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,

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

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

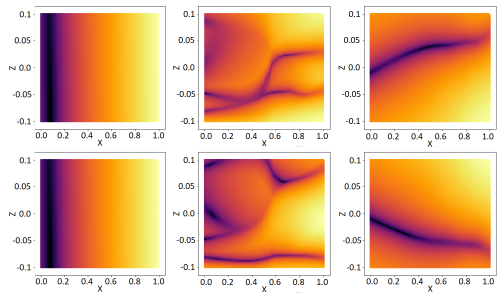
Consider the inner loop. It is straightforward to see that $V(G, D)$ is maximized w.r.t. D when $D(y) = \frac{p(y^g_{\cdot,j})}{p(y^g_{\cdot,j}) + p(\bar{y}_{\cdot,j})}$. Then,

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

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

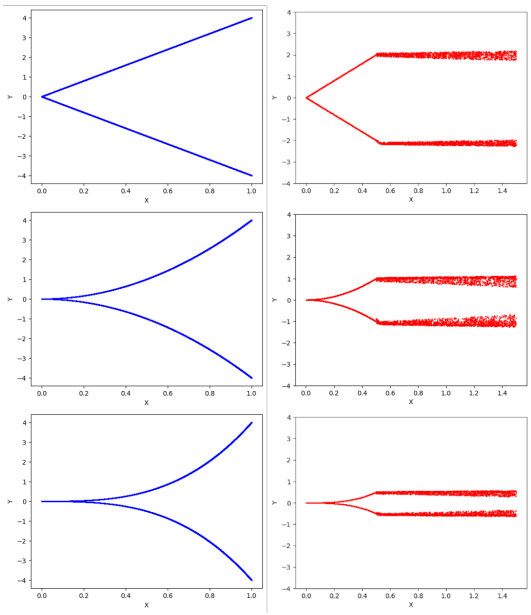


Figure 22: The uncertainty measurement illustration with the toy example. *(left-column: ground truth, right-column: prediction)*. We train the model with $x \in [0, 0.5]$ and test with $x \in [0, 1.5]$. During the testing, we add a small Gaussian noise to z^* at each x and get stochastic outputs. As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

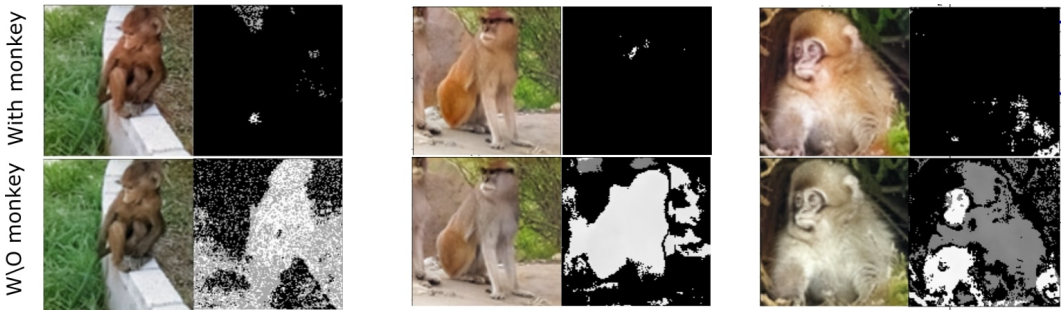


Figure 23: Colorization predictions for models trained with and without monkey class. Output images are shown side by side with corresponding uncertainty maps. For models trained without monkey data, high uncertainty is predicted for pixels belonging to the monkey portion (intensity is higher for high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

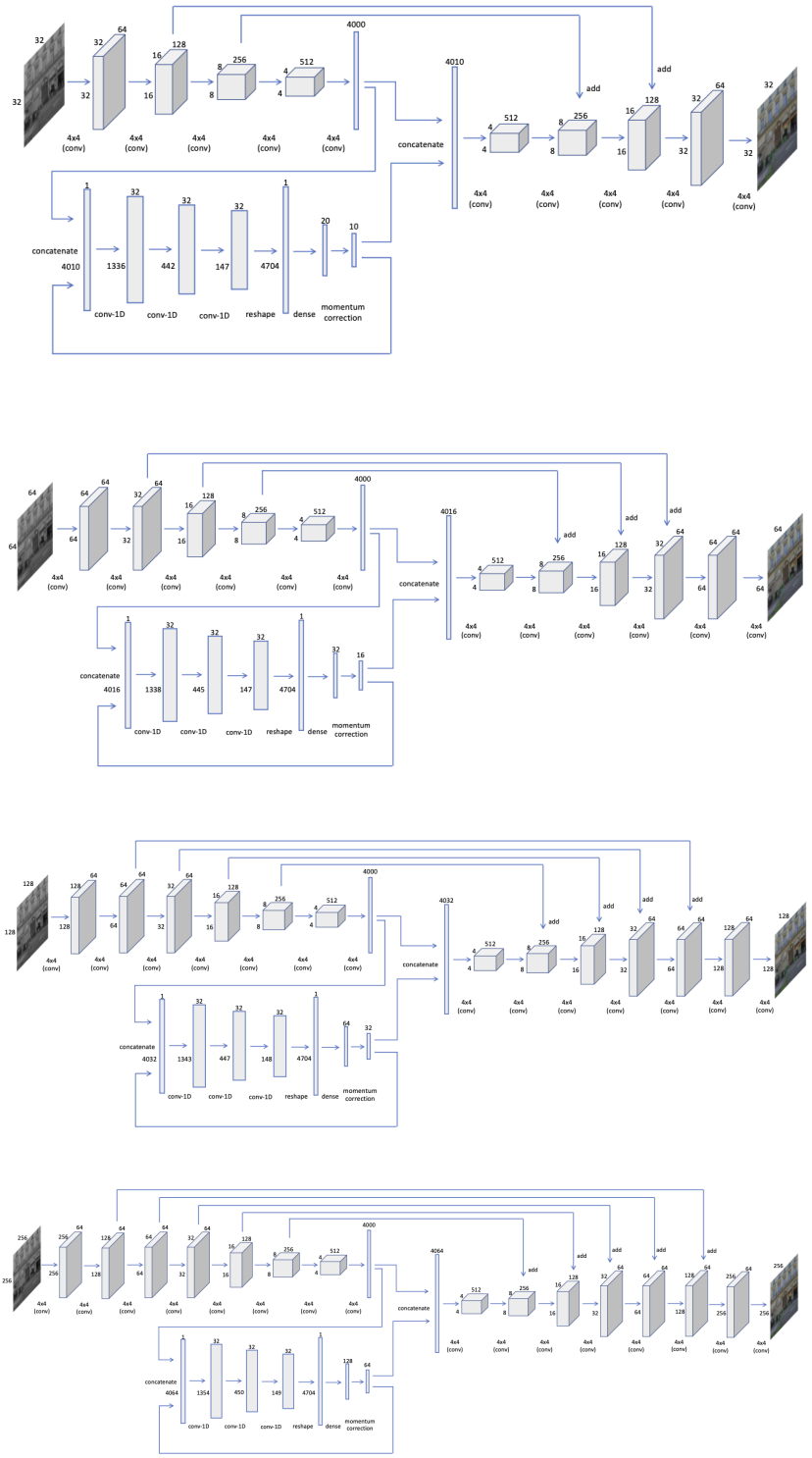


Figure 24: The model architecture for various input sizes. The same general structure in maintained with minimal changes to accomodate for the changing input size.

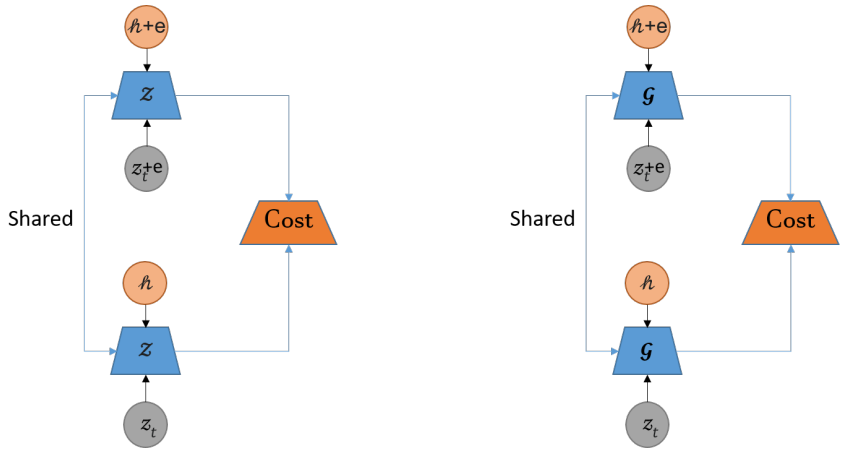


Figure 25: We enforce the Lipschitz continuity on both $\mathcal{G}$ and $\mathcal{Z}$.

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,

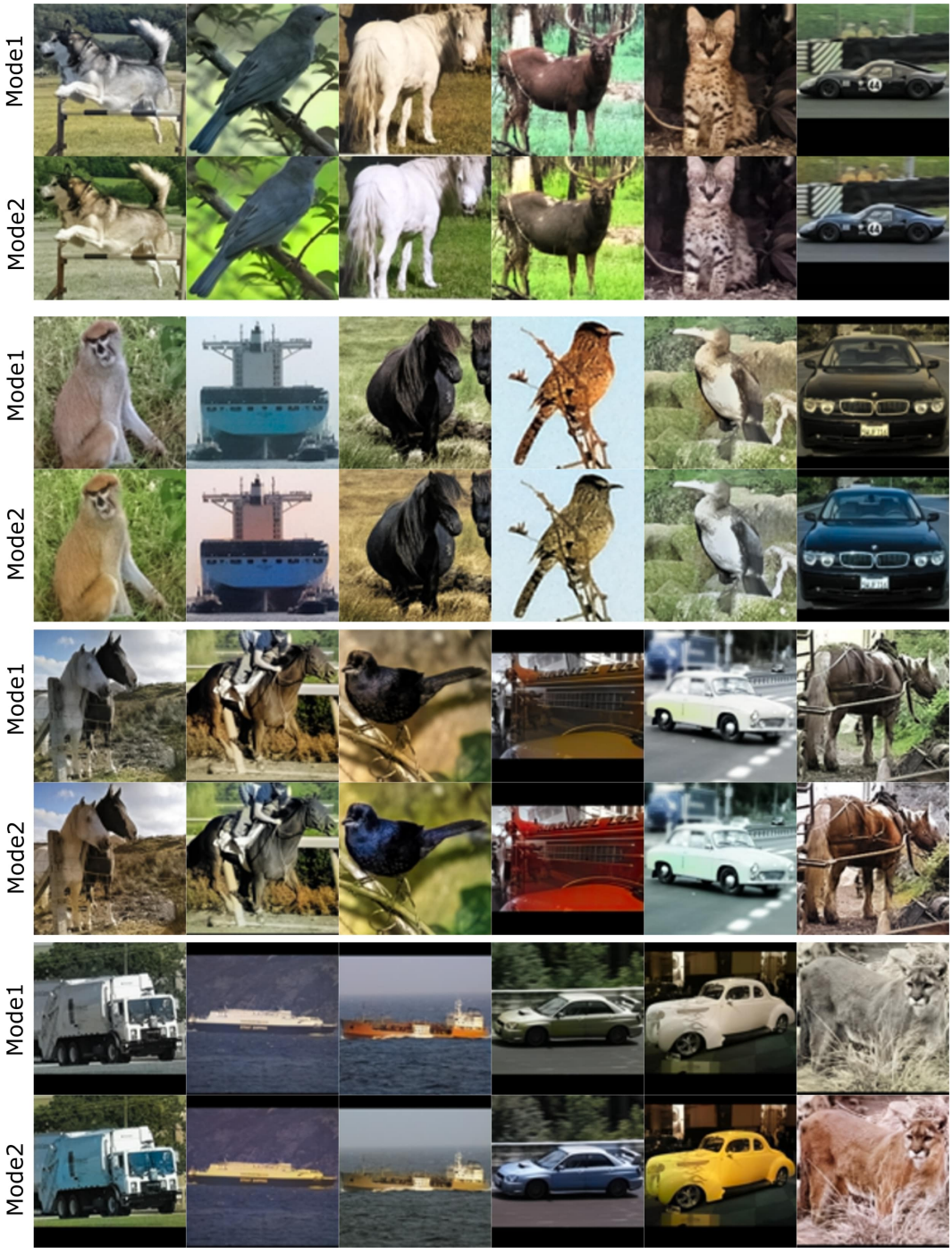


Figure 26: Multimodel predictions of our model in colorization

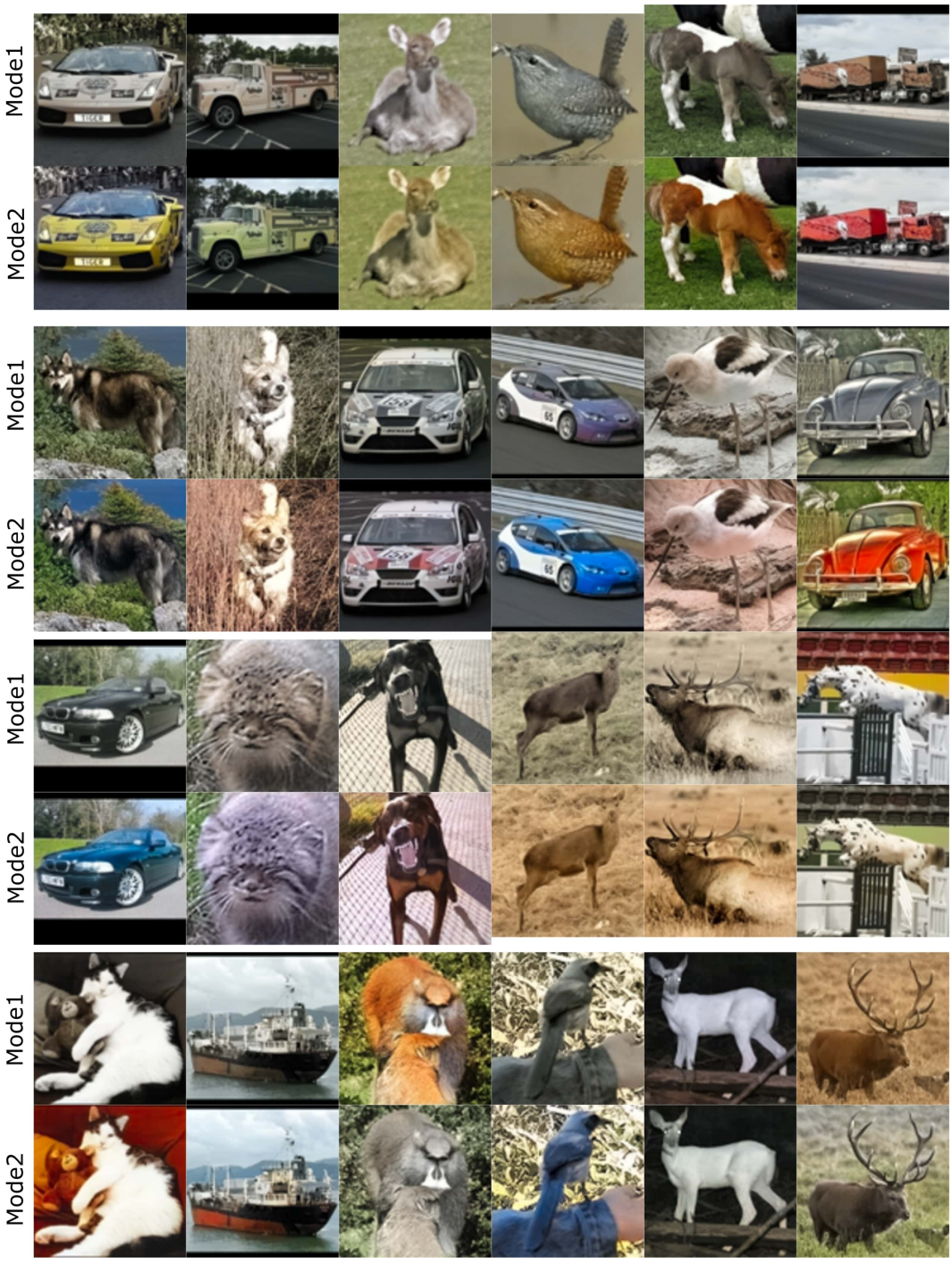


Figure 27: Multimodel predictions of our model in colorization

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

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

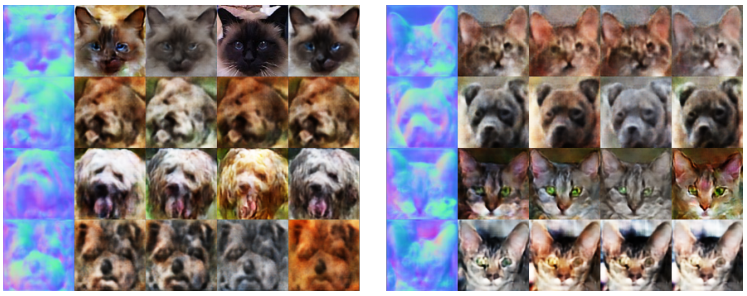


Figure 30: Multimodel predictions of our model in surface-normals-to-pet-faces. Note that this is generally a difficult task due to the diverse texture.

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.

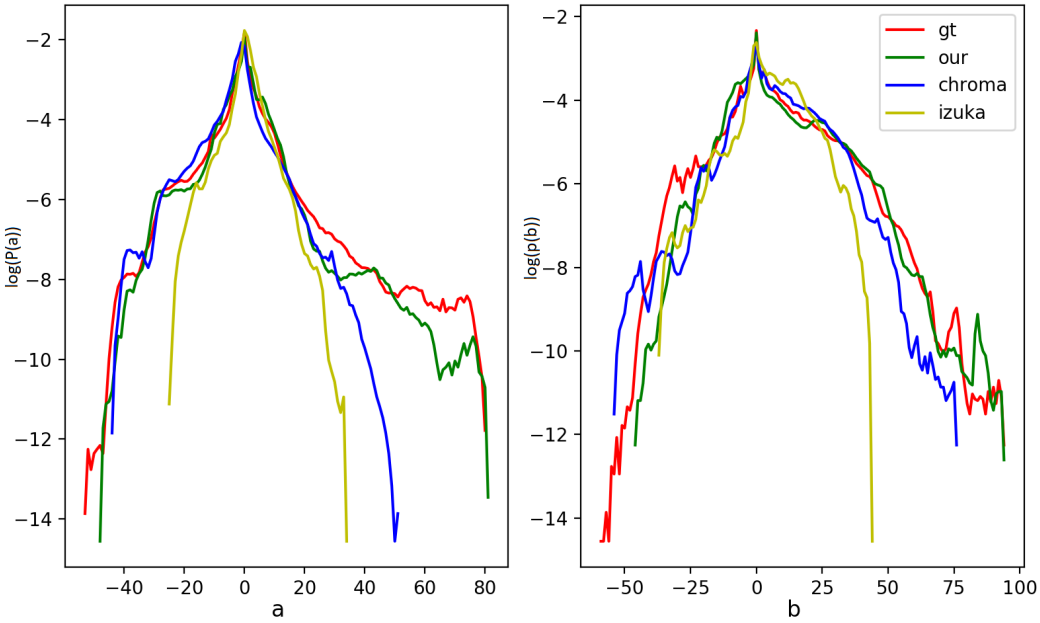


Figure 33: Color distribution comparison of a, b planes. Our method produces the closest distribution to the ground truth.

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

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

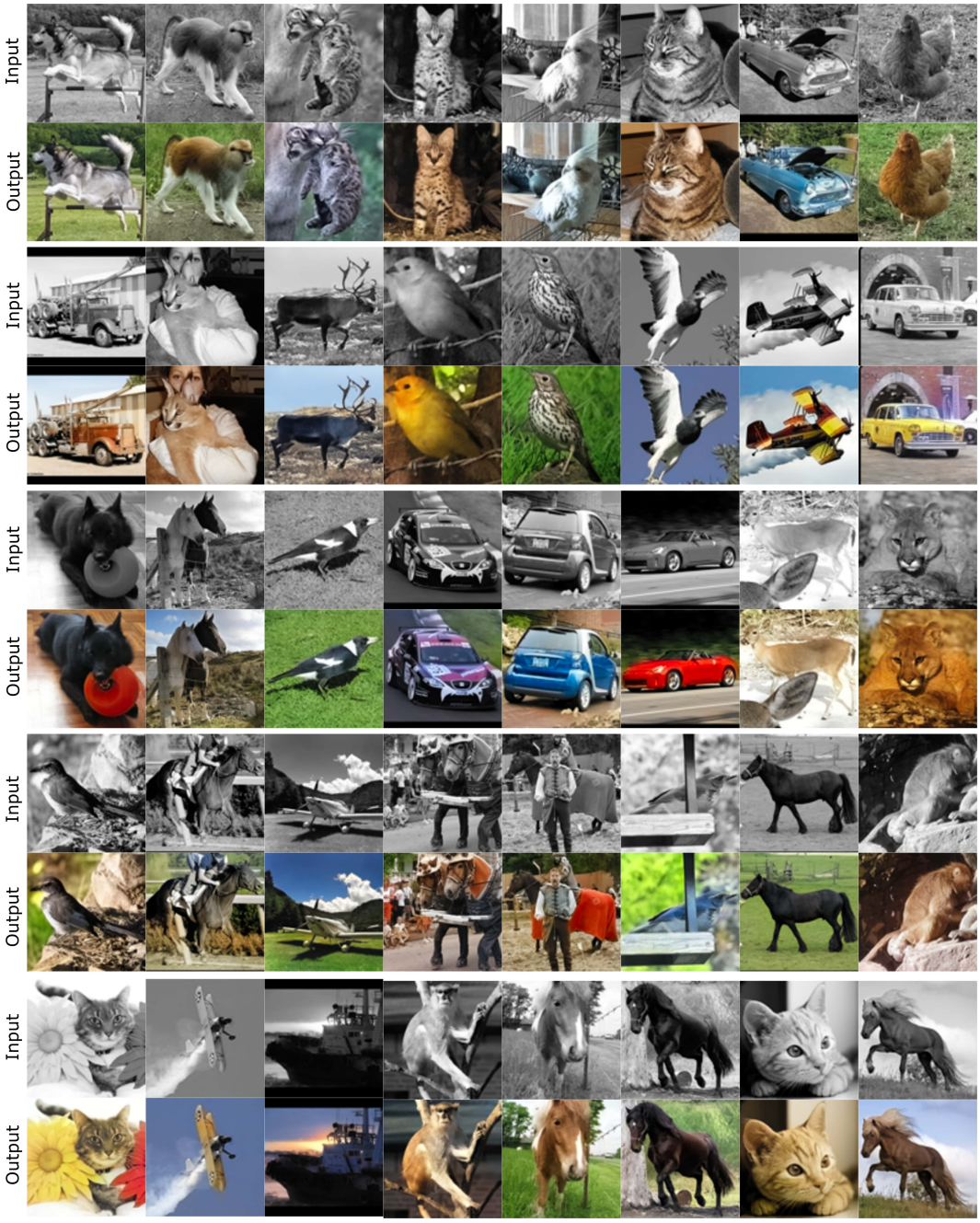


Figure 34: Qualitative results of our model in the colorization task on STL dataset.

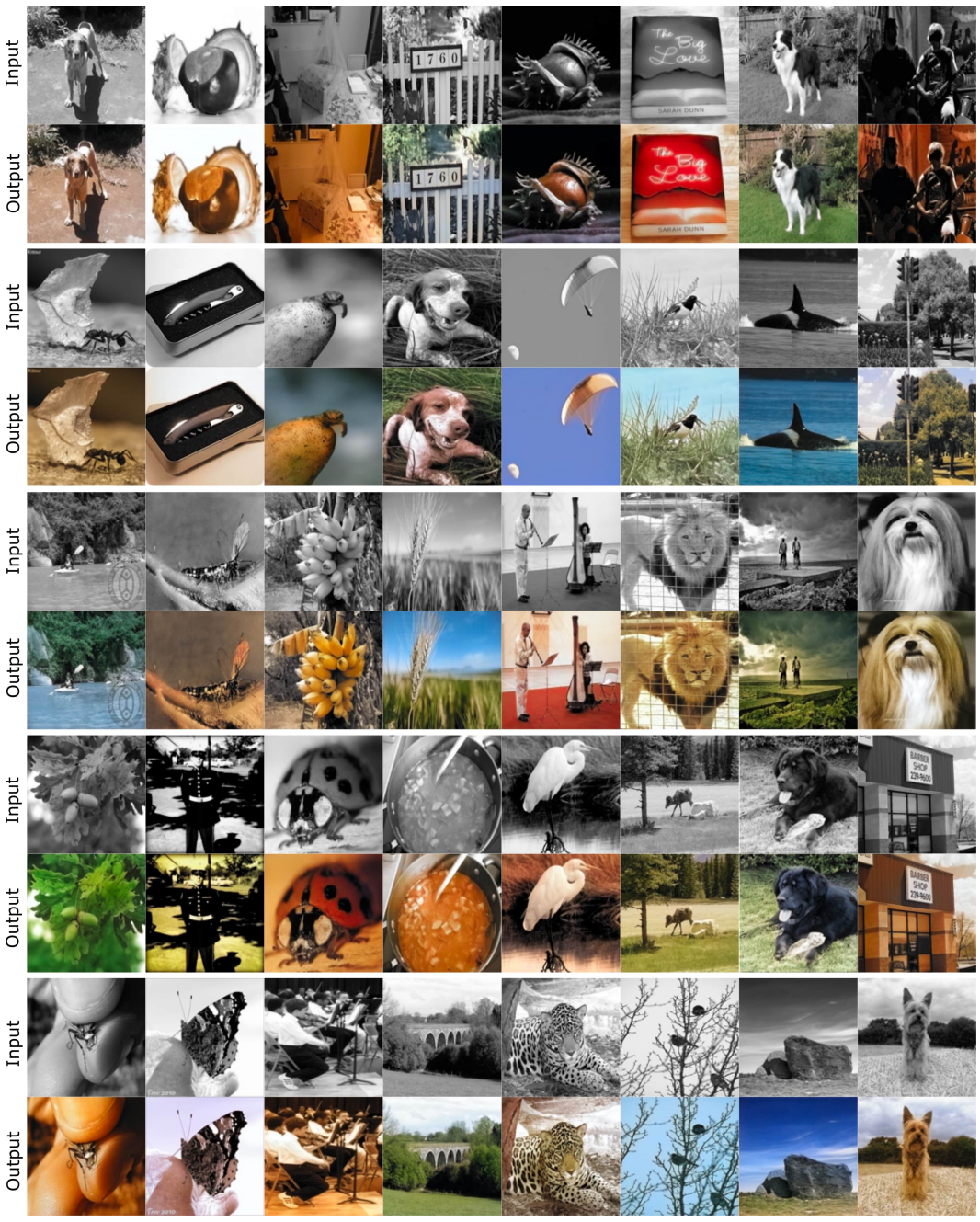


Figure 35: Qualitative results of our model in the colorization task on ImageNet dataset.

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.

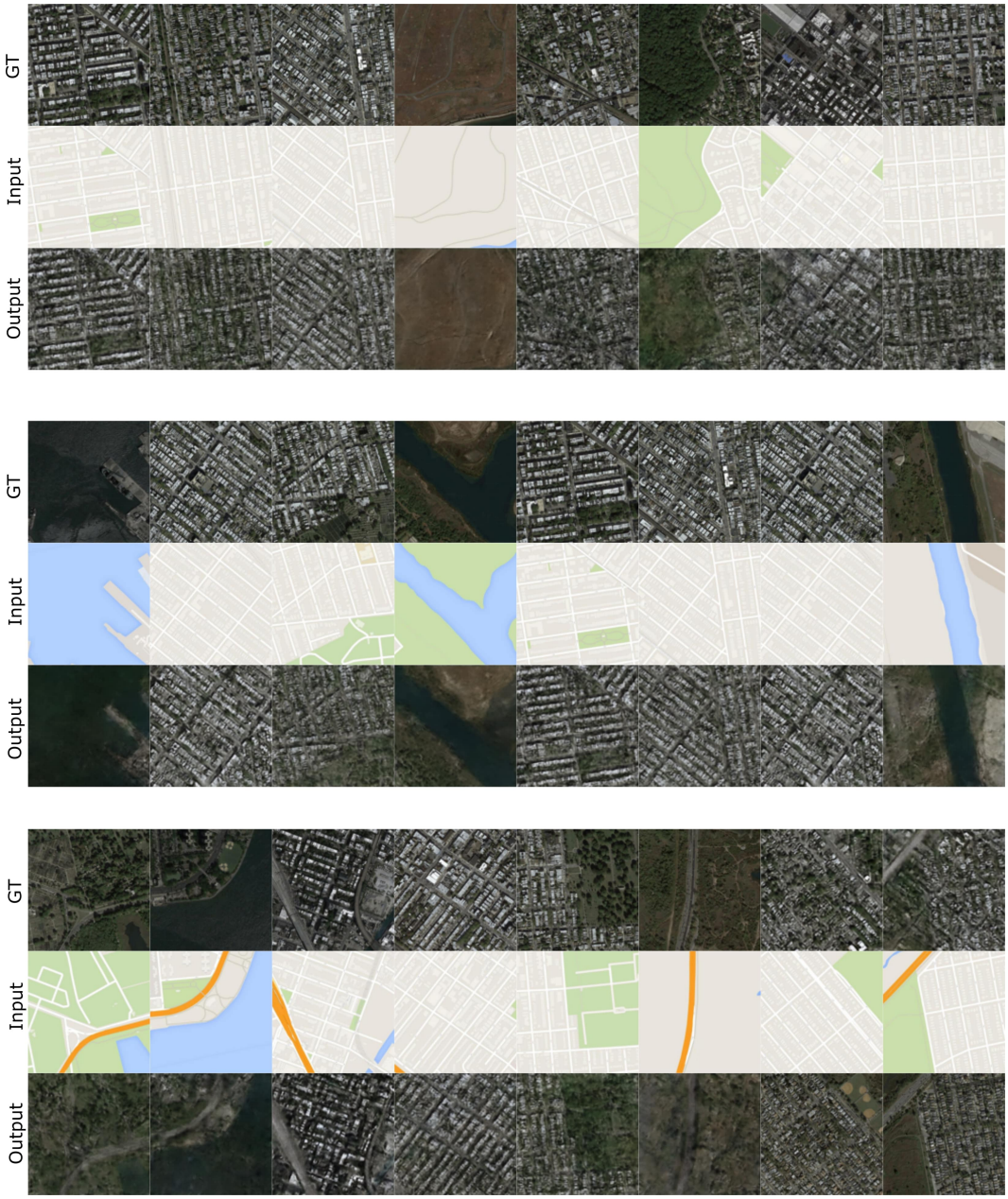


Figure 39: Qualitative results of our model in map-to-photo translation.

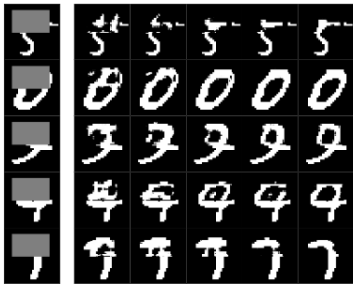


Figure 40: Output gets better as the z traverse to the optimum position at inference. Left column is the input. Five right columns show outputs at iterations 2, 4, 6, 8 and 10 (from left to right).

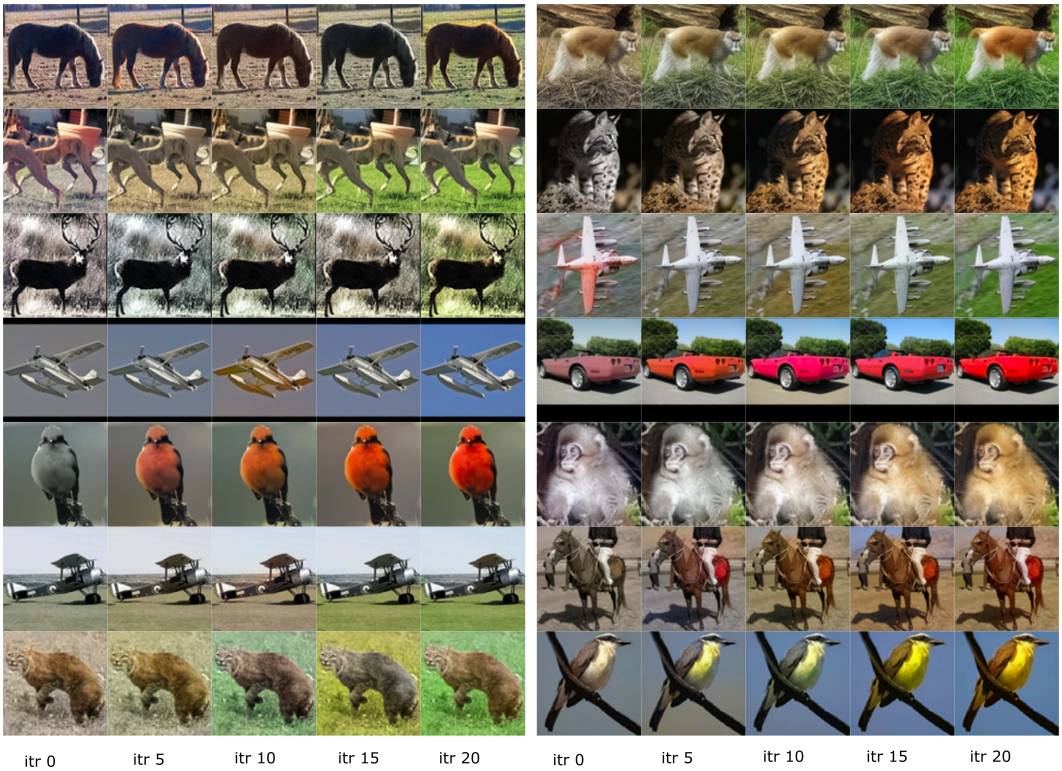


Figure 41: Output quality increases as $z \rightarrow z^*$ at inference.