CONDITIONAL GENERATIVE MODELING VIA LEARNING THE LATENT SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Although deep learning has achieved appealing results on several machine learning tasks, most of the models are deterministic at inference, limiting their application to single-modal settings. We propose a novel general-purpose framework for conditional generation in multimodal spaces, that uses latent variables to model generalizable learning patterns while minimizing a family of regression cost functions. At inference, the latent variables are optimized to find optimal solutions corresponding to multiple output modes. Compared to existing generative solutions, our approach demonstrates faster and stable convergence, and can learn better representations for downstream tasks. Importantly, it provides a simple generic model that can beat highly engineered pipelines tailored using domain expertise on a variety of tasks, while generating diverse outputs. Our codes will be released.

1 INTRODUCTION

Conditional generative models provide a natural mechanism to jointly learn a data distribution and optimize predictions. In contrast, discriminative models improve predictions by modeling the label distribution. Learning to model the data distribution allows generating novel samples and is considered a preferred way to understand the real world. Existing conditional generative models have generally been explored in single-modal settings, where a one-to-one mapping between input and output domains exists (Nalisnick et al., 2019; Fetaya et al., 2020). Here, we investigate continuous multimodal (CMM) spaces for generative modeling, where one-to-many mappings exist between input and output domains. This is critical since many real world situations are inherently multi-modal, e.g., humans can imagine several outcomes for a given occluded image. In a discrete setting, this problem becomes relatively easy to tackle using techniques such as maximum-likelihood-estimation, since the output can be predicted as a vector (Zhang et al., 2016), which is not possible in continuous domains. One way to model CMM spaces is by using variational inference, e.g., variational autoencoders (VAE) (Kingma & Welling, 2013). However, the approximated posterior distribution of VAEs are often restricted to the Gaussian family, which hinders its ability to model more complex distributions. As a solution, Maaløe et al. (2016) suggested using auxiliary variables to improve the variational distribution. To this end, the latent variables are hierarchically correlated through injected auxiliary variables, which can produce non-Gaussian distributions. A slightly similar work by Rezende & Mohamed (2015) proposed Normalizing Flows, that can hierarchically generate more complex probability distributions by applying a series of bijective mappings to an original simpler distribution. Recently, Chang et al. (2019) proposed a model, where a separate variable can be used to vary the impact of different loss components at inference, which allows diverse outputs. For a more detailed discussion on these methods please see App. 1.

In addition to the aforesaid methods, in order to model CMM spaces, a prominent approach in the literature is to use a combination of reconstruction and adversarial losses (Isola et al., 2017; Zhang et al., 2016; Pathak et al., 2016). However, this entails key shortcomings. 1) The goals of adversarial and reconstruction losses are contradictory (Sec. 4), hence model engineering and numerous regularizers are required to support convergence (Lee et al., 2019; Mao et al., 2019), thereby resulting in less-generic models tailored for specific applications (Zeng et al., 2019; Vitoria et al., 2020). 2) The adversarial loss based models are notorious for difficult convergence due to the challenge of finding Nash equilibrium of a non-convex min-max game in high-dimensions (Barnett, 2018; Chu et al., 2020; Kodali et al., 2017). 3) The convergence is heavily dependent on the architecture, hence such models show lack of scalability (Thanh-Tung et al., 2019; Arora &

Zhang, 2017). 4) The promise of assisting downstream tasks remains challenging, with a large gap in performance between the generative modelling approaches and their discriminative counterparts (Grathwohl et al., 2020; Jing & Tian, 2020).

In this work, we propose a general-purpose framework—Conditional Generation by Modeling the Latent Space (cGML)—for modeling CMM spaces using a set of domain-agnostic regression cost functions instead of the adversarial loss. This improves both the stability and eliminates the incompatibility between the adversarial and reconstruction losses, allowing more precise outputs while maintaining diversity. The underlying notion is to learn the '*behaviour of the latent variables*' in minimizing these cost functions while converging to an optimum mode during the training phase, and mimicking the same at inference. Despite being a novel direction, the proposed framework showcases promising attributes by: (a) achieving state-of-the-art results on a diverse set of tasks using a generic model, implying generalizability, (b) rapid convergence to optimal modes despite architectural changes, (c) learning useful features for downstream tasks, and (d) producing diverse outputs via traversal through multiple output modes at inference.

2 Proposed Methodology

We define a family of cost functions $\{E_{i,j} = d(y^g_{i,j}, \mathcal{G}(x_j, w))\} \in \xi$, where $x_j \sim \chi$ is the input, $y^g_{i,j} \sim \Upsilon$ is the i^{th} ground-truth mode for x_j, $\mathcal{G}$ is a generator function with weights w, and $d(\cdot,\cdot)$ is a distance function. Note that the number of cost functions $E_{(\cdot,j)}$ for a given x_j can vary over χ. Our aim here is to come up with a generator function $\mathcal{G}(x_j, w)$, that can minimize each $E_{i,j}, \forall i$ as $\mathcal{G}(x_j, w) \rightarrow y^g_{i,j}$. However, since $\mathcal{G}$ is a deterministic function (x and w are both fixed at inference), it can only produce a single output. Therefore, we introduce a latent vector z to the generator function, that can be used to converge $\bar{y}_{i,j} = \mathcal{G}(x_j, w, z_{i,j})$ towards a $y^g_{(i,j)}$ at inference, and possibly, to multiple solutions. Formally, the family of cost functions now becomes: $\{\hat{E}_{i,j} = d(y^g_{i,j}, \mathcal{G}(x_j, w, z_{i,j}))\}, \forall z_{i,j} \sim \zeta$. Then, our training objective can be defined as finding a set of optimal $z^*_i \in \zeta$ and $w^* \in \omega$ by minimizing $\mathbb{E}_{i\sim I}[\hat{E}_i]$, where I is the number of possible solutions for x_j. Note that w^* is fixed for all i and a different z^*_i exists for each i. Considering all the training samples $x_j \sim \chi$, our training objective becomes,

$$\{\{z^*_{i,j}\}, w^*\} = \underset{z_{i,j}\in\zeta, w\in\omega}{\arg\min}\ \mathbb{E}_{i\in I, j\in J}[\hat{E}_{i,j}]. \tag{1}$$

Eq. 1 can be optimized via Algorithm 1 (proof in App. 2.2). Intuitively, the goal of Eq. 1 is to obtain a family of optimal latent codes $\{z^*_{i,j}\}$, each causing a global minima in the corresponding $\hat{E}_{i,j}$ as $y^g_{i,j} = \mathcal{G}(x_j, w, z^*_{i,j})$. Consequently, at inference, we can optimize $\bar{y}_{i,j}$ to converge to an optimal mode in the output space by varying z. Therefore, we predict an estimated $\bar{z}_{i,j}$ at inference,

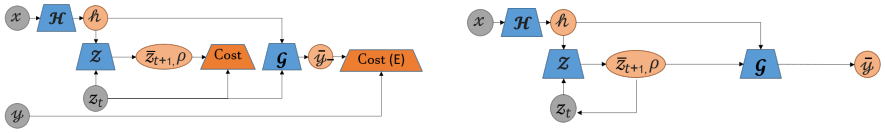
$$\bar{z}_{i,j} \approx \min_z \hat{E}_{i,j}, \tag{2}$$

for each $y^g_{i,j}$, which in turn can be used to obtain the prediction $\mathcal{G}(x_j, \bar{z}_{i,j}, w) \approx y^g_{i,j}$. In other words, for a selected x_j, let $\bar{y}^t_{i,j}$ be the initial estimate for $\bar{y}_{i,j}$. At inference, z can traverse gradually towards an optimum point $y^g_{i,j}$ in the space, forcing $\bar{y}^{t+n}_{i,j} \rightarrow y^g_{i,j}$, in finite steps ($n$).

However, still a critical problem exists: Eq. 2 depends on $y^g_{i,j}$, which is not available at inference. As a remedy, we enforce Lipschitz constraints on $\mathcal{G}$ over $(x_j, z_{i,j})$, which bounds the gradient norm as,

$$\frac{\left\|\mathcal{G}(x_j, w^*, z^*_{i,j}) - \mathcal{G}(x_j, w^*, z_0)\right\|}{\left\|z^*_{i,j} - z_0\right\|} \leq \int \left\|\nabla_z \mathcal{G}(x_j, w^*, \gamma(t))\right\| dt \leq C, \tag{3}$$

where $z_0 \sim \zeta$ is an arbitrary random initialization, C is a constant, and $\gamma(\cdot)$ is a straight path from z_0 to $z^*_{i,j}$ (proof in App. 2.1) . Intuitively, Eq. 3 implies that the gradients $\nabla_z \mathcal{G}(x_j, w^*, z_0)$ along the path $\gamma(\cdot)$ do not tend to vanish or explode, hence, finding the path to optimal $z^*_{i,j}$ in the space ζ becomes a fairly straight forward regression problem. Moreover, enforcing the Lipschitz constraint encourages meaningful structuring of the latent space: suppose $z^*_{1,j}$ and $z^*_{2,j}$ are two optimal codes corresponding to two ground truth modes for a particular input. Since $\|z^*_{2,j} - z^*_{1,j}\|$ is lower bounded by $\frac{\left\|\mathcal{G}(x_j, w^*, z^*_{2,j}) - \mathcal{G}(x_j, w^*, z^*_{1,j})\right\|}{L}$, where L is the Lipschitz constant, the minimum distance between the two latent codes is proportional to the difference between the corresponding ground truth modes.



(a) Training (b) Inference

Figure 1: Training and inference process. Refer to Algorithm 1 for the training process. At inference, z is iteratively updated using the predictions of $\mathcal{Z}$ and fed to $\mathcal{G}$ to obtain increasingly fine-tuned outputs (see Sec. 3).

In practice, we observed that this encourages the optimum latent codes to be placed sparsely (visual illustration in App. 2), which helps a network to learn distinctive paths towards different modes.

2.1 Convergence at Inference

We formulate finding the convergence path of z at inference as a regression problem, i.e., $z_{t+1} = r(z_t, x_j)$. We implement $r(\cdot)$ as a recurrent neural network (RNN). The series of predicted values $\{z_{(t+k)} : k = 1, 2, .., N\}$ can be modeled as a first-order Markov chain requiring no memory for the RNN. We observe that enforcing Lipschitz continuity on $\mathcal{G}$ over z leads to smooth trajectories even in high dimensional settings, hence, memorizing more than one step in to the history is redundant. However, z_{t+1} is not a state variable, i.e., the existence of multiple modes for output prediction $\bar{y}$ leads to multiple possible solutions for z_{t+1}. On the contrary, $\mathbb{E}[z_{t+1}]$ is a state variable w.r.t. the state (z_t, x), which can be used as an approximation to reach the optimal z^* at inference. Therefore, instead of directly learning $r(\cdot)$, we learn a simplified version $r'(z_t, x) = \mathbb{E}[z_{t+1}]$. Intuitively, the whole process can be understood as observing the behavior of z on a smooth surface at the training stage, and predicting the movement at inference. A key aspect of $r'(z_t, x)$ is that the model is capable of converging to multiple possible optimum modes at inference based on the initial position of z.

2.2 Momentum as a Supplementary Aid

Based on Sec. 2.1, z can now traverse to an optimal position z^* during inference. However, there can exist rare symmetrical positions in the ζ where $\mathbb{E}[z_{t+1}] - z_t \approx 0$, although far away from $\{z^*\}$, forcing $z_{t+1} \approx z_t$. Simply, the above phenomenon can occur if some z_{t+1} has traveled in many non-orthogonal directions, so the vector addition of $z_{t+1} \approx 0$. This can *fool* the system to falsely identify convergence points, forming *phantom* optimum point distributions amongst the true distribution (see Fig. 3). To avoid such behavior, we consider $\vec{v}(z_t, x_j) = (z_{t+1} - z_t)_{x_j}$. Then, we learn the expected momentum $\mathbb{E}[\rho(z_t, x_j)] = \alpha\mathbb{E}[|\vec{v}(z_t, x_j)|]$ at each (z_t, x_j) during the training phase, where α is an empirically chosen scalar. In practice, $\mathbb{E}[\rho(z_t, x_j)] \to 0$ as $z_{t+1}, z_t \to \{z^*\}$. Thus, to avoid *phantom* distributions, we improve the z update as,

$$z_{t+1} = z_t + \mathbb{E}[\rho(z_t, x_j)] \left[\frac{r'(z_t, x_j) - z_t}{\left\| r'(z_t, x_j) - z_t \right\|} \right]. \tag{4}$$

Since both $\mathbb{E}[\rho(z_t, x_j)]$ and $r'(z_t, x_j)$ are functions on (z_t, x_j), we jointly learn these two functions using a single network $\mathcal{Z}(z_t, x_j)$. Note that coefficient $\mathbb{E}[\rho(z_t, x_j)]$ serves two practical purposes: 1) slows down the movement of z near true distributions, 2) pushes z out of the phantom distributions.

3 Overall Design

The proposed model consists of three major blocks as shown in Fig. 1: an encoder $\mathcal{H}$, a generator $\mathcal{G}$, and $\mathcal{Z}$. The detailed architecture diagram for 128×128 is shown in Fig. 2. Note that for derivations in Sec. 2, we used x instead of $h = \mathcal{H}(x)$, as h is a high-level representation of x. The training process is illustrated in Algorithm 1. At each optimization $z_{t+1} = z_t - \beta\nabla_{z_t}[\hat{E}_{i,j}]$, $\mathcal{Z}$ is trained separately to approximate (z_{t+1}, ρ). At inference, x is fed to $\mathcal{H}$, and then the $\mathcal{Z}$ optimizes the output $\bar{y}$ by updating z for a pre-defined number of iterations of Eq. 4. For $\hat{E}(\cdot, \cdot)$, we use L_1 loss. Furthermore, it is important to limit the search space for z_{t+1}, to improve the performance of $\mathcal{Z}$. To this end, we sample z from the surface of the n-dimensional sphere ($\mathbb{S}^n$). Moreover, to ensure faster convergence of the model, we force the Lipschitz continuity on both $\mathcal{Z}$ and the $\mathcal{G}$ (App. 2.4) . For hyper-parameters and training details, see App. 3.1.

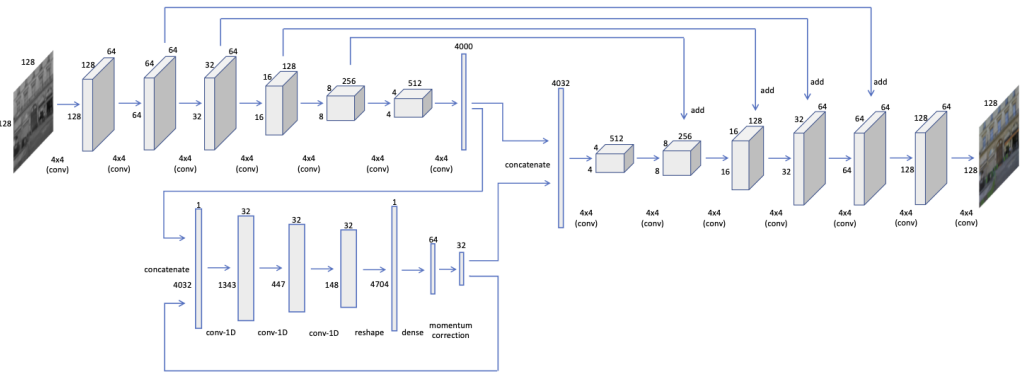


Figure 2: *Overall architecture for* 128×128 *inputs.*

Algorithm 1: Training algorithm

sample inputs $\{x_1, x_2, ..., x_J\} \in \chi$; sample outputs $\{y_1, y_2, ..., y_J\} \in \Upsilon$;
for k *epochs* **do**
  for x *in* χ **do**
    for l *steps* **do**
      update $z = \{z_1, z_2, ..., z_J\}$: $\nabla_z \hat{E}$ ▷ Freeze $\mathcal{H}, \mathcal{G}, \mathcal{Z}$ and update z
      update $\mathcal{Z}$: $\nabla_w L_1[(z_{t+1}, \rho), \mathcal{Z}(z_t, \mathcal{H}(x))]$ ▷ Freeze $\mathcal{H}, \mathcal{G}, z$ and update $\mathcal{Z}$
    update $\mathcal{G}, \mathcal{H}$: $\nabla_w \hat{E}$ ▷ Freeze $\mathcal{Z}, z$ and update $\mathcal{H}, \mathcal{G}$

4 MOTIVATION

Here, we explain the drawbacks of conditional GAN methods and illustrate our idea via a toy example.

Incompatibility of adversarial and reconstruction losses: cGANs use a combination of adversarial and reconstruction losses. We note that this combination is suboptimal to model CMM spaces.
Remark: *Consider a generator* $G(x, z)$ *and a discriminator* $D(x, z)$*, where* x *and* z *are the input and the noise vector, respectively. Then, consider an arbitrary input* x_j *and the corresponding set of ground-truths* $\{y^g_{i,j}\}, i = 1, 2, ..N$*. Further, let us define the optimal generator* $G^*(x_j, z) = \hat{y}, \hat{y} \in \{y^g_{i,j}\}$*,* $L_{GAN} = \mathbb{E}_i[\log D(y^g_{i,j})] + \mathbb{E}_z[\log(1 - D(G(x_j, z))]$ *and* $L_\ell = \mathbb{E}_{i,z}[|y^g_{i,j} - G(x_j, z)|]$*. Then,* $G^* \neq \hat{G}^*$ *where* $\hat{G}^* = \arg\min_G \max_D L_{GAN} + \lambda L_\ell, \forall \lambda \neq 0$. (Proof in App. 2.3).

Generalizability: The incompatibility of above mentioned loss functions demands domain specific design choices from models that target high realism in CMM settings. This hinders the generalizability across different tasks (Vitoria et al., 2020; Zeng et al., 2019). We further argue that due to this discrepancy, cGANs learn sub-optimal features which are less useful for downstream tasks (Sec. 5.3).

Convergence and the sensitivity to the architecture: The difficulty of converging GANs to the Nash equilibrium of a non-convex min-max game in high-dimensional spaces is well explored (Barnett, 2018; Chu et al., 2020; Kodali et al., 2017). Goodfellow et al. (2014b) underlines *if the discriminator has enough capacity, and is optimal at every step of the GAN algorithm, then the generated distribution converges to the real distribution*; that cannot be guaranteed in a practical scenario. In fact, Arora et al. (2018) confirmed that the adversarial objective can easily approach to an equilibrium even if the generated distribution has very low support, and further, the number of training samples required to avoid mode collapse can be in order of $\exp(d)$ (d is the data dimension).

Multimodality: The ability to generate diverse outputs, i.e., convergence to multiple modes in the output space, is an important requirement. Despite the typical noise input, cGANs generally lack the ability to generate diverse outputs (Lee et al., 2019). Pathak et al. (2016) and Iizuka et al. (2016) even state that better results are obtained when the noise is completely removed. Further, variants of cGAN that target diversity often face a trafe-off between the realism and diversity (He et al., 2018), as they have to compromise between the reconstruction and adversarial losses.

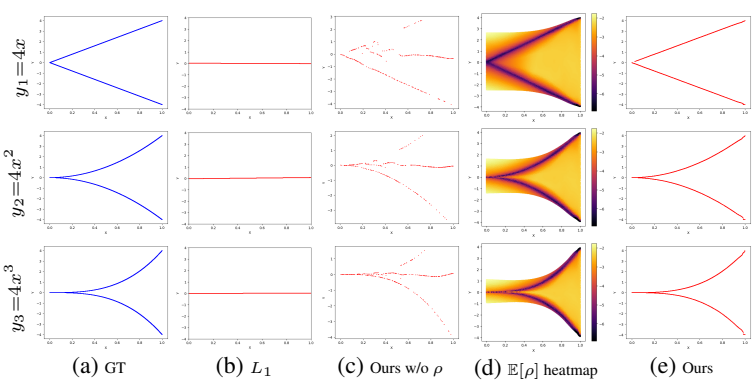


Figure 3: *Toy Example:* Plots generated for each dimension of the CMM space Υ. (a) Ground-truth distributions. (b) Model outputs for L_1 loss. (c) Output when trained with the proposed objective (without ρ correction). Note the *phantom distribution* identified by the model. (d) $\mathbb{E}[\rho]$ as a heatmap on (x, y). $\mathbb{E}[\rho]$ is lower near the true distribution and higher otherwise. (e) Model outputs after ρ correction.

A toy example: Here, we experiment with the formulations in Sec. 2. Consider a 3D CMM space $y = \pm 4(x, x^2, x^3)$. Then, we construct three layer multi-layer perceptrons (MLP) to represent each of the functions, $\mathcal{H}$, $\mathcal{G}$, and $\mathcal{Z}$, and compare the proposed method against the L_1 loss. Figure 3 illustrates the results. As expected, L_1 loss generates the line $y = 0$, and is inadequate to model the multimodal space. As explained in Sec. 2.2, without momentum correction, the network is fooled by a phantom distribution where $\mathbb{E}[z_{t+1}] \approx 0$ at training time. However, the *push* of momentum removes the phantom distribution and refines the output to closely resemble the input distribution. As implied in Sec. 2.2, the momentum is maximized near the true distribution and minimized otherwise.

5 Experiments and Discussions

The distribution of natural images lies on a high dimensional manifold, making the task of modelling it extremely challenging. Moreover, conditional image generation poses an additional challenge with their constrained multimodal output space (a single input may correspond to multiple outputs while not all of them are available for training). In this section, we experiment on several such tasks. For a fair comparison with a similar capacity GAN, we use the encoder and decoder architectures used in Pathak et al. (2016) for $\mathcal{H}$ and $\mathcal{G}$ respectively. We make two minor modifications: the channel-wise fully connected (FC) layers are removed and U-Net style skip connections are added (see App. 3.1). We train the existing models for a maximum of 200 epochs where pretrained weights are not provided, and demonstrate the generalizability of our theoretical framework in diverse practical settings by using a generic network for all the experiments. Models used for comparisons are denoted as follows: PN (Zeng et al., 2019), CA (Yu et al., 2018b), DSGAN (Yang et al., 2019), CIC (Zhang et al., 2016), RFR (Li et al., 2020), Chroma (Vitoria et al., 2020), P2P (Isola et al., 2017), Izuka (Iizuka et al., 2016), CE (Pathak et al., 2016), CRN (Chen & Koltun, 2017a), and B-GAN (Zhu et al., 2017b).

5.1 Corrupted Image Recovery

We design this task as image completion, i.e., given a masked image as input, our goal is to recover the masked area. Interestingly, we observed that the MNIST dataset, in its original form, does not have a multimodal behaviour, i.e., a fraction of the input image only maps to a single output. Therefore, we modify the training data as follows: first, we overlap the top half of an input image with the top half of another randomly sampled image. We carry out this corruption for 20% of the training data. Corrupted samples are not fixed across epochs. Then, we apply a random sized mask to the top half, and ask the network to predict the missing pixels. We choose two competitive baselines here: our network with the L_1 loss and CE. Fig. 4 illustrates the predictions. As shown, our model converges to the most probable non-corrupted mode without any ambiguity, while other baselines give sub-optimal results. In the next experiment, we add a small white box to the top part of the ground-truth images at different rates. At inference, our model was able to converge to both the modes (Fig. 5), depending on the initial position of z, as the probability of the alternate mode reaches 0.3.

5.2 Automatic Image Colorization

Deep models have tackled this problem using semantic priors (Iizuka et al., 2016; Vitoria et al., 2020), adversarial and L_1 losses (Isola et al., 2017; Zhu et al., 2017a; Lee et al., 2019), or by conversion to a discrete form through binning of color values (Zhang et al., 2016). Although these methods provide

Method	User study		Turing test
	STL	ImageNet	ImageNet
Izuka	21.89	32.28	-
Chroma	32.40	31.67	-
Ours	**45.71**	**36.05**	31.66

Table 1: Colorization: Psychophysical study and Turing test results. All performances are in %.

Method	STL				ImageNet			
	LPIP ↓	PieAPP ↓	SSIM ↑	PSNR ↑	LPIP ↓	PieAPP ↓	SSIM ↑	PSNR ↑
Izuka	0.18	2.37	0.81	24.30	0.17	2.47	0.87	18.43
P2P	1.21	2.69	0.73	17.80	2.01	2.80	0.87	18.43
CIC	0.18	2.81	0.71	22.04	0.19	2.56	0.71	19.11
Chroma	0.16	2.06	0.91	25.57	**0.16**	2.13	0.90	23.33
Ours	**0.12**	**1.47**	**0.95**	**27.03**	**0.16**	**2.04**	**0.92**	**24.51**
Ours (w/o ρ)	0.16	1.90	0.89	25.02	0.20	2.11	0.88	23.21

Table 2: Colorization: Quantitative analysis of our method against the state-of-the-art. Ours perform better on a variety of metrics.

Method	10% corruption				15% corruption				25% corruption			
	LPIP ↓	PieAPP ↓	PSNR ↑	SSIM ↑	LPIP ↓	PieAPP ↓	PSNR ↑	SSIM ↑	LPIP ↓	PieAPP ↓	PSNR ↑	SSIM ↑
DSGAN	0.101	1.577	20.13	0.67	0.189	2.970	18.45	0.55	0.213	3.54	16.44	0.49
PN	**0.045**	**0.639**	27.11	0.88	0.084	**0.680**	20.50	0.71	0.147	0.764	19.41	0.63
CE	0.092	1.134	22.34	0.71	0.134	2.134	19.11	0.63	0.189	2.717	17.44	0.51
P2P	0.074	0.942	22.33	0.79	0.101	1.971	19.34	0.70	0.185	2.378	17.81	0.57
CA	0.048	0.731	26.45	0.83	0.091	0.933	20.12	0.72	0.166	0.822	21.43	0.72
RFR	0.051	0.743	**29.31**	0.85	0.097	1.033	19.22	0.70	0.171	1.127	18.42	0.61
Ours (w/o ρ)	0.053	0.799	27.77	0.83	0.085	0.844	23.22	0.76	0.141	0.812	22.31	0.74
Ours	0.051	0.727	27.83	**0.89**	**0.080**	0.740	**26.43**	**0.80**	**0.129**	**0.760**	**24.16**	**0.77**

Table 3: *Image completion:* Quantitative analysis of our method against state-of-the-art on a variety of metrics.

compelling results, several inherent limitations exist: (a) use of semantic priors results in complex models, (b) adversarial loss suffers from drawbacks (see Sec. 4), and (c) discretization reduces the precision. In contrast, we achieve better results using a simpler model.

The input and the output of the network are l and (a, b) planes respectively (LAB color space). However, since the color distributions of a and b spaces are highly imbalanced over a natural dataset (Zhang et al., 2016), we add another constraint to the cost function E to push the predicted a and b colors towards a uniform distribution: $E = \|a_{gt} - a\| + \|b_{gt} - b\| + \lambda(loss_{kl,a} + loss_{kl,b})$, where $loss_{kl,\cdot} = \mathrm{KL}(\cdot||u(0,1))$. Here, $\mathrm{KL}(\cdot||\cdot)$ is the KL divergence and $u(0,1)$ is a uniform distribution (see App. 3.3). Fig. 7 and Table 2 depict our qualitative and quantitative results, respectively. We demonstrate the superior performance of our method against four metrics: LPIP, PieAPP, SSIM and PSNR (App. 3.2). Fig. 10 depicts examples of multimodality captured by our model (more examples in App. 3.4). Fig. 6 shows colorization behaviour as the z converges during inference.

User study: We also conduct two user studies to further validate the quality of generated samples (Table 1). **a)** In the PSYCHOPHYSICAL STUDY, we present volunteers with batches of 3 images, each generated with a different method. A batch is displayed for 5 secs and the user has to pick the most realistic image. After 5 secs, the next image batch is displayed. **b)** We conduct a TURING TEST to validate our output quality against the ground-truth, following the setting proposed by Zhang et al. (2016). The volunteers are presented with a series of paired images (ground-truth and our output). The images are visible for 1 sec, and then the user has an unlimited time to pick the realistic image.

5.3 IMAGE COMPLETION

In this case, we show that our generic model outperforms a similar capacity GAN (CE) as well as task-specific GANs. In contrast to task-specific models, we do not use any domain-specific modifications to make our outputs perceptually pleasing. We observe that with random irregular and fixed-sized masks, all the models perform well, and we were not able to visually observe a considerable difference (Fig. 8, see App. 3.11 for more results). Therefore, we presented models with a more challenging task: train with random sized square-shaped masks and evaluate the performance

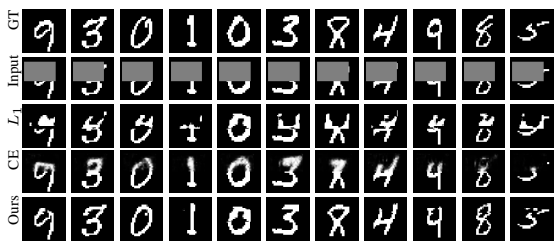


Figure 4: Performance with 20% corrupted data. Our model demonstrates better convergence compared to L_1 loss and a similar capacity GAN (Pathak et al., 2016).

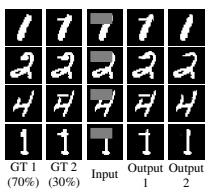


Figure 5: With >30% alternate mode data, our model can converge to both the input modes (cols 4-5).

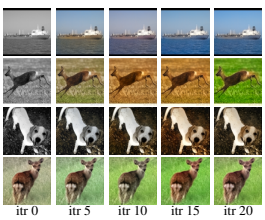


Figure 6: The prediction quality increases as the z traverses to an optimum position at the inference.

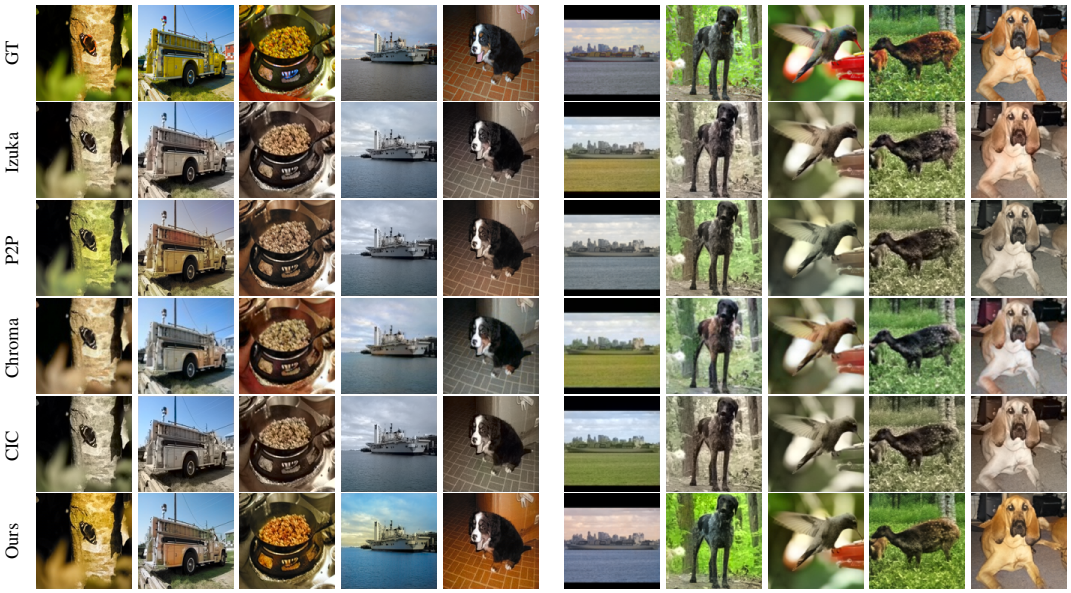


Figure 7: Qualitative comparison against the state-of-the-art on ImageNet (left 5 columns) and STL (right 5 columns) datasets. Our model generally produces more vibrant and balanced color distributions.

Figure 8: Image completion on Celeb-HQ (left) and Facade (right) datasets. We used fixed center masks and random irregular masks (Liu et al., 2018) for Celeb-HQ and Facades datasets, respectively.

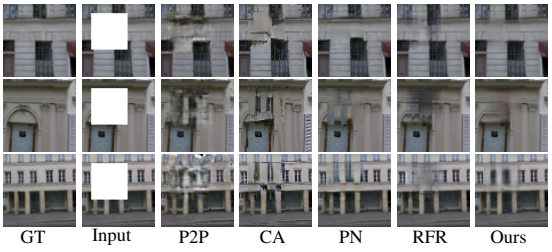


Figure 9: Qualitative comparison for image completion with 25% missing data (models trained with random sized square masks).

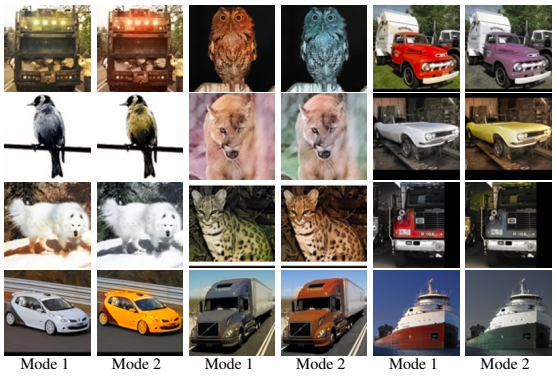


Figure 10: Multiple colorization modes predicted by our model for a single input. *(Best viewed in color).*

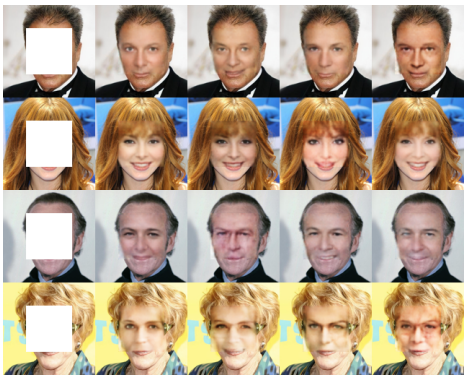


Figure 11: Multi-modality of our predictions on Celeb-HQ dataset. *(Best viewed with zoom)*

Figure 12: Translation from hand bag sketches to images.

Figure 13: Translation from hand shoe sketches to images.

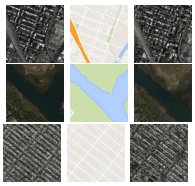
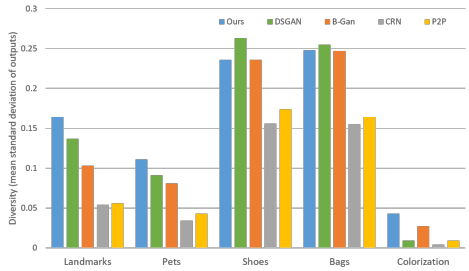


Figure 14: Map to aerial image translation. *From left*: GT, Input and Output. Also see App. 5.2.



.Figure 15: Diversity: Quantitative comparisons.

Figure 16: Translation from facial landmarks to faces.

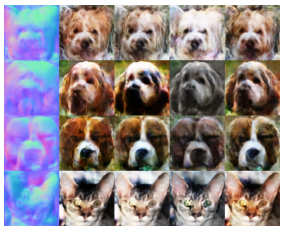


Figure 17: Translation from surface-normals to pet faces.

against masks of varying sizes. Fig. 9 illustrates qualitative results of the models with 25% masked data. As evident, our model recovers details more accurately compared to the state-of-the-art. Notably, all models produce comparable results when trained with a fixed sized center mask, but find this setting more challenging. Table 3 includes a quantitative comparison. Observe that in the case of smaller sized masks, PN performs slightly better than ours, but worse otherwise. We also evaluate the learned features of the models against a downstream classification task (Table 5). First, we train all the models on Facades (Tyleček & Šára, 2013) against random masks, and then apply the trained models on CIFAR10 (Krizhevsky et al., 2009) to extract bottleneck features, and finally pass them through a FC layer for classification (App. 3.7). We compare PN and ours against an oracle (AlexNet features pre-trained on ImageNet) and show our model performs closer to the oracle.

5.3.1 Diversity and other compelling attributes

We also experiment on a diverse set of image translation tasks to demonstrate our generalizability. Fig. 12, 13, 14, 16 and 17 illustrate the qualitative results of *sketch-to-handbag*, *sketch-to-shoes*, *map-to-arial*, *lanmarks-to-faces* and *surface-normals-to-pets* tasks. Fig. 10, 11, 12, 13, 16 and 17 show the ability of our model to converge to multiple modes, depending on the z initialization. Fig. 15 demonstrates the quantitative comparison against other models. See App. 3.4 for further details on experiments. Another appealing feature of our model is its strong convergence properties irrespective of the architecture, hence, *scalability* to different input sizes. Fig. 19 shows examples from image completion and colorization for varying input sizes. We add layers to the architecture to be trained

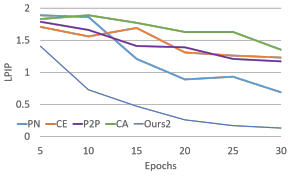


Figure 18: Convergence on image completion (Paris view). Our model exhibits rapid and stable convergence compared to state-of-the-art (PN, CE, P2P, CA).

Method	M10	M40
Sharma et al. (2016)	80.5%	75.5%
Han et al. (2019)	92.2%	90.2%
Achlioptas et al. (2017)	**95.3%**	85.7%
Yang et al. (2018)	94.4%	88.4%
Sauder & Sievers (2019)	94.5%	90.6%
Ramasinghe et al. (2019c)	93.1%	-
Khan et al. (2019)	92.2%	-
Ours	92.4%	**90.9%**

Table 4: Downstream 3D object classification results on ModelNet10 and ModelNet40 using features learned in an unsupervised manner. All results in % accuracy.

Method	Pretext	Acc. (%)
ResNet*	ImageNet Cls.	74.2
PN	Im. Completion	40.3
Ours	Im. Completion	**62.5**

Table 5: Comparison on downstream task (CIFAR10 cls). (*) denotes the oracle case.

Method	M10	M40
CE	10.3	4.6
cVAE	8.7	4.2
Ours	**84.2**	**79.4**

Table 6: Reconstruction mAP of 3d spectral denoising.

Model	CE	PN	Chroma	CIC	P2P	Izuka	RFR	Ours
Flops (1×10^9)	0.634	0.946	1.275	52.839	0.732	14.082	25.64	0.638

Table 7: Model complexity comparison.

on increasingly high-resolution inputs, where our model was able to converge to optimal modes at each scale (App. 3.8). Fig. 18 demonstrates our faster and stable *convergence*. Table. 7 compares the number of flops required by the models for a batch size of 10.

5.4 Denoising of 3D objects in spectral space

Spectral moments of 3D objects provide a compact representation, and help building light-weight networks (Ramasinghe et al., 2020; 2019b; Cohen et al., 2018; Esteves et al., 2018). However, spectral information of 3D objects has not been used before for self-supervised learning, a key reason being the difficulty of learning representations in the spectral domain due to the complex structure and unbounded spectral coefficients. Here, we present an efficient pretext task that is conducted in the spectral domain: denoising 3D spectral maps. We use two types of spectral spaces: spherical harmonics and Zernike polynomials (App. 4). We first convert the 3D point clouds to spherical harmonic coefficients, arrange the values as a 2D map, and mask or add noise to a map portion (App. 3.12). The goal is to recover the original spectral map. Fig. 20 and Table 6 depicts our qualitative and quantitative results. We perform favorably well against other methods. To evaluate the learned features, we use Zernike polynomials, as they are more discriminative compared to spherical harmonics (Ramasinghe et al., 2019a). We first train the network on the 55k ShapeNet objects by denoising spectral maps, and then apply the trained network on the ModelNet10 & 40. The features are then extracted from the bottleneck (similar to Sec. 5.3), and fed to a FC classifier (Table 4). We achieve the state-of-the-art results in ModelNet40 with a simple pretext task.

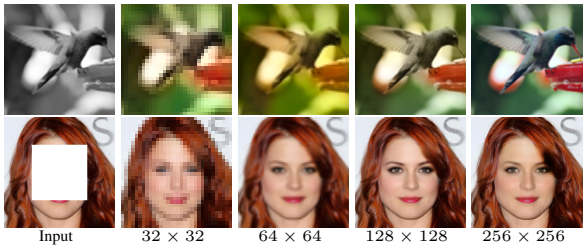


Figure 19: Scalability: we subsequently add layers to the architecture to be trained on increasingly high-resolution inputs

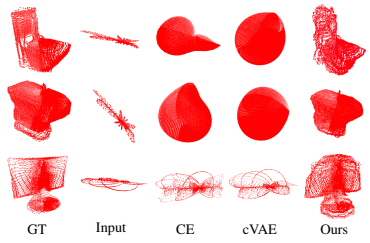


Figure 20: Qualitative comparison of 3D spectral denoising. The results are converted to the spatial domain for a clear visualization.

6 Conclusion

Conditional generation in multimodal domains is a challenging task due to its ill-posed nature. In this paper, we propose a novel generative framework that minimize a family of cost functions during training. Further, it observes the convergence patterns of latent variables and applies this knowledge during inference to traverse to multiple output modes during inference. Despite using a simple and generic architecture, we show impressive results on a diverse set of tasks. The proposed approach demonstrates faster convergence, scalability, generalizability, diversity and superior representation learning capability for downstream tasks.

References

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017. 8

Sanjeev Arora and Yi Zhang. Do gans actually learn the distribution? an empirical study. *arXiv preprint arXiv:1706.08224*, 2017. 1

Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*, 2018. 4

Aayush Bansal, Xinlei Chen, Bryan Russell, Abhinav Gupta, and Deva Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. *arXiv preprint arXiv:1702.06506*, 2017a. 21

Aayush Bansal, Yaser Sheikh, and Deva Ramanan. Pixelnn: Example-based image synthesis. *arXiv preprint arXiv:1708.05349*, 2017b. 21

Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. Cvae-gan: Fine-grained image generation through asymmetric training. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 15

Samuel A Barnett. Convergence problems with generative adversarial networks (gans). *arXiv preprint arXiv:1806.11382*, 2018. 1, 4

Simyung Chang, SeongUk Park, John Yang, and Nojun Kwak. Sym-parameterized dynamic inference for mixed-domain image translation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4803–4811, 2019. 1, 16

Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1511–1520, 2017a. 5

Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017b. 15

Casey Chu, Kentaro Minami, and Kenji Fukumizu. Smoothness and stability in gans. *arXiv preprint arXiv:2002.04185*, 2020. 1, 4

Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018. 9

Aditya Deshpande, Jiajun Lu, Mao-Chuang Yeh, Min Jin Chong, and David Forsyth. Learning diverse image colorization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 15

James R Driscoll and Dennis M Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994. 28

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 3608–3618. Curran Associates, Inc., 2019. 15

Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 52–68, 2018. 9

Ethan Fetaya, Jörn-Henrik Jacobsen, Will Grathwohl, and Richard Zemel. Understanding the limitations of conditional generative models. In *International Conference on Learning Representations*, 2020. 1

Arnab Ghosh, Viveka Kulharia, Vinay P. Namboodiri, Philip H.S. Torr, and Puneet K. Dokania. Multi-agent diverse generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 15

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 27*, pp. 2672–2680. 2014a. 15

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*, pp. 2672–2680, 2014b. 4, 17

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2020. 2

Zhizhong Han, Mingyang Shang, Yu-Shen Liu, and Matthias Zwicker. View inter-prediction gan: Unsupervised representation learning for 3d shapes by learning global shape memories to support local view predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 8376–8384, 2019. 8

Yang He, Bernt Schiele, and Mario Fritz. Diverse conditional image generation by stochastic regression with latent drop-out codes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 406–421, 2018. 4

Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *The European Conference on Computer Vision (ECCV)*, September 2018. 15

Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics (ToG)*, 35(4):1–11, 2016. 4, 5, 15, 28

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017. 1, 5, 15, 21

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2

Salman H Khan, Yulan Guo, Munawar Hayat, and Nick Barnes. Unsupervised primitive discovery for improved 3d generative modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9739–9748, 2019. 8

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

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014. 15

Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017. 1, 4

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 2009. URL http://www.cs.toronto.edu/~kriz/cifar.html. 8

Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *The European Conference on Computer Vision (ECCV)*, September 2018. 15

Soochan Lee, Junsoo Ha, and Gunhee Kim. Harmonizing maximum likelihood with GANs for multimodal conditional generation. In *International Conference on Learning Representations*, 2019. 1, 4, 5, 15

Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7760–7768, 2020. 5

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 85–100, 2018. 7

Antonio Loquercio, Mattia Segù, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *arXiv preprint arXiv:1907.06890*, 2019. 19

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016. 1, 15

Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. In *Advances in neural information processing systems*, pp. 6548–6558, 2019. 15

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*, pp. 1429–1437, 2019. 1

Michaël Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *CoRR*, abs/1511.05440, 2015. 15

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *ArXiv*, abs/1411.1784, 2014. 15

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *International Conference on Learning Representations*, 2019. 1

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 21

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. 2016. 1, 4, 5, 6, 15

Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. Deepsphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. *Astronomy and Computing*, 27:130–146, 2019. 31

Ekta Prashnani, Hong Cai, Yasamin Mostofi, and Pradeep Sen. Pieapp: Perceptual image-error assessment through pairwise preference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1808–1817, 2018. 21, 23

Sameera Ramasinghe, Salman Khan, and Nick Barnes. Volumetric convolution: Automatic representation learning in unit ball. *arXiv preprint arXiv:1901.00616*, 2019a. 9, 31

Sameera Ramasinghe, Salman Khan, Nick Barnes, and Stephen Gould. Representation learning on unit ball with 3d roto-translational equivariance. *International Journal of Computer Vision*, pp. 1–23, 2019b. 9

Sameera Ramasinghe, Salman Khan, Nick Barnes, and Stephen Gould. Spectral-gans for high-resolution 3d point-cloud generation. *arXiv preprint arXiv:1912.01800*, 2019c. 8, 31

Sameera Ramasinghe, Salman Khan, Nick Barnes, and Stephen Gould. Blended convolution and synthesis for efficient discrimination of 3d shapes. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 21–31, 2020. 9

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015. 1, 15

Herbert E. Robbins. A stochastic approximation method. *Annals of Mathematical Statistics*, 22: 400–407, 2007. 17

Min-cheol Sagong, Yong-goo Shin, Seung-wook Kim, Seung Park, and Sung-jea Ko. Pepsi : Fast image inpainting with parallel decoding network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 15

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*, pp. 2234–2242, 2016. 23

Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. In *Advances in Neural Information Processing Systems*, pp. 12942–12952, 2019. 8

Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *European Conference on Computer Vision*, pp. 236–250. Springer, 2016. 8

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*. 2015. 15

Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. *arXiv preprint arXiv:1902.03984*, 2019. 1

Radim Tyleček and Radim Šára. Spatial pattern templates for recognition of objects with regular structure. In *Proc. GCPR*, Saarbrucken, Germany, 2013. 8

Patricia Vitoria, Lara Raad, and Coloma Ballester. Chromagan: Adversarial picture colorization with semantic class distribution. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 2445–2454, 2020. 1, 4, 5, 15, 28

Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *Advances in Neural Information Processing Systems 31*. 2018. 15

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pp. 1398–1402. Ieee, 2003. 21

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 21

You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow. *ACM Transactions on Graphics (TOG)*, 37(4):95, 2018. 15

Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tianchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. *arXiv preprint arXiv:1901.09024*, 2019. 5

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 206–215, 2018. 8

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018a. 15

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5505–5514, 2018b. 5

Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1486–1494, 2019. 1, 4, 5, 15, 23

Lin Zhang, Lei Zhang, Xuanqin Mou, and David Zhang. Fsim: A feature similarity index for image quality assessment. *IEEE transactions on Image Processing*, 20(8):2378–2386, 2011. 21

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pp. 649–666. Springer, 2016. 1, 5, 6, 15, 21, 28

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 21, 23

Song-Yang Zhang, Zhifei and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017. 21

Xian Zhang, Xin Wang, Bin Kong, Youbing Yin, Qi Song, Siwei Lyu, Jiancheng Lv, Canghong Shi, and Xiaojie Li. Domain embedded multi-model generative adversarial networks for image-based face inpainting. *ArXiv*, abs/2002.02909, 2020. 15

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 15

Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems 30*. 2017a. 5, 15

Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in neural information processing systems*, pp. 465–476, 2017b. 5

APPENDIX

1 RELATED WORK

Conditional Generative Modeling. Conditional generation involves modeling the data distribution given a set of conditioning variables that control of modes of the generated samples. With the success of *VAEs* (Kingma & Welling, 2014) and *GANs* (Goodfellow et al., 2014a) in standard generative modeling tasks, their conditioned counterparts (Sohn et al., 2015; Mirza & Osindero, 2014) have dominated conditional generative tasks recently (Vitoria et al., 2020; Zhang et al., 2016; Isola et al., 2017; Pathak et al., 2016; Lee et al., 2019; Zhu et al., 2017a; Bao et al., 2017; Lee et al., 2018; Zeng et al., 2019). While probabilistic latent variable models such as VAEs generate relatively low quality samples and poor likelihood estimates at inference (Maaløe et al., 2019), GAN based models perform significantly better at high dimensional distributions like natural images but demonstrate unstable training behaviour. A distinct feature of GANs is its mapping of points from a random noise distribution to the various modes of the output distribution. However, in the conditional case where an additional loss is incorporated to enforce the conditioning on the input, the significantly better performance of GANs is achieved at the expense of multimodality; the conditioning loss pushes the GAN to learn to mostly ignore its noise distribution. In fact, some works intentionally ignore the noise input in order to achieve more stable training (Isola et al., 2017; Pathak et al., 2016; Mathieu et al., 2015; Xie et al., 2018).

Multimodality. *Conditional VAE-GANs* are one popular approach for generating multimodal outputs (Bao et al., 2017; Zhu et al., 2017a) using the VAE's ability to enforce diversity through its latent variable representation and the GAN's ability to enforce output fidelity through its learnt discrimanator model. *Mixture models* (Chen & Koltun, 2017b; Ghosh et al., 2018; Deshpande et al., 2017) that discretize the output space are another approach. Domain specific *disentangled representations* (Lee et al., 2018; Huang et al., 2018) and explicit encoding of multiple modes as inputs Zhu et al. (2016); Isola et al. (2017) have also been successful in generating diverse outputs. *Sampling-based loss* functions enforcing similarity at a distribution level (Lee et al., 2019) have also been successful in multimoal generative tasks. Further, the use of additional *specialized reconstruction losses* (often using higher-level features extracted from the data distribution) and attention mechanisms also achieves multimodality through intricate model architectures in domain specific cases (Zeng et al., 2019; Chen & Koltun, 2017b; Vitoria et al., 2020; Zhang et al., 2016; Iizuka et al., 2016; Zhang et al., 2020; Yu et al., 2018a; Sagong et al., 2019; Wang et al., 2018; Iizuka et al., 2016).

We propose a simpler direction through our domain-independent energy function based approach that is also capable of learning generic representations that better support downstream tasks. Notably, our work contrasts from energy based models previously investigated for likelihood modeling due to their simplicity, however, such models are notoriously difficult to train especially on high-dimensional spaces (Du & Mordatch, 2019).

VAEs and other related work. Variational auto encoders (VAE) are a special class of autoencoders that are trained in a manner that ensures the latent space has good properties allowing the generation of new data. In VAEs, for an input x, the latent space is modeled as a probability distribution $q(z|x)$, which is sampled from a known family of distributions (typically Gaussian), as the true posterior distribution $p(z|x)$ is intractable. However, assuming the posterior distribution of the latent space as a Gaussian distribution constrains the quality of the generated data distribution, as the true distribution may be far from a Gaussian. Therefore, it is beneficial to model $q(z|x)$ as a more complex distribution, in order to generate high dimensional data distributions.

As a solution, Maaløe et al. (2016) suggested using a set of auxiliary variables a to improve the flexibility of $q(z|x)$. The key idea is to obtain a complex marginal $q(z|x) = \int q(z|a,x)p(a,x)da$, which can be non-Gaussian. On the other hand, Normalizing flows (NF) (Rezende & Mohamed, 2015), among other benefits, provides an ideal mechanism for the above task. NFs apply a series of bijective mappings $NF : P(z_i) \rightarrow P(z_{i+1})$, where $P(z_{i+1})$ is typically more complex compared to $P(z_i)$. In contrast, we do not explicitly model our latent space as a probability distribution. However, we draw some interesting analogies from a probabilistic perspective as follows: our latent space ζ can be interpreted as a set of energy surfaces $E_{x_j} : \zeta \rightarrow \mathbb{R}$, as $E_{x_j} = ||y_j^g - G(x, z_j)||$ for each ground truth mode y_j^g. From this perspective, Fig. 32 in the appendix illustrates the energy heatmaps for the toy example. As shown, high energies are indicated by a brighter color. Since our system has a finite

energy, the combined energy $E_x = \sum_j E_{x_j}$ can be transformed to a probability distribution via the Gibbs measure as $p'(z) = \frac{1}{T(\beta)} \exp(-\beta E_x(z))$, where $T(\cdot)$ is the partition function. Note that this probability is not restricted to a simple distribution.

A critical difference between the VAEs and our model is that we do not sample directly from $p'(z)$, since to obtain $p'(z)$, we need to integrate E_x over the latent space. However, our predictor network $\mathcal{Z}$ learns the high probability coordinates $\{z^*\}$ of $p'(z)$, and is able to converge to such locations at inference. This probabilistic perspective of our latent space (or the corresponding energy surface) is intuitively justified by the convergence samples shown in Fig. 52 in appendix. The intermediate samples we obtain as we go from z to z^* also produce plausible results, however, the visual quality at the $z*$ is maximized, indicating high $p'(z = z^*|x)$. In contrast to NFs, our model does not explicitly learn the probability distributions, rather, the predictor network learns to converge to the high probability areas in complex distributions. Also, the complexity of the $p'(z)$ increases with the complexity and the multimodality of the corresponding higher dimensional target data distribution. Therefore, the required dimensionality of the z tends to increase in such cases. A property of NFs is that each transformation affects only a small volume in the original space, hence, we need a higher number of layers to work with high dimensional spaces (the volume grows exponentially with the dimension of the space). In contrast, we did not observe such an increase in the required capacity of the predictor with the dimension of z.

The model proposed by Chang et al. (2019) also uses a separate input S, that can vary the output of the generator, in cases where multiple loss components are used. The variable S is used both as an input to the generator and also to control the weights of the loss components while training. Interestingly, at the inference, the model is able to approximate the change in loss based on the input S, and generate diverse outputs. However, this method is more useful in scenarios where the required diversity is in the form of different styles, which can be induced by different loss functions.

2 Theoretical results

2.1 Proof for Eq. 3

$$||\mathcal{G}(x_j, w^*, z^*_{i,j}) - \mathcal{G}(x_j, w^*, z_0)|| = || \int_{z_0}^{z^*_{i,j}} \nabla_z \mathcal{G}(x_j, w^*, z) dz|| \tag{5}$$

Let $\gamma(t)$ be a straight path from z_0 to $z^*_{i,j}$, where $\gamma(0) = z_0$ and $\gamma(1) = z^*_{i,j}$. Then,

$$= || \int_0^1 \nabla_z \mathcal{G}(x_j, w^*, \gamma(t)) \frac{d\gamma}{dt} dt|| \tag{6}$$

$$= || \int_0^1 \nabla_z \mathcal{G}(x_j, w^*, \gamma(t))(z^*_{i,j} - z^*_0) dt|| \tag{7}$$

$$= ||(z^*_{i,j} - z^*_0) \int_0^1 \nabla_z \mathcal{G}(x_j, w^*, \gamma(t)) dt|| \tag{8}$$

$$\leq \left\|(z^*_{i,j} - z^*_0)\right\| \left\| \int_0^1 \nabla_z \mathcal{G}(x_j, w^*, \gamma(t)) dt \right\| \tag{9}$$

On the other hand the Lipschitz constraint ensures,

$$\left\|\nabla_z \mathcal{G}(x_j, w^*, \gamma(t))\right\| \leq \lim_{\epsilon \to 0} \frac{\left\|\mathcal{G}(x_j, w^*, \gamma(t)) - \mathcal{G}(x_j, w^*, \gamma(t+\epsilon))\right\|}{\|z_t - z_{t+\epsilon}\|} \leq C, \tag{10}$$

where C is a constant. Combining Eq. 9 and 10 we get,

$$\frac{\left\|\mathcal{G}(x_j, w^*, z^*_{i,j}) - \mathcal{G}(x_j, w^*, z_0)\right\|}{\left\|z^*_{i,j} - z_0\right\|} \leq \int_0^1 \left\|\nabla_z \mathcal{G}(x_j, w^*, \gamma(t))\right\| dt \leq C. \tag{11}$$

2.2 CONVERGENCE OF THE TRAINING ALGORITHM.

Proof: Let us consider a particular input x_j and an associated ground truth $y^g_{i,j}$. Then, for this particular case, we denote our cost function to be $\hat{E}_{i,j} = d(w, z)$. Further, a family of cost functions can be defined as,

$$f_w(z) = d(w, z), \tag{12}$$

for each $w \sim \omega$. Further, let us consider an arbitrary initial setting (z_{init}, w_{init}). Then, with enough iterations, gradient descent by $\nabla_z f_w(z)$ converges z_{init} to,

$$\bar{z} = \arg\inf_{z \in \zeta} f_w. \tag{13}$$

Next, with enough iterations, gradient descent by $\nabla_w f_w(\bar{z})$ converges w to,

$$\bar{w} = \arg\inf_{w \in \omega} f_w(\bar{z}). \tag{14}$$

Observe that $f_{\bar{w}}(\bar{z}) \leq f_{w_{init}}$, where the equality occurs when $\nabla_z f_w(z) = \nabla_w f_w(\bar{z}) = 0$. If $f_w(z)$ has a unique global minima, repeating Equation 13 and 14 converges to that global minima, giving $\{z^*_{i,j}, w^*_{i,j}\}$. It is straight forward to see that using a small number of iterations (usually one in our case) for each sample set for Equation 14, i.e., stochastic gradient descent, gives us,

$$\{z^*_{i,j}, w^*\} = \arg\min_{z_{i,j} \in \zeta, w \in \omega} \mathbb{E}_{i \in I, j \in J}[\hat{E}_{i,j}], \tag{15}$$

where w^* is fixed for all samples and modes (Robbins, 2007). Note that the proof is valid only for the convex case, and we rely on stochastic gradient descent to converge to at least a good local minima, as commonly done in many deep learning settings.

2.3 PROOF FOR REMARK

Remark: *Consider a generator $G(x, z)$ and a discriminator $D(x, z)$ with a finite capacity, where x and z are input and the noise vector, respectively. Then, consider an arbitrary input x_j and the corresponding set of ground truths $\{y^g_{i,j}\}, i = 1, 2, ..N$. Further, let us define the optimal generator $G^*(x_j, z) = \hat{y}, \hat{y} \in \{y^g_{i,j}\}$, $L_{GAN} = \mathbb{E}_i[\log D(y^g_{i,j})] + \mathbb{E}_z[\log(1 - D(G(x_j, z))]$ and $L_\ell = \mathbb{E}_{i,z}[|y^g_{i,j} - G(x_j, z)|]$. Then, $G^* \neq \hat{G}^*$ where $\hat{G}^* = \arg\min_G \max_D L_{GAN} + \lambda L_\ell, \forall \lambda \neq 0$.*
Proof.

It is straightforward to derive the equilibrium point of $\arg\min_G \max_D L_{GAN}$ from the original GAN formulation. However, for clarity, we show some steps here.

Let,

$$V(G, D) = \arg\min_G \max_D \mathbb{E}_i[\log D(y^g_{i,j})] + \mathbb{E}_z[\log(1 - D(G(x_j, z))] \tag{16}$$

Let $p(\cdot)$ denote the probability distribution. Then,

$$V(G, D) = \arg\min_G \max_D \int_\Upsilon p(y^g_{\cdot,j}) \log D(y^g_{\cdot,j}) + p(\bar{y}_{\cdot,j})(\log(1 - D(G(x_j, z))dy \tag{17}$$

$$V(G, D) = \arg\min_G \max_D \mathbb{E}_{y \sim y^g_{\cdot,j}}[\log D(y)] + \mathbb{E}_{y \sim \bar{y}_{\cdot,j}}[\log(1 - D(y)] \tag{18}$$

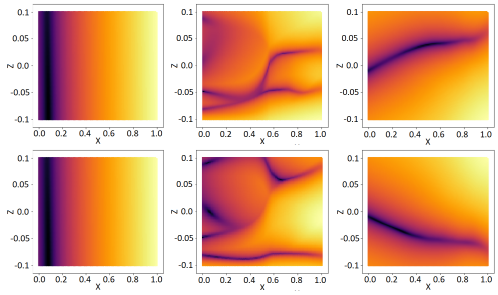
Consider the inner loop. It is straightforward to see that $V(G, D)$ is maximized w.r.t. D when $D(y) = \frac{p(y^g_{\cdot,j})}{p(y^g_{\cdot,j}) + p(\bar{y}_{\cdot,j})}$. Then,

$$C(G) = V(G, D) = \arg\min_G \mathbb{E}_{y \sim y^g_{\cdot,j}}[\log \frac{p(y^g_{\cdot,j})}{p(y^g_{\cdot,j}) + p(\bar{y}_{\cdot,j})}] + \mathbb{E}_{y \sim \bar{y}_{\cdot,j}}[\log \frac{p(\bar{y}_{\cdot,j})}{p(y^g_{\cdot,j}) + p(\bar{y}_{\cdot,j})}] \tag{19}$$

Then, following the **Theorem 1** from Goodfellow et al. (2014b), it can be shown that the global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p(y^g_{\cdot,j}) = p(\bar{y}_{\cdot,j})$.

Next, consider the L_1 loss for x_j,

$$L_1 = \frac{1}{N} \sum_i \left| y^g_{i,j} - G(x_j, z, w) \right| \tag{20}$$



(a) 0 epochs (b) 5 epochs (c) 10 epochs

Figure 21: The behaviour of cost heatmaps $\hat{E}$ against (x, z) as the training progresses (toy example). The latent space gets increasingly structured as $w \rightarrow w^*$. Also, in (c) the network intelligently puts the optimal latent codes further apart as the distance between the two ground truth modes ($m = 4$ and $m = -4$) keeps increasing.

$$\nabla_w L_1 = -\frac{1}{N} \sum_i \text{sgn}(y^g_{i,j} - G(x_j, z, w)) \nabla_w (G(x_j, z, w)) \tag{21}$$

For L_1 to approach to a minima, $\nabla_w L_1 \rightarrow 0$. Since $\{y^g_{i,j}\}$ is not a singleton, when $L_1 \rightarrow 0$, $G(x_j, z, w) \neq \hat{y} \in \{y^g_{i,j}\}$.

Now, let us consider the L_2 loss,

$$L_2 = \frac{1}{N} \sum_i \left\| y^g_{i,j} - G(x_j, z, w) \right\|^2 \tag{22}$$

$$\nabla_w L_2 = -\frac{2}{N} \sum_i (y^g_{i,j} - G(x_j, z, w)) \nabla_w (G(x_j, z, w)) \tag{23}$$

For $\nabla_w L_2 \rightarrow 0$, $G(x_j, z, w) \rightarrow \frac{1}{N} \sum_i y^g_{i,j}$. However, omitting the very specific case where $(\frac{1}{N} \sum_i y^g_{i,j}) \in \{y^g_{i,j}\}$, which is highly unlikely in a complex distribution, as $L_2 \rightarrow 0$, $G(x_j, z, w) \neq \hat{y} \in \{y^g_{i,j}\}$. Therefore, the goals of $\arg \min_G \max_D L_{GAN}$ and λL_ℓ are contradictory and $G^* \neq \hat{G}^*$. Note that we do not extend our proof to high order L losses as it is intuitive.

2.4 Lipschitz continuity and structuring of the latent space

Enforcing the Lipschitz constraint encourages meaningful structuring of the latent space: suppose $z^*_{1,j}$ and $z^*_{2,j}$ are two optimal codes corresponding to two ground truth modes for a particular input. Since $\left\| z^*_{2,j} - z^*_{1,j} \right\|$ is lower bounded by $\frac{\left\| \mathcal{G}(x_j, w^*, z^*_{2,j}) - \mathcal{G}(x_j, w^*, z^*_{1,j}) \right\|}{L}$, where L is the Lipschitz constant, the minimum distance between the two latent codes is proportional to the difference between the corresponding ground truth modes. Also, in practice, we observed that this encourages the optimum latent codes to be placed sparsely. Fig. 21 illustrates a visualization from the toy example. As the training progresses, the optimal $\{z^*\}$ corresponding to minimas of $\hat{E}$ are identified and placed sparsely. Note that as expected, at the 10^{th} epoch the distance between the two optimum z^* increases as x goes from 0 to 1, in other words, as the $\left\| 4(x, x^2, x^3) - (-4(x, x^2, x^3)) \right\|$ increases.

Practical implementation is done as follows: during the training phase, a small noise e is injected to the inputs of $\mathcal{Z}$ and $\mathcal{G}$, and the networks are penalized for any difference in output. More formally, $L_{\mathcal{Z}}$ and $\hat{E}$ now become, $L_1[z_{t+1}, \mathcal{Z}(z_t, h)] + \alpha L_1[\mathcal{Z}(z_t + e, h + e), \mathcal{Z}(z_t, h)]$ and $L_1[y^g, \mathcal{G}(h, z)] + \alpha L_1[\mathcal{G}(h + e, z + e), \mathcal{G}(h, z)]$, respectively. Fig. 25 illustrates the procedure.

2.5 Towards a measurement of uncertainty

In Bayesian approaches, the uncertainty is represented using the distribution of the network parameters ω. Since a network output is unique for fixed $\bar{w} \sim \omega$, sampling from the output is equivalent to sampling from ω. Often, ω is modeled as a parametric distribution or obtained through sampling, and at inference, the model uncertainty can be estimated as $\mathbb{VAR}_{p(y|x)}(y)$. One intuition behind this is that for more confident inputs, $p(y|x, w)$ will showcase less variance over the distribution

of ω—hence lower $\mathbb{VAR}_{p(y|x)}(y)$—as the network parameters have learned redundant information (Loquercio et al., 2019).

As opposed to sampling from the distribution of network parameters, we model the optimal z^* for a particular input as a probability distribution $p(z^*)$, and measure $\mathbb{VAR}_{p(y|x)}(y)$ where $p(y|x) = \int p(y|x,z^*)p(z^*|x)dz$. Our intuition is that in the vicinity of well observed data $\mathbb{VAR}_{p(y|x)}(y)$ is lower, since for training data 1) we enforce the Lipschitz constraint on $\mathcal{G}(x,z)$ over (x,z) and 2) $\hat{E}(y^g, \mathcal{G}(x,z))$ resides in a relatively stable local minima against z^* for observed data, as in practice, $z^* = \mathbb{E}_{epochs}[z^*] + \epsilon$ for a given x, where ϵ is some random noise which is susceptible to change over each epoch. Further, Let (x, z^*) and y^g be the inputs to a network $\mathcal{G}$ and the corresponding ground truth label, respectively.

Formally, let $p(y^g|x,z^*) = \mathcal{N}(y^g; \mathcal{G}(x,z^*), \alpha\mathbb{I})$ and $z^* \sim \mathcal{U}(|z^* - \mathbb{E}(z^*)| < \delta)$, where α is some variable describing the noise in the input x and δ is a small positive scalar. Then,

$$\mathbb{COV}_{p(y^g|x)}(y^g) \approx= \frac{1}{K}\sum_{k=1}^{K}[\alpha_k\mathbb{I}] + \overline{\mathbb{COV}}(\mathcal{G}(x,z^*)). \tag{24}$$

where $\overline{\mathbb{COV}}$ is the sample covariance.

proof: $\mathbb{E}_{p(y^g|x)}(y^g) = \int y^g p(y^g|x)dy^g$.

$= \int y^g[\int \mathcal{N}(y^g; \mathcal{G}(x,z^*), \alpha\mathbb{I})p(z^*|x)dz^*]dy^g$

$= \int[\int y^g \mathcal{N}(y^g; \mathcal{G}(x,z^*), \alpha\mathbb{I})p(z^*|x)dy^g]dz^*$

$= \int[\int y^g \mathcal{N}(y^g; \mathcal{G}(x,z^*), \alpha\mathbb{I})dy^g]p(z^*|x)dz^*$

$= \int \mathcal{G}(x,z^*)p(z^*|x)dz^*$

Let $\pi\delta^2 = A$, and $p(z^*|x) \approx \frac{1}{A}$. Then, by Monte-Carlo approximation,

$\approx \frac{1}{K}\sum_{k=1}^{K}\mathcal{G}(x, z_k^*)$

Next, consider,

$\mathbb{COV}_{p(y^g|x)}(y^g) = \mathbb{E}_{p(y^g|x)}((y^g)(y^g)^T) - \mathbb{E}_{p(y^g|x)}(y^g)\mathbb{E}_{p(y^g|x)}(y^g)^T$

$= \int\int (y^g)(y^g)^T p(y^g|x,z^*)p(z^*|x)dz^*dy^g - \mathbb{E}_{p(y^g|x)}(y^g)\mathbb{E}_{p(y^g|x)}(y^g)^T$

$= \int[\mathbb{COV}_{p(y^g|x,z^*)} + \mathbb{E}_{p(y^g|x,z^*)}\mathbb{E}^T_{p(y^g|x,z^*)}p(z^*|x)dz - \mathbb{E}_{p(y^g|x)}(y^g)\mathbb{E}_{p(y^g|x)}(y^g)^T$

$\approx \frac{1}{K}\sum_{k=1}^{K}[\alpha_k\mathbb{I} + G(x,z_k^*)G(x,z_k^*)^T] - \frac{1}{K^2}[(\sum_{k=1}^{K}G(x,z_k^*))(\sum_{k=1}^{K}G(x,z_k^*))^T]$.

$= \frac{1}{K}\sum_{k=1}^{K}[\alpha_k\mathbb{I}] + \overline{\mathbb{COV}}(\mathcal{G}(x,z^*))$

Note that in similar to Bayesian uncertainty estimations, where an approximate distribution $q(w)$ is used to estimate $p(w|D)$, where D is data, our model sample from the an empirical distribution $p(z^*|x)$. In practice, we treat α_k as a constant over all the samples–hence omit from the calculation—and use stochastic forward passes to obtain Eq. 24. Then, the diagonal entries are used to calculate the uncertainty in the each dimension of the output. We test this hypothesis on the toy example and the colorization task, as shown in Fig. 22 and Fig. 23, respectively.

3 Experiments

3.1 Experimental architectures

For the experiments on images, we mainly use 128×128 size inputs. However, to demonstrate the scalability, we use several different architectures and show that the proposed framework is capable of converging irrespective of the architecture. Fig. 24 shows the architectures for different input sizes.

For training, we use the Adam optimizer with hyper-parameters $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$, and a learning rate $lr = 1 \times 10^{-5}$. We use batch normalization after each convolution layer, and

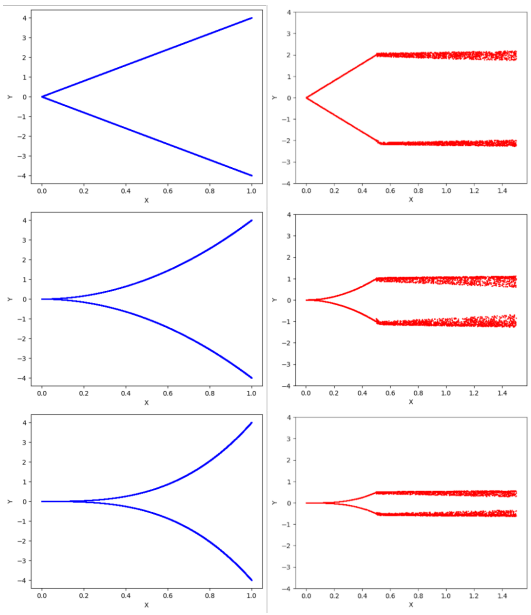


Figure 22: The uncertainty measurement illustration with the toy example. *(left-column: ground truth, right-column: prediction)*. We train the model with $x \in [0, 0.5]$ and test with $x \in [0, 1.5]$. During the testing, we add a small Gaussian noise to z^* at each x and get stochastic outputs. As illustrated, the sample variance (the uncertainty measurement) increases as x deviates from the observed data portion.

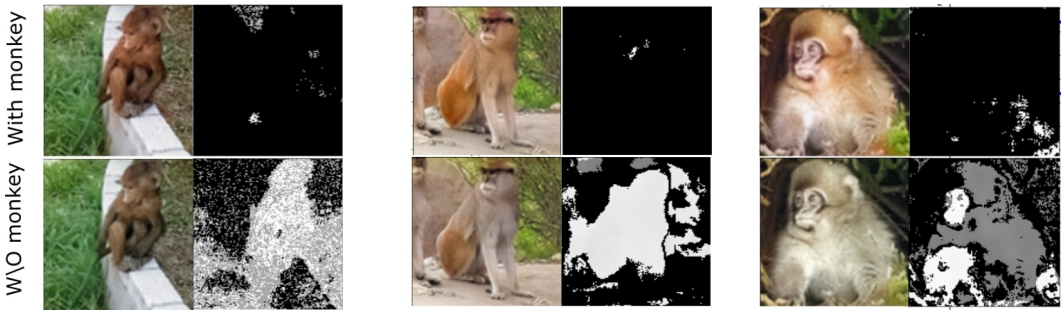


Figure 23: Colorization predictions for models trained with and without monkey class. Output images are shown side by side with corresponding uncertainty maps. For models trained without monkey data, high uncertainty is predicted for pixels belonging to the monkey portion (intensity is higher for high uncertainty).

Method	a	b
Chroma	0.71	0.78
Izuka	0.68	0.63
Ours	0.82%	0.80%

Table 8: IOU of the predicted color distributions against the ground truth. Our method shows better results.

leaky ReLu as the activation, except the last layer where we use $tanh$. All the weights are initialized using a random normal distribution with 0 mean and 0.5 standard deviation. Furthermore, we use a batch size of 20 for training, though we did not observe much change in performance for different batch sizes. We choose the dimensions of z to be $10, 16, 32, 64$ for 32×32, 64×64, 128×128, 256×256 input sizes, respectively. An important aspect to note here is that the dimension of z should not be increased too much, as it would increase the search space for z unnecessarily. While training, z is updated 20 times for a single $\mathcal{G}, \mathcal{H}$ update. Similarly, at inference, we use 20 update steps for z , in order to converge to the optimal solution. All the values are chosen empirically.

3.2 Evaluation metrics

Although heavily used in the literature, per pixel metrics such as PSNR does not effectively capture the perceptual quality of an image. To overcome this shortcoming, more perceptually motivated metrics have been proposed such as SSIM Wang et al. (2004), MSSIM Wang et al. (2003), and FSIM Zhang et al. (2011). However the similarity of two images is largely context dependant, and may not be captured by the aforementioned metrics. As a solution, recently, two deep feature based perceptual metrics–LPIP Zhang et al. (2018) and PieAPP Prashnani et al. (2018)–were proposed, which coincide well with the human judgement. To cover all these aspects, we evaluate our experiments against four metrics: LPIP, PieAPP, PSNR and SSIM.

3.3 Unbalanced color distributions

The color distribution of a natural dataset in a and b planes (LAB space) are strongly biased towards low values. If not taken into account, the loss function can be dominated by these desaturated values. Richard *et al.* Zhang et al. (2016) addressed this problem by rebalancing class weights according to the probability of color occurrence. However, this is only possible in a case where the output domain is discretized. To tackle this problem in the continuous domain, we push the output color distribution towards a uniform distribution as explained in Sec. 5.2 in the main paper.

3.4 Multimodality

An appealing attribute of our network is its ability to converge to multiple optimal modes at inference. A few such examples are shown in Fig. 27, Fig. 26, Fig. 31 Fig. 32, Fig. 29, Fig. 30 and Fig. 28. For the *facial-land-marks-to-faces* experiment, we used the UTKFace dataset (Zhang & Qi, 2017). For the *surface-normals-to-pets* experiment, we used the Oxford Pet dataset (Parkhi et al., 2012). In order to get the surface normal images, we follow Bansal Bansal et al. (2017b). First, we crop the bounding boxes of pet faces and then apply PixelNet (Bansal et al., 2017a) to extract surface normals. For *maps-to-ariel* and *edges-to-photos* experiments, we used the datasets provided by Isola et al. (2017).

For measuring the diversity, we adapt the following procedure: 1) we generate 20 random samples from the model. 2) calculate the mean pixel value μ_i of each sample. 3) pick the closest sample s_m to the average of all the mean pixels $\lambda = \frac{1}{20}\sum_{i=1}^{20} \mu_i$. 4) pick the 10 samples which have maximum mean pixel distance from s_m. 5) calculate the mean standard deviation of the 10 samples picked in step 4. 6) repeat the experiment 5 times for each model and get the expected standard deviation.

3.5 Colorization on STL dataset

Additional colorization examples on the STL dataset are shown in Fig. 34. We also compare the color distributions of the predicted a, b planes with state-of-the-art. The results are shown in Fig. 33 and Table 8. As evident, our method predicts the closest color distribution to the ground truth.

3.6 Colorization on ImageNet dataset

Additional colorization examples on the ImageNet dataset are shown in Fig. 35.

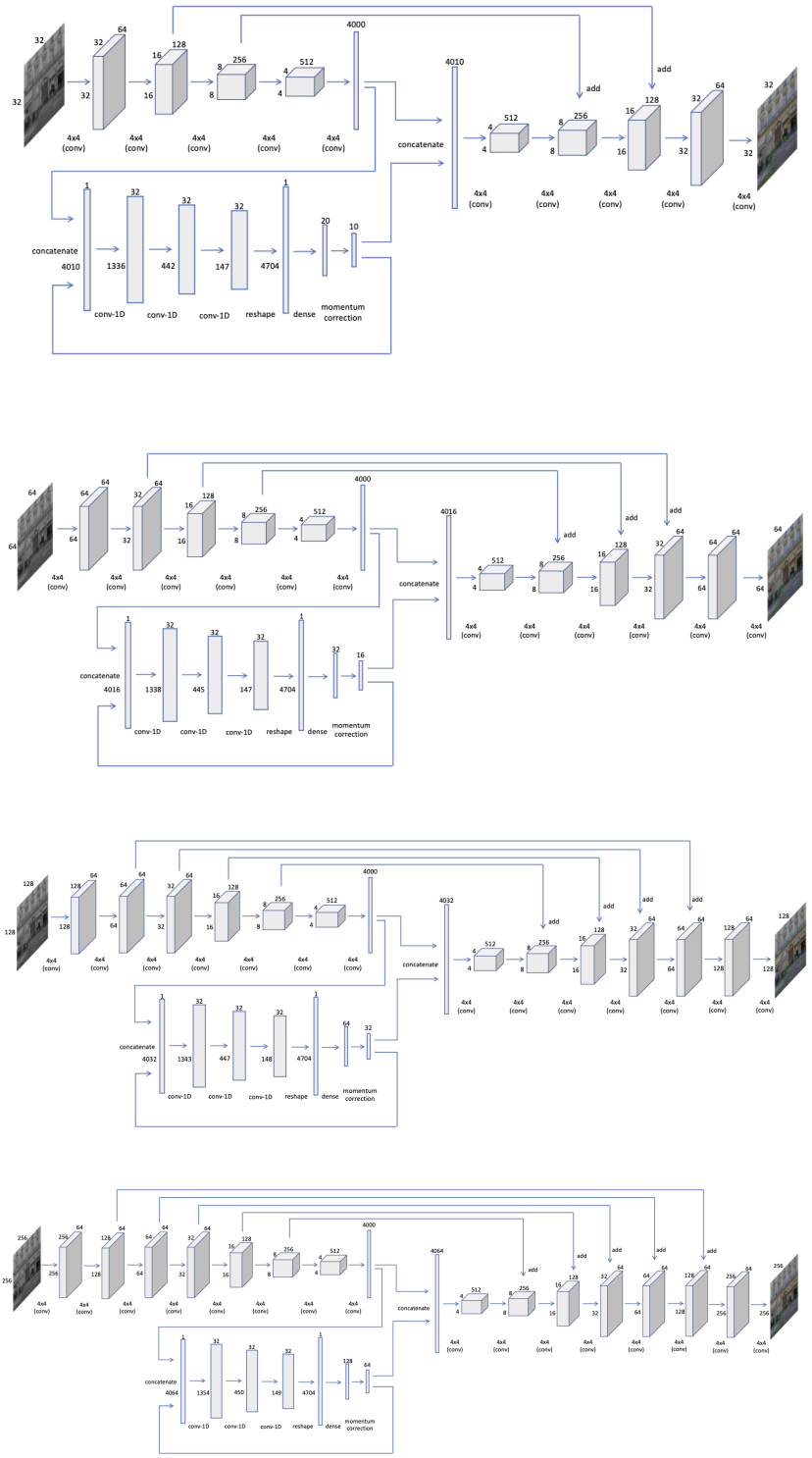


Figure 24: The model architecture for various input sizes. The same general structure in maintained with minimal changes to accomodate for the changing input size.

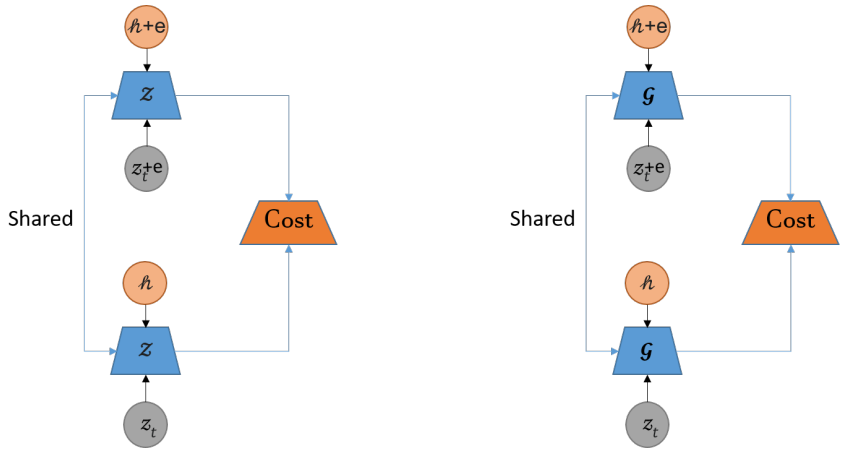


Figure 25: We enforce the Lipschitz continuity on both $\mathcal{G}$ and $\mathcal{Z}$.

Dimensionality	LPIP	PieAPP	Diversity
5	1.05	3.40	0.01
10	0.58	2.91	0.018
16	0.14	1.89	0.021
32	**0.12**	**1.47**	0.043
64	0.27	1.71	**0.048**
128	0.69	2.12	0.043

Table 9: Ablation study against the dimension of z for the colorization task (128×128 inputs).

3.7 Self-supervised learning setup

Here we evaluate the performance of our model on down-stream tasks, using three distinct setups involving bottleneck features of trained models. The bottleneck layer features (of models trained on some dataset) are fed to a fully-connected layer and trained on a different dataset.

The baseline experiment uses the output of the penultimate layer in a Resnet-50 trained on ImageNet for classification as the bottleneck features. The comparison to state-of-the-art experiment involves Zeng et al. (2019) where the five outputs of its multi-scale decoder are max-pooled and concatenated to use as the bottleneck features. The outputs of layers before this were also experimented with, and the highest performance was obtained for these selected features. In our network, the output of the encoder network was used as the bottleneck features.

3.8 Scalability

One promising attribute of the proposed method compared to the state-of-the-art is its scalability. In other words, we propose a generic framework which is not bound to the architecture, hence, the model can be scaled to different input sizes without affecting the convergence behaviour. To demonstrate this, we use 4 different architectures and train them on 4 different input sizes (32×32, 64×64, 128×128, 256×256) on the same tasks: image completion and colorization. The different architectures we use are shown in Fig. 24.

3.9 Ablation study on the z dimension

To demonstrate the effect of dimension of z on the model accuracy, we conduct an ablation study for the colorization task for the input size 128×128. Table 9 shows the results. The quality of the outputs increases to a maximum when $dim(z) = 32$, and then decreases. This is intuitive because when the search space of z gets unnecessarily high, it becomes difficult for $\mathcal{Z}$ to learn the paths to optimum modes, due to limited capacity.

3.10 User studies

Evaluation of synthesized images is an open problem (Salimans et al., 2016). Although recent metrics such as LPIP (Zhang et al., 2018) and PieAPP (Prashnani et al., 2018) have been proposed,

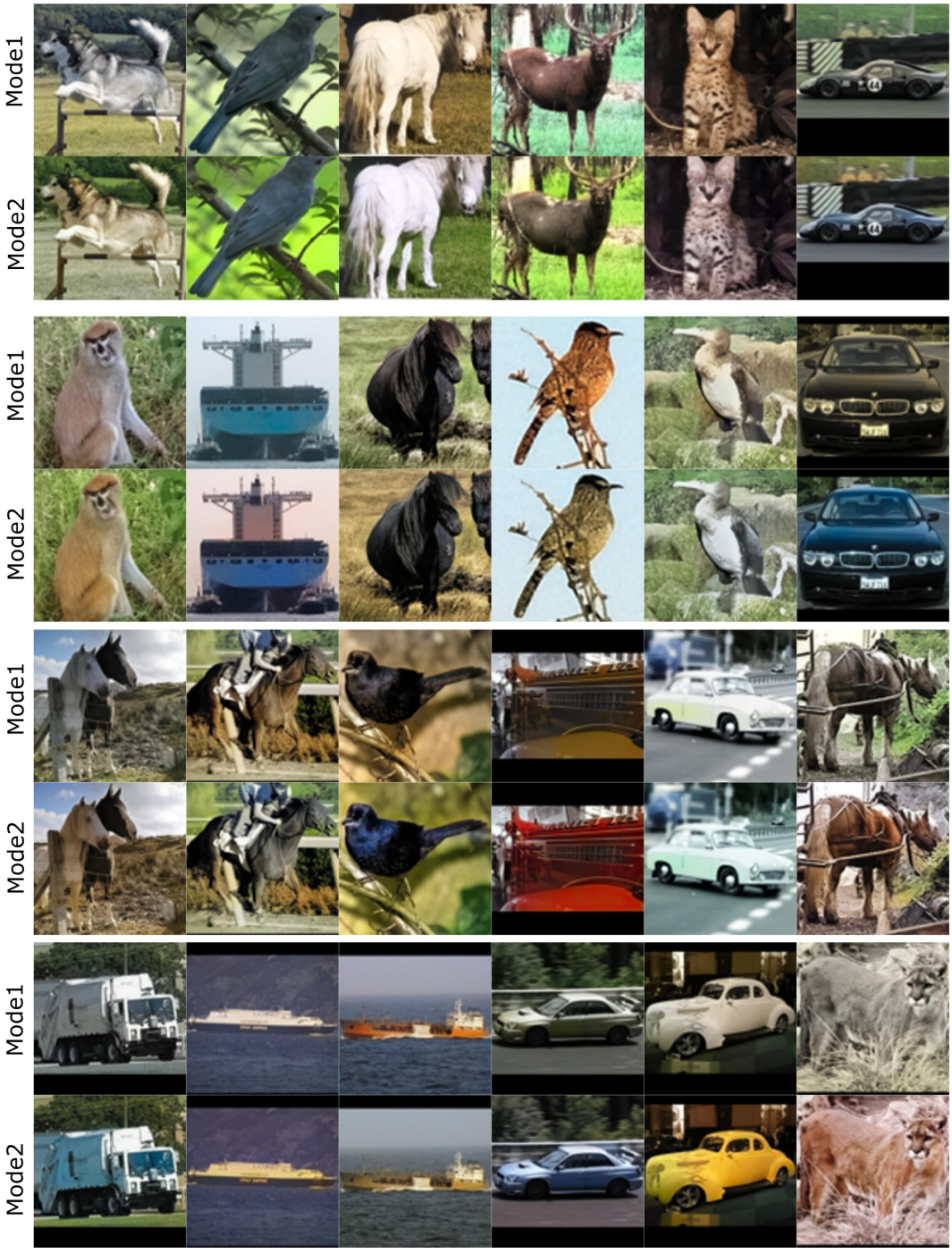


Figure 26: Multimodel predictions of our model in colorization

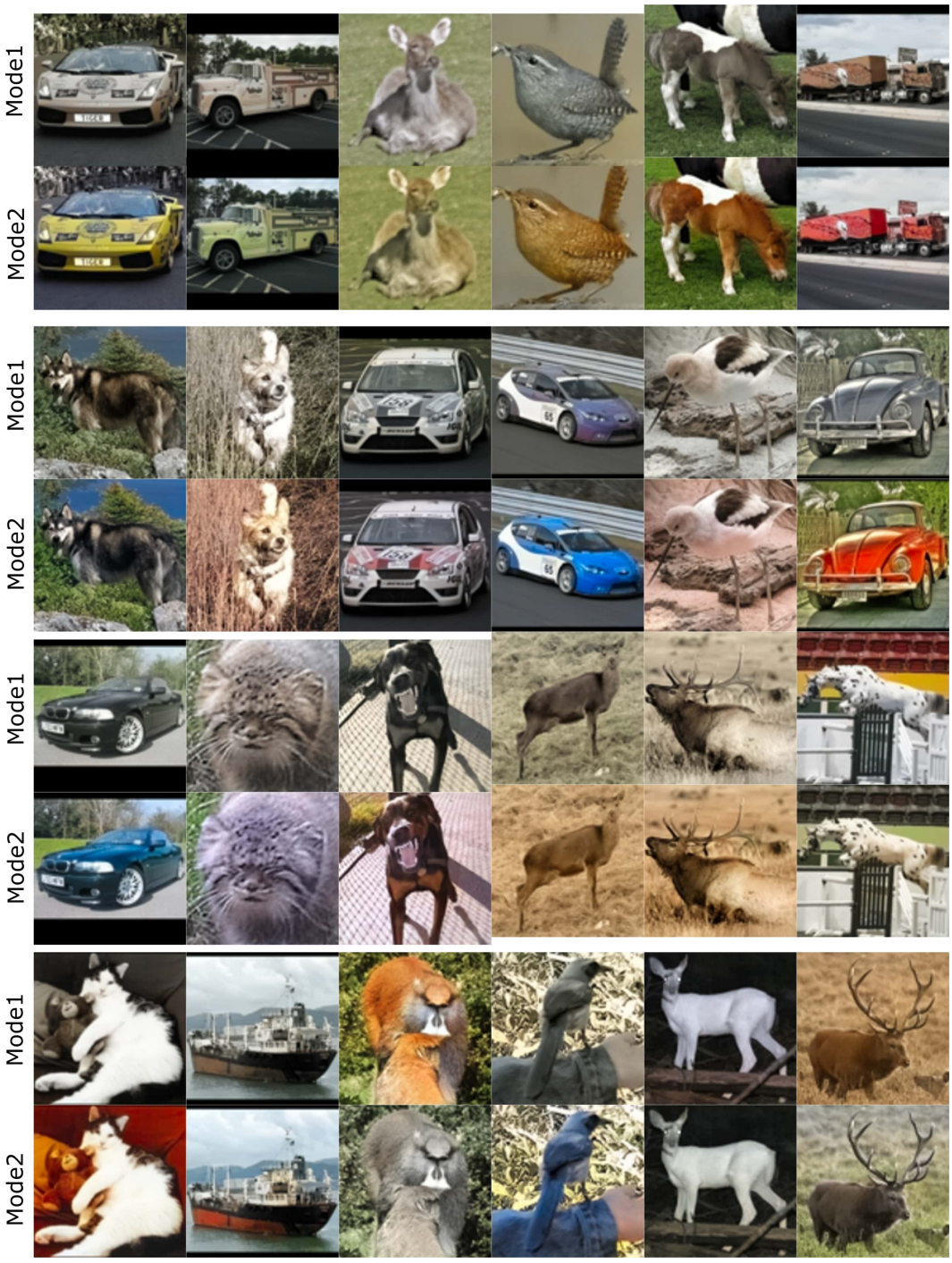


Figure 27: Multimodel predictions of our model in colorization

Figure 28: Multimodel predictions of our model in landmarks-to-faces.

Figure 29: Multimodel predictions of our model in face inpainting.

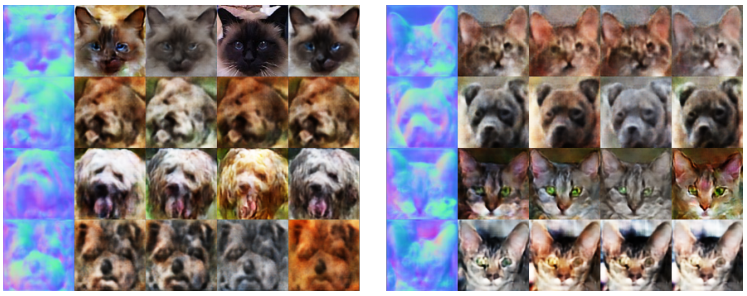


Figure 30: Multimodel predictions of our model in surface-normals-to-pet-faces. Note that this is generally a difficult task due to the diverse texture.

Dataset	Celeb-HQ	Facades
GT	59.11%	55.75%
Ours	40.89%	44.25%

Table 10: Turing Test for GT vs ours on popular image datasets Celeb-HQ and Facades.

which coincide closely with human judgement, perceptual user studies remain the preferred method. Therefore, to evaluate the quality of our synthesized images in the colorization task, we conduct two types of user studies: a Turing test and a psychophysical study. In the Turing test, we show the users a series of paired images, ground truth and our predictions, and ask the users to pick the most realistic image. Here, following Zhang et al. (2016), we display each image for 1 second, and then give the users an unlimited amount of time to make the choice. For the psychophysical study, we choose the two best performing methods according to the LPIP metric: Vitoria et al. (2020) and Iizuka et al. (2016). We create a series of batches of three images, Vitoria et al. (2020), Iizuka et al. (2016) and ours, and ask the users to pick the best quality image. In this case, each batch is shown to the users for 5 seconds, and the users have to make this decision during that time. We conduct the Turing test on ImageNet, and the psychophysical study on both ImageNet and STL datasets. For each test, we use 500 randomly sampled batches and ~ 15 users.

We also conduct Turing tests to evaluate the image completion tasks on Facades and Celeb-HQ datasets. The results are shown in Table 10.

3.11 Image completion

The additional image completion examples are provided in Figs. 36 and 37. Our turing test results on Celeb-HQ and Facades are shown in Table 10.

3.12 3D spectral map denoising

In this experiment, we use two types of spectral moments: spherical harmonics and Zernike polynomials (see App. 4). The minimum number of sample points required to accurately represent a finite energy function in a particular function space depends on the used sampling theorem. According to Driscoll and Healy's theorem Driscoll & Healy (1994), $4N^2$ equiangular sampled points are needed to represent a function on $\mathbb{S}^2$ using spherical moments at a maximum degree N. Therefore, we compute the first 16384 spherical moments of 3D objects where $l \leq 128$ by sampling 256×256 equiangular points in θ and ϕ directions, where $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$. Afterwards, we arrange the spherical moments as a 128×128 feature map, and convolve with a 2×2 kernel with stride size 2 to downsample the feature map to 64×64 size. The output is then fed to 64-size architecture. We add Gaussian noise and mask portions of the spectral map to corrupt it. Afterwards, the model is trained to de-noise the input.

For Zernike polynomials, we compute the first 100 moments for each 3D object where $n \leq 9$, and arrange the moments as a 10×10 feature map. Then, the feature map is upsampled using transposed convolution by using a 5×5 kernel and with a stride size 3. The upsamapled feature map is fed to a 32-size network and trained end-to-end to denoise. We first train the network on 55k objects in ShapeNet, and then apply the trained network on the Modelnet10 and Modelnet40 to extract the bottleneck features. These features are then fed to a single fully connected layer for classification.

Figure 31: Multimodel predictions of our model in sketch-to-shoes translation.

Figure 32: Multimodel predictions of our model in sketch-to-bag translation.

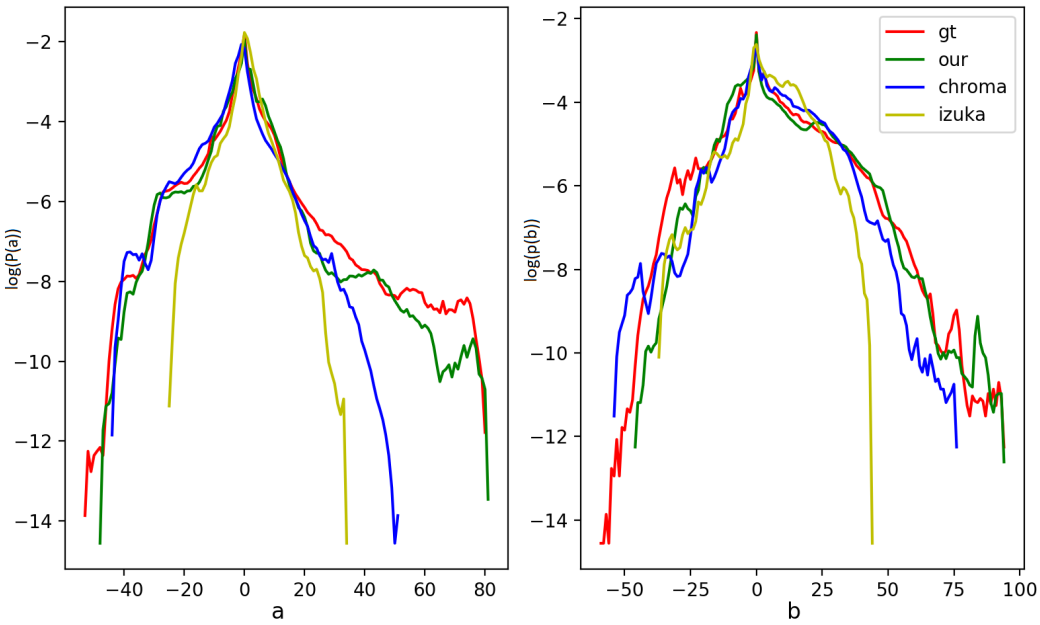


Figure 33: Color distribution comparison of a, b planes. Our method produces the closest distribution to the ground truth.

4 Spectral domain representation of 3D objects

Spherical harmonics and Zernike polynomials are orthogonal and complete functions in $\mathbb{S}^2$ and $\mathbb{B}^3$, respectively, hence, 3D point clouds can be represented by a set of coefficients corresponding to a linear combination of these functions Perraudin et al. (2019); Ramasinghe et al. (2019a;c).

4.1 Spherical harmonics

Spherical harmonics are complete and orthogonal functions defined on the unit sphere ($\mathbb{S}^2$) as,

$$Y_{l,m}(\theta,\phi) = (-1)^m \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}} P_l^m(\cos\phi) e^{im\theta}, \tag{25}$$

where $\theta \in [0, 2\pi]$ is the azimuth angle, $\phi \in [0, \pi]$ is the polar angle, $l \in \mathbb{Z}^+$, $m \in \mathbb{Z}$, and $|m| < l$. Here, $P_l^m(\cdot)$ is the associated Legendre function defined as,

$$P_l^m(x) = (-1)^m \frac{(1-x^2)^{m/2}}{2^l l!} \frac{d^{l+m}}{dx^{l+m}}(x^2-1)^l. \tag{26}$$

Spherical harmonics demonstrate the following orthogonal property,

$$\int_0^{2\pi}\int_0^{\pi} Y_l^m(\theta,\phi) Y_{l'}^{m'}(\theta,\phi)^\dagger \sin\phi \, d\phi d\theta = \delta_{l,l'}\delta_{m,m'}, \tag{27}$$

where † denotes the complex conjugate and,

$$\delta_{m,m'} = \begin{cases} 1, & \text{if } m = m' \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

Since spherical harmonics are complete in $\mathbb{S}^2$, any function $f : \mathbb{S}^2 \to \mathbb{R}$ with finite energy can be rewritten as

$$f(\theta,\phi) = \sum_l \sum_{m=-l}^{l} \hat{f}(l,m) Y_{l,m}(\theta,\phi), \tag{29}$$

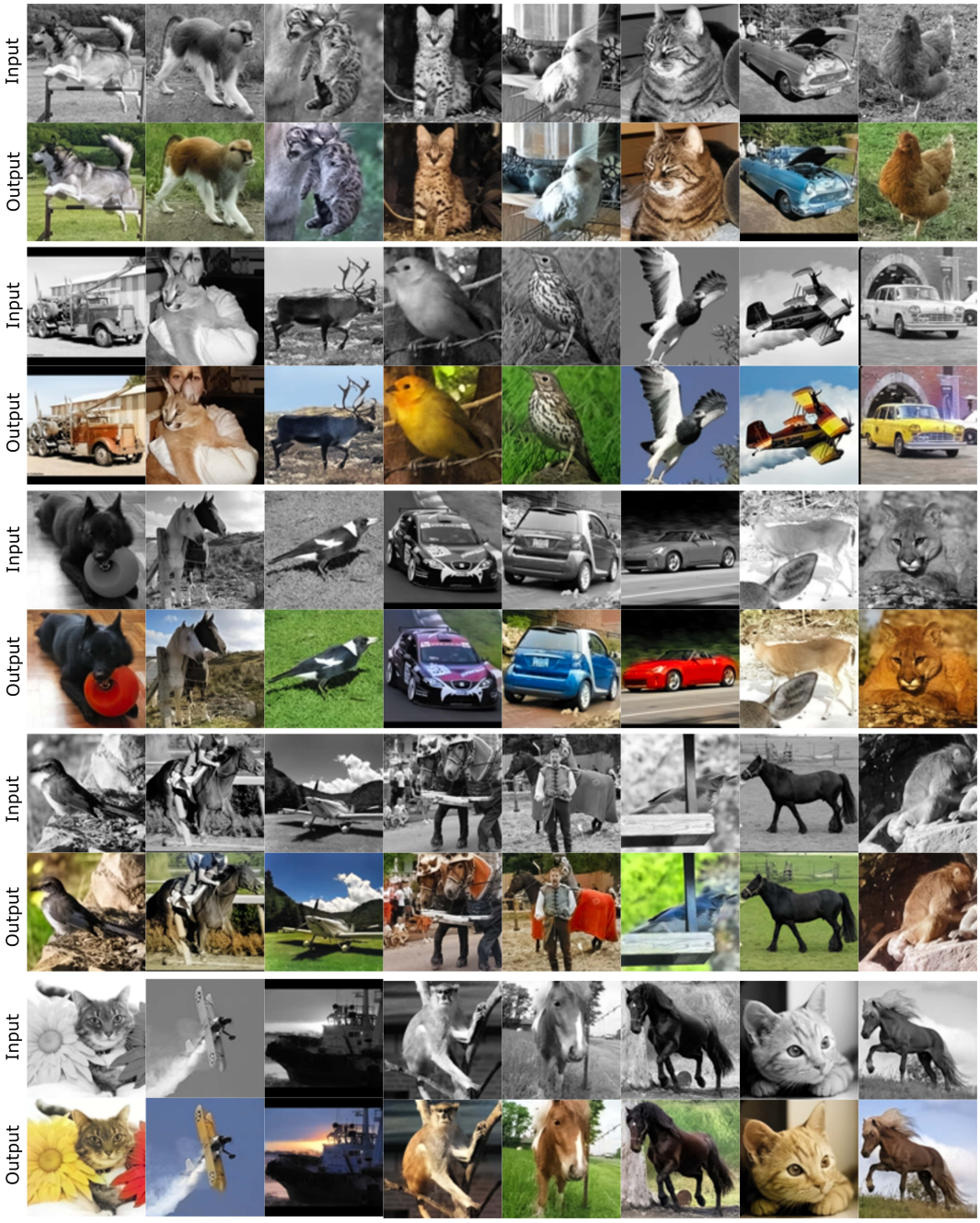


Figure 34: Qualitative results of our model in the colorization task on STL dataset.

Figure 35: Qualitative results of our model in the colorization task on ImageNet dataset.

Figure 36: Qualitative results of our model in the image completion task on Celeb-HQ dataset.

Figure 37: Qualitative results of our model in the image completion task on Facades dataset.

where,

$$\hat{f}(l,m) = \int_0^{\pi} \int_0^{2\pi} f(\theta,\phi) Y_l^m(\theta,\phi)^{\dagger} \sin\phi \, d\phi d\theta. \tag{30}$$

4.2 3D ZERNIKE POLYNOMIALS

3D Zernike polynomials are complete and orthogonal on $\mathbb{B}^3$ and defined as,

$$Z_{n,l,m}(r,\theta,\phi) = R_{n,l}(r) Y_{l,m}(\theta,\phi), \tag{31}$$

where,

$$R_{n,l}(r) = \sum_{v=0}^{(n-1)/2} q_{nl}^v r^{2v+l}, \tag{32}$$

and q_{nl}^v is a scaler defined as

$$q_{nl}^v = \frac{(-1)^{\frac{(n-l)}{2}}}{2^{(n-l)}} \sqrt{\frac{2n+3}{3}} \binom{(n-l)}{\frac{(n-l)}{2}} (-1)^v \frac{\binom{\frac{(n-l)}{2}}{v} \binom{2(\frac{(n-l)}{2}+l+v)+1}{(n-l)}}{\binom{\frac{(n-l)}{2}+l+v}{\frac{(n-l)}{2}}}. \tag{33}$$

Here $Y_{l,m}(\theta,\phi)$ is the spherical harmonics function, $n \in \mathbb{Z}^+$, $l \in [0,n]$, $m \in [-l,l]$ and $n-l$ is even. 3D Zernike polynomials also show orthogonal properties as,

$$\begin{aligned} \int_0^1 \int_0^{2\pi} \int_0^{\pi} Z_{n,l,m}(\theta,\phi,r) Z_{n',l',m'}^{\dagger} r^2 \sin\phi \, dr d\phi d\theta \\ = \frac{4\pi}{3} \delta_{n,n'} \delta_{l,l'} \delta_{m,m'}, \end{aligned} \tag{34}$$

Since Zernike polynomials are complete in $\mathbb{B}^3$, any function $f : \mathbb{B}^3 \to \mathbb{R}$ with finite energy can be rewritten as,

$$f(\theta,\phi,r) = \sum_{n=0}^{\infty} \sum_{l=0}^{n} \sum_{m=-l}^{l} \Omega_{n,l,m}(f) Z_{n,l,m}(\theta,\phi,r) \tag{35}$$

where $\Omega_{n,l,m}(f)$ can be obtained using

$$\Omega_{n,l,m}(f) = \int_0^1 \int_0^{2\pi} \int_0^{\pi} f(\theta,\phi,r) Z_{n,l,m}^{\dagger} r^2 \sin\phi \, dr d\phi d\theta. \tag{36}$$

5 IMAGE-TO-IMAGE TRANSLATION

5.1 SKETCH-TO-SHOES QUALITATIVE RESULTS

Additional qualitative results of the sketch-to-shoe translation task are shown in Fig. 38.

5.2 MAP-TO-PHOTO QUALITATIVE RESULTS

Additional qualitative results of the map-to-photo translation task are shown in Fig. 39.

6 CONVERGENCE AT INFERENCE

A key aspect of our method is the optimization of the predictions at inference. Fig. 40 and Fig. 41 demonstrate this behaviour on the MNIST image completion and STL colorization tasks, respectively.

Figure 38: Qualitative results of our model in sketch-to-shoe translation.

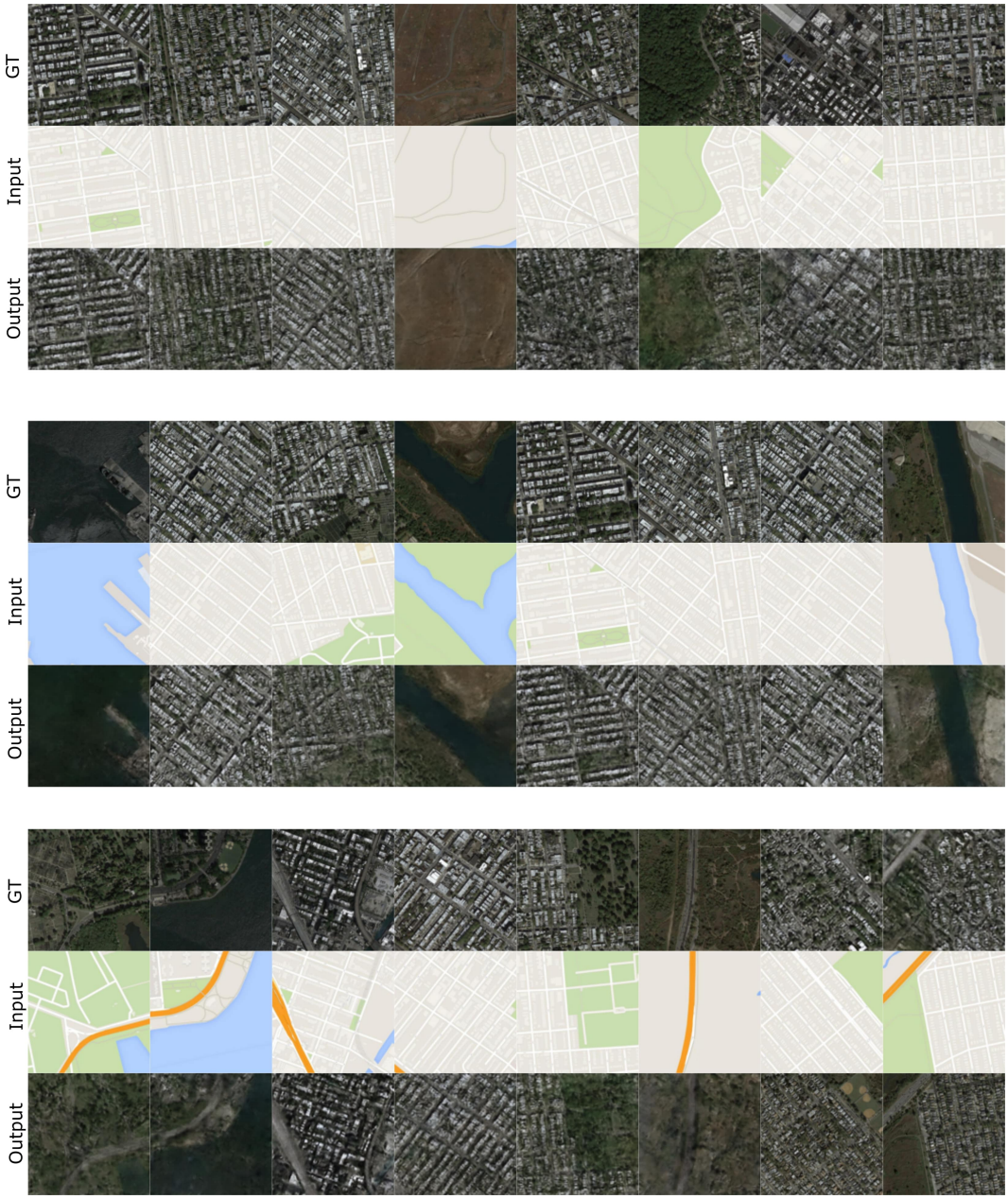


Figure 39: Qualitative results of our model in map-to-photo translation.

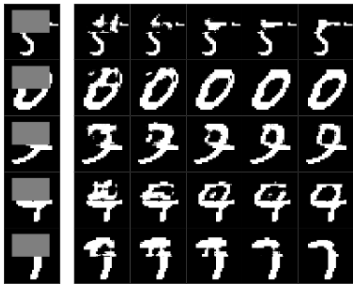


Figure 40: Output gets better as the z traverse to the optimum position at inference. Left column is the input. Five right columns show outputs at iterations 2, 4, 6, 8 and 10 (from left to right).

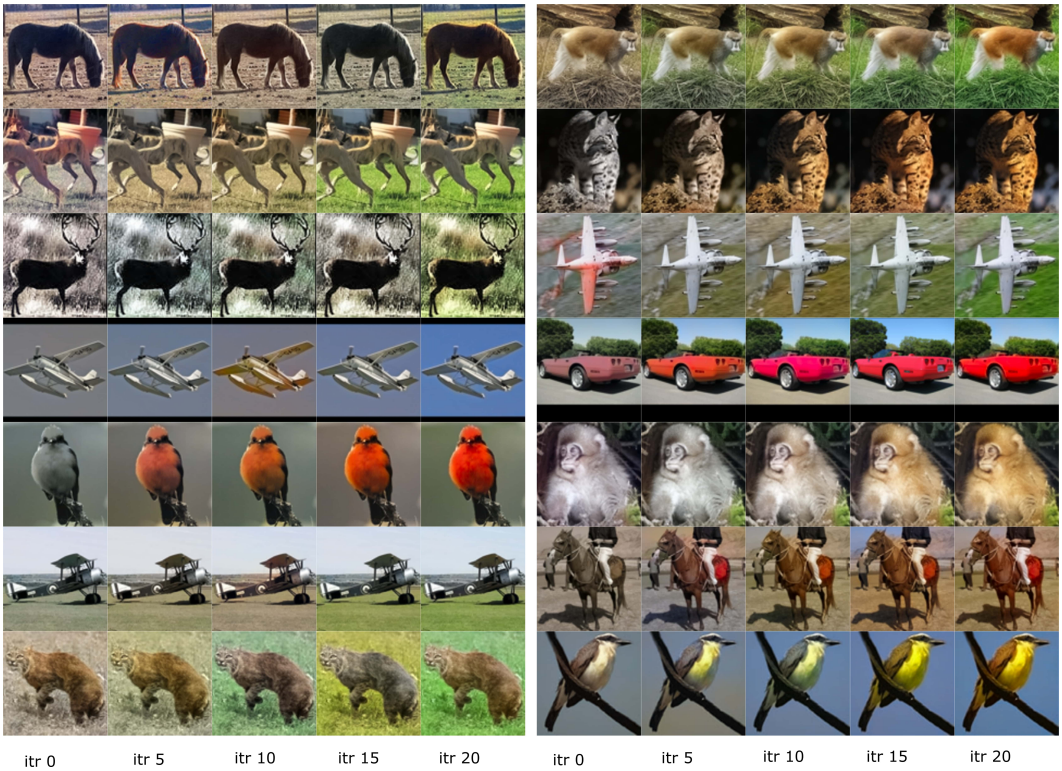


Figure 41: Output quality increases as $z \rightarrow z^*$ at inference.