

# Multimodal Controller for Generative Models

Enmao Diao  
Duke University  
Durham, NC, USA  
enmao.diao@duke.edu

Jie Ding  
University of Minnesota  
Minneapolis, MN, USA  
dingj@umn.edu

Vahid Tarokh  
Duke University  
Durham, NC, USA  
vahid.tarokh@duke.edu

## Abstract

*Class-conditional generative models are crucial tools for data generation from user-specified class labels. Existing approaches for class-conditional generative models require nontrivial modifications of backbone generative architectures to model conditional information fed into the model. This paper introduces a plug-and-play module named ‘multimodal controller’ to generate multimodal data without introducing additional learning parameters. In the absence of the controllers, our model reduces to non-conditional generative models. We test the efficacy of multimodal controllers on CIFAR10, COIL100, and Omniglot benchmark datasets. We demonstrate that multimodal controlled generative models (including VAE, PixelCNN, Glow, and GAN) can generate class-conditional images of significantly better quality when compared with the state-of-the-art conditional generative models. Moreover, we show that multimodal controlled models can also create novel modalities of images.*

## 1. Introduction

In recent years, many generative models based on neural networks have been proposed and achieved remarkable performance. The main backbones of generative models include Autoencoder, Autoregression, Normalization Flow, and Adversarial generative models. Perhaps the most well-known representatives of them are Variational Autoencoder (VAE) [15], PixelCNN [27], Glow [14], and Generative Adversarial Network (GAN) [9], respectively. VAE learns a parametric distribution over an encoded latent space, samples from this distribution, and then constructs generations from decoded samples. PixelCNN uses autoregressive connections to factorize the joint image distribution as a product of conditionals over sub-pixels. Glow optimizes the exact log-likelihood of the data with a deterministic and invertible transformation. GAN was introduced as a generative framework where intractable probabilistic distributions are approximated through adversarial training.

In many application scenarios, we are interested in constructing generations based on a conditional distribution. For instance, we may be interested in generating human face images conditional on some given characteristics of faces such as hair color, eye size, gender, etc. A systematic way to incorporate conditional information may enable us to control the data generating process with more flexibility. In this direction, conditional generative models including Conditional Variational Autoencoder (CVAE) [30], Conditional Generative Adversarial Network (CGAN) [22], and Conditional PixelCNN (CPixelCNN) [34] have been proposed which model conditional information by learning the associated embeddings. The learned features are usually concatenated or added with non-conditional features at various network layers. Conditional Glow (CGlow) learns a class-conditional prior distribution and an optional auxiliary classifier.

In this paper, we propose a plug-and-play module named Multimodal Controller (MC) to allocate uniformly sampled subnetwork for each mode of data. Instead of introducing additional learning parameters to model conditional information, multimodal controlled generative models generate each mode of data from its corresponding unique subnetwork. Our main contributions of this work are three-fold.

- We introduce a novel method to transform non-conditional generative models into class-conditional generative models, by simply attaching a multimodal controller at each layer. Unlike existing methods, our method does not introduce additional learning parameters, and it can be easily incorporated into existing implementations.
- We empirically demonstrate that our method outperforms various well-known conditional generative models for datasets with small intra-class variation and a large number of data modalities. We achieve this advantage by allocating specialized subnetworks for corresponding data modalities.
- We show that multimodal controlled generative models can create novel data modalities by allocating un-

trained subnetworks from genetic crossover or resampling of a codebook.

We experiment with CIFAR10, COIL100, and Omniglot datasets [16, 25, 17]. We compare our method with the conditional generative models with different backbone generative models such as VAE, PixelCNN, Glow, and GAN.

The rest of the paper is organized as follows. In Section 2, we review the related work. In Section 3, we introduce our proposed multimodal controller. In Section 4, we provide experimental results demonstrating the performance of our approach. Finally, we make our concluding remarks in Section 5.

## 2. Related Work

We use four distinct backbone generative models including Variational Autoencoder (VAE) [15], PixelCNN [27], Glow [14] and Generative Adversarial Network (GAN) [9] to demonstrate general compatibility of our method. VAE is a directed generative model with probabilistic latent variables. PixelCNN is an autoregressive generative model that fully factorizes the joint probability density function of images into a product of conditional distributions over all subpixels. Glow is a flow-based generative model that enables exact and tractable log-likelihood and latent space inference by ensuring a bijective neural network mapping. GAN consists of a Generator ( $G$ ) network and a Discriminator ( $D$ ) network that trains to find a Nash equilibrium for generating realistic-looking images.

**Conditional generative models** treat class-conditional information  $h$  as input to the model. Typically, modeling such  $h$  requires additional learning parameters and the resulting objective is the conditional distribution  $p_\theta(x | h)$ . Conditional VAE (CVAE) [30] and Conditional GAN (CGAN) [22] concatenate trainable embeddings to both encoder (discriminator) and decoder (generator). Apart from trainable embeddings, this approach also requires additional learning parameters on the backbone generative models to incorporate the embeddings. Instead of concatenating, Conditional PixelCNN (CPixelCNN) [27] and Conditional Glow (CGlow) [14] adds trainable embeddings to the features. This method requires the size of embedding to match the channel size of features. There also exist many other ways of modeling  $h$ . ACGAN [26] introduces an auxiliary classifier with a class-conditional objective function to model  $h$ . The Conditional Normalization [6, 3] learns class-conditional affine parameters, which can be considered another way to incorporate embeddings. [24] proposed a projection discriminator to mode  $h$  by measuring the cosine similarity between features and a learned embedding. A hybrid approach that combines the previous two methods was used in BigGAN [1]. Recently, StyleGAN [13] enables scale-specific synthesis by transforming style information

with fully connected layers into affine parameters for conditional normalization. MSGAN [20] models  $h$  to generate a mode-seeking regularization term. STGAN [18] enables image attribute editing by modeling the difference between the source and target  $h$ .

We aim to introduce a novel and generic alternative method to generate data class-conditionally. Intuitively, training separate models for each mode of data can guarantee class-conditional data generation. However, we do not want the model complexity to grow with the number of data modalities.

Recently, a few works pay attention to subnetworks. The lottery ticket hypothesis [8, 39] states that there may exist a subnetwork that can match the performance of the original network when trained in isolation. PathNet [7] demonstrates successful transfer learning through migration from one subnetwork to the other. Piggyback and some other works in continual learning literature also adopt similar methods to train subnetworks for various tasks of data [19, 28, 35].

**Multimodal controlled generative models** uniformly sample subnetworks from a backbone generative model and allocate a unique computational path for each data modality. Our method is different from the existing weight masking approach [19, 28, 35] because our mask is applied to the network representations of networks rather than the model parameters. Specifically, our approach is able to train data from multiple data modalities in the same batch of data simultaneously. However, existing weight masking methods can only optimize one data modality at a time. This paper empirically justifies the following claim. Uniformly sampled subnetworks through masking network representations can well-represent a substantial number of data modalities in class-conditional generative models by allocating a unique computational path for each mode of data.

## 3. Multimodal Controller

Suppose that there is a dataset  $X$  with  $C$  data modalities. Each mode of data  $X_c = \{x_c^i\}_{i=1}^{N_c}$  consists of  $N_c$  i.i.d. samples of a (continuous or discrete) random variable. Given a set of learning model parameters  $\theta \in \mathbb{R}^D$  with size  $D$ , each mode of data is modeled with a random subset  $\theta_c \subset \theta$ . For notational convenience, we will interchangeably use the notions of subset and subvector. In this way, the allocated parameters for each mode will represent both the inter-mode association and intra-mode variation, thanks to parameter sharing and specialization.

Next, we discuss technical details in the specific context of neural networks. Suppose a uniformly sampled subnetwork takes input  $X_c \in \mathbb{R}^{N_c \times K_c}$ , where  $N_c$  and  $K_c$  are the batch size and the input channel size for data mode  $c$ . Suppose that the subnetwork is parameterized by a weight matrix  $W_c \in \mathbb{R}^{D_c \times K_c}$  and bias vector  $b_c \in \mathbb{R}^{D_c}$ , where  $D_c$  is

the output channel size for data mode  $c$ , then we have output  $y_c \in \mathbb{R}^{N_c \times D_c}$  where

$$y_c = \phi(\text{BN}_c(X_c \times W_c^T + b_c))$$

$\text{BN}_c(\cdot)$  denotes the Batch Normalization (BN) [12] with affine parameters for corresponding data mode  $c$  and  $\phi(\cdot)$  is the activation function.

Existing methods [19] allocate subnetworks by masking out model parameters, i.e.  $W_c = W \odot e_c$  where  $e_c$  is a binary mask and  $\odot$  indicates Hadamard product. However, the computation cost of the above formulation increases with the number of data modalities, as we need to sequentially backward the subnetwork of each data modality. Therefore we propose a nonparametric module named **Multimodal Controller (MC)** using a masking method similar to Dropout [31], which allocates subnetworks by masking out network representations. We choose to uniformly sample the codebook because we want to create a large number of unique subnetworks. The number of unique subnetworks we can generate with MC is  $2^D$ . We do not train the codebook, because it is possible for some data modalities to have the same binary masks.

We uniformly draw  $C$  unique modality codewords  $e_c \in \mathbb{F}_2^D$  to construct a modality codebook  $e \in \mathbb{F}_2^{C \times D}$ , where  $\mathbb{F}_2$  denotes the binary field. Note that each row of the codebook is a binary mask allocated for each mode of data. Let  $\times$  denote the usual matrix multiplication. Let  $X \in \mathbb{R}^{N \times K}$  denote the output from the previous layer. Suppose that the original network is parameterized by a weight matrix  $W \in \mathbb{R}^{D \times K}$  and bias vector  $b \in \mathbb{R}^D$ . Then for a specific mode of data  $c$ , we have multimodal controlled output  $y \in \mathbb{R}^{N \times D}$  where

$$\begin{aligned} \hat{W}_c &= W \odot e_c, \quad \hat{b}_c^D = b \odot e_c, \\ \hat{\text{BN}}_c &= \text{BN} \odot e_c, \quad \hat{\phi}_c(\cdot) = \phi(\cdot) \odot e_c \\ y &= \hat{\phi}_c(\hat{\text{BN}}_c(X \times \hat{W}_c^T + \hat{b}_c)) \\ &= \phi(\text{BN}(X \times W^T + b)) \odot e_c \end{aligned}$$

Note that  $\hat{W}_c$ ,  $\hat{b}_c^D$ ,  $\hat{\text{BN}}_c$  and  $\hat{\phi}_c$  are uniformly masked from original network. Because we are masking out the network representations, we can factorize  $e_c$  to avoid interfering with the calculation of running statistics and activation function because  $(e_c)^n = e_c$ . Suppose we have class-conditional information  $h \in \mathbb{F}_2^{N \times C}$  where each row is a one-hot vector. The multimodal controller can optimize all data modalities of one batch of data in parallel as follows

$$y = \phi(\text{BN}(X \times W^T + b)) \odot (h \times e).$$

Suppose that the above formulation is for an intermediate layer. Then  $X$  is the output masked from the multimodal controller in the previous layer. Denote  $\tilde{e}_c \in \mathbb{F}_2^K$  as

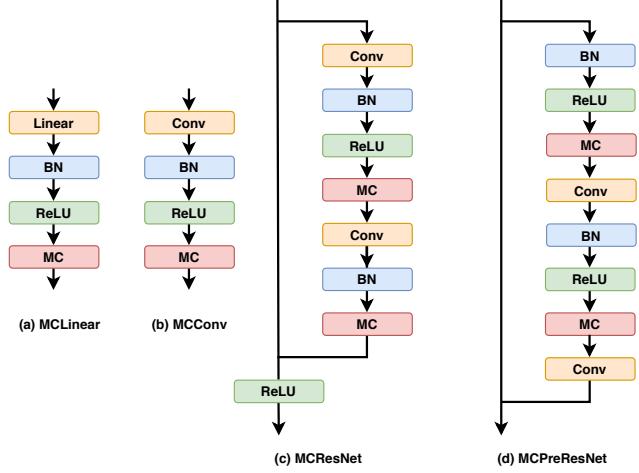


Figure 1: Multimodal Controlled Neural Networks.

the codeword for data mode  $c$  in previous multimodal controller, then the effective subnetwork weight matrix  $\tilde{W}$  is  $W \odot (e_c \times \tilde{e}_c)$ . Since each codeword is uniformly sampled, the size of  $\tilde{W}$  is approximately  $\frac{1}{4}$  the original  $W$ . Indeed, the multimodal controller is a nonparametric computation module modulating model architecture and trading memory space for speedup in runtime. Although the above formulation only describes the interaction of multimodal controller and linear network. It can be readily extended to other parametric modules, such as convolution layers. We demonstrate the suggested way of attaching this plug-and-play module in practice in Figure 1.

## 4. Experiments

This section demonstrates applications of multimodal controlled generative models to data generation and creation. We compare the result of our proposed method with that of conditional generative models. We illustrate our results for four different types of multimodal controlled generative models, including VAE, PixelCNN, Glow, and GAN on CIFAR10, COIL100, and Omniglot datasets [16, 25, 17]. Due to our proposed method's random nature, we conduct 12 random experiments for each generative model on each dataset. The standard errors in parentheses show that uniformly sampled subnetworks are robust enough to produce stable results for either a small or large number of data modalities. Details regarding the experimental settings and network architecture for comparisons are described in the supplementary document.

Evaluating generative models is generally challenging [33]. Throughout our experiments, we use the Inception Score (IS) [29] and Fréchet Inception Distance (FID) [11] in order to approximately measure the quality of generated samples and to compare with the baseline. We also provide

	CIFAR10	COIL100	Omniglot
CVAE	0.6 (0.00)	0.4 (0.00)	0.1 (0.00)
MCVAE	0.6 (0.00)	0.4 (0.00)	0.1 (0.00)
CPixelCNN	2.8 (0.11)	0.1 (0.00)	0.1 (0.05)
MCPixelCNN	3.0 (0.12)	0.1 (0.01)	0.5 (0.15)
CGlow	3.3 (0.00)	1.6 (0.03)	0.8 (0.01)
MCGlow	3.4 (0.01)	1.5 (0.02)	0.9 (0.01)

Table 1: Negative Log-Likelihood (NLL) of multimodal controlled and conditional VAE, PixelCNN and Glow.

	CIFAR10	COIL100	Omniglot
CVAE	7.8	7.8	7.9
MCVAE	7.6	7.6	7.6
CPixelCNN	6.4	6.8	12.6
MCPixelCNN	6.4	6.4	6.4
CGlow	22.0	22.0	15.8
MCGlow	22.0	22.0	15.7
CGAN	5.5	8.7	8.8
MCGAN	5.3	8.4	8.4

Table 2: Number of learning parameters used in generative models in M (millions). Half of parameters of CPixelCNN (Omniglot) are used for learning conditional embeddings which increase with the number of data modalities.

likelihood measurements for VAE, PixelCNN, and Glow in Table 1. The results show that multimodal controlled generative models perform comparably to conditional generative models in likelihood estimation.

#### 4.1. Generation

In this section, we present quantitative and qualitative results of generations from conditional and multimodal controlled generative models. More results are included in the supplementary document. Based on our results, multimodal controlled generative models can generate samples of comparable or better fidelity and diversity compared with conditional counterparts. We report our quantitative results in Table 3 with Inception Score (IS) [29] and Fréchet Inception Distance (FID) [11] which are perhaps the two most common metrics for comparing generative models.

Both conditional and multimodal controlled generative models share the same backbone generative models for a fair comparison. Therefore, it is unrealistic to expect multimodal controlled generative models to outperform their conditional counterparts in all cases. It is possible for conditional generative models to introduce more learning parameters for incorporating embeddings and thus produce better results. Most generative models in our experiments have

similar model complexity as shown in Table 2.

**Quantitative** results show that multimodal controlled generative models perform comparably with conditional generative models for the CIFAR10 dataset, which has sufficient shots for a small number of data modalities. On the other hand, our method performs considerably better than the conditional generative models for COIL100 and Omniglot datasets. The major difference among them is that CIFAR10 has more intra-class variation but less number of data modalities than COIL100 and Omniglot do. Note that uniformly sampled subnetworks are using approximately a quarter of the number of learning parameters as the original network. Therefore, it is difficult for a small subnetwork to learn one mode of data with a high intra-class variation. When intra-class variation is small, and the number of data modalities is large as in COIL100 and Omniglot datasets, our proposed method has a significant advantage because we can specialize subnetworks for their corresponding data modalities by allocating unique computational paths. It is worth mentioning that CPixelCNN for Omniglot outperforms MCPixelCNN because it has almost twice the learning parameters as the later, as shown in Table 2. The additional parameters are used for learning conditional embeddings, which increase with the number of data modalities.

**Qualitative** results regarding Multimodal Controlled GAN (MCGAN) and Conditional GAN (CGAN) for CIFAR10, COIL100, and Omniglot datasets are shown in Figure 3, 4, and 5 respectively. For CIFAR10 datasets, MCGAN and CGAN generate similar qualitative results. For the COIL100 dataset, the rotation of different objects can be readily recognized by MCGAN. However, CGAN fails to generate objects of diverse rotations, as shown in red lines. For the Omniglot dataset, the intra-class variation and inter-class distinction can also be identified by MCGAN. However, CGAN fails to generate some data modalities, as shown in red lines.

We show the learning curves of one of the experiments of MCGAN and CGAN for COIL100 and Omniglot datasets in Figure 2. The learning curves also show that MCGAN consistently outperforms CGAN. Both our quantitative and qualitative results demonstrate the efficacy and advantage of our proposed multimodal controller for datasets with small intra-class variation and a large number of data modalities.

#### 4.2. Creation

In this section, we provide quantitative and qualitative results of data creation from conditional and multimodal controlled generative models. We show that multimodal controlled generative models can class-conditionally synthesize from a novel data modality not prescribed in the training dataset. A common way to create novel data modalities for conditional generative models is to interpolate a convex combination between a pre-trained source and tar-

	CIFAR10		COIL100		Omniglot	
	IS	FID	IS	FID	IS	FID
CVAE	3.4 (0.07)	133.7 (3.13)	89.4 (1.94)	37.6 (2.30)	539.8 (21.61)	367.5 (14.16)
MCAVE	<b>3.4 (0.05)</b>	<b>128.6 (1.32)</b>	<b>95.2 (0.50)</b>	<b>29.5 (1.91)</b>	<b>889.3 (16.94)</b>	<b>328.5 (9.82)</b>
CPixelCNN	<b>5.1 (0.06)</b>	<b>70.8 (1.78)</b>	94.2 (0.68)	8.1 (0.51)	<b>1048.5 (160.33)</b>	<b>23.4 (5.38)</b>
MCPixelCNN	4.8 (0.04)	75.2 (1.60)	<b>98.1 (0.75)</b>	<b>4.7 (0.69)</b>	762.4 (86.07)	43.3 (6.20)
CGlow	4.4 (0.05)	<b>63.9 (1.34)</b>	78.3 (2.25)	<b>35.1 (2.67)</b>	616.9 (6.71)	<b>40.8 (1.28)</b>
MCGlow	<b>4.8 (0.05)</b>	65.2 (1.21)	<b>89.6 (1.63)</b>	42.0 (5.19)	<b>998.5 (29.10)</b>	47.2 (2.69)
CGAN	<b>8.0 (0.10)</b>	<b>18.1 (0.75)</b>	97.9 (0.78)	24.4 (9.86)	677.3 (40.07)	51.0 (12.65)
MCGAN	7.9 (0.13)	21.4 (0.83)	<b>98.8 (0.09)</b>	<b>7.8 (0.44)</b>	<b>1288.8 (5.38)</b>	<b>23.9 (0.73)</b>

Table 3: Inception Score (IS) and Fréchet Inception Distance (FID) for generative models.

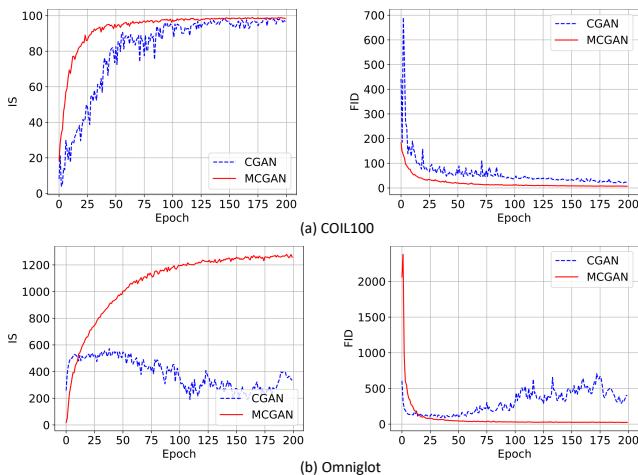


Figure 2: Learning curves of CGAN and MCGAN for COIL100 and Omniglot datasets.

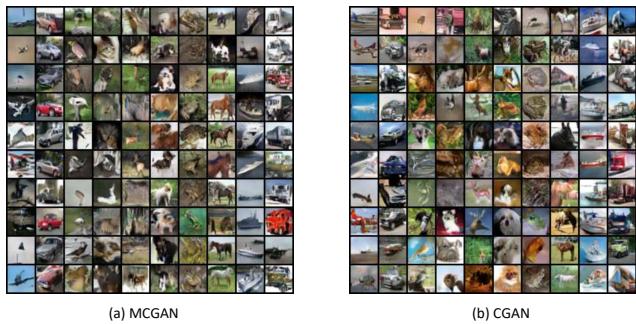


Figure 3: (a) MCGAN (b) CGAN trained with CIFAR10 dataset. Generations in each column are from one data modality.

get embeddings. However, multimodal controlled generative models do not contain learned embeddings for interpolation. Because the subnetwork architecture can be solely

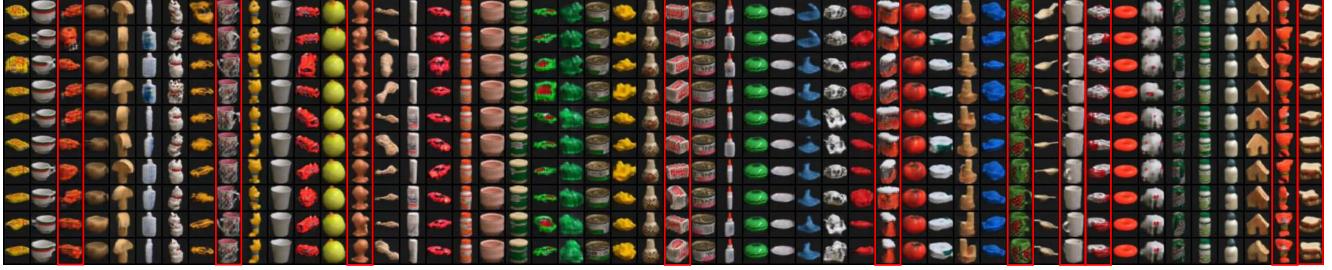
	CIFAR10	COIL100	Omniglot
Raw dataset	12.0	2.7	5.4
CVAE	37.9 (5.10)	15.5 (0.60)	8.5 (0.04)
MCAVE	<b>2.1 (0.17)</b>	<b>1.8 (0.06)</b>	<b>3.0 (0.03)</b>
CPixelCNN	27.6 (1.84)	17.9 (0.59)	9.0 (0.19)
MCPixelCNN	<b>3.8 (0.34)</b>	<b>4.8 (0.18)</b>	<b>4.6 (0.19)</b>
CGlow	40.3 (5.09)	14.3 (0.87)	8.0 (0.02)
MCGlow	<b>5.4 (0.46)</b>	<b>2.8 (0.12)</b>	<b>5.1 (0.14)</b>
CGAN	33.2 (3.46)	10.4 (2.88)	7.8 (0.08)
MCGAN	<b>2.0 (0.30)</b>	<b>1.5 (0.07)</b>	<b>3.6 (0.02)</b>

Table 4: Davies-Bouldin Index (DBI) for conditional and multimodal controlled generative models on the raw and created datasets. The created dataset has the same number of modalities as the raw dataset. Small DBI values indicate that data creations are closely clustered on novel data modality.

represented by their corresponding codewords, we can produce transitions between two data modalities by using genetic crossover between the source  $e_s$  and target  $e_t$  codeword, as illustrated in Figure 6.

We also propose an unbiased way of creating novel data modalities for our proposed method. Because the pre-trained codewords are uniformly sampled binary masks, we can naturally create unbiased novel data modalities by uniformly resampling the codebooks of multimodal controllers plugged in each layer. To compare with conditional generative models, we uniformly create new data modalities for conditional generative models by sampling the weights of a convex combination of pre-trained embeddings from Dirichlet(1) distribution.

**Quantitative** results are shown in Table 4. We evaluate the quality of uniform data creation with Davies-Bouldin Index (DBI) [2]. Small DBI values indicate that data creations are closely clustered on novel data modalities. Be-

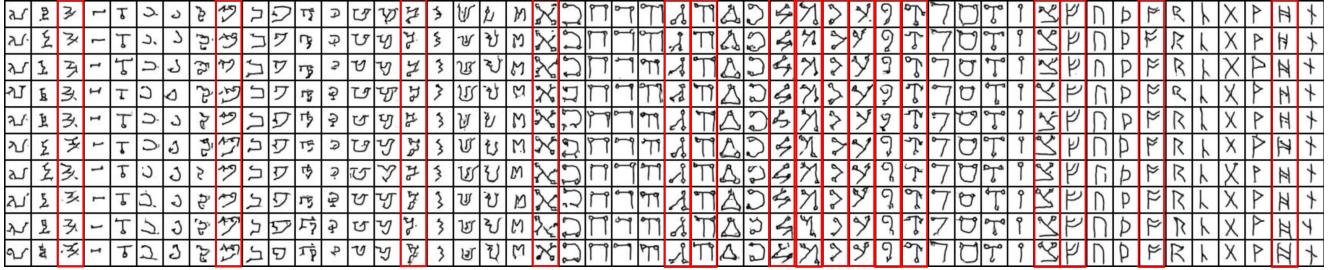


(a) MCGAN

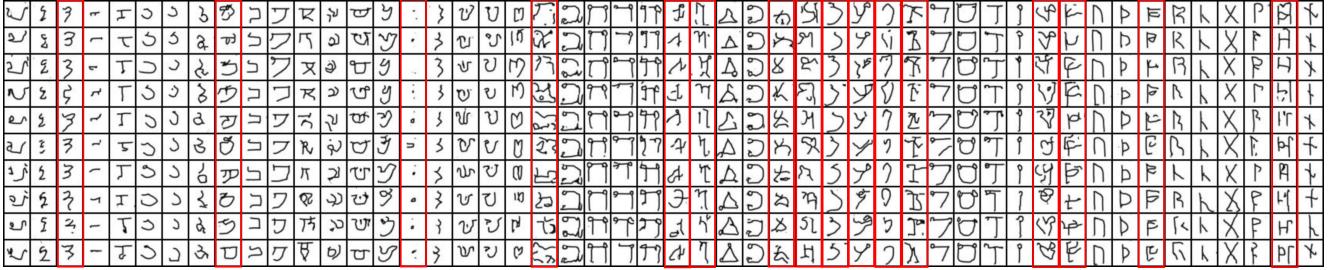


(b) CGAN

Figure 4: (a) MCGAN (b) CGAN trained with COIL100 dataset. Generations in each column are from one data modality.



(a) MCGAN



(b) CGAN

Figure 5: (a) MCGAN (b) CGAN trained with Omniglot dataset. Generations in each column are from one data modality.

cause novel data modalities are parameterized by resampled subnetworks, data created by our proposed method can be closely clustered together. Data created by conditional generative models are not closely clustered together and have much higher DBI. It shows that a random convex combination of embeddings is not enough to create unbiased novel

data modalities from pre-trained class-conditional generative models.

**Qualitative** results are shown in Figure 7, 8 and 9. The results show that interpolation between two selected pre-trained data modalities is achievable with both methods. However, subnetworks can create unbiased novel data

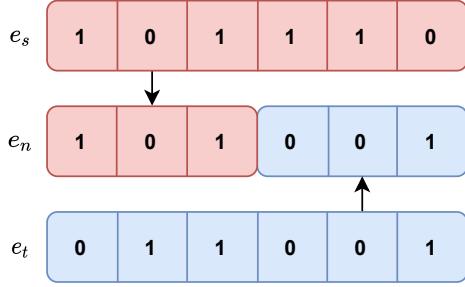


Figure 6: Illustration of genetic crossover between source  $e_s$  and target  $e_t$  codewords to create novel codeword  $e_n$ .

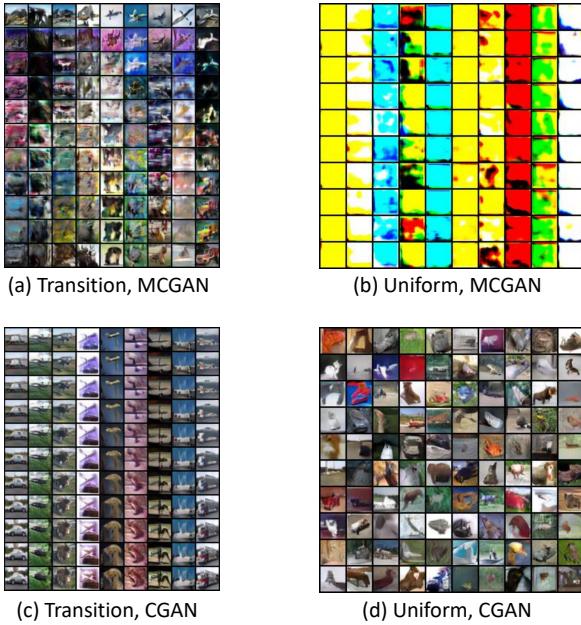


Figure 7: (a,b) MCGAN and (c,d) CGAN trained with CIFAR10 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.

modalities that have never been trained before. For the CIFAR10 dataset, although the created data from MCGAN are closely clustered, they are not structurally similar to the original training data. A possible reason is that a small number of data modalities is insufficient to learn adequate subnetworks variations. For COIL100 and Omniglot datasets, our proposed method can create novel data modalities with high fidelity and diversity because both datasets have a large number of data modalities. In particular, the learning parameters of those resampled subnetworks have been sufficiently exploited by a large number of pre-trained subnetworks.

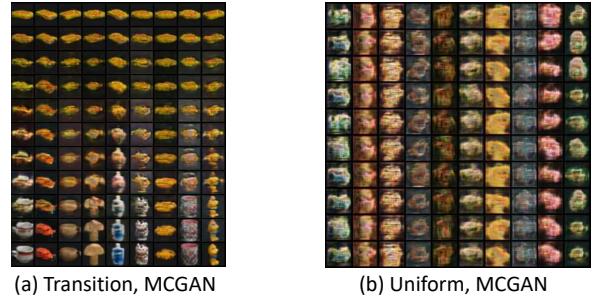


Figure 8: (a,b) MCGAN and (c,d) CGAN trained with COIL100 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.

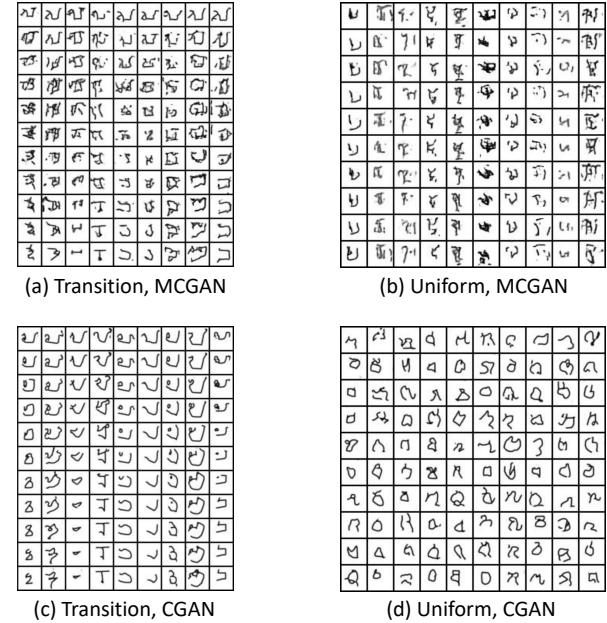


Figure 9: (a,b) MCGAN and (c,d) CGAN trained with Omniglot dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.

## 5. Conclusion

In this work, we proposed a plug-and-play nonparametric module named Multimodal Controller (MC) to equip generative models with class-conditional data generation. Unlike classical conditional generative models that introduce additional learning parameters to model class-conditional information, our method allocates a unique computation path for each data modality with a uniformly sampled subnetwork. The multimodal controller is a general method applicable to various well-known backbone generative models, and it works particularly well for a substantial number of modalities (e.g., the Omniglot challenge). Multimodal controlled generative models are also capable of creating novel data modalities. We believe that this work will shed light on the use of subnetworks for large-scale and multimodal deep learning. An interesting future direction is to study how the subnetwork can be adapted for cooperative learning scenarios such as assisted learning [36], federated learning [21, 4] and continual learning [38, 37].

## Acknowledgement

This work was supported by the Office of Naval Research (ONR) under grant number N00014-18-1-2244, and the Army Research Office (ARO) under grant number W911NF-20-1-0222.

## References

- [1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.
- [2] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.
- [3] Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems*, pages 6594–6604, 2017.
- [4] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. *arXiv preprint arXiv:2010.01264*, 2020.
- [5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.
- [6] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.
- [7] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.
- [8] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.
- [9] 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.
- [10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.
- [11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.
- [12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- [13] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.
- [14] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.
- [15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [16] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.
- [17] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.
- [18] Ming Liu, Yukang Ding, Min Xia, Xiao Liu, Errui Ding, Wangmeng Zuo, and Shilei Wen. Stgan: A unified selective transfer network for arbitrary image attribute editing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3673–3682, 2019.
- [19] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.
- [20] Qi Mao, Hsin-Ying Lee, Hung-Yu Tseng, Siwei Ma, and Ming-Hsuan Yang. Mode seeking generative adversarial networks for diverse image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1429–1437, 2019.
- [21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
- [22] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

- [23] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.
- [24] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.
- [25] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-100).
- [26] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.
- [27] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.
- [28] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for incremental learning, 2019.
- [29] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.
- [30] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.
- [31] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- [32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [33] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.
- [34] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.
- [35] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *arXiv preprint arXiv:2006.14769*, 2020.
- [36] Xun Xian, Xinran Wang, Jie Ding, and Reza Ghanadan. Assisted learning: A framework for multi-organization learning. *Advances in Neural Information Processing Systems*, 33, 2020.
- [37] Mengyao Zhai, Lei Chen, Jiawei He, Megha Nawhal, Frederick Tung, and Greg Mori. Piggyback gan: Efficient lifelong learning for image conditioned generation. In *European Conference on Computer Vision*, pages 397–413. Springer, 2020.
- [38] Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2759–2768, 2019.
- [39] Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. *arXiv preprint arXiv:1905.01067*, 2019.

## Appendix

In this supplementary document, we provide 1) more experimental details including the metrics and network architecture, 2) more qualitative results to demonstrate the multimodal controlled generative models presented in the main paper as well as conditional generative models.

## 6. Experimental Details

### 6.1. Metric

Inception Score (IS) was originally introduced by [29] and formulated as

$$\exp [\mathbb{E} \{ D_{\text{KL}}(p(y|x) \| p(y)) \}]$$

where  $p(y|x)$  is the output of pretrained Inception network [32]. It is known that IS is strongly correlated with subjective human views of image quality. Fréchet Inception Distance (FID) [11] uses the information of the final layer of the inception model to measure the quality of the generated examples. It measures the Wasserstein distance between two distributions  $p_1$  and  $p_2$  assuming that they are both multivariate Gaussian distributions, expressed by

$$\| \mu_{p_1} - \mu_{p_2} \|_2^2 + \text{trace} \left( C_{p_1} + C_{p_2} - 2 (C_{p_1} C_{p_2})^{1/2} \right),$$

where  $(\mu_{p_1}, \mu_{p_2})$  and  $(C_{p_1}, C_{p_2})$  are the mean and covariance of samples from  $p_1$  and  $p_2$  respectively.

We evaluate IS for CIFAR10 datasets with the standard Inception network pre-trained with the ImageNet dataset. We train our classifier as shown in Table 11 for COIL100 and Omniglot datasets with all the training and test data. We calculate the scores from randomly generated 1000 examples per data mode for CIFAR10, 100 for COIL100, and 20 for the Omniglot dataset respectively. We repeat each experiment 12 times with different random seeds and report the mean and standard deviations (in brackets).

### 6.2. Network Architecture

We show hyperparameters for training generative models in Table 12. The network architectures of MCVAE and MCGAN used in our experiments are shown in Table 5 to 10, where  $n_c$  is the number of image channels and  $M$  is the shape of the input image. We use the standard architecture of PixelCNN and Glow described in the original work [34, 14]. The number of layers, embedding size, and hidden channel size of PixelCNN are 15, 512, and 128 respectively.

The depth of flows  $K$ , the number of levels  $L$ , and the hidden channel size of Glow are 3, 16, and 512 respectively. We do not train an auxiliary classifier for Glow networks for fair comparisons with other generative models.

The embedding size of embeddings used for modeling conditional information  $h$  in CVAE and CGAN is 32. We concatenate conditional embeddings to the first layer of the encoder (discriminator) and decoder (generator) as described in [30, 22]. Following the implementation suggested in [34], we add conditional embeddings to every gated activation in CPixelCNN. As a result, CPixelCNN has more learning parameters for modeling  $h$  than other conditional generative models do. As a result, the CPixelCNN for Omniglot also has twice the number of learning parameters as MCPixelCNN does. We add conditional embeddings to the trained prior in the first level of CGlow [14]. Multimodal controlled generative models always have less learning parameters than conditional generative models because we do not train any additional learning parameters and the uniformly sampled codebook  $e$  is merely an indicator for modulating subnetworks.

Multimodal controlled generative models attach the multimodal controller to various parametric layers, with some customization depending on their architectures. By controlling the allocation of subnetworks, uniformly sampled subnetworks are able to synthesize class-conditional data. We describe the usage of the multimodal controller for the most well-known representatives of generative models including VAE, PixelCNN, Glow, and GAN.

**Multimodal Controlled Variational Autoencoder (MCVAE).** MCVAE appends a multimodal controller to every layer of VAE. The number of model parameters remains the same as non-conditional VAE. If  $e = 1$ , the all-one matrix, our proposed method produces a non-conditional VAE. For each mode of data  $x_c$ , we allocate a unique subnetwork with learning parameters  $\theta_c$  uniformly sampled from the overall model parameters  $\theta$ , and we also obtain class-conditional latent variables  $z_c$ . The objective can be written as:

$$\min_{\theta} \left\{ \sum_{c=1}^C \frac{N_c}{N} \left[ \mathbb{E}_{q_{\theta_c}(z_c|x_c)} \log p_{\theta_c}(x_c|z_c) - D_{\text{KL}}(q_{\theta_c}(z_c|x_c) \| p_{\theta_c}(z_c)) \right] \right\}.$$

Thanks to the end-to-end structure of VAE, MCVAE can uniformly sample subnetworks by appending a multimodal controller after every parametric layer. For the bottleneck layer, we obtain class-conditional latent variables  $z_c$  by attaching the multimodal controller after applying the reparameterization trick, instead of directly attaching to the parameters of the latent distribution. The subnetworks  $\theta_c$  can be optimized in the same way as in non-conditional VAE.

**Multimodal Controlled PixelCNN (MCPixelCNN).** PixelCNN combines two gated convolutional network stacks in order to remove blind spots in the receptive field [34]. MCPixelCNN allocates subnetworks for each data mode by introducing the following multimodal controlled gated activation unit:

$$y = \{ \tanh(W_{k,f} * x) \odot (h \times e) \} \odot \sigma(W_{k,g} * x),$$

where  $\sigma$  denotes the sigmoid function,  $k$  is the number of layers, and  $*$  is the convolution operator. Our various experimental results show that controlling the  $\tanh(\cdot)$  activation unit is sufficient for synthesizing class-conditional data. Other convolution layers used in connecting two stacks and residuals do not necessarily need to be multimodally controlled. The class-conditional likelihood function can be written as

$$\begin{aligned} p_{\theta_c}(x_c) &= p_{\theta_c}(x_{1,c}, \dots, x_{n,c}) \\ &= \prod_{i=1}^n p_{\theta_c}(x_{i,c}|x_{1,c}, \dots, x_{i-1,c}) \end{aligned}$$

**Multimodal Controlled Glow (MCGlow).** Glow has three major components including Actnorm, Invertible  $1 \times 1$  convolution, and affine coupling layer. The number of channels of features passing Actnorm and Invertible  $1 \times 1$  convolution is usually a multiple of the number of color channels due to the squeeze operation. It leads to a small number of channels of features which are incapable of allocating subnetworks for a large number of data modalities. MCGlow allocates subnetworks only for  $\text{NN}(\cdot)$  in affine coupling layer, where  $\text{NN}(\cdot)$  is a shallow and wide convolutional neural network block often used in ResNet [10] and RealNVP [5]. As a result, the parameters used in Actnorm and Invertible  $1 \times 1$  convolution are shared across all data modalities. The class-conditional likelihood function for a  $k$ -layer Glow can be written as:

$$\begin{aligned} \log p_{\theta_c}(x_c) &= \log p(z) + \sum_{i=1}^k \log |\det(J(f_{i,c}^{-1}(x_c)))| \\ z &= f_{k,c}^{-1} \circ f_{k-1,c}^{-1} \circ \dots \circ f_{0,c}^{-1}(x_c). \end{aligned}$$

**Multimodal Controlled Generative Adversarial Network (MCGAN).** MCGAN appends a multimodal controller after every layer of the discriminator and generator. We experimentally found that using the multimodal controller on either generator or discriminator only does not produce a class-conditional synthesis. The objective function of a multimodal controlled two-player minimax game is formulated as:

$$\min_G \max_D \left\{ \sum_{c=1}^C \frac{N_c}{N} \left[ \mathbb{E}_{q_{\theta_c}(x_c)} [\log D_{\theta_c}(x_c)] + \mathbb{E}_{p_{\theta_c}(z)} [\log(1 - D_{\theta_c}(G_{\theta_c}(z)))] \right] \right\}.$$

Image $x \in \mathbb{R}^{n_c \times M \times M}$
MCCConv ( $n_c, 64, 4, 2, 1$ )
MCCConv ( $64, 128, 4, 2, 1$ )
MCCConv ( $128, 256, 4, 2, 1$ )
MCResBlock ( $256, 256, 3, 1, 1$ )
MCResBlock ( $256, 256, 3, 1, 1$ )
Linear ( $\mu$ ) ( $256 \times 4 \times 4, 128$ )
Linear ( $\log \sigma^2$ ) ( $256 \times 4 \times 4, 128$ )

Table 5: Encoder used in MCVAE for all datasets.

$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$
MC (128, 128)
Linear ( $128, 256 \times 4 \times 4$ ), BN, ReLU
MC (256, 256)
MCResBlock ( $256, 256, 3, 1, 1$ )
MCResBlock ( $256, 256, 3, 1, 1$ )
MCConvTranspose ( $256, 128, 4, 2, 1$ )
MCConvTranspose ( $128, 64, 4, 2, 1$ )
ConvTranspose ( $64, n_c, 4, 2, 1$ ), Sigmoid

Table 6: Decoder used in MCVAE for all datasets.

$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$
MCLinear ( $128, 256 \times 4 \times 4$ )
MCPreResBlock Up ( $256, 256, 3, 1, 1$ )
MCPreResBlock Up ( $256, 256, 3, 1, 1$ )
MCPreResBlock Up ( $256, 256, 3, 1, 1$ )
BN, ReLU, Conv ( $256, n_c, 3, 1, 1$ ), Tanh

Table 7: Generator used in MCGAN for CIFAR10 dataset. ResBlock Up interpolates input feature map at the beginning of block by scale 2.

## Qualitative Results

In this section, we provide additional qualitative results from our experiments from Figure 10 to 27. The qualitative results are well aligned with the quantitative results. For example, conditional generative models trained with the COIL100 dataset are not able to produce diverse orientation of objects. Due to the unique subnetwork allocated

Image $x \in \mathbb{R}^{M \times M \times n_c}$
MCPreResBlock Down ( $n_c, 128, 3, 1, 1$ )
MCPreResBlock Down ( $128, 128, 3, 1, 1$ )
MCPreResBlock ( $128, 128, 3, 1, 1$ )
MCPreResBlock ( $128, 128, 3, 1, 1$ )
ReLU, Global Sum Pooling
Linear (512, 1)

Table 8: Discriminator used MCGAN for CIFAR10 dataset. All BN layers are replaced with Spectral Normalization (SN) [23]. ResBlock Down average pools feature map at the end of block by scale 2.

$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$
MCLinear ( $128, 512 \times 4 \times 4$ )
MCPreResBlock Up ( $512, 256, 3, 1, 1$ )
MCPreResBlock Up ( $256, 128, 3, 1, 1$ )
MCPreResBlock Up ( $128, 64, 3, 1, 1$ )
BN, ReLU, Conv ( $64, n_c, 3, 1, 1$ ), Tanh

Table 9: Generator used in MCGAN for COIL100 and Omnipiglot datasets. ResBlock Up interpolates input feature map at the beginning of block by scale 2.

Image $x \in \mathbb{R}^{n_c \times M \times M}$
MCPreResBlock Down ( $n_c, 64, 3, 1, 1$ )
MCPreResBlock Down ( $64, 128, 3, 1, 1$ )
MCPreResBlock Down ( $128, 256, 3, 1, 1$ )
MCPreResBlock ( $256, 512, 3, 1, 1$ )
ReLU, Global Sum Pooling
Linear (512, 1)

Table 10: Discriminator used MCGAN for COIL100 and Omnipiglot datasets. All BN layers are replaced with Spectral Normalization (SN) [23]. ResBlock Down average pools feature map at the end of block by scale 2.

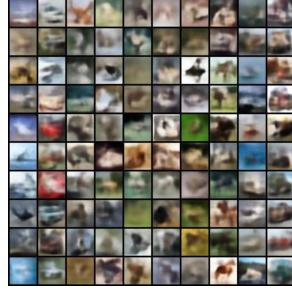
for each mode of data, the qualitative results show that creations from multimodal generative models are more closely clustered than those from conditional generative models. Transition results of PixelCNN for both methods are not available because we do not have a fixed latent variable. Results from Glow suffer from numerical overflow (black areas) due to its invertibility.

Image $x \in \mathbb{R}^{n_c \times M \times M}$
Conv ( $n_c, 8, 3, 1, 1$ )
MaxPool ( $8, 8, 2$ )
Conv ( $8, 16, 3, 1, 1$ )
MaxPool ( $16, 16, 2$ )
Conv ( $16, 32, 3, 1, 1$ )
MaxPool ( $32, 32, 2$ )
Conv ( $32, 64, 3, 1, 1$ )
Linear ( $64 \times 4 \times 4, C$ )

Table 11: Classifier used to train COIL100 and Omniglot for evaluating IS and FID.



(a) MCVAE



(b) CVAE

Figure 10: (a) MCVAE (b) CVAE trained with CIFAR10 dataset. Generations in each column are from one data modality.



(a) MCPixelCNN



(b) CPixelCNN

Figure 11: (a) MCPixelCNN (b) CPixelCNN trained with CIFAR10 dataset. Generations in each column are from one data modality.



(a) MCGlow



(b) CGlow

Figure 12: (a) MCGlow (b) CGlow trained with CIFAR10 dataset. Generations in each column are from one data modality.

	Batch size	Loss	Optimizer	Learning rate	beta
CVAE	128	BCE	Adam	3e-4	(0.9,0.999)
MCVAE	128	BCE	Adam	3e-4	(0.9,0.999)
CPixelCNN	128	NLL	Adam	3e-4	(0.9,0.999)
MCPixelCNN	128	NLL	Adam	3e-4	(0.9,0.999)
CGlow	128	NLL	Adam	3e-4	(0.9,0.999)
MCGlow	128	NLL	Adam	3e-4	(0.9,0.999)
CGAN	128	Hinge	Adam	2e-4	(0,0.9)
MCGAN	128	Hinge	Adam	2e-4	(0.5,0.999)

Table 12: Hyperparameters for training generative models.

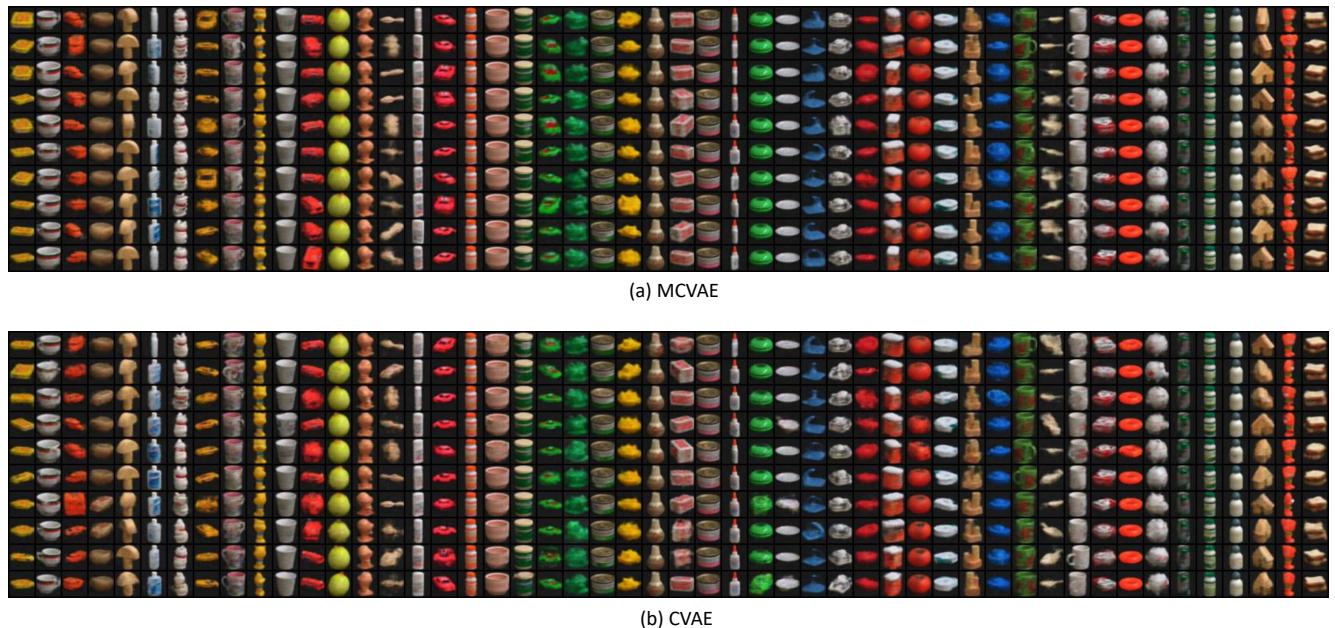
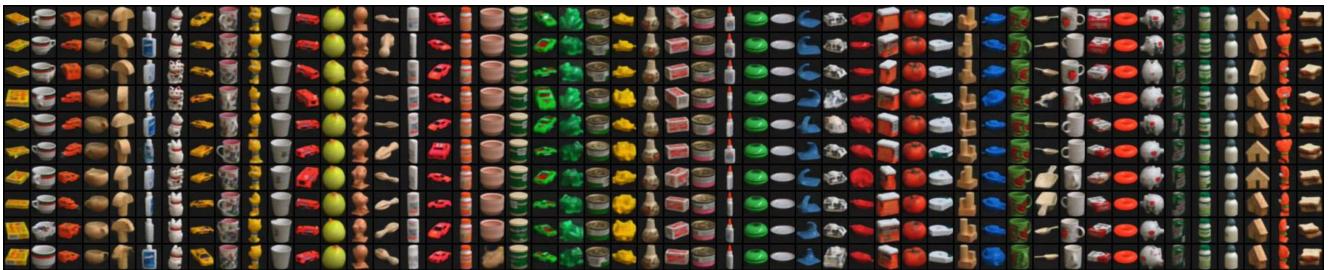
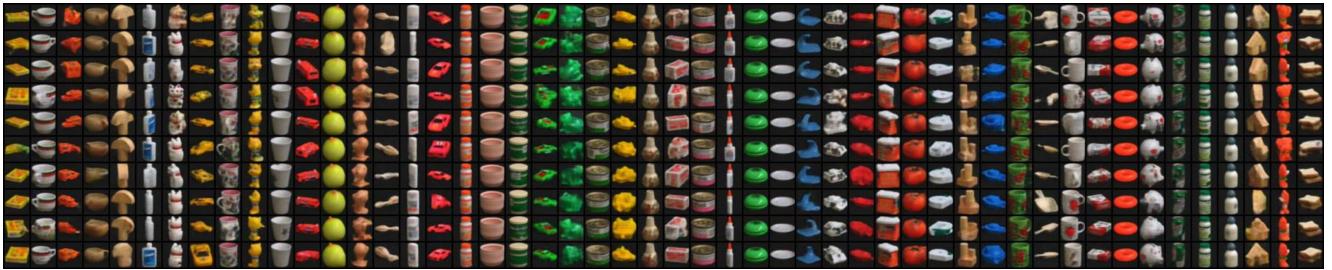


Figure 13: (a) MCVAE (b) CVAE trained with COIL100 dataset. Generations in each column are from one data modality.

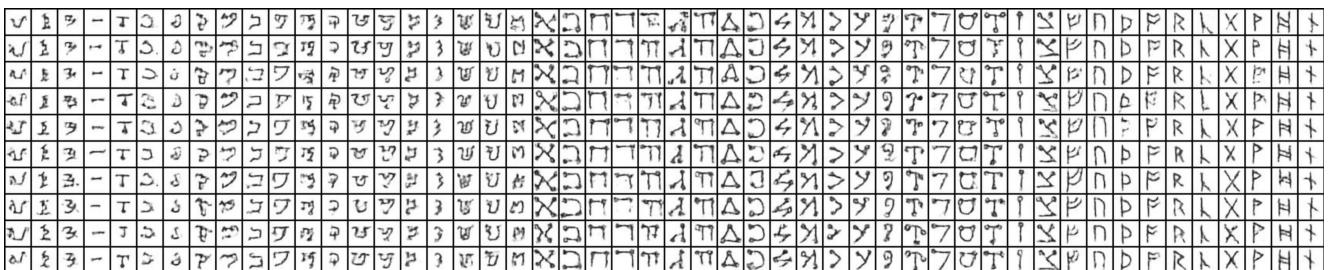


(a) MCPixelCNN



(b) CPixelCNN

Figure 14: (a) MCPixelCNN (b) CPixelCNN trained with COIL100 dataset. Generations in each column are from one data modality.

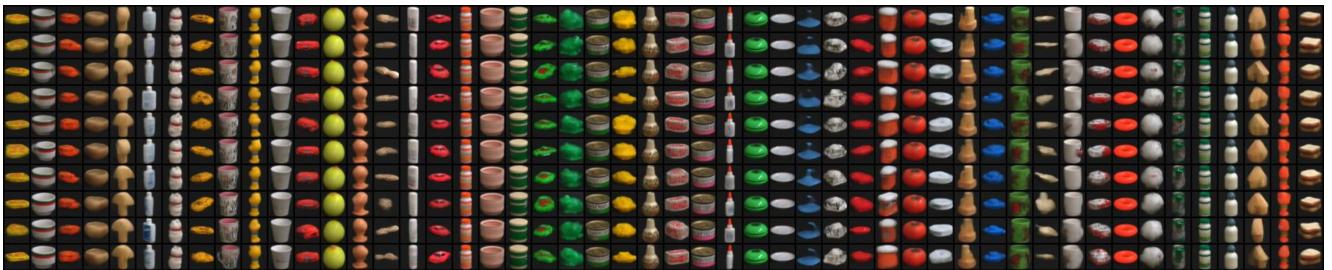


(a) MCVAE

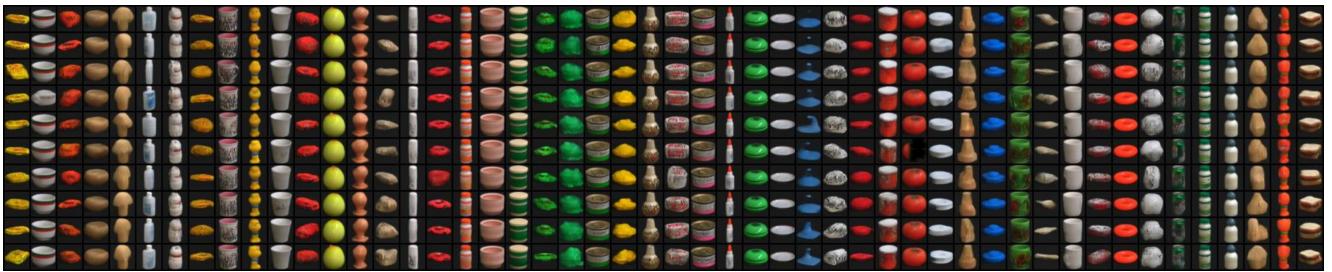


(b) CVAE

Figure 15: (a) MCVAE (b) CVAE trained with Omniglot dataset. Generations in each column are from one data modality.

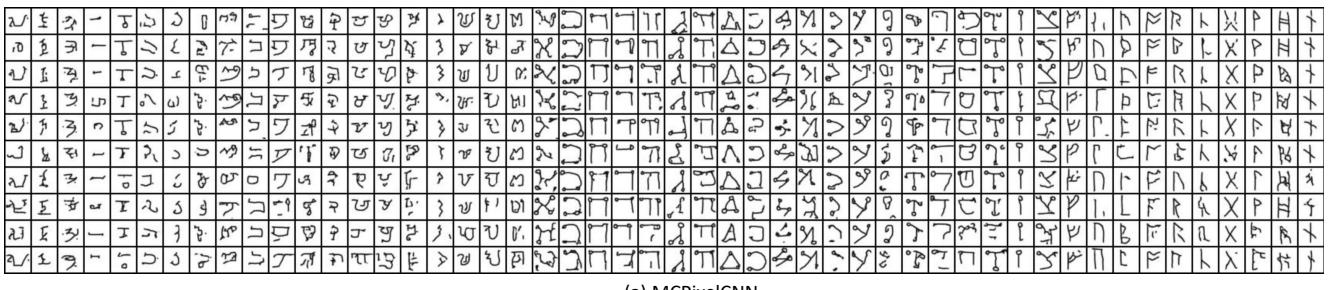


(a) MCGlow

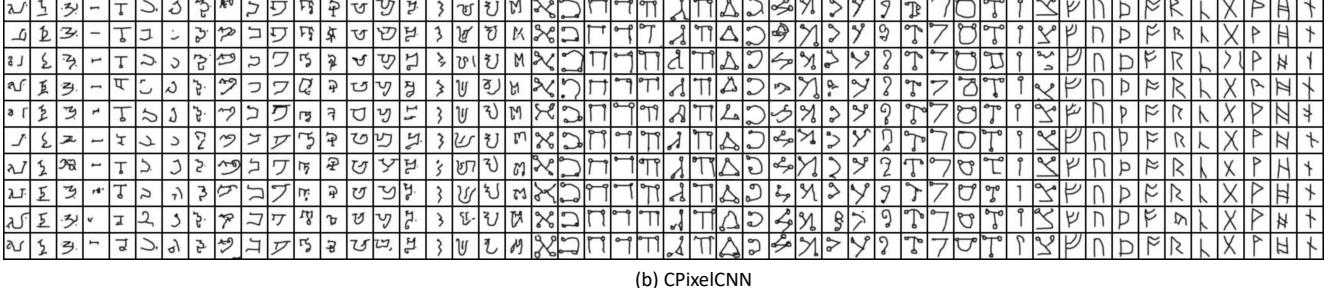


(b) CGlow

Figure 16: (a) MCGlow (b) CGlow trained with COIL100 dataset. Generations in each column are from one data modality.

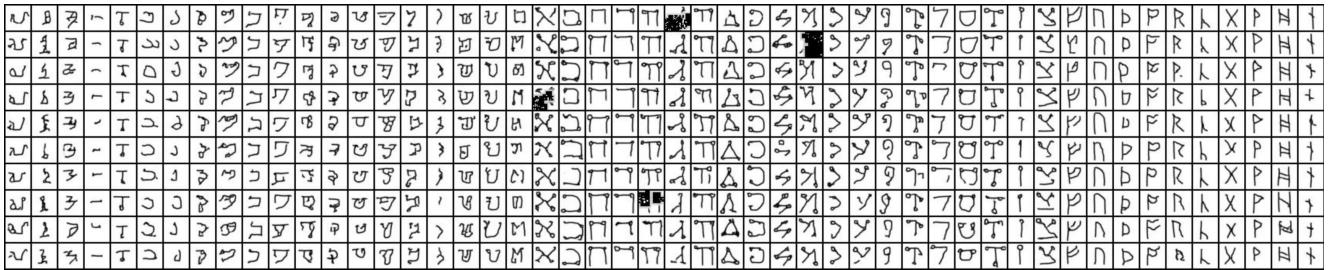


(a) MCPixelCNN



(b) CPixelCNN

Figure 17: (a) MCPixelCNN (b) CPixelCNN trained with Omniglot dataset. Generations in each column are from one data modality.



(a) MCGlow



(b) CGlow

Figure 18: (a) MCGlow (b) CGlow trained with Omniglot dataset. Generations in each column are from one data modality.



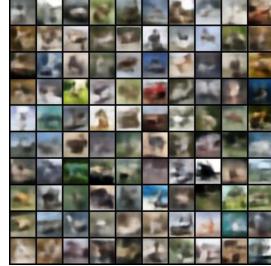
(a) Transition, MCVae



(b) Uniform, MCVae



(c) Transition, CVAE



(d) Uniform, CVAE

Figure 19: (a,b) MCVae and (c,d) CVAE trained with CI-FAR10 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.



(a) Uniform, MCPixelCNN



(b) Uniform, CPixelCNN

Figure 20: (a) MCPixelCNN and (b) CPixelCNN trained with CIFAR10 dataset. Uniform data modalities are created from resampling of pre-trained data modalities.



(a) Transition, MCGlow



(b) Uniform, MCGlow

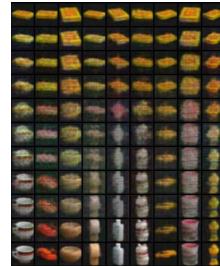


(c) Transition, CGlow

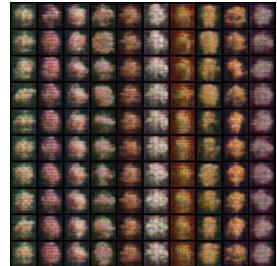


(d) Uniform, CGlow

Figure 21: (a,b) MCGlow and (c,d) CGlow trained with CI-FAR10 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.



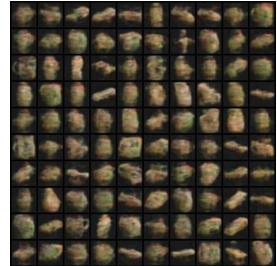
(a) Transition, MCVae



(b) Uniform, MCVae

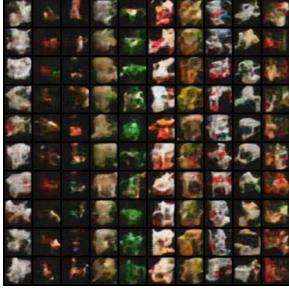


(c) Transition, CVAE



(d) Uniform, CVAE

Figure 22: (a,b) MCVae and (c,d) CVAE trained with COIL100 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.



(a) Uniform, MCPixelCNN



(b) Uniform, CPixelCNN

Figure 23: (a,b) MCPixelCNN and (c,d) CPixelCNN trained with COIL100 dataset. Uniform data modalities are created from resampling of pre-trained data modalities.



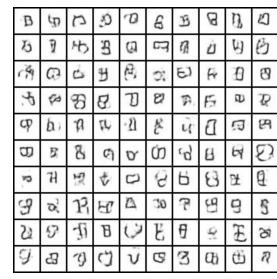
(a) Transition, MCVAE



(b) Uniform, MCVAE



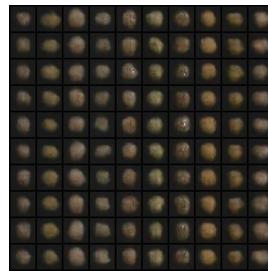
(c) Transition, CVAE



(d) Uniform, CVAE



(a) Transition, MCGlow



(b) Uniform, MCGlow



(c) Transition, CGlow



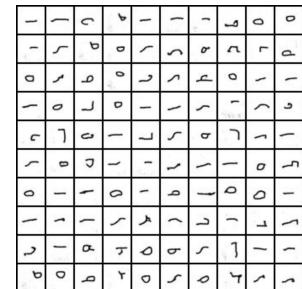
(d) Uniform, CGlow

Figure 24: (a,b) MCGlow and (c,d) CGlow trained with COIL100 dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.

Figure 25: (a,b) MCVAE and (c,d) CVAE trained with Omniglot dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.

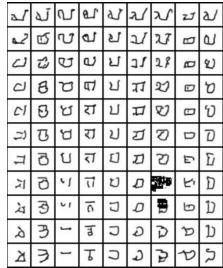


(a) Uniform, MCVAE

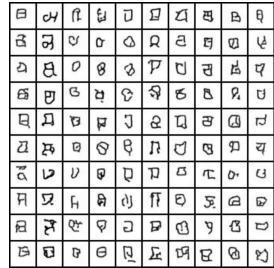


(b) Uniform, CVAE

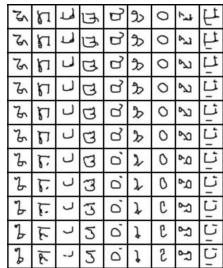
Figure 26: (a,b) MCPixelCNN and (c,d) CPixelCNN trained with Omniglot dataset. Uniform data modalities are created from resampling of pre-trained data modalities.



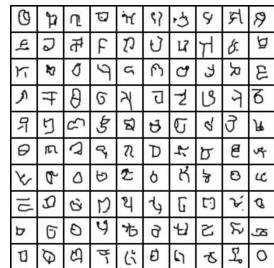
(a) Transition, MCGlow



(b) Uniform, MCGlow



(c) Transition, CGlow



(d) Uniform, CGlow

Figure 27: (a,b) MCGlow and (c,d) CGlow trained with Omniglot dataset. (a,c) Transitions in each column are created from interpolations from the first data modality to others. (b,d) Uniform data modalities are created from resampling of pre-trained data modalities.