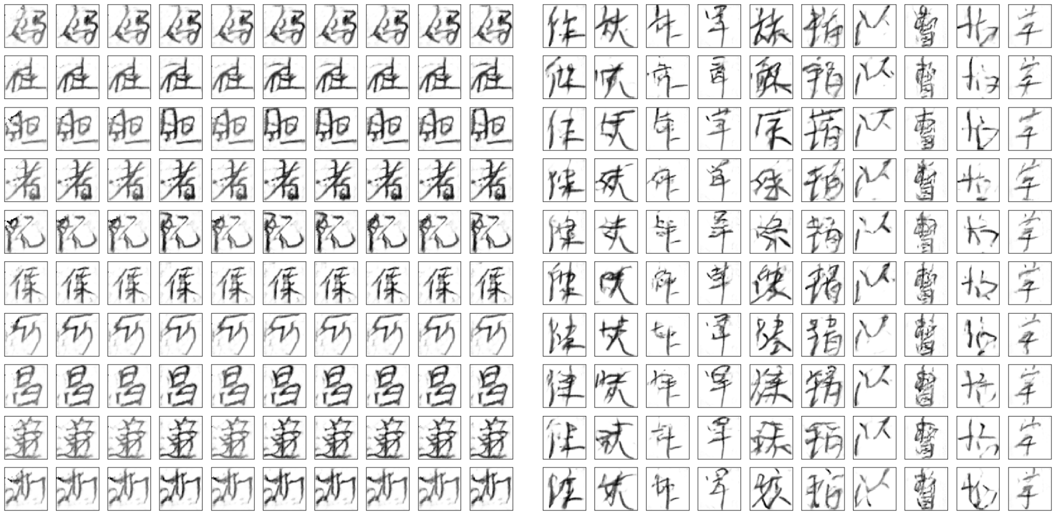
$$L(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x)}[\log p_\phi(x|z,y)] + KL(q_\theta(z|x,y)||p(z)). \qquad (4)$$

3.3 Conditional convolutional GAN and conditional convolutional VAE

The deep convolutional GAN model [17] removes all the fully connected layers in the traditional GAN model [2]. Instead, its generator consists of multiple transposed convolutional layers, and its discriminator contains multiple convolutional layers. Fig. 1 shows the structure of the conditional convolutional GAN (cCGAN) model used for our task, where c is the dimension of the label vector y and d is the dimension of the latent vector z. We use the same network structure for our conditional convolutional VAE (cCVAE) model. Fig. 2a and Fig. 2b shows the structures of the encoder and decoder of the cCVAE used for our task.

3.4 Combine cCVAE and cCGAN

While the generator in a GAN model learns to map from a latent space to the data sample space, it does not exploit the semantics of the latent space during training (the latent vector z is sampled from a standard Gaussian). We can add an encoder to the cCGAN model so as to make use of such semantics. The combined cCVAE-cCGAN model is straightforward, when training the model, we



(a) Sample results of the cCGAN model trained with 10 labels.

(b) Sample results of the cCGAN model trained with 100 labels.

Figure 3: Generated handwritten Chinese characters by the cCGAN model. The ground-truths of (a) are “妈", “硅", “胆", “者", “阮", “傈", “何", “昌", “递", “浙" . The ground-truths of (b) are “撅", “掌", “隅", “萤", “笑", “胆", “忙", “们", “吉", “羚".

first use the encoder in cCVAE to generate the latent vector z, and then we feed this encoded z into the cCGAN model. The combined cCVAE-cCGAN model is trained end-to-end. During test, we only use the generator of this model.

4 Experiments

4.1 Dataset

We build our dataset based on the CASIA offline Chinese handwriting databases[3]. We use the CASIA-HWDB1.1 database. This database contains about one million scanned handwritten characters of 3755 different Chinese characters.

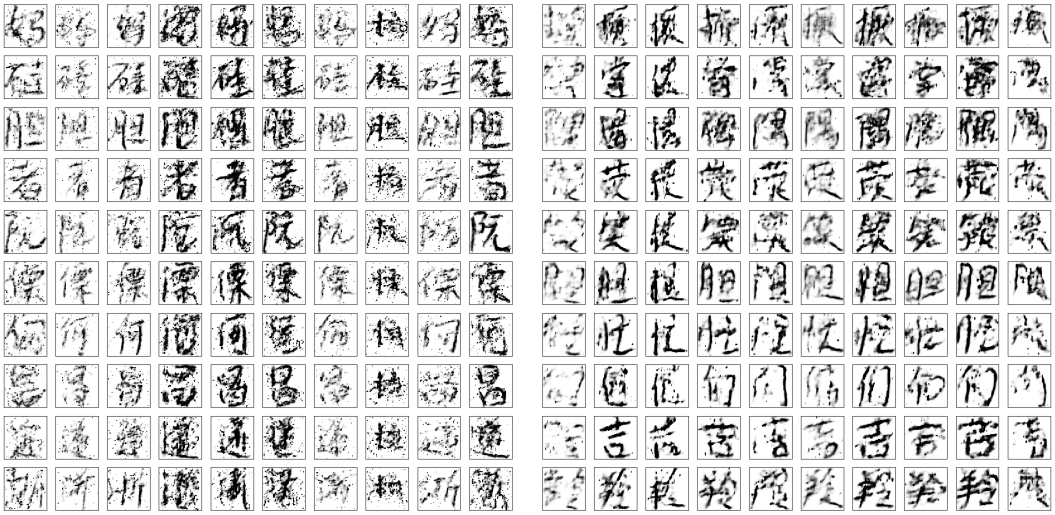
To control the complexity of our task, we randomly pick 10 different Chinese characters with 2048 training samples (around 200 samples per Chinese character) to train our model. Each character is a 128×128 gray scale image. We preprocess the characters by reshaping them to 64×64 and normalizing the intensities. We further extend our models to generate 100 different Chinese characters in Section 4.5.

4.2 Evaluate the cCGAN model

In our implementation, we set the dimension of the latent vector z to 10. Since we have 10 labels, the label size c is also 10. Note that in Fig. 1b, the label y is a $64 \times 64 \times c$ tensor. It can be interpreted as the one hot encoding in 3D. We use a batch size of 256 and Adam optimizer to train the model. The initial learning rates for both generator and discriminator are 0.0002.

To evaluate the performance, we generate a 10×10 grid of characters. Each row of the grid shares the same conditioning label and each column has the same latent vector. That is, the 10 rows represent the 10 labels, and the 10 columns reflect how different latent vectors affect the style of the generated characters. Fig. 3a shows one such grid generated by our cCGAN model. It manages to learn the structure of each character and the generated characters are relatively sharp. Focusing on each row, one observation is that the styles of the generated characters do not change much over the columns despite we have different latent vectors for different columns.

[3] http://www.nlpr.ia.ac.cn/databases/handwriting/Offline_database.html



(a) Sample results of the cCVAE model trained with 10 labels.

(b) Sample results of the cCVAE model trained with 100 labels.

Figure 4: Generated handwritten Chinese characters by the cCVAE model. The ground-truths of (a) are “妈", “硅", “胆", “者", “阮", “僳", “何", “昌", “递", “浙" . The ground-truths of (b) are “撅", “掌", “隅", “萤", “笑", “胆", “忙", “们", “吉", “羚".

4.3 Evaluate the cCVAE model

For our cCVAE model, the dimension of the latent vector z is also 10. We still use a batch size of 256 with Adam optimizer to train the model. The initial learning rates for both encoder and decoder is 0.001. Note that the decoder here has the same structure as the generator in our cCGAN model, and the encoder here has the same structure as the discriminator of our cCGAN model except that the output of the encoder is a vector of size 10, rather than a saclar.

Fig. 4a shows one grid of characters generated by the cCVAE model. The model also learns the structure of each label very well. Compared with the results of cCGAN, cCVAE can generate characters in diverse styles. Given different latent vectors conditioned on the same label, some of the generated characters are like children’s writing whereas some look like written by people with years of experience. Although cCVAE preserves the diversity of the generated characters, overall the generated characters are a bit blurry. This is a well-known problem for VAE based models.

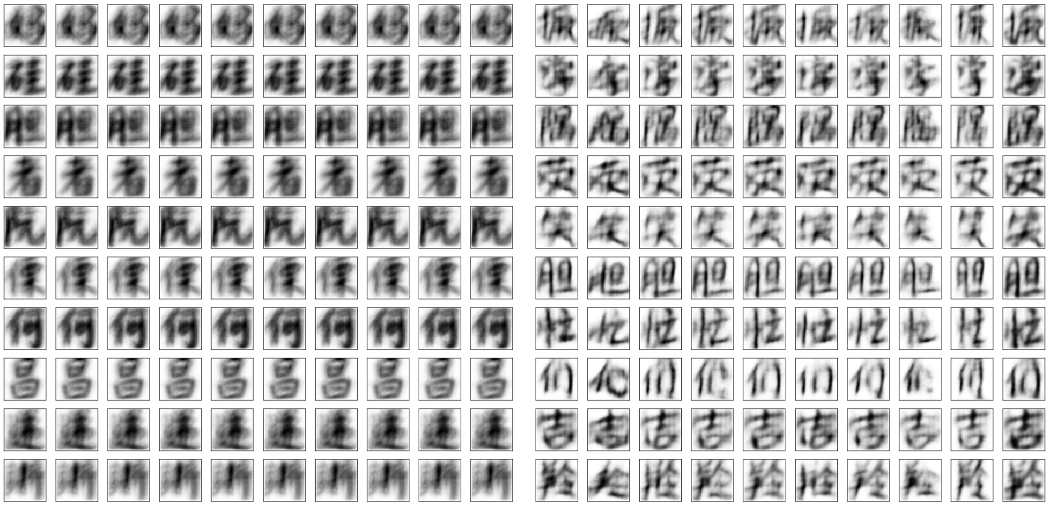
4.4 Evaluating the combined cCVAE-cCGAN model

When training the combined cCVAE-cCGAN model, we use the same settings as the previous two models, i.e., the same initial learning rates, batch size and optimizer. However, after training the combined model for about 8 epochs, the discriminator loss drops to zero. What is worse, the computation for the KL divergence loss becomes numerical unstable: we always encounter values with -inf after training about 8 epochs. Even though, the generated characters from the first few epochs still look good. Fig. 5a shows one grid of generated characters. Despite the blurriness, the generated characters are pretty good looking. It feels like zooming in a low resolution character written by a calligrapher.

4.5 Extending the number of labels to 100

Success in generating characters of 10 labels, we try our three models to generate 100 different characters. That is, we extend the number of labels from 10 to 100. We randomly select 100 different characters from the CASIA-HWDB1.1 database with 20921 data samples to train our models.

All the training settings here are the same as training the 10 labels described above except that the dimension of y is 100. The dimension of the latent vector z is still 10. We randomly pick 10 different characters from the 100 labels to display the results. Fig. 3b and Fig. 4b show the results



(a) Sample results of the cCVAE-cCGAN model trained with 10 labels.

(b) Sample results of the cCVAE-cCGAN model trained with 100 labels.

Figure 5: Generated handwritten Chinese characters by the combined cCVAE-cCGAN model. The ground-truths of (a) are “妈", “硅", “胆", “者", “阮", “傈", “何", “昌", “递", “浙" . The ground-truths of (b) are “撅", “掌", “隅", “萤", “笑", “胆", “忙", “们", “吉", “羚".

of the cCGAN model and cCVAE model respectively. Though a bit ugly, cCVAE can still generate characters for different labels successfully. On the contrary, cCGAN suffers from mode collapse. It almost generates the same thing for different labels.

In terms of the combined cCVAE-cCGAN model, again we encounter the numerical issue and can not train the model for many epochs. But as Fig. 5b shows, the results after training a few epochs are still encouraging. It does not have the mode collapse problem and looks more like human writing than the cCVAE model.

5 Discussion and Future Work

For the task with small label size, generating 10 different characters in our experiments, both cCGAN and cCVAE models generate well-structured characters. cCGAN produces sharp characters while cCVAE preserves the diversity. As the label size becomes larger, cCGAN has the issue of mode collapse.

Combining cCVAE and cCGAN together is a straightforward idea for better performance. But it has two problems, the instability in training the cCGAN model and the numerical issue in computing the KL divergence loss. Future work should be done to solve these problems and extend our models to generate handwritten Chinese characters of larger label sets.

To avoid the discriminator loss from getting to zero, we can perform more than once gradient descent on the generator, for every time performing gradient descent on the discriminator. This will make the generator have a good chance to always catch up to the discriminator. Besides, we can set some thresholds on the generator loss and the discriminator loss when training the discriminator. That is, we only update the weights of the discriminator when the generator loss is below some upper bound and the discriminator loss is greater than some lower bound. In this way the discriminator is not too strong towards the generator and the generator is not too weak against the discriminator. For the numerical instability issue, we believe that tuning the hyper-parameters such as the initial learning rates and the latent vector dimension, and using some clamping tricks will help.

References

[1] Insup Taylor and M Martin Taylor. *Writing and Literacy in Chinese, Korean and Japanese: Revised Edition*, volume 14. John Benjamins Publishing Company, 2014. 1

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1, 2, 3

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 2, 3

[4] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015. 1, 2

[5] SM Eslami, N Heess, and T Weber. Attend, infer, repeat: Fast scene understanding with generative models. arxiv preprint arxiv:..., 2016. *URL http://arxiv. org/abs/1603.08575*, 2016. 1

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016. 1

[7] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE Int. Conf. Comput. Vision (ICCV)*, pages 5907–5915, 2017. 1

[8] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. *arXiv preprint arXiv:1801.05091*, 2018. 1

[9] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1, 2, 3

[10] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016. 2

[11] Levent Karacan, Zeynep Akata, Aykut Erdem, and Erkut Erdem. Learning to generate images of outdoor scenes from attributes and semantic layouts. *arXiv preprint arXiv:1612.00215*, 2016. 2

[12] Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. A hybrid convolutional variational autoencoder for text generation. *arXiv preprint arXiv:1702.02390*, 2017. 2

[13] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851. Springer, 2016. 2

[14] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang. Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks. *arXiv preprint arXiv:1704.00849*, 2017. 2

[15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015. 2

[16] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 2

[17] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2, 3