Large Scale Adversarial Representation Learning

Jeff Donahue
DeepMind
jeffdonahue@google.com

Karen Simonyan
DeepMind
simonyan@google.com

Abstract

Adversarially trained generative models (GANs) have recently achieved compelling image synthesis results. But despite early successes in using GANs for unsupervised representation learning, they have since been superseded by approaches based on self-supervision. In this work we show that progress in image generation quality translates to substantially improved representation learning performance. Our approach, BigBiGAN, builds upon the state-of-the-art BigGAN model, extending it to representation learning by adding an encoder and modifying the discriminator. We extensively evaluate the representation learning and generation capabilities of these BigBiGAN models, demonstrating that these generation-based models achieve the state of the art in unsupervised representation learning on ImageNet, as well as in unconditional image generation. Pretrained BigBiGAN models – including image generators and encoders – are available on TensorFlow Hub[1].

1 Introduction

In recent years we have seen rapid progress in generative models of visual data. While these models were previously confined to domains with single or few modes, simple structure, and low resolution, with advances in both modeling and hardware they have since gained the ability to convincingly generate complex, multimodal, high resolution image distributions [2, 20, 22].

Intuitively, the ability to generate data in a particular domain necessitates a high-level understanding of the semantics of said domain. This idea has long-standing appeal as raw data is both cheap – readily available in virtually infinite supply from sources like the Internet – and rich, with images comprising far more information than the class labels that typical discriminative machine learning models are trained to predict from them. Yet, while the progress in generative models has been undeniable, nagging questions persist: what semantics have these models learned, and how can they be leveraged for representation learning?

The dream of generation as a means of true understanding from raw data alone has hardly been realized. Instead, the most successful approaches for unsupervised learning leverage techniques adopted from the field of supervised learning, a class of methods known as *self-supervised* learning [6, 42, 39, 11]. These approaches typically involve changing or holding back certain aspects of the data in some way, and training a model to predict or generate aspects of the missing information. For example, [41, 42] proposed colorization as a means of unsupervised learning, where a model is given a subset of the color channels in an input image, and trained to predict the missing channels.

Generative models as a means of unsupervised learning offer an appealing alternative to self-supervised tasks in that they are trained to model the full data distribution without requiring any modification of the original data. One class of generative models that has been applied to representation learning is generative adversarial networks (GANs) [13]. The generator in the GAN

[1] Models available at https://tfhub.dev/s?publisher=deepmind&q=bigbigan, with a Colab notebook demo at https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/bigbigan_with_tf_hub.ipynb.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

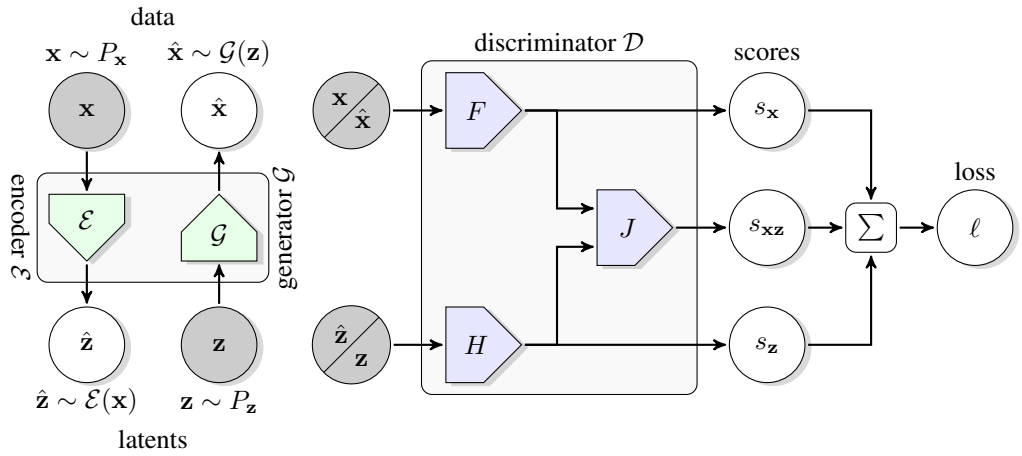


Figure 1: The structure of the BigBiGAN framework. The joint discriminator $\mathcal{D}$ is used to compute the loss ℓ. Its inputs are data-latent pairs, either $(\mathbf{x} \sim P_\mathbf{x}, \hat{\mathbf{z}} \sim \mathcal{E}(\mathbf{x}))$, sampled from the data distribution $P_\mathbf{x}$ and encoder $\mathcal{E}$ outputs, or $(\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z}), \mathbf{z} \sim P_\mathbf{z})$, sampled from the generator $\mathcal{G}$ outputs and the latent distribution $P_\mathbf{z}$. The loss ℓ includes the unary data term $s_\mathbf{x}$ and the unary latent term $s_\mathbf{z}$, as well as the joint term $s_\mathbf{xz}$ which ties the data and latent distributions.

framework is a feed-forward mapping from randomly sampled latent variables (also called "noise") to generated data, with learning signal provided by a *discriminator* trained to distinguish between real and generated data samples, guiding the generator's outputs to follow the data distribution. The *adversarially learned inference* (ALI) [10] or *bidirectional GAN* (BiGAN) [7] approaches were proposed as extensions to the GAN framework that augment the standard GAN with an *encoder* module mapping real data to latents, the inverse of the mapping learned by the generator.

In the limit of an optimal discriminator, [7] showed that a deterministic BiGAN behaves like an autoencoder minimizing ℓ_0 reconstruction costs; however, the shape of the reconstruction error surface is dictated by a parametric discriminator, as opposed to simple pixel-level measures like the ℓ_2 error. Since the discriminator is usually a powerful neural network, the hope is that it will induce an error surface which emphasizes "semantic" errors in reconstructions, rather than low-level details.

In [7] it was demonstrated that the encoder learned via the BiGAN or ALI framework is an effective means of visual representation learning on ImageNet for downstream tasks. However, it used a DCGAN [31] style generator, incapable of producing high-quality images on this dataset, so the semantics the encoder could model were in turn quite limited. In this work we revisit this approach using BigGAN [2] as the generator, a modern model that appears capable of capturing many of the modes and much of the structure present in ImageNet images. Our contributions are as follows:

- We show that BigBiGAN (BiGAN with BigGAN generator) matches the state of the art in unsupervised representation learning on ImageNet.
- We propose a more stable version of the joint discriminator for BigBiGAN.
- We perform a thorough empirical analysis and ablation study of model design choices.
- We show that the representation learning objective also improves unconditional image generation, and demonstrate state-of-the-art results in unconditional ImageNet generation.
- We open source pretrained BigBiGAN models on TensorFlow Hub[2].

2 BigBiGAN

The BiGAN [7] or ALI [10] approaches were proposed as extensions of the GAN [13] framework which enable the learning of an encoder that can be employed as an inference model [10] or feature representation [7]. Given a distribution $P_\mathbf{x}$ of data $\mathbf{x}$ (e.g., images), and a distribution $P_\mathbf{z}$ of latents $\mathbf{z}$

[2]See footnote [1].

(usually a simple continuous distribution like an isotropic Gaussian $\mathcal{N}(0, I)$), the generator $\mathcal{G}$ models a conditional distribution $P(\mathbf{x}|\mathbf{z})$ of data $\mathbf{x}$ given latent inputs $\mathbf{z}$ sampled from the latent prior $P_\mathbf{z}$, as in the standard GAN generator [13]. The encoder $\mathcal{E}$ models the inverse conditional distribution $P(\mathbf{z}|\mathbf{x})$, predicting latents $\mathbf{z}$ given data $\mathbf{x}$ sampled from the data distribution $P_\mathbf{x}$.

Besides the addition of $\mathcal{E}$, the other modification to the GAN in the BiGAN framework is a joint discriminator $\mathcal{D}$, which takes as input data-latent pairs $(\mathbf{x}, \mathbf{z})$ (rather than just data $\mathbf{x}$ as in a standard GAN), and learns to discriminate between pairs from the data distribution and encoder, versus the generator and latent distribution. Concretely, its inputs are pairs $(\mathbf{x} \sim P_\mathbf{x}, \hat{\mathbf{z}} \sim \mathcal{E}(\mathbf{x}))$ and $(\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z}), \mathbf{z} \sim P_\mathbf{z})$, and the goal of the $\mathcal{G}$ and $\mathcal{E}$ is to "fool" the discriminator by making the two joint distributions $P_{\mathbf{x}\mathcal{E}}$ and $P_{\mathcal{G}\mathbf{z}}$ from which these pairs are sampled indistinguishable. The adversarial minimax objective in [7, 10], analogous to that of the GAN framework [13], was defined as follows:

$$\min_{\mathcal{G}\mathcal{E}} \max_{\mathcal{D}} \left\{ \mathbb{E}_{\mathbf{x} \sim P_\mathbf{x}, \mathbf{z} \sim \mathcal{E}_\Phi(\mathbf{x})} \left[\log(\sigma(\mathcal{D}(\mathbf{x}, \mathbf{z}))) \right] + \mathbb{E}_{\mathbf{z} \sim P_\mathbf{z}, \mathbf{x} \sim \mathcal{G}_\Phi(\mathbf{z})} \left[\log(1 - \sigma(\mathcal{D}(\mathbf{x}, \mathbf{z}))) \right] \right\}$$

Under this objective, [7, 10] showed that with an optimal $\mathcal{D}$, $\mathcal{G}$ and $\mathcal{E}$ minimize the Jensen-Shannon divergence between the joint distributions $P_{\mathbf{x}\mathcal{E}}$ and $P_{\mathcal{G}\mathbf{z}}$, and therefore at the global optimum, the two joint distributions $P_{\mathbf{x}\mathcal{E}} = P_{\mathcal{G}\mathbf{z}}$ match, analogous to the results from standard GANs [13]. Furthermore, [7] showed that in the case where $\mathcal{E}$ and $\mathcal{G}$ are deterministic functions (i.e., the learned conditional distributions $P_\mathcal{G}(\mathbf{x}|\mathbf{z})$ and $P_\mathcal{E}(\mathbf{z}|\mathbf{x})$ are Dirac δ functions), these two functions are inverses at the global optimum: e.g., $\forall_{\mathbf{x} \in \text{supp}(P_\mathbf{x})}\, \mathbf{x} = \mathcal{G}(\mathcal{E}(\mathbf{x}))$, with the optimal joint discriminator effectively imposing ℓ_0 reconstruction costs on $\mathbf{x}$ and $\mathbf{z}$.

While the crux of our approach, BigBiGAN, remains the same as that of BiGAN [7, 10], we have adopted the generator and discriminator architectures from the state-of-the-art BigGAN [2] generative image model. Beyond that, we have found that an improved discriminator structure leads to better representation learning results without compromising generation (Figure 1). Namely, in addition to the joint discriminator loss proposed in [7, 10] which ties the data and latent distributions together, we propose additional unary terms in the learning objective, which are functions only of either the data $\mathbf{x}$ or the latents $\mathbf{z}$. Although [7, 10] prove that the original BiGAN objective already enforces that the learnt joint distributions match at the global optimum, implying that the marginal distributions of $\mathbf{x}$ and $\mathbf{z}$ match as well, these unary terms intuitively guide optimization in the "right direction" by explicitly enforcing this property. For example, in the context of image generation, the unary loss term on $\mathbf{x}$ matches the original GAN objective and provides a learning signal which steers only the generator to match the image distribution independently of its latent inputs. (In our evaluation we will demonstrate empirically that the addition of these terms results in both improved generation and representation learning.)

Concretely, the discriminator loss $\mathcal{L}_\mathcal{D}$ and the encoder-generator loss $\mathcal{L}_{\mathcal{E}\mathcal{G}}$ are defined as follows, based on scalar discriminator "score" functions s_* and the corresponding per-sample losses ℓ_*:

$$\begin{aligned}
s_\mathbf{x}(\mathbf{x}) &= \theta_\mathbf{x}^\intercal F_\Theta(\mathbf{x}) \\
s_\mathbf{z}(\mathbf{z}) &= \theta_\mathbf{z}^\intercal H_\Theta(\mathbf{z}) \\
s_{\mathbf{x}\mathbf{z}}(\mathbf{x}, \mathbf{z}) &= \theta_{\mathbf{x}\mathbf{z}}^\intercal J_\Theta(F_\Theta(\mathbf{x}), H_\Theta(\mathbf{z})) \\
\ell_{\mathcal{E}\mathcal{G}}(\mathbf{x}, \mathbf{z}, y) &= y \left(s_\mathbf{x}(\mathbf{x}) + s_\mathbf{z}(\mathbf{z}) + s_{\mathbf{x}\mathbf{z}}(\mathbf{x}, \mathbf{z}) \right) && y \in \{-1, +1\} \\
\mathcal{L}_{\mathcal{E}\mathcal{G}}(P_\mathbf{x}, P_\mathbf{z}) &= \mathbb{E}_{\mathbf{x} \sim P_\mathbf{x}, \hat{\mathbf{z}} \sim \mathcal{E}_\Phi(\mathbf{x})} \left[\ell_{\mathcal{E}\mathcal{G}}(\mathbf{x}, \hat{\mathbf{z}}, +1) \right] + \mathbb{E}_{\mathbf{z} \sim P_\mathbf{z}, \hat{\mathbf{x}} \sim \mathcal{G}_\Phi(\mathbf{z})} \left[\ell_{\mathcal{E}\mathcal{G}}(\hat{\mathbf{x}}, \mathbf{z}, -1) \right] \\
\ell_\mathcal{D}(\mathbf{x}, \mathbf{z}, y) &= h(y s_\mathbf{x}(\mathbf{x})) + h(y s_\mathbf{z}(\mathbf{z})) + h(y s_{\mathbf{x}\mathbf{z}}(\mathbf{x}, \mathbf{z})) && y \in \{-1, +1\} \\
\mathcal{L}_\mathcal{D}(P_\mathbf{x}, P_\mathbf{z}) &= \mathbb{E}_{\mathbf{x} \sim P_\mathbf{x}, \hat{\mathbf{z}} \sim \mathcal{E}_\Phi(\mathbf{x})} \left[\ell_\mathcal{D}(\mathbf{x}, \hat{\mathbf{z}}, +1) \right] + \mathbb{E}_{\mathbf{z} \sim P_\mathbf{z}, \hat{\mathbf{x}} \sim \mathcal{G}_\Phi(\mathbf{z})} \left[\ell_\mathcal{D}(\hat{\mathbf{x}}, \mathbf{z}, -1) \right]
\end{aligned}$$

where $h(t) = \max(0, 1 - t)$ is a "hinge" used to regularize the discriminator [26, 37] [3], also used in BigGAN [2]. The discriminator $\mathcal{D}$ includes three submodules: F, H, and J. F takes only $\mathbf{x}$ as input and H takes only $\mathbf{z}$, and learned projections of their outputs with parameters $\theta_\mathbf{x}$ and $\theta_\mathbf{z}$ respectively give the scalar unary scores $s_\mathbf{x}$ and $s_\mathbf{z}$. In our experiments, the data $\mathbf{x}$ are images and latents $\mathbf{z}$ are unstructured flat vectors; accordingly, F is a ConvNet and H is an MLP. The joint score $s_{\mathbf{x}\mathbf{z}}$ tying $\mathbf{x}$ and $\mathbf{z}$ is given by the remaining $\mathcal{D}$ submodule, J, a function of the outputs of F and H.

[3] We also considered an alternative discriminator loss $\ell'_\mathcal{D}$ which invokes the "hinge" h just once on the sum of the three loss terms – $\ell'_\mathcal{D}(\mathbf{x}, \mathbf{z}, y) = h(y \left(s_\mathbf{x}(\mathbf{x}) + s_\mathbf{z}(\mathbf{z}) + s_{\mathbf{x}\mathbf{z}}(\mathbf{x}, \mathbf{z}) \right))$ – but found that this performed significantly worse than $\ell_\mathcal{D}$ above which clamps each of the three loss terms separately.

The $\mathcal{E}$ and $\mathcal{G}$ parameters Φ are optimized to minimize the loss $\mathcal{L}_{\mathcal{EG}}$, and the $\mathcal{D}$ parameters Θ are optimized to minimize loss $\mathcal{L}_{\mathcal{D}}$. As usual, the expectations $\mathbb{E}$ are estimated by Monte Carlo samples taken over minibatches.

Like in BiGAN [7] and ALI [10], the discriminator loss $\mathcal{L}_{\mathcal{D}}$ intuitively trains the discriminator to distinguish between the two joint data-latent distributions from the encoder and the generator, pushing it to predict positive values for encoder input pairs $(\mathbf{x}, \mathcal{E}(\mathbf{x}))$ and negative values for generator input pairs $(\mathcal{G}(\mathbf{z}), \mathbf{z})$. The generator and encoder loss $\mathcal{L}_{\mathcal{EG}}$ trains these two modules to fool the discriminator into incorrectly predicting the opposite, in effect pushing them to create matching joint data-latent distributions. (In the case of deterministic $\mathcal{E}$ and $\mathcal{G}$, this requires the two modules to invert one another [7].)

3 Evaluation

Most of our experiments follow the standard protocol used to evaluate unsupervised learning techniques, first proposed in [41]. We train a BigBiGAN on unlabeled ImageNet, freeze its learned representation, and then train a linear classifier on its outputs, fully supervised using all of the training set labels. We also measure image generation performance, reporting Inception Score [35] (IS) and Fréchet Inception Distance [18] (FID) as the standard metrics there.

3.1 Ablation

We begin with an extensive ablation study in which we directly evaluate a number of modeling choices, with results presented in Table 1. Where possible we performed three runs of each variant with different seeds and report the mean and standard deviation for each metric.

We start with a relatively fully-fledged version of the model at 128×128 resolution (row *Base*), with the $\mathcal{G}$ architecture and the F component of $\mathcal{D}$ taken from the corresponding 128×128 architectures in BigGAN, including the skip connections and shared noise embedding proposed in [2]. $\mathbf{z}$ is 120 dimensions, split into six groups of 20 dimensions fed into each of the six layers of $\mathcal{G}$ as in [2]. The remaining components of $\mathcal{D}$ – H and J – are 8-layer MLPs with ResNet-style skip connections (four residual blocks with two layers each) and size 2048 hidden layers. The $\mathcal{E}$ architecture is the ResNet-v2-50 ConvNet originally proposed for image classification in [16], followed by a 4-layer MLP (size 4096) with skip connections (two residual blocks) after ResNet's globally average pooled output. The unconditional BigGAN training setup corresponds to the "Single Label" setup proposed in [27], where a single "dummy" label is used for all images (theoretically equivalent to learning a bias in place of the class-conditional batch norm inputs). We then ablate several aspects of the model, with results detailed in the following paragraphs. Additional architectural and optimization details are provided in Appendix A. Full learning curves for many results are included in Appendix D.

Latent distribution $P_{\mathbf{z}}$ and stochastic $\mathcal{E}$. As in ALI [10], the encoder $\mathcal{E}$ of our *Base* model is non-deterministic, parametrizing a distribution $\mathcal{N}(\mu, \sigma)$. μ and $\hat{\sigma}$ are given by a linear layer at the output of the model, and the final standard deviation σ is computed from $\hat{\sigma}$ using a non-negative "softplus" non-linearity $\sigma = \log(1 + \exp(\hat{\sigma}))$ [9]. The final $\mathbf{z}$ uses the reparametrized sampling from [23], with $\mathbf{z} = \mu + \epsilon\sigma$, where $\epsilon \sim \mathcal{N}(0, I)$. Compared to a deterministic encoder (row *Deterministic* $\mathcal{E}$) which predicts $\mathbf{z}$ directly without sampling (effectively modeling $P(\mathbf{z}|\mathbf{x})$ as a Dirac δ distribution), the non-deterministic *Base* model achieves significantly better classification performance (at no cost to generation). We also compared to using a uniform $P_{\mathbf{z}} = \mathcal{U}(-1, 1)$ (row *Uniform* $P_{\mathbf{z}}$) with $\mathcal{E}$ deterministically predicting $\mathbf{z} = \tanh(\hat{\mathbf{z}})$ given a linear output $\hat{\mathbf{z}}$, as done in BiGAN [7]. This also achieves worse classification results than the non-deterministic *Base* model.

Unary loss terms. We evaluate the effect of removing one or both unary terms of the loss function proposed in Section 2, $s_{\mathbf{x}}$ and $s_{\mathbf{z}}$. Removing both unary terms (row *No Unaries*) corresponds to the original objective proposed in [7, 10]. It is clear that the $\mathbf{x}$ unary term has a large positive effect on generation performance, with the *Base* and $\mathbf{x}$ *Unary Only* rows having significantly better IS and FID than the $\mathbf{z}$ *Unary Only* and *No Unaries* rows. This result makes intuitive sense as it matches the standard generator loss. It also marginally improves classification performance. The $\mathbf{z}$ unary term makes a more marginal difference, likely due to the relative ease of modeling relatively simple distributions like isotropic Gaussians, though also does result in slightly improved classification and

generation in terms of FID – especially without the $\mathbf{x}$ term ($\mathbf{z}$ *Unary Only* vs. *No Unaries*). On the other hand, IS is worse with the $\mathbf{z}$ term. This may be due to IS roughly measuring the generator's coverage of the major modes of the distribution (the classes) rather than the distribution in its entirety, the latter of which may be better captured by FID and more likely to be promoted by a good encoder $\mathcal{E}$. The requirement of invertibility in a (Big)BiGAN could be encouraging the generator to produce distinguishable outputs across the entire latent space, rather than "collapsing" large volumes of latent space to a single mode of the data distribution.

$\mathcal{G}$ capacity. To address the question of the importance of the generator $\mathcal{G}$ in representation learning, we vary the capacity of $\mathcal{G}$ (with $\mathcal{E}$ and $\mathcal{D}$ fixed) in the *Small $\mathcal{G}$* rows. With a third of the capacity of the *Base $\mathcal{G}$* model (*Small $\mathcal{G}$ (32)*), the overall model is quite unstable and achieves significantly worse classification results than the higher capacity base model[4] With two-thirds capacity (*Small $\mathcal{G}$ (64)*), generation performance is substantially worse (matching the results in [2]) and classification performance is modestly worse. These results confirm that a powerful image generator is indeed important for learning good representations via the encoder. Assuming this relationship holds in the future, we expect that better generative models are likely to lead to further improvements in representation learning.

Standard GAN. We also compare BigBiGAN's image generation performance against a standard unconditional BigGAN with no encoder $\mathcal{E}$ and only the standard F ConvNet in the discriminator, with only the $s_{\mathbf{x}}$ term in the loss (row *No $\mathcal{E}$ (GAN)*). While the standard GAN achieves a marginally better IS, the BigBiGAN FID is about the same, indicating that the addition of the BigBiGAN $\mathcal{E}$ and joint $\mathcal{D}$ does not compromise generation with the newly proposed unary loss terms described in Section 2. (In comparison, the versions of the model without unary loss term on $\mathbf{x}$ – rows $\mathbf{z}$ *Unary Only* and *No Unaries* – have substantially worse generation performance in terms of FID than the standard GAN.) We conjecture that the IS is worse for similar reasons that the $s_{\mathbf{z}}$ unary loss term leads to worse IS. Next we will show that with an enhanced $\mathcal{E}$ taking higher input resolutions, generation with BigBiGAN in terms of FID is substantially improved over the standard GAN.

High resolution $\mathcal{E}$ with varying resolution $\mathcal{G}$. BiGAN [7] proposed an asymmetric setup in which $\mathcal{E}$ takes higher resolution images than $\mathcal{G}$ outputs and $\mathcal{D}$ takes as input, showing that an $\mathcal{E}$ taking 128×128 inputs with a 64×64 $\mathcal{G}$ outperforms a 64×64 $\mathcal{E}$ for downstream tasks. We experiment with this setup in BigBiGAN, raising the $\mathcal{E}$ input resolution to 256×256 – matching the resolution used in typical supervised ImageNet classification setups – and varying the $\mathcal{G}$ output and $\mathcal{D}$ input resolution in $\{64, 128, 256\}$. Our results in Table 1 (rows *High Res $\mathcal{E}$ (256)* and *Low/High Res $\mathcal{G}$ (*)*) show that BigBiGAN achieves better representation learning results as the $\mathcal{G}$ resolution increases, up to the full $\mathcal{E}$ resolution of 256×256. However, because the overall model is much slower to train with $\mathcal{G}$ at 256×256 resolution, the remainder of our results use the 128×128 resolution for $\mathcal{G}$.

Interestingly, with the higher resolution $\mathcal{E}$, generation improves significantly (especially by FID), despite $\mathcal{G}$ operating at the same resolution (row *High Res $\mathcal{E}$ (256)* vs. *Base*). This is an encouraging result for the potential of BigBiGAN as a means of improving adversarial image synthesis itself, besides its use in representation learning and inference.

$\mathcal{E}$ architecture. Keeping the $\mathcal{E}$ input resolution fixed at 256, we experiment with varied and often larger $\mathcal{E}$ architectures, including several of the ResNet-50 variants explored in [24]. In particular, we expand the capacity of the hidden layers by a factor of 2 or 4, as well as swap the residual block structure to a reversible variant called *RevNet* [12] with the same number of layers and capacity as the corresponding ResNets. (We use the version of RevNet described in [24].) We find that the base ResNet-50 model (row *High Res $\mathcal{E}$ (256)*) outperforms RevNet-50 (row *RevNet*), but as the network widths are expanded, we begin to see improvements from RevNet-50, with double-width RevNet outperforming a ResNet of the same capacity (rows *RevNet* $\times 2$ and *ResNet* $\times 2$). We see further gains with an even larger quadruple-width RevNet model (row *RevNet* $\times 4$), which we use for our final results in Section 3.2.

[4] Though the generation performance by IS and FID in row *Small $\mathcal{G}$ (32)* is very poor at the point we measured – when its best validation classification performance (43.59%) is achieved – this model was performing more reasonably for generation earlier in training, reaching IS 14.69 and FID 60.67.

	Encoder ($\mathcal{E}$)						Gen. ($\mathcal{G}$)		Loss $\mathcal{L}_*$				Results		
	A.	D.	C.	R.	Var.	η	C.	R.	$s_{\mathbf{xz}}$	$s_{\mathbf{x}}$	$s_{\mathbf{z}}$	$P_{\mathbf{z}}$	IS (↑)	FID (↓)	Cls. (↑)
Base	S	50	1	128	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	22.66 ± 0.18	31.19 ± 0.37	48.10 ± 0.13
Deterministic $\mathcal{E}$	S	50	1	128	(-)	1	96	128	✓	✓	✓	$\mathcal{N}$	22.79 ± 0.27	31.31 ± 0.30	46.97 ± 0.35
Uniform $P_{\mathbf{z}}$	S	50	1	128	(-)	1	96	128	✓	✓	✓	($\mathcal{U}$)	22.83 ± 0.24	31.52 ± 0.28	45.11 ± 0.93
$\mathbf{x}$ Unary Only	S	50	1	128	✓	1	96	128	✓	✓	(-)	$\mathcal{N}$	23.19 ± 0.28	31.99 ± 0.30	47.74 ± 0.20
$\mathbf{z}$ Unary Only	S	50	1	128	✓	1	96	128	✓	(-)	✓	$\mathcal{N}$	19.52 ± 0.39	39.48 ± 1.00	47.78 ± 0.28
No Unaries (BiGAN)	S	50	1	128	✓	1	96	128	✓	(-)	(-)	$\mathcal{N}$	19.70 ± 0.30	42.92 ± 0.92	46.71 ± 0.88
Small $\mathcal{G}$ (32)	S	50	1	128	✓	1	(32)	128	✓	✓	✓	$\mathcal{N}$	3.28 ± 0.18	247.30 ± 10.31	43.59 ± 0.34
Small $\mathcal{G}$ (64)	S	50	1	128	✓	1	(64)	128	✓	✓	✓	$\mathcal{N}$	19.96 ± 0.15	38.93 ± 0.39	47.54 ± 0.33
No $\mathcal{E}$ (GAN) *	(-)						96	128	(-)	✓	(-)	$\mathcal{N}$	23.56 ± 0.37	30.91 ± 0.23	-
High Res $\mathcal{E}$ (256)	S	50	1	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.45 ± 0.14	27.86 ± 0.13	50.80 ± 0.30
Low Res $\mathcal{G}$ (64)	S	50	1	(256)	✓	1	96	(64)	✓	✓	✓	$\mathcal{N}$	19.40 ± 0.19	15.82 ± 0.06	47.51 ± 0.09
High Res $\mathcal{G}$ (256)	S	50	1	(256)	✓	1	96	(256)	✓	✓	✓	$\mathcal{N}$	24.70	38.58	51.49
ResNet-101	S	(101)	1	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.29	28.01	51.21
ResNet $\times 2$	S	50	(2)	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.68	27.81	52.66
RevNet	(V)	50	1	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.33 ± 0.09	27.78 ± 0.06	49.42 ± 0.18
RevNet $\times 2$	(V)	50	(2)	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.21	27.96	54.40
RevNet $\times 4$	(V)	50	(4)	(256)	✓	1	96	128	✓	✓	✓	$\mathcal{N}$	23.23	28.15	57.15
ResNet (↑ $\mathcal{E}$ LR)	S	50	1	(256)	✓	(10)	96	128	✓	✓	✓	$\mathcal{N}$	23.27 ± 0.22	28.51 ± 0.44	53.70 ± 0.15
RevNet $\times 4$ (↑ $\mathcal{E}$ LR)	(V)	50	(4)	(256)	✓	(10)	96	128	✓	✓	✓	$\mathcal{N}$	23.08	28.54	60.15

Table 1: Results for variants of BigBiGAN, given in Inception Score [35] (IS) and Fréchet Inception Distance [18] (FID) of the generated images, and ImageNet top-1 classification accuracy percentage (Cls.) of a supervised logistic regression classifier trained on the encoder features [41], computed on a split of 10K images randomly sampled from the training set, which we refer to as the "$\text{train}_{\text{val}}$" split. The *Encoder ($\mathcal{E}$)* columns specify the $\mathcal{E}$ architecture (A.) as ResNet (S) or RevNet (V), the depth (D., e.g. 50 for ResNet-50), the channel width multiplier (C.), with 1 denoting the original widths from [16], the input image resolution (R.), whether the variance is predicted and a $\mathbf{z}$ vector is sampled from the resulting distribution (Var.), and the learning rate multiplier η relative to the $\mathcal{G}$ learning rate. The *Generator ($\mathcal{G}$)* columns specify the BigGAN $\mathcal{G}$ channel multiplier (C.), with 96 corresponding to the original width from [2], and output image resolution (R.). The *Loss* columns specify which terms of the BigBiGAN loss are present in the objective. The P_z column specifies the input distribution as a standard normal $\mathcal{N}(0, 1)$ or continuous uniform $\mathcal{U}(-1, 1)$. Changes from the *Base* setup in each row are highlighted in blue. Results with margins of error (written as "$\mu \pm \sigma$") are the means and standard deviations over three runs with different random seeds. (Experiments requiring more computation were run only once.) (* Result for vanilla GAN (*No $\mathcal{E}$ (GAN)*) selected with early stopping based on best FID; other results selected with early stopping based on validation classification accuracy (Cls.).)

Decoupled $\mathcal{E}$/$\mathcal{G}$ optimization. As a final improvement, we decoupled the $\mathcal{E}$ optimizer from that of $\mathcal{G}$, and found that simply using a $10\times$ higher learning rate for $\mathcal{E}$ dramatically accelerates training and improves final representation learning results. For ResNet-50 this improves linear classifier accuracy by nearly 3% (*ResNet (↑ $\mathcal{E}$ LR)* vs. *High Res $\mathcal{E}$ (256)*). We also applied this to our largest $\mathcal{E}$ architecture, RevNet-50 $\times 4$, and saw similar gains (*RevNet $\times 4$ (↑ $\mathcal{E}$ LR)* vs. *RevNet $\times 4$*).

3.2 Comparison with prior methods

Representation learning. We now take our best model by $\text{train}_{\text{val}}$ classification accuracy from the above ablations and present results on the official ImageNet validation set, comparing against the state of the art in recent unsupervised learning literature. For comparison, we also present classification results for our best performing variant with the smaller ResNet-50-based $\mathcal{E}$. These models correspond to the last two rows of Table 1, *ResNet (↑ $\mathcal{E}$ LR)* and *RevNet $\times 4$ (↑ $\mathcal{E}$ LR)*.

Results are presented in Table 2. (For reference, the fully supervised accuracy of these architectures is given in Appendix A, Table 4.) Compared with a number of modern self-supervised approaches [30, 5, 41, 39, 11, 17] and combinations thereof [6], our BigBiGAN approach based purely on generative models performs well for representation learning, state-of-the-art among recent unsupervised learning results, improving upon a recently published result from [24] of 55.4% to 60.8% top-1 accuracy using rotation prediction pre-training with the same representation learning architecture [5] and feature,

[5] Our RevNet $\times 4$ architecture matches the widest architectures used in [24], labeled as $\times 16$ there.

Method	Architecture	Feature	Top-1	Top-5
BiGAN [7, 42]	AlexNet	Conv3	31.0	-
SS-GAN [4]	ResNet-19	Block6	38.3	-
Motion Segmentation (MS) [30, 6]	ResNet-101	AvePool	27.6	48.3
Exemplar (Ex) [8, 6]	ResNet-101	AvePool	31.5	53.1
Relative Position (RP) [5, 6]	ResNet-101	AvePool	36.2	59.2
Colorization (Col) [41, 6]	ResNet-101	AvePool	39.6	62.5
Combination of MS+Ex+RP+Col [6]	ResNet-101	AvePool	-	69.3
CPC [39]	ResNet-101	AvePool	48.7	73.6
Rotation [11, 24]	RevNet-50 $\times 4$	AvePool	55.4	-
Efficient CPC [17]	ResNet-170	AvePool	61.0	83.0
BigBiGAN (ours)	ResNet-50	AvePool	55.4	77.4
	ResNet-50	BN+CReLU	56.6	78.6
	RevNet-50 $\times 4$	AvePool	60.8	81.4
	RevNet-50 $\times 4$	BN+CReLU	61.3	81.9

Table 2: Comparison of BigBiGAN models on the official ImageNet validation set against recent competing approaches with a supervised logistic regression classifier. BigBiGAN results are selected with early stopping based on highest accuracy on our $\text{train}_{\text{val}}$ subset of 10K training set images. *ResNet-50* results correspond to row *ResNet ($\uparrow \mathcal{E}$ LR)* in Table 1, and *RevNet-50* $\times 4$ corresponds to *RevNet* $\times 4$ *($\uparrow \mathcal{E}$ LR)*.

Method	Steps	IS ($\uparrow$)	FID vs. Train ($\downarrow$)	FID vs. Val. ($\downarrow$)
BigGAN + SL [27]	500K	20.4 (15.4 $\pm$ 7.57)	-	25.3 (71.7 $\pm$ 66.32)
BigGAN + Clustering [27]	500K	22.7 (22.8 $\pm$ 0.42)	-	23.2 (22.7 $\pm$ 0.80)
BigBiGAN + SL (ours)	500K	25.38 (25.33 $\pm$ 0.17)	22.78 (22.63 $\pm$ 0.23)	23.60 (23.56 $\pm$ 0.12)
BigBiGAN High Res $\mathcal{E}$ + SL (ours)	500K	25.43 (25.45 $\pm$ 0.04)	22.34 (22.36 $\pm$ 0.04)	22.94 (23.00 $\pm$ 0.15)
BigBiGAN High Res $\mathcal{E}$ + SL (ours)	1M	27.94 (27.80 $\pm$ 0.21)	20.32 (20.27 $\pm$ 0.09)	21.61 (21.62 $\pm$ 0.09)

Table 3: Comparison of our BigBiGAN for unsupervised (unconditional) generation vs. previously reported results for unsupervised BigGAN from [27]. We specify the "pseudo-labeling" method as *SL* (Single Label) or *Clustering*. For comparison we train BigBiGAN for the same number of steps (500K) as the BigGAN-based approaches from [27], but also present results from additional training to 1M steps in the last row and observe further improvements. All results above include the median m as well as the mean μ and standard deviation σ across three runs, written as "m ($\mu \pm \sigma$)". The BigBiGAN result is selected with early stopping based on best FID vs. Train.

labeled as *AvePool* in Table 2, and matches the results of the concurrent work in [17] based on contrastic predictive coding (CPC).

We also experiment with learning linear classifiers on a different rendering of the *AvePool* feature, labeled *BN+CReLU*, which boosts our best results with RevNet $\times 4$ to 61.3% top-1 accuracy. Given the global average pooling output a, we first compute $h = \text{BatchNorm}(a)$, and the final feature is computed by concatenating $[\text{ReLU}(h), \text{ReLU}(-h)]$, sometimes called a "CReLU" (concatened ReLU) non-linearity [36]. BatchNorm denotes parameter-free Batch Normalization [19], where the scale (γ) and offset (β) parameters are not learned, so training a linear classifier on this feature does not involve any additional learning. The CReLU non-linearity retains all the information in its inputs and doubles the feature dimension, each of which likely contributes to the improved results.

Finally, in Appendix C we consider evaluating representations by zero-shot k nearest neighbors classification, achieving 43.3% top-1 accuracy in this setting. Qualitative examples of nearest neighbors are presented in Figure 13.

Unsupervised image generation. In Table 3 we show results for unsupervised generation with BigBiGAN, comparing to the BigGAN-based [2] unsupervised generation results from [27]. Note that these results differ from those in Table 1 due to the use of the data augmentation method of [27][6]

[6]See the "distorted" preprocessing method from the Compare GAN framework: https://github.com/google/compare_gan/blob/master/compare_gan/datasets.py.

Figure 2: Selected reconstructions from an unsupervised BigBiGAN model (Section 3.3). Top row images are real data $\mathbf{x} \sim P_\mathbf{x}$; bottom row images are generated reconstructions of the above image $\mathbf{x}$ computed by $\mathcal{G}(\mathcal{E}(\mathbf{x}))$. Unlike most explicit reconstruction costs (e.g., pixel-wise), the reconstruction cost implicitly minimized by a (Big)BiGAN [7, 10] tends to emphasize more semantic, high-level details. Additional reconstructions are presented in Appendix B.

(rather than ResNet-style preprocessing used for all results in our Table 1 ablation study). The lighter augmentation from [27] results in better image generation performance under the IS and FID metrics. The improvements are likely due in part to the fact that this augmentation, on average, crops larger portions of the image, thus yielding generators that typically produce images encompassing most or all of a given object, which tends to result in more representative samples of any given class (giving better IS) and more closely matching the statistics of full center crops (as used in the real data statistics to compute FID). Besides this preprocessing difference, the approaches in Table 3 have the same configurations as used in the *Base* or *High Res $\mathcal{E}$ (256)* row of Table 1.

These results show that BigBiGAN significantly improves both IS and FID over the baseline unconditional BigGAN generation results with the same (unsupervised) "labels" (a single fixed label in the *SL* (Single Label) approach – row *BigBiGAN + SL* vs. *BigGAN + SL*). We see further improvements using a high resolution $\mathcal{E}$ (row *BigBiGAN High Res $\mathcal{E}$ + SL*), surpassing the previous unsupervised state of the art (row *BigGAN + Clustering*) under both IS and FID. (Note that the image generation results remain comparable: the generated image resolution is still 128×128 here, despite the higher resolution $\mathcal{E}$ input.) The alternative "pseudo-labeling" approach from [27], *Clustering*, which uses labels derived from unsupervised clustering, is complementary to BigBiGAN and combining both could yield further improvements. Finally, observing that results continue to improve significantly with training beyond 500K steps, we also report results at 1M steps in the final row of Table 3.

3.3 Reconstruction

As shown in [7, 10], the (Big)BiGAN $\mathcal{E}$ and $\mathcal{G}$ can reconstruct data instances $\mathbf{x}$ by computing the encoder's predicted latent representation $\mathcal{E}(\mathbf{x})$ and then passing this predicted latent back through the generator to obtain the reconstruction $\mathcal{G}(\mathcal{E}(\mathbf{x}))$. We present BigBiGAN reconstructions in Figure 2. These reconstructions are far from pixel-perfect, likely due in part to the fact that no reconstruction cost is explicitly enforced by the objective – reconstructions are not even computed at training time. However, they may provide some intuition for what features the encoder $\mathcal{E}$ learns to model. For example, when the input image contains a dog, person, or a food item, the reconstruction is often a different instance of the same "category" with similar pose, position, and texture – for example, a similar species of dog facing the same direction. The extent to which these reconstructions tend to retain the high-level semantics of the inputs rather than the low-level details suggests that BigBiGAN training encourages the encoder to model the former more so than the latter. Additional reconstructions are presented in Appendix B.

4 Related work

A number of approaches to unsupervised representation learning from images based on self-supervision have proven very successful. Self-supervision generally involves learning from tasks designed to resemble supervised learning in some way, but in which the "labels" can be created automatically from the data itself with no manual effort. An early example is relative location prediction [5], where a model is trained on input pairs of image patches and predicts their relative

locations. Contrastive predictive coding (CPC) [39, 17] is a recent related approach where, given an image patch, a model predicts which patches occur in other image locations. Other approaches include colorization [41, 42], motion segmentation [30], rotation prediction [11, 4], GAN-based discrimination [31, 4], and exemplar matching [8]. Rigorous empirical comparisons of many of these approaches have also been conducted [6, 24]. A key advantage offered by BigBiGAN and other approaches based on generative models, relative to most self-supervised approaches, is that their input may be the full-resolution image or other signal, with no cropping or modification of the data needed (though such modifications may be beneficial as data augmentation). This means the resulting representation can typically be applied directly to full data in the downstream task with no domain shift.

A number of relevant autoencoder and GAN variants have also been proposed. Associative compression networks (ACNs) [15] learn to compress at the dataset level by conditioning data on other previously transmitted data which are similar in code space, resulting in models that can "daydream" semantically similar samples, similar to BigBiGAN reconstructions. VQ-VAEs [40] pair a discrete (vector quantized) encoder with an autoregressive decoder to produce faithful reconstructions with a high compression factor and demonstrate representation learning results in reinforcement learning settings. In the adversarial space, adversarial autoencoders [28] proposed an autoencoder-style encoder-decoder pair trained with pixel-level reconstruction cost, replacing the KL-divergence regularization of the prior used in VAEs [23] with a discriminator. In another proposed VAE-GAN hybrid [25] the pixel-space reconstruction error used in most VAEs is replaced with feature space distance from an intermediate layer of a GAN discriminator. Other hybrid approaches like AGE [38] and α-GAN [33] add an encoder to stabilize GAN training. An interesting difference between many of these approaches and the BiGAN [10, 7] framework is that BiGAN does not train the encoder or generator with an explicit reconstruction cost. Though it can be shown that (Big)BiGAN implicitly minimizes a reconstruction cost, qualitative reconstruction results (Section 3.3) suggest that this reconstruction cost is of a different flavor, emphasizing high-level semantics over pixel-level details.

5 Discussion

We have shown that BigBiGAN, an unsupervised learning approach based purely on generative models, achieves state-of-the-art results in image representation learning on ImageNet. Our ablation study lends further credence to the hope that powerful generative models can be beneficial for representation learning, and in turn that learning an inference model can improve large-scale generative models. In the future we hope that representation learning can continue to benefit from further advances in generative models and inference models alike, as well as scaling to larger image databases.

Acknowledgments

The authors would like to thank Aidan Clark, Olivier Hénaff, Aäron van den Oord, Sander Dieleman, and many other colleagues at DeepMind for useful discussions and feedback on this work.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. In *arXiv:1603.04467*, 2015.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

[3] Peter Buchlovsky, David Budden, Dominik Grewe, Chris Jones, John Aslanides, Frederic Besse, Andy Brock, Aidan Clark, Sergio Gómez Colmenarejo, Aedan Pope, Fabio Viola, and Dan Belov. TF-Replicator: Distributed machine learning for researchers. In *arXiv:1902.00465*, 2019.

[4] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised GANs via auxiliary rotation loss. In *CVPR*, 2019.

[5] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[6] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *ICCV*, 2017.

[7] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017.

[8] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. In *NeurIPS*, 2014.

[9] Charles Dugas, Yoshua Bengio, François Belisle, Claude Nadeau, and Rene Garcia. Incorporating second-order functional knowledge for better option pricing. In *NeurIPS*, 2000.

[10] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. In *ICLR*, 2017.

[11] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

[12] Aidan N. Gomez, Mengye Ren, Raquel Urtasun, and Roger B. Grosse. The reversible residual network: Backpropagation without storing activations. In *NeurIPS*, 2017.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[14] Google. Cloud TPU. https://cloud.google.com/tpu/, 2019.

[15] Alex Graves, Jacob Menick, and Aäron van den Oord. Associative compression networks. In *arXiv:1804.02476*, 2018.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

[17] Olivier J. Hénaff, Ali Razavi, Carl Doersch, S. M. Ali Eslami, and Aäron van den Oord. Data-efficient image recognition with contrastive predictive coding. In *arXiv:1905.09272*, 2019.

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NeurIPS*, 2017.

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

[20] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[22] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *arXiv:1807.03039*, 2018.

[23] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *arXiv:1312.6114*, 2013.

[24] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *arXiv:1901.09005*, 2019.

[25] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

[26] Jae Hyun Lim and Jong Chul Ye. Geometric GAN. In *arXiv:1705.02894*, 2017.

[27] Mario Lucic, Michael Tschannen, Marvin Ritter, Xiaohua Zhai, Olivier Bachem, and Sylvain Gelly. High-fidelity image generation with fewer labels. In *ICML*, 2019.

[28] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. In *ICLR*, 2016.

[29] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[30] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017.

[31] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[32] Malcolm Reynolds, Gabriel Barth-Maron, Frederic Besse, Diego de Las Casas, Andreas Fidjeland, Tim Green, Adrià Puigdomènech, Sébastien Racanière, Jack Rae, and Fabio Viola. Open sourcing Sonnet - a new library for constructing neural networks. https://deepmind.com/blog/open-sourcing-sonnet/, 2017.

[33] Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, and Shakir Mohamed. Variational approaches for auto-encoding generative adversarial networks. In *arXiv:1706.04987*, 2017.

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

[35] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *arXiv:1606.03498*, 2016.

[36] Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *ICML*, 2016.

[37] Dustin Tran, Rajesh Ranganath, and David M. Blei. Hierarchical implicit models and likelihood-free variational inference. In *NeurIPS*, 2017.

[38] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. It takes (only) two: Adversarial generator-encoder networks. In *arXiv:1704.02304*, 2017.

[39] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *arXiv:1807.03748*, 2018.

[40] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *arXiv:1711.00937*, 2017.

[41] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *ECCV*, 2016.

[42] Richard Zhang, Phillip Isola, and Alexei A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, 2016.

Appendix A Model and optimization details

Our optimizer matches that of BigGAN [2] – we use Adam [21] with batch size 2048 and the same learning rates and other hyperparameters, using the $\mathcal{G}$ optimizer to update $\mathcal{E}$ simultaneously, with the same alternating optimization: two $\mathcal{D}$ updates followed by a single joint update of $\mathcal{G}$ and $\mathcal{E}$. (We do not use orthogonal regularization used in [2], finding it gave worse results in the unconditional setting, matching the findings of [27].) Spectral normalization [29] is used in $\mathcal{G}$ and $\mathcal{D}$, but not in $\mathcal{E}$. Full cross-replica batch normalization is used in both $\mathcal{G}$ and $\mathcal{E}$ (including for the linear classifier training on $\mathcal{E}$ features used for evaluations). We also apply exponential moving averaging (EMA) with a decay of 0.9999 to the $\mathcal{G}$ and $\mathcal{E}$ weights in all evaluations. (We find this results in only a small improvement for $\mathcal{E}$ evaluations, but a substantial one for $\mathcal{G}$ evaluations.)

At BigBiGAN training time, as well as linear classification evaluation training time, we preprocess inputs with ResNet [16]-style data augmentation, though with crops of size 128 or 256 rather than 224[7].

For linear classification evaluations in the ablations reported in Table 1, we hold out 10K randomly selected images from the official ImageNet [34] training set as a validation set and report accuracy on that validation set, which we call $\text{train}_{\text{val}}$. All results in Table 1 are run for 500K steps, with early stopping based on linear classifier accuracy on our $\text{train}_{\text{val}}$ split. In all of these models the linear classifier is initialized to 0 and trained for 5K Adam steps with a (high) learning rate of 0.01 and EMA smoothing with decay 0.9999. We have found it helpful to monitor representation learning progress during BigBiGAN training by periodically rerunning this linear classification evaluation from scratch given the current $\mathcal{E}$ weights, resetting the classifier weights to 0 before each evaluation.

In Table 2 we extend the BigBiGAN training time to 1M steps, and report results on the official validation set of 50K images for comparison with prior work. The classifier in these results is trained for 100K Adam steps, sweeping over learning rates $\{10^{-4}, 3 \cdot 10^{-4}, 10^{-3}, 3 \cdot 10^{-3}, 10^{-2}\}$, again applying EMA with decay 0.9999 to the classifier weights. Hyperparameter selection and early stopping is again based on classification accuracy on $\text{train}_{\text{val}}$. As in [2], FID is reported against statistics over the full ImageNet training set, preprocessed by resizing the minor axis to the $\mathcal{G}$ output resolution and taking the center crop along the major axis, except as noted in Table 3, where we also report FID against the validation set for comparison with [27].

All models were trained via TensorFlow [1] and Sonnet [32] with data parallelism on TPU pod slices [14] using 32 to 512 cores, coordinated by TF-Replicator [3].

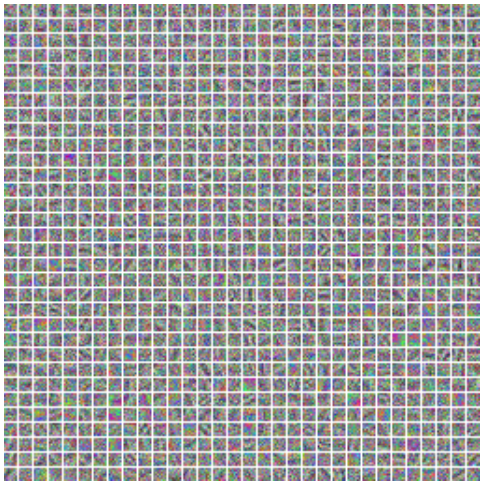
Supervised model performance. In Table 4 we present the results of fully supervised training with the model architectures used in our experiments in Section 3 for comparison purposes.

Architecture	Top-1	Top-5
ResNet-50	76.3	93.1
ResNet-101	77.8	93.8
RevNet-50	71.8	90.5
RevNet-50 $\times 2$	74.9	92.2
RevNet-50 $\times 4$	76.6	93.1

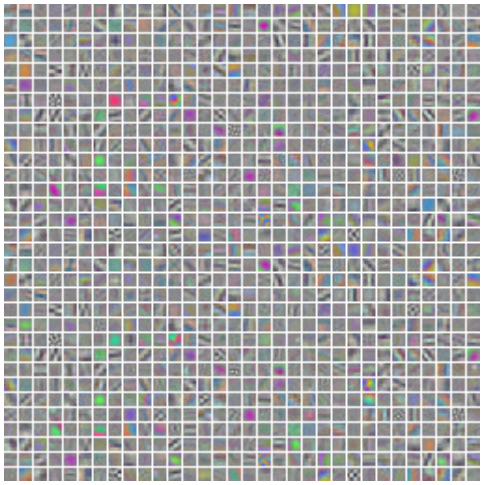
Table 4: ImageNet validation set accuracy for fully supervised end-to-end training of the model architectures used in our representation learning experiments.

First layer convolutional filters. In Figure 3 we visualize the learned convolutional filters for the first convolutional layer of our BigBiGAN encoders $\mathcal{E}$ using the largest RevNet $\times 4$ $\mathcal{E}$ architecture. Note the difference between the filters in (a) and (b) (corresponding to rows *RevNet* $\times 4$ and *RevNet* $\times 4$ *(*$\uparrow$ $\mathcal{E}$ *LR)* in Table 1). In (b) we use the higher $\mathcal{E}$ learning rate and see a corresponding qualitative improvement in the appearance of the learned filters, with less noise and more Gabor-like and color filters, as observed in BiGAN [7]. This suggests that examining the convolutional filters of the input layer can serve as a diagnostic for undertrained models.

[7] Preprocessing code from the TensorFlow ResNet TPU model: https://github.com/tensorflow/tpu/tree/master/models/official/resnet.



(a) RevNet $\times 4$



(b) + $\uparrow \mathcal{E}$ LR

Figure 3: Visualization of first layer convolutional filters for our unsupervised BigBiGAN models with the RevNet $\times 4$ $\mathcal{E}$ architecture, which includes 1024 filters. (Best viewed with zoom.)

Model	Samples			Reconstructions	
	Image	IS (↑)	FID (↓)	Image	Rel. ℓ_1 Error % (↓)
Base	Figure 4	24.10	30.14	Figure 5	70.54
Light Augmentation	Figure 6	27.09	20.96	Figure 7	72.53
High Res $\mathcal{E}$ (256)	Figure 8	24.91	26.56	Figure 9	70.60
High Res $\mathcal{G}$ (256)	Figure 10	25.73	37.21	Figure 11	77.70

Table 5: Links to BigBiGAN samples and reconstructions with associated metrics.

Appendix B Samples and reconstructions

In this Appendix we present BigBiGAN samples and reconstructions from several variants of the method. Table 5 includes pointers to samples and reconstruction images, as well as relevant metrics. The samples were selected by best FID vs. training set statistics, and we show the IS and FID along with sample images at that point. The reconstructions were selected by best (lowest) relative pixel-wise ℓ_1 error, the error metric presented in Table 5, computed as:

$$E_{\text{Rel}\ell_1} = \frac{\mathbb{E}_{\mathbf{x} \sim P_\mathbf{x}} ||\mathbf{x} - \mathcal{G}(\mathcal{E}(\mathbf{x}))||_1}{\mathbb{E}_{\mathbf{x}, \mathbf{x}' \sim P_\mathbf{x}} ||\mathbf{x}' - \mathcal{G}(\mathcal{E}(\mathbf{x}))||_1},$$

where $\mathbf{x}$ and $\mathbf{x}'$ are independent data samples, and $||\mathbf{x}' - \mathcal{G}(\mathcal{E}(\mathbf{x}))||_1$ serves as a "baseline" reconstruction error relative to a "random" input. For example, with a random initialization of $\mathcal{G}$ and $\mathcal{E}$, we have $E_{\text{Rel}\ell_1} \approx 1$. This relative metric penalizes degenerate reconstructions, such as the mean image, which would sometimes achieve low absolute reconstruction error despite having no perceptual similarity to the inputs. despite that the resulting images having no perceptual similarity to the inputs. In practice, given N data samples $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{N-1}$ (we use $N = 50$K), we estimate the denominator by comparing each sample $\mathbf{x}_i$ with a single neighbor $\mathbf{x}_{(i+1) \bmod N}$, computing:

$$E_{\text{Rel}\ell_1} \approx \frac{\sum_{i=0}^{N-1} ||\mathbf{x}_i - \mathcal{G}(\mathcal{E}(\mathbf{x}_i))||_1}{\sum_{i=0}^{N-1} ||\mathbf{x}_{(i+1) \bmod N} - \mathcal{G}(\mathcal{E}(\mathbf{x}_i))||_1}$$

Iterated reconstruction To further explore the behavior of a BigBiGAN (or any other model capable of approximately reconstructing its input), we can "iterate" the reconstruction operation. In particular, let $R_i(\mathbf{x})$ be defined for non-negative integers i and input images $\mathbf{x}$ as:

$$R_0(\mathbf{x}) = \mathbf{x}$$
$$R_{i+1}(\mathbf{x}) = \mathcal{G}(\mathcal{E}(R_i(\mathbf{x})))$$

In Figure 12 we show the results of up to 500 steps of this process for a few sample images. Qualitatively, the first several steps of this process often appear to retain some semantics of the input image $\mathbf{x}$. After dozens or hundreds of iterations, however, little content from the original input apparently remains intact.

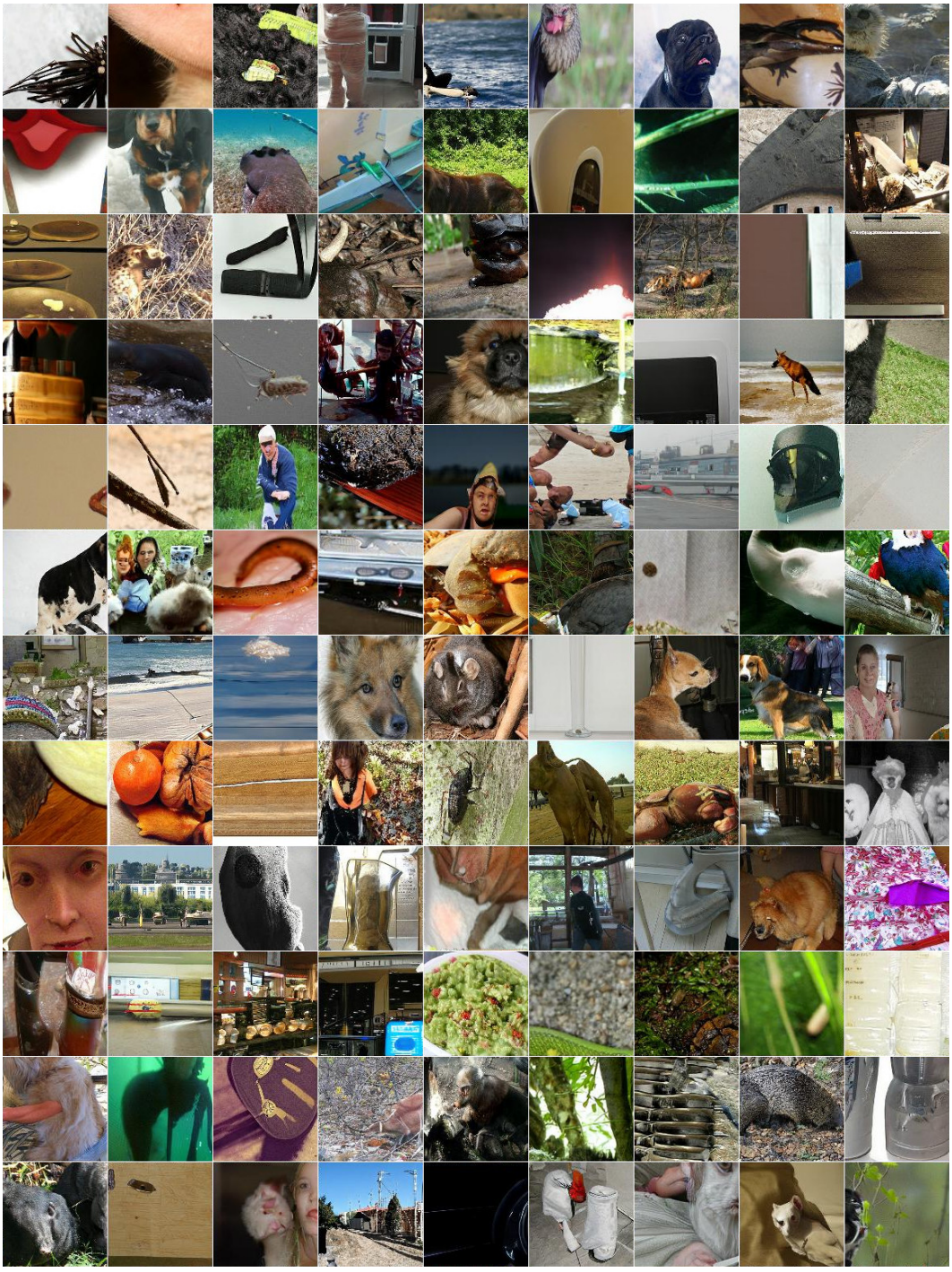


Figure 4: 128×128 samples $\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z})$ from an unsupervised BigBiGAN generator $\mathcal{G}$, trained using the *Base* method from Table 1.

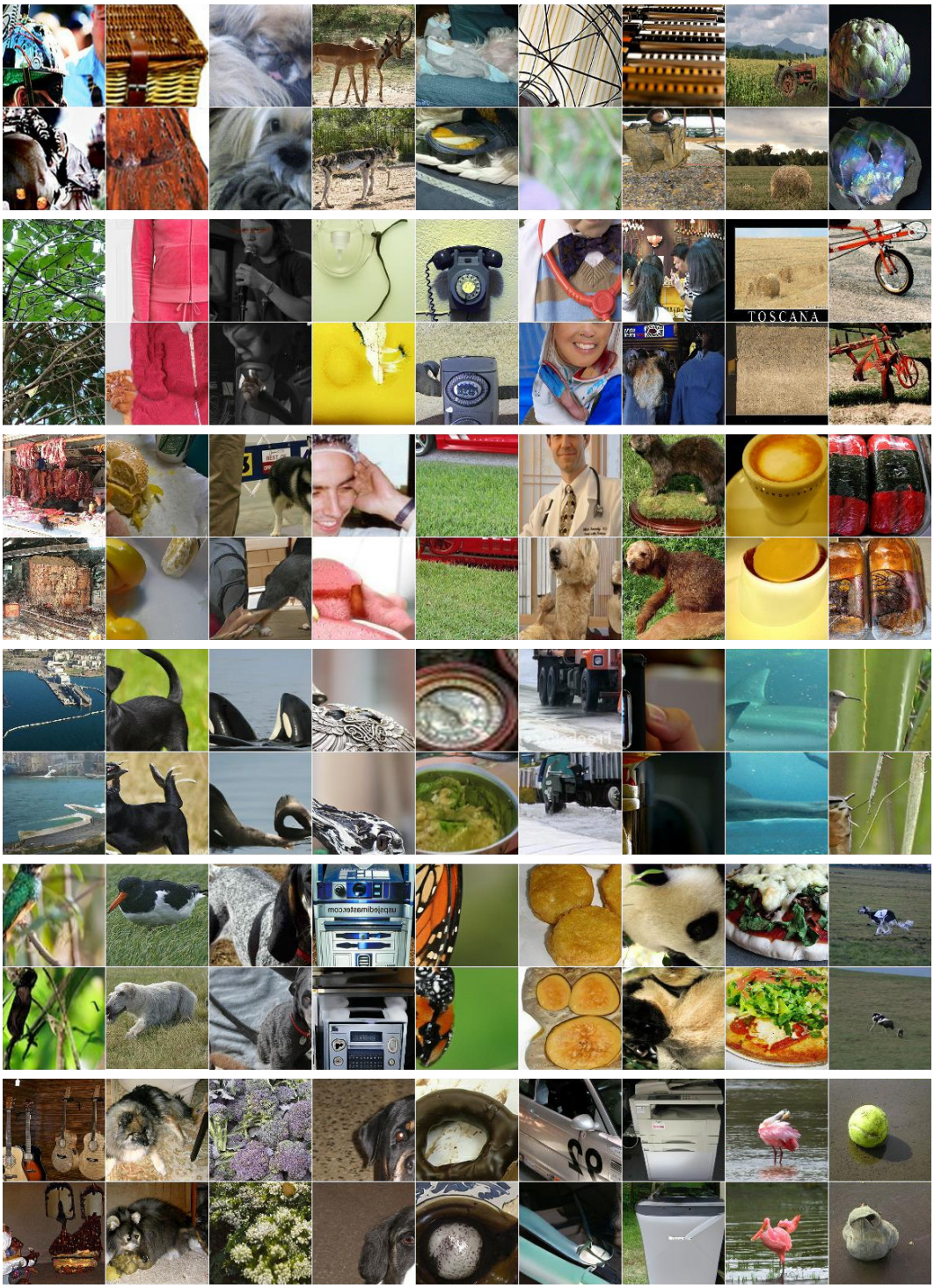


Figure 5: 128×128 reconstructions from an unsupervised BigBiGAN model, trained using the *Base* method from Table 1. The top rows of each pair are real data $\mathbf{x} \sim P_{\mathbf{x}}$, and bottom rows are generated reconstructions computed by $\mathcal{G}(\mathcal{E}(\mathbf{x}))$.

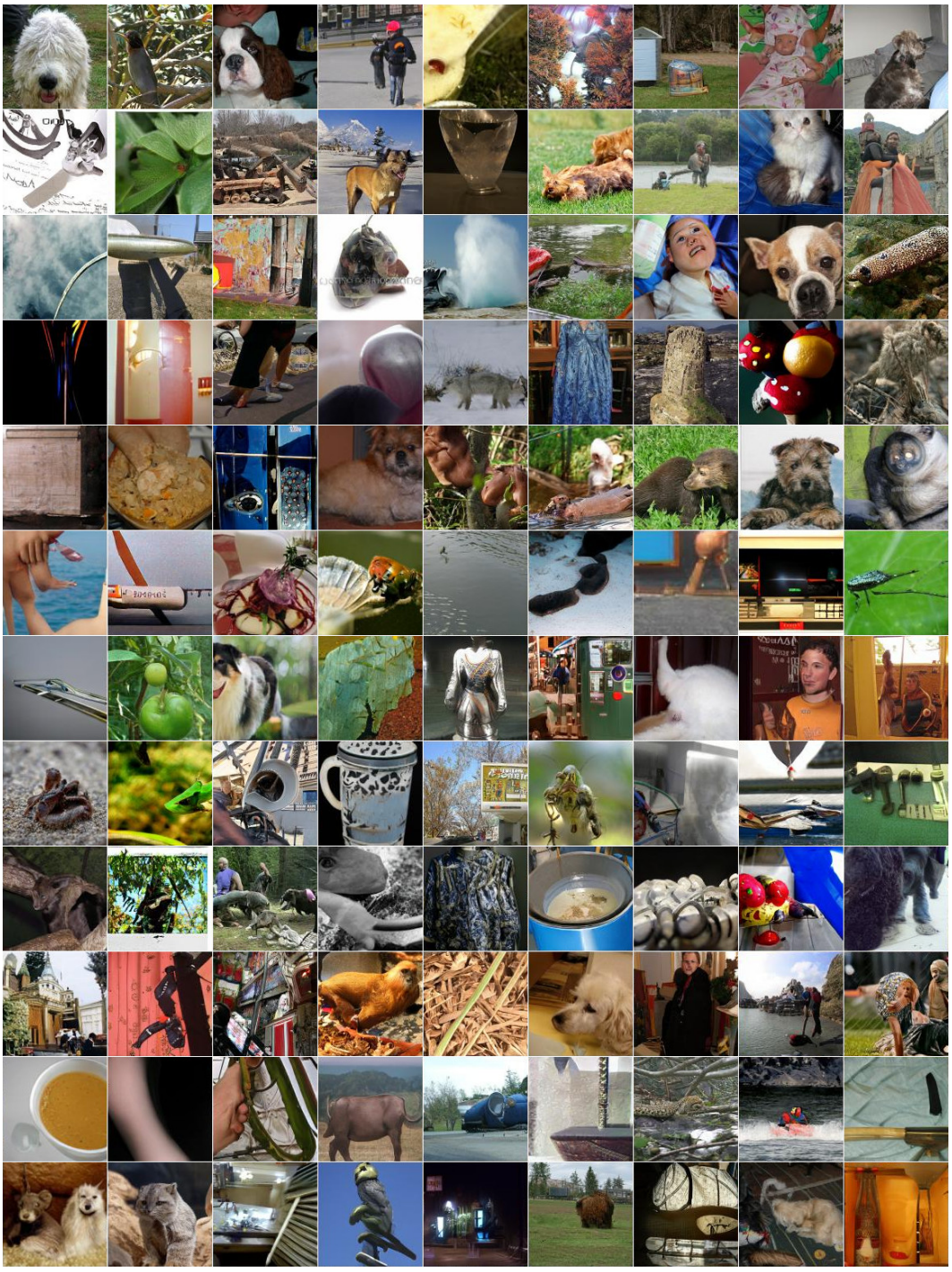


Figure 6: 128×128 samples $\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z})$ from an unsupervised BigBiGAN generator $\mathcal{G}$, trained using the lighter augmentation from [27] with generation results reported in Table 3.

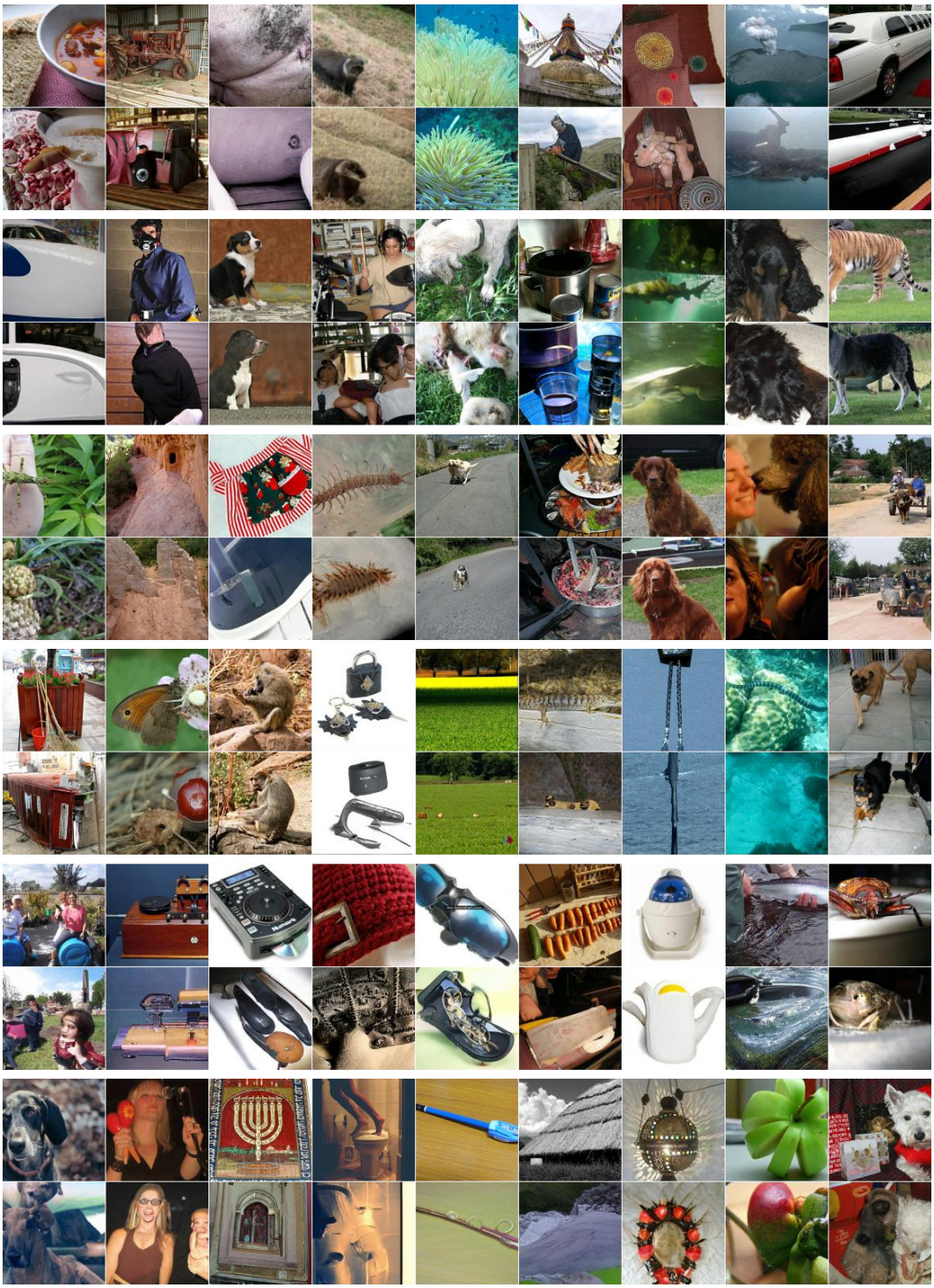


Figure 7: 128×128 reconstructions from an unsupervised BigBiGAN model, trained using the lighter augmentation from [27] with generation results reported in Table 3. The top rows of each pair are real data $\mathbf{x} \sim P_{\mathbf{x}}$, and bottom rows are generated reconstructions computed by $\mathcal{G}(\mathcal{E}(\mathbf{x}))$.

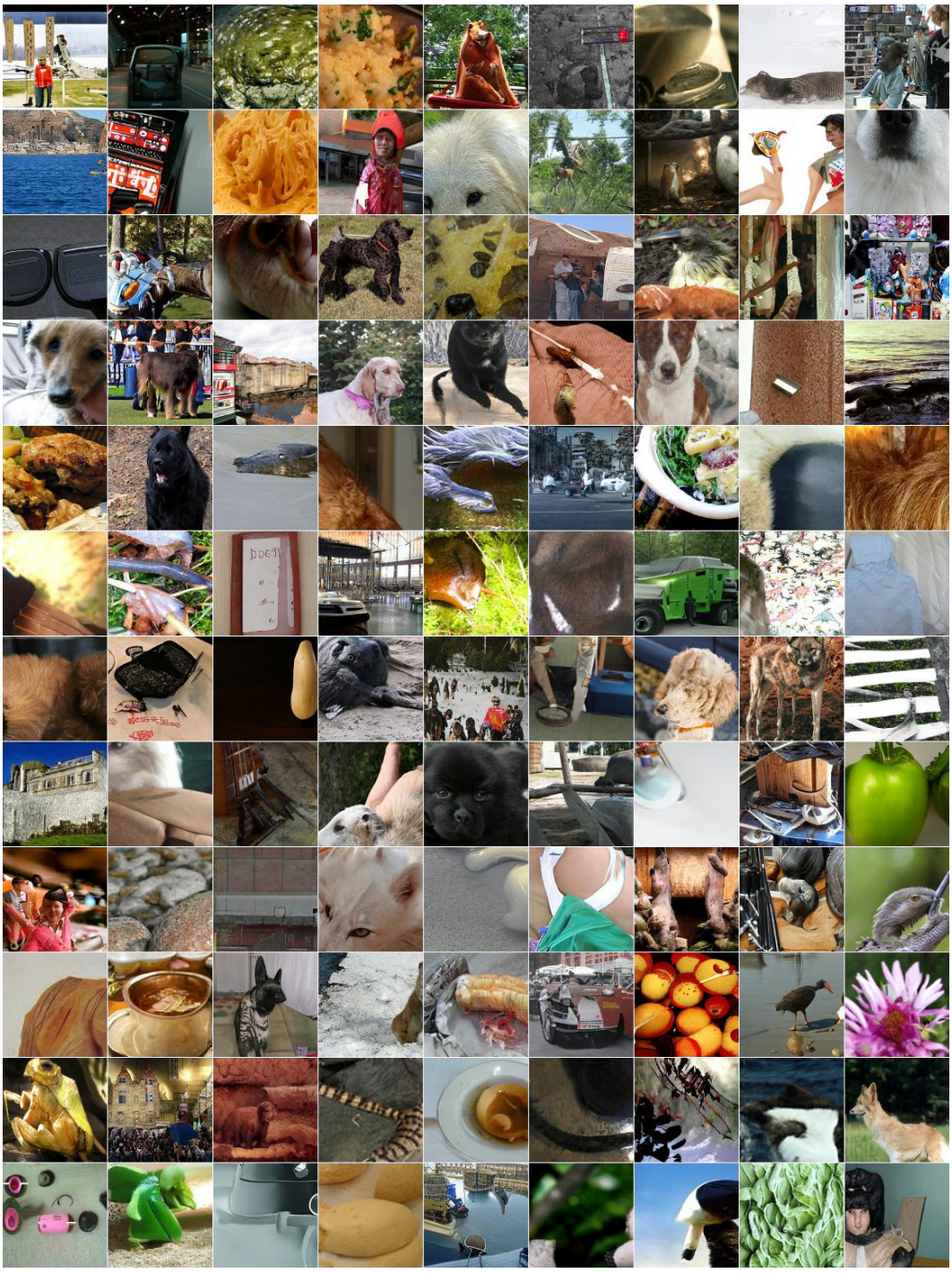


Figure 8: 128×128 samples $\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z})$ from an unsupervised BigBiGAN generator $\mathcal{G}$, trained using the *High Res $\mathcal{E}$ (256)* configuration from Table 1.

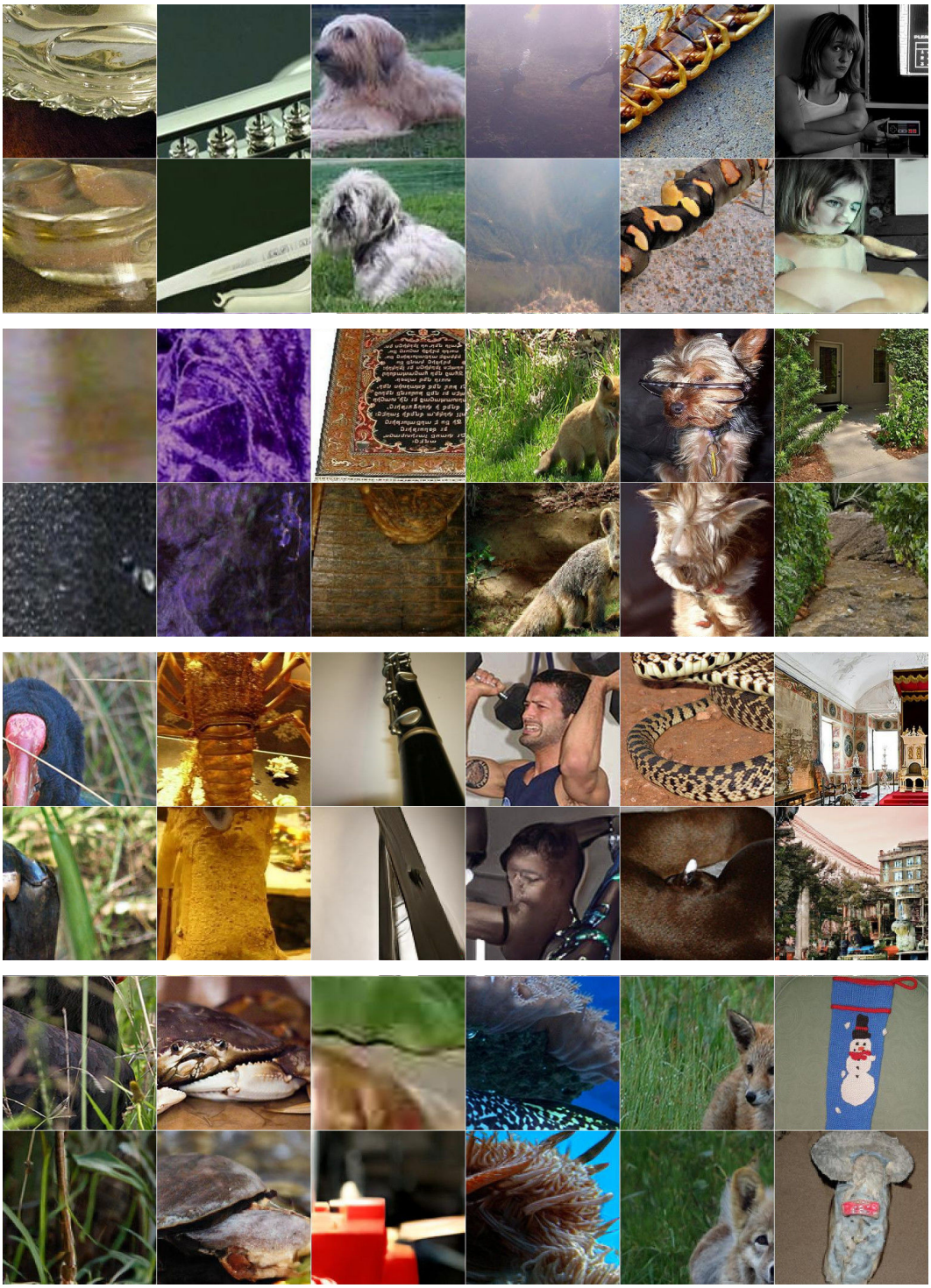


Figure 9: 128×128 reconstructions of 256×256 encoder input images from an unsupervised BigBiGAN model, trained using the *High Res $\mathcal{E}$ (256)* configuration from Table 1. Reconstructions are upsampled from 128×128 to 256×256 for visualization. The top rows of each pair are real data $\mathbf{x} \sim P_{\mathbf{x}}$, and bottom rows are generated reconstructions computed by $\mathcal{G}(\mathcal{E}(\mathbf{x}))$.

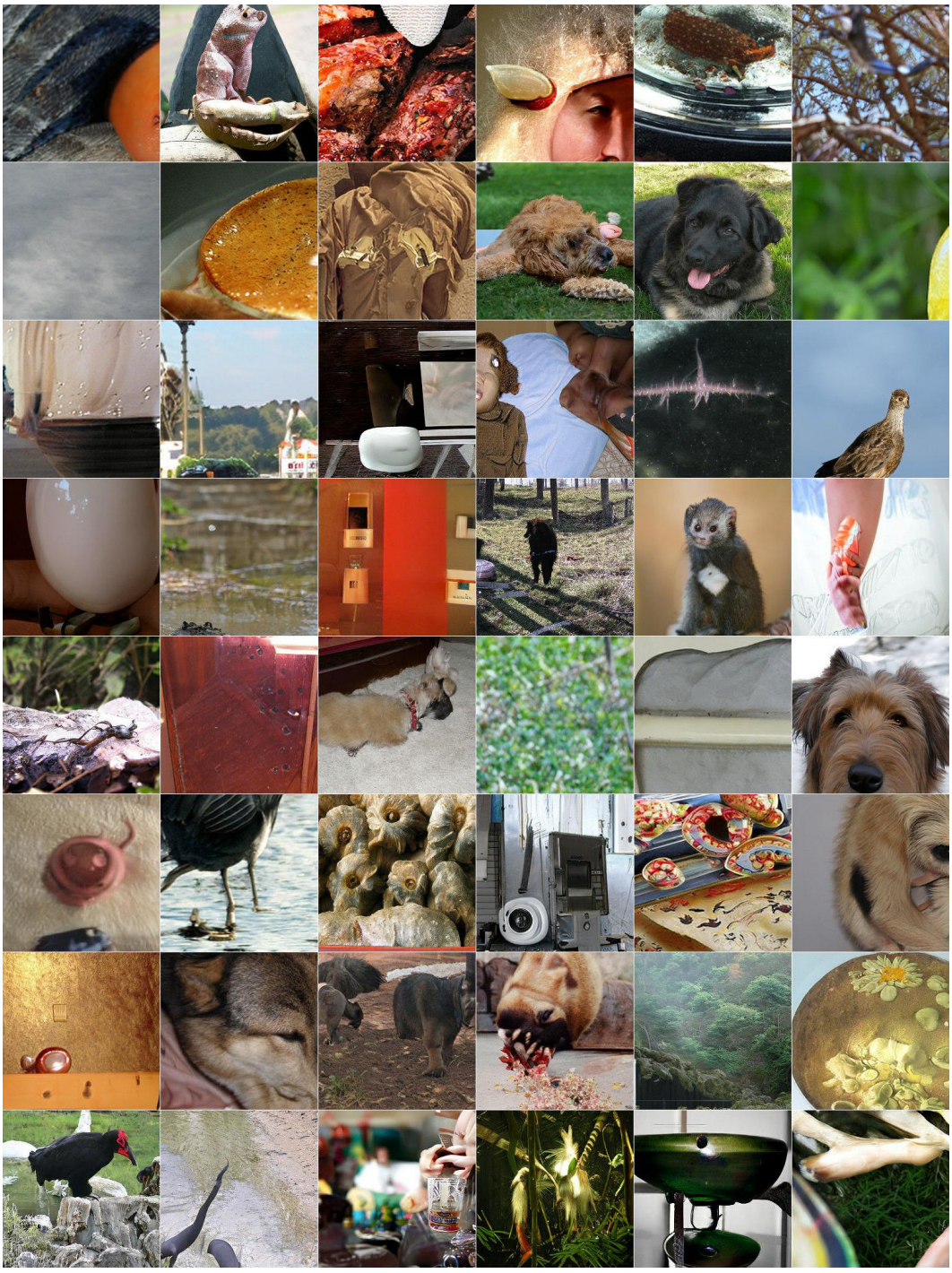


Figure 10: 256×256 samples $\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z})$ from an unsupervised BigBiGAN generator $\mathcal{G}$, trained with a high-resolution $\mathcal{E}$ and $\mathcal{G}$ (*High Res $\mathcal{G}$ (256)* from Table 1).

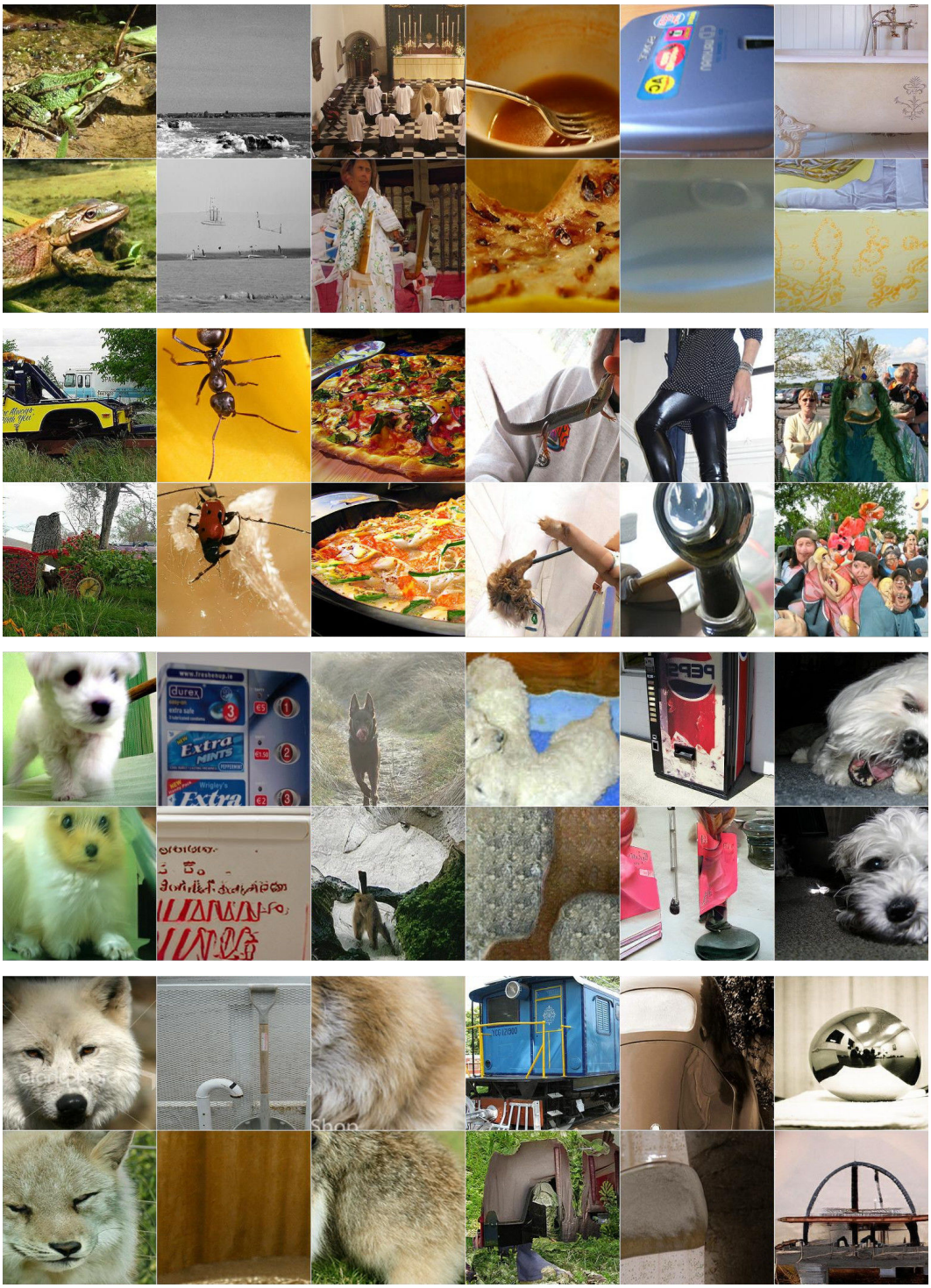


Figure 11: 256×256 reconstructions from an unsupervised BigBiGAN model, trained with a high-resolution $\mathcal{E}$ and $\mathcal{G}$ (*High Res $\mathcal{G}$ (256)* from Table 1). The top rows of each pair are real data $\mathbf{x} \sim P_{\mathbf{x}}$, and bottom rows are generated reconstructions computed by $\mathcal{G}(\mathcal{E}(\mathbf{x}))$.

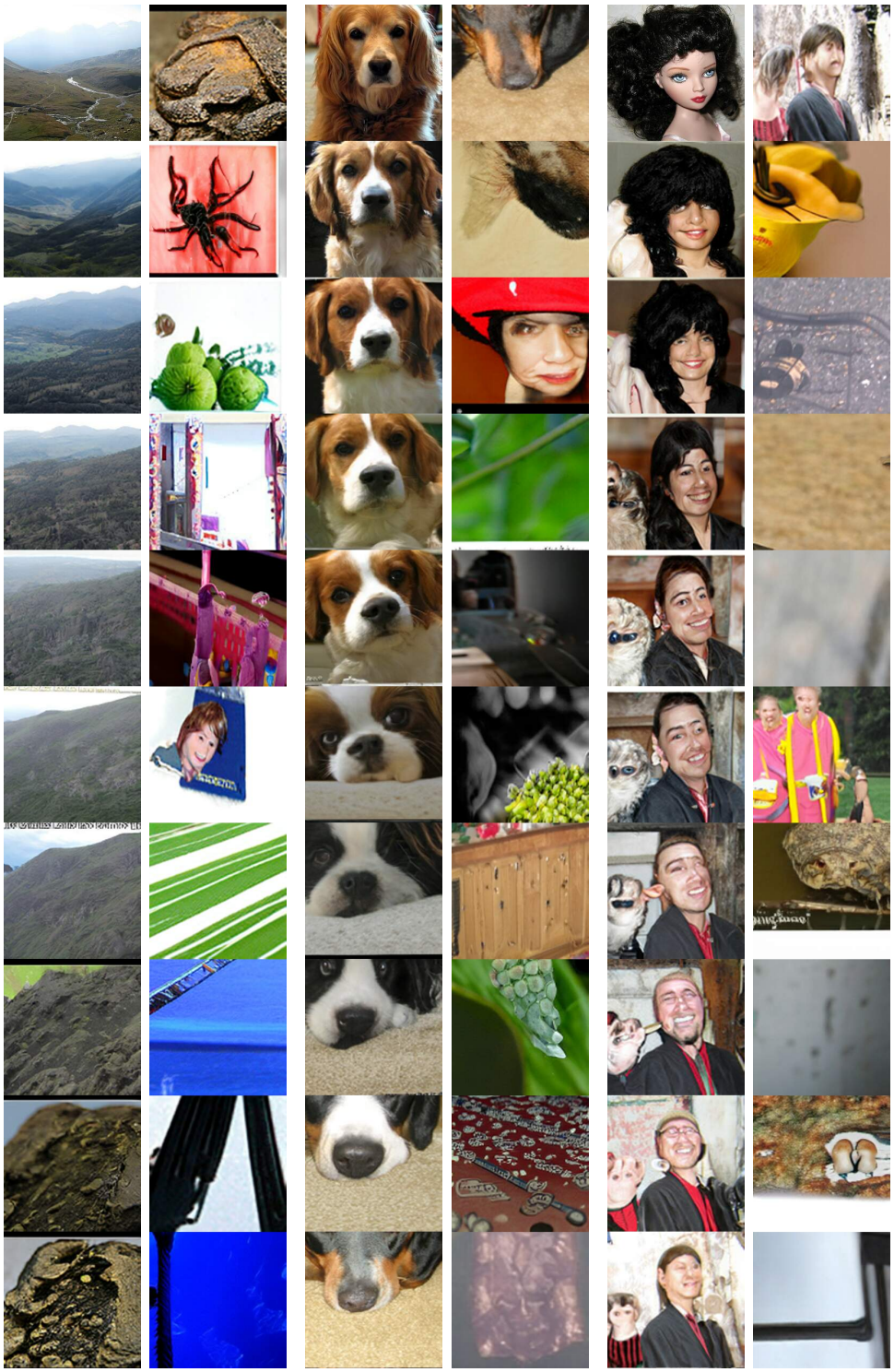


Figure 12: Iterated reconstructions from an unsupervised BigBiGAN model, trained using the *ResNet ($\uparrow \mathcal{E}$ LR)* method from Table 1, computed by recursively running the reconstruction operation $\mathcal{G}(\mathcal{E}(\cdot))$ on its own output as described in Appendix B. In each pair of columns, the left column shows a real input image R_0 at the top, and R_1 through R_9 in the remaining rows, the results of iterating reconstruction one to nine times, The right column shows the result of up to 500 iterations sampled at longer intervals, displaying R_{10}, R_{20}, R_{30}, R_{40}, R_{50}, R_{100}, R_{200}, R_{300}, R_{400}, and R_{500}.

Metric	Top-1 / Top-5 Acc. (%) $k = 1$	$k = 5$	$k = 25$	$k = 50$
D_1	38.09 / -	41.28 / 58.56	43.32 / 65.12	42.73 / 66.22
D_2	35.68 / -	38.61 / 55.59	40.65 / 62.23	40.15 / 63.42

Table 6: Accuracy of k nearest neighbors classifiers in BigBiGAN feature space on the ImageNet validation set. We report results under the normalized ℓ_1 distance D_1 as well as the normalized ℓ_2 (cosine) distance D_2.

Appendix C Nearest neighbors

In this Appendix we consider an alternative way of evaluating representations — by means of k nearest neighbors classification, which does not involve learning any parameters during evaluation and is even simpler than learning a linear classifier as done in Section 3. For all results in this section, we use the outputs of the global average pooling layer (a flat 8192D feature) of our best performing model, *RevNet* $\times 4$, $\uparrow \mathcal{E}$ *LR*. We do not do any data augmentation for either the training or validation sets: we simply crop each image at the center of its larger axis and resize to 256×256.

We use a normalized ℓ_1 or ℓ_2 distance metric as our nearest neighbors criterion, defined as $D_p(a, b) = \left|\left| \frac{a}{||a||_p} - \frac{b}{||b||_p} \right|\right|_p$, for $p \in \{1, 2\}$. (D_2 corresponds to cosine distance.) For label predictions with multiple neighbors ($k > 1$), we use a simple counting scheme: the label with the most votes is selected as the prediction. Ties (multiple labels with the same number of votes) are broken by $k = 1$ nearest neighbor classification among the data with the tied labels.

Quantitative results. In Table 6 we present k nearest neighbors classification results for $k \in \{1, 5, 25, 50\}$. Across all k, the ℓ_1-based metric D_1 outperforms D_2, and the remainder of our discussion refers to the D_1 results. With just a single neighbor ($k = 1$) we achieve a top-1 accuracy around 38%. Top-1 accuracy reaches 43% with $k = 25$, dropping off slightly at $k = 50$ as votes from more distant neighbors are added.

Qualitative results. Figure 13 shows sample nearest neighbors in the ImageNet training set for query images in the validation set. Despite being fully unsupervised, the neighbors in many cases match the query image in terms of high-level semantic content such as the category of the object of interest, demonstrating BigBiGAN's ability to capture high-level attributes of the data in its unsupervised representations. Where applicable, the object's pose and position in the image appears to be important as well – for example, the nearest neighbors of the RV (row 2, column 2) are all RVs facing roughly the same direction. In other cases, the nearest neighbors appear to be selected primarily based on the background or color scheme.

Discussion. While our quantitative k nearest neighbors classification results are far from the state of the art for ImageNet classification and significantly below the linear classifier-based results reported in Table 2, note that in this setup, no supervised learning of model parameters from labels occurs at any point: labels are predicted purely based on distance in a feature space learned from BigBiGAN training on image pixels alone. We believe this makes nearest neighbors classification an interesting additional benchmark for future approaches to unsupervised representation learning.

Figure 13: Nearest neighbors in BigBiGAN $\mathcal{E}$ feature space, from our best performing model (*RevNet* $\times 4$, $\uparrow \mathcal{E}$ *LR*). In each row, the first (left) column is a query image, and the remaining columns are its three nearest neighbors from the training set (the leftmost being the nearest, next being the second nearest, etc.). The query images above are the first 24 images in the ImageNet validation set.

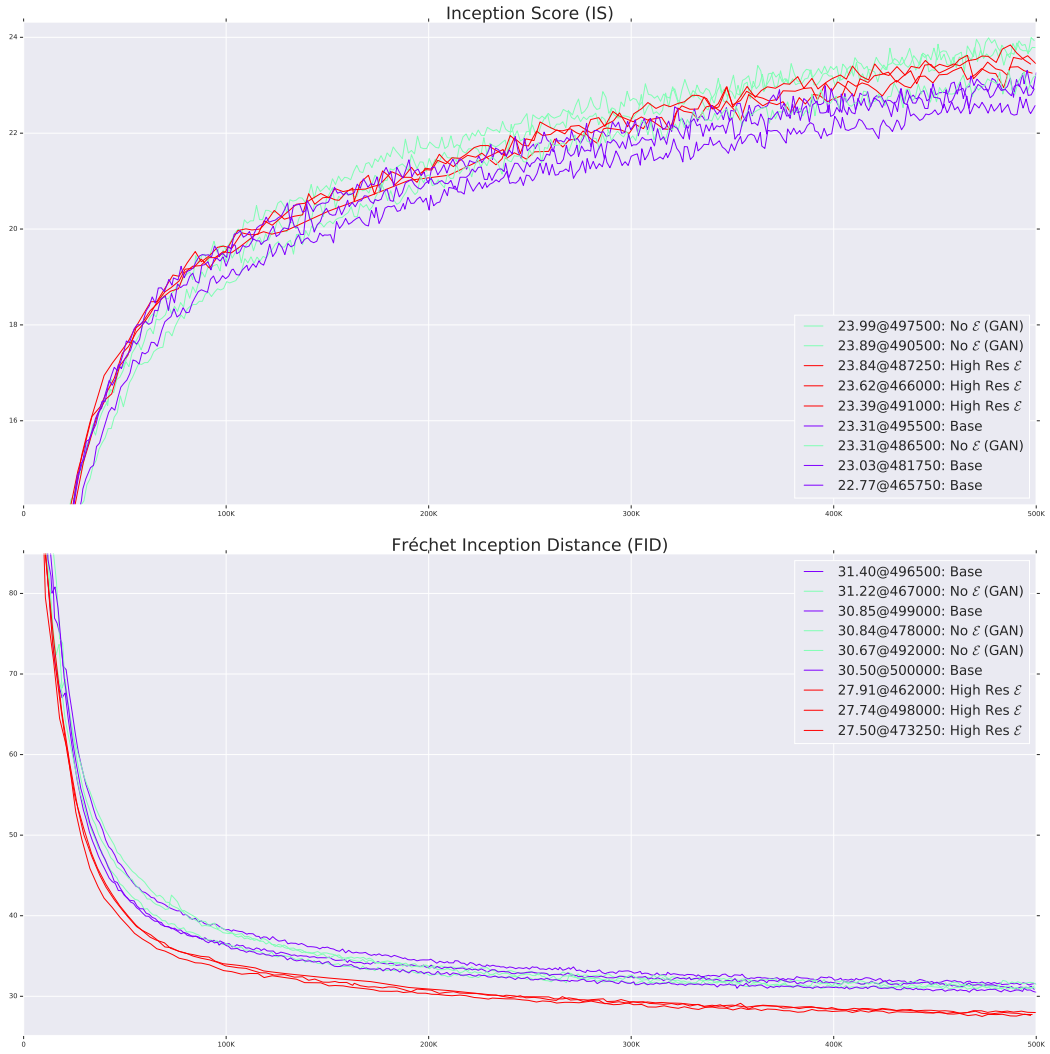


Figure 14: Image generation learning curves for several of the ablations in Section 3, including a comparison of BigBiGAN to standard GAN. Legend entries correspond to the following rows in Table 1: *Base*, *No $\mathcal{E}$ (GAN)*, and *High Res $\mathcal{E}$ (256)*.

Appendix D Learning curves

In this Appendix we present learning curves showing how the image generation and representation learning metrics that we measured evolve throughout training, as a more detailed view of the results in Section 3, Table 1. We include plots for the following results:

- Image generation (Figure 14)
- Latent distribution $P_{\mathbf{z}}$ and stochastic $\mathcal{E}$ (Figure 15)
- Unary loss terms (Figure 16)
- $\mathcal{G}$ capacity (Figure 17)
- High resolution $\mathcal{E}$ with varying resolution $\mathcal{G}$ (Figure 18)
- $\mathcal{E}$ architecture (Figure 19)
- Decoupled $\mathcal{E}$/$\mathcal{G}$ learning rates (Figure 20)

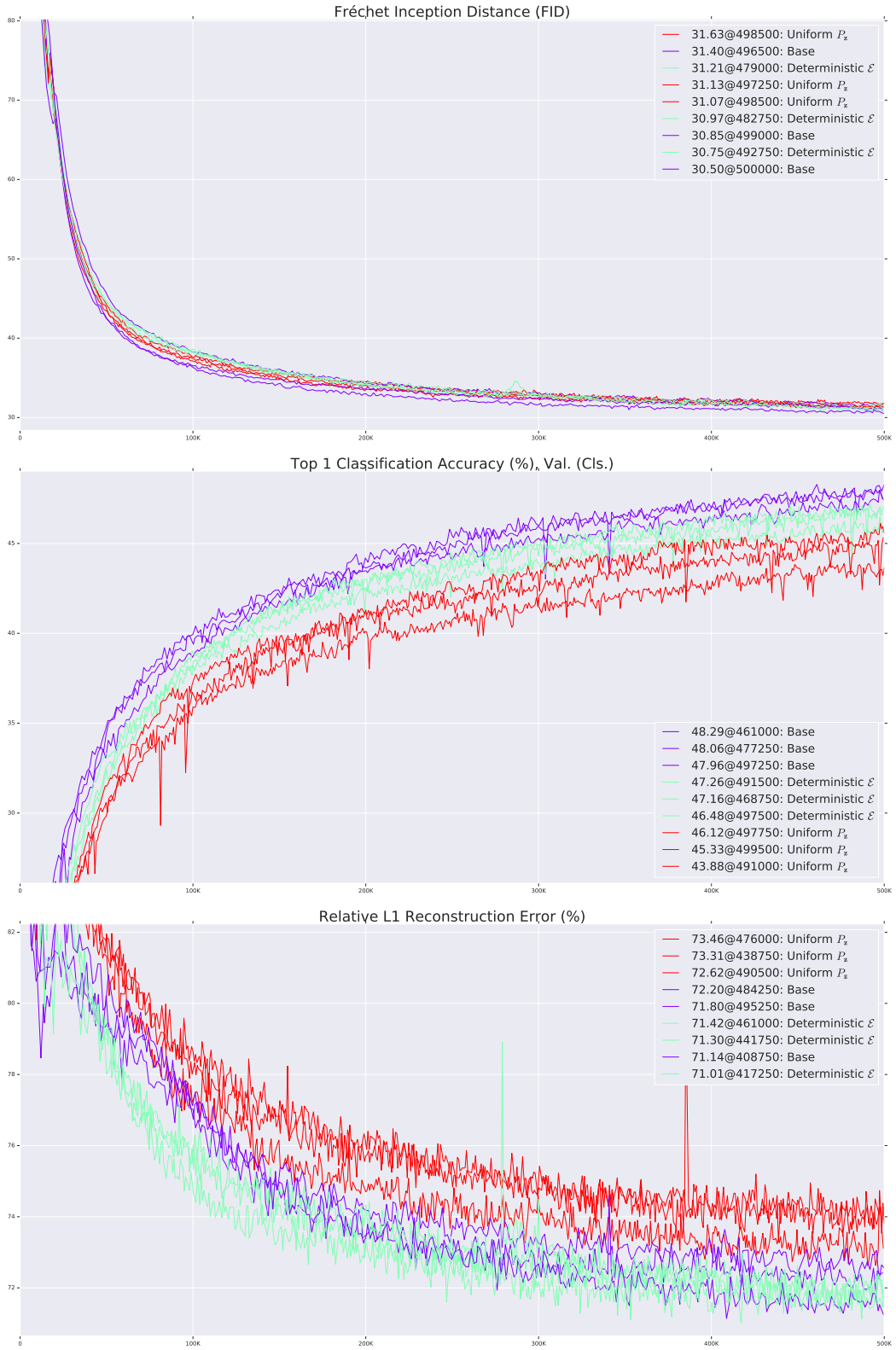


Figure 15: Image generation and representation learning curves for the latent space variations explored in Section 3. Legend entries correspond to the following rows in Table 1: *Base*, *Deterministic* $\mathcal{E}$, and *Uniform* $P_{\mathbf{z}}$.

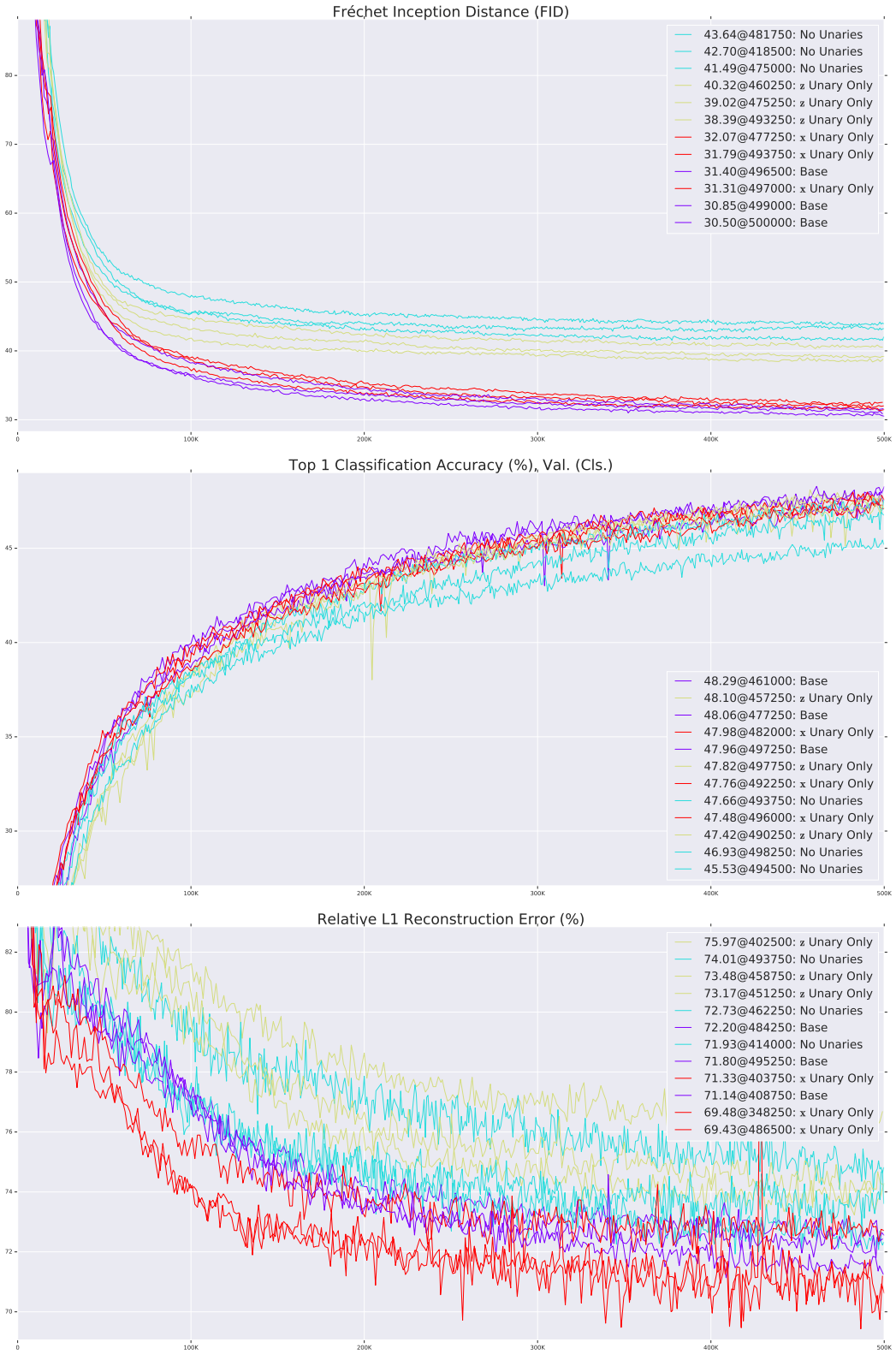


Figure 16: Image generation and representation learning curves for the unary loss component variations explored in Section 3. Legend entries correspond to the following rows in Table 1: *Base*, $\mathbf{x}$ *Unary Only*, $\mathbf{z}$ *Unary Only*, and *No Unaries (BiGAN)*.

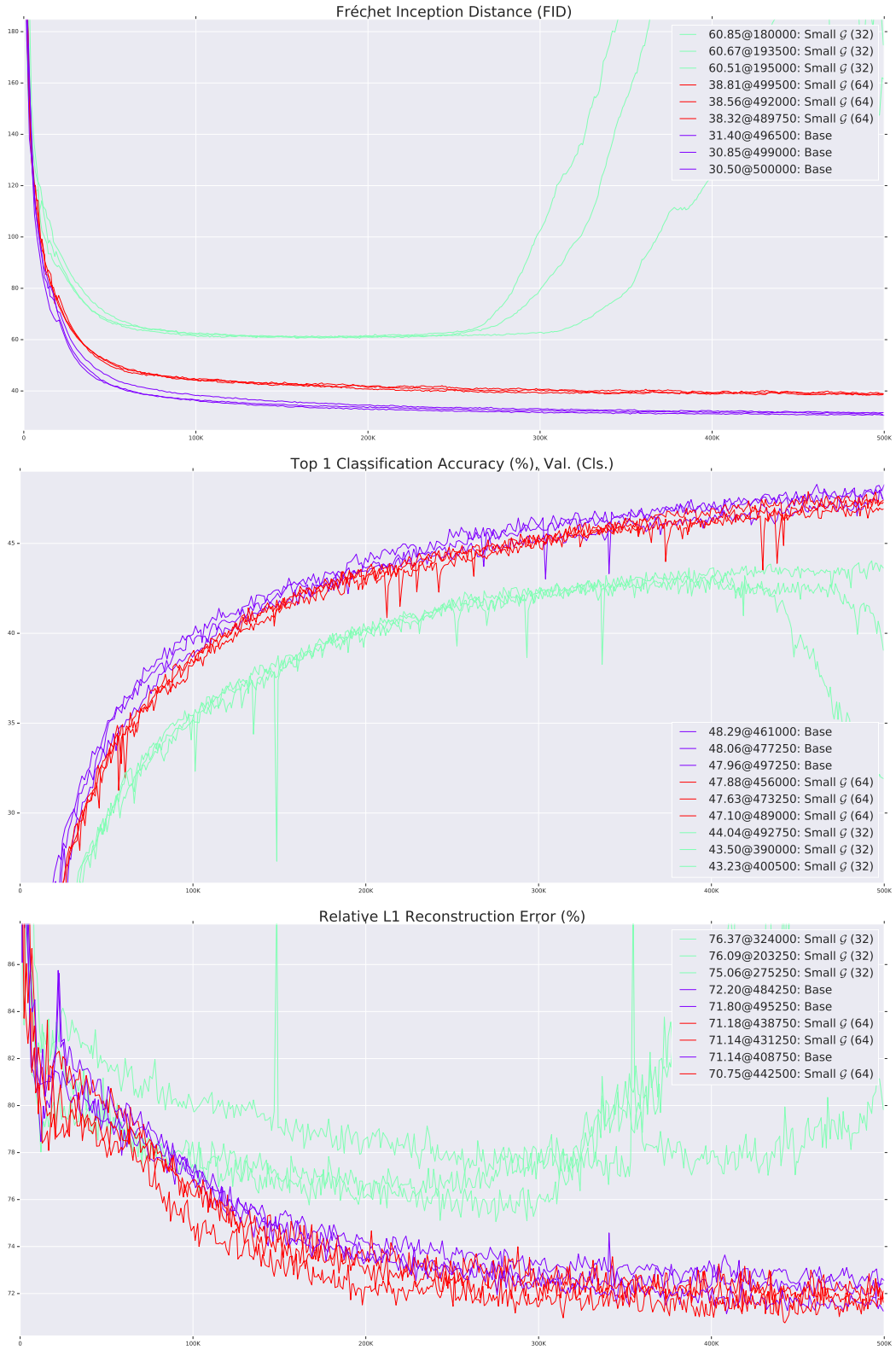


Figure 17: Image generation and representation learning curves for the $\mathcal{G}$ size variations explored in Section 3. Legend entries correspond to the following rows in Table 1: *Base*, *Small $\mathcal{G}$ (32)*, and *Small $\mathcal{G}$ (64)*.

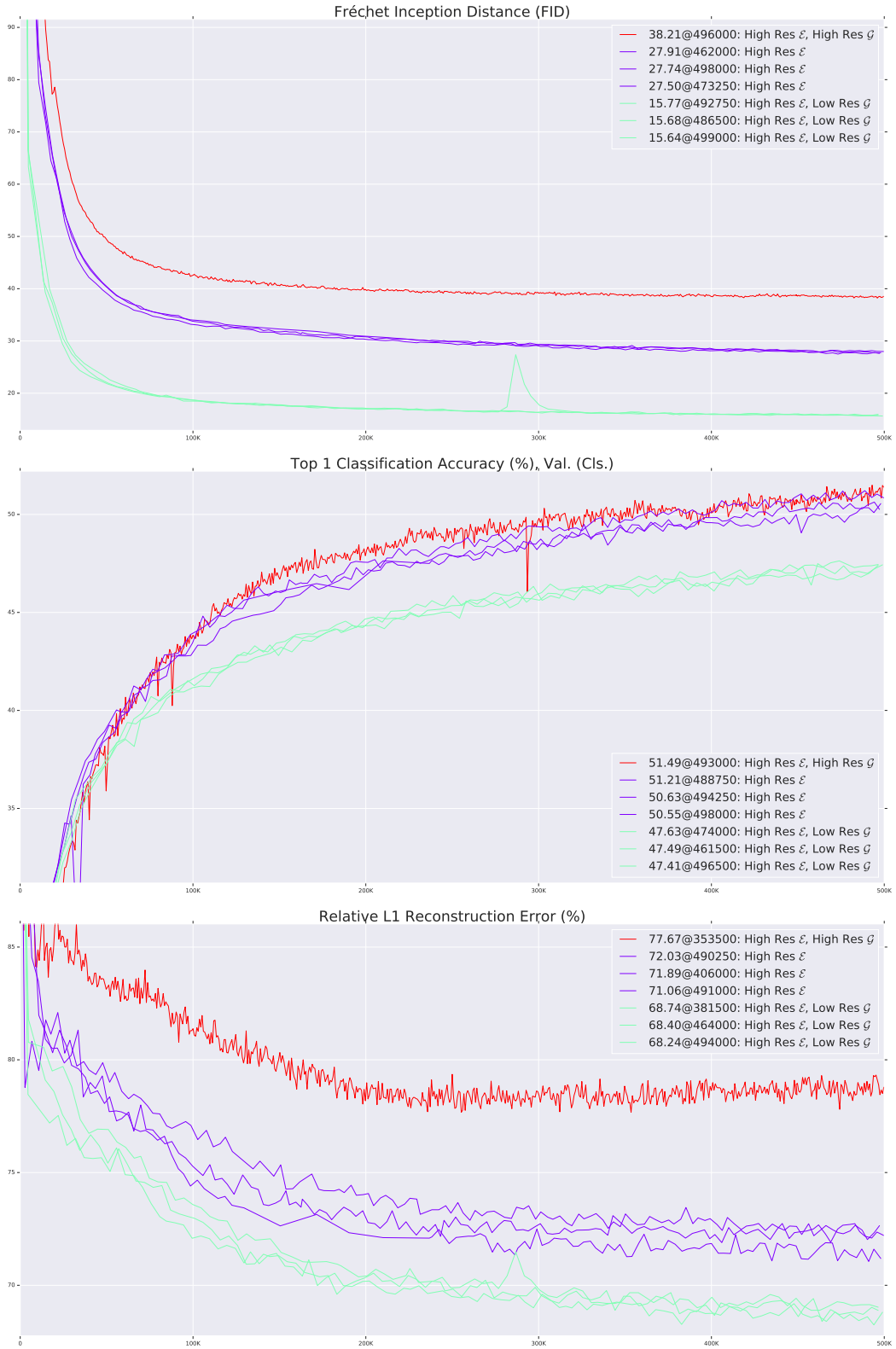


Figure 18: Image generation and representation learning curves for high resolution $\mathcal{E}$ with varying resolution $\mathcal{G}$ explored in Section 3. Legend entries correspond to the following rows in Table 1: *High Res $\mathcal{E}$ (256)*, *Low Res $\mathcal{G}$ (64)*, and *High Res $\mathcal{G}$ (256)*.

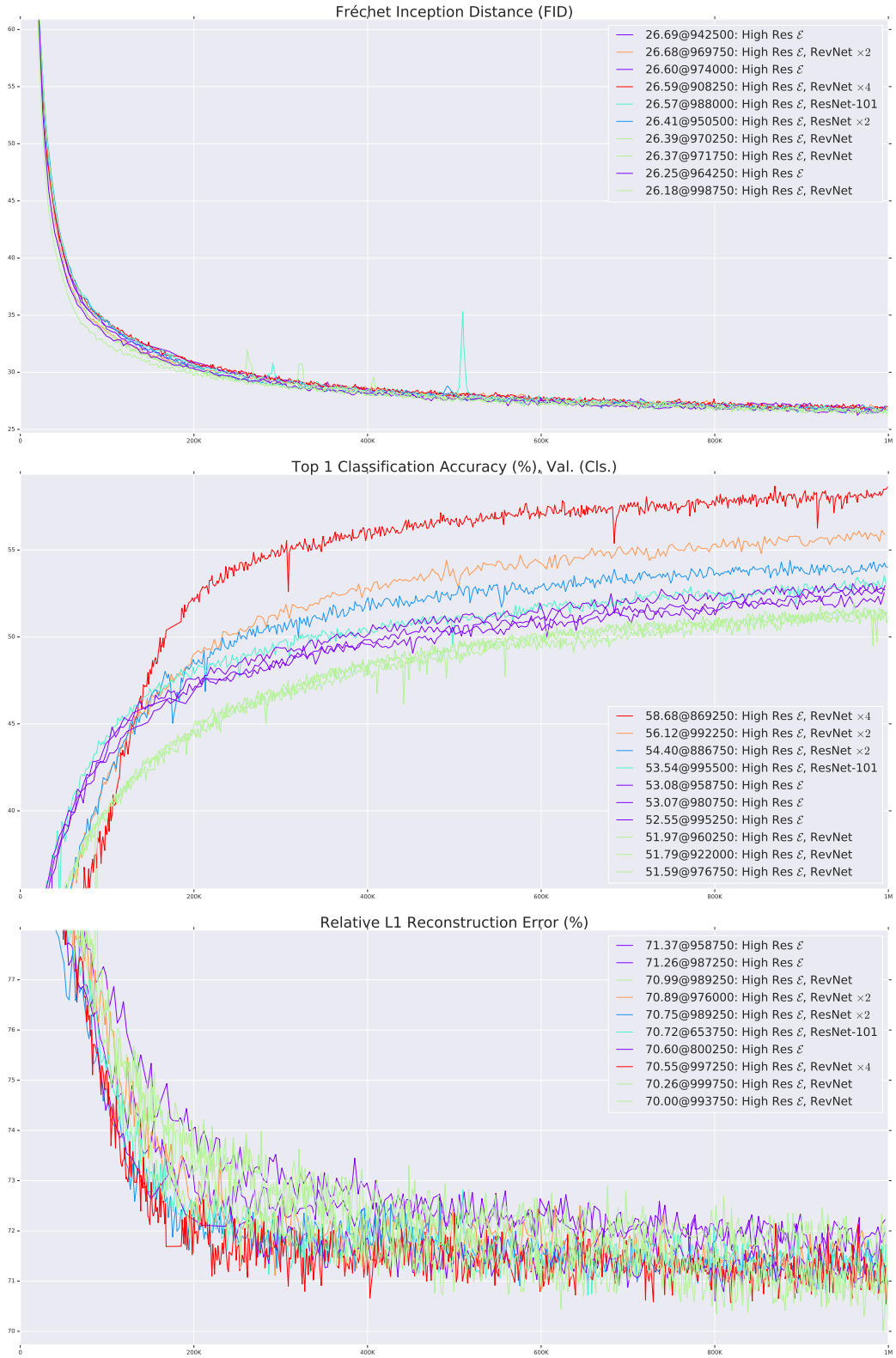


Figure 19: Image generation and representation learning curves for the $\mathcal{E}$ architecture variations explored in Section 3. Legend entries correspond to the following rows in Table 1: *High Res $\mathcal{E}$ (256)*, *ResNet-101*, *ResNet* $\times 2$, *RevNet*, *RevNet* $\times 2$, and *RevNet* $\times 4$.

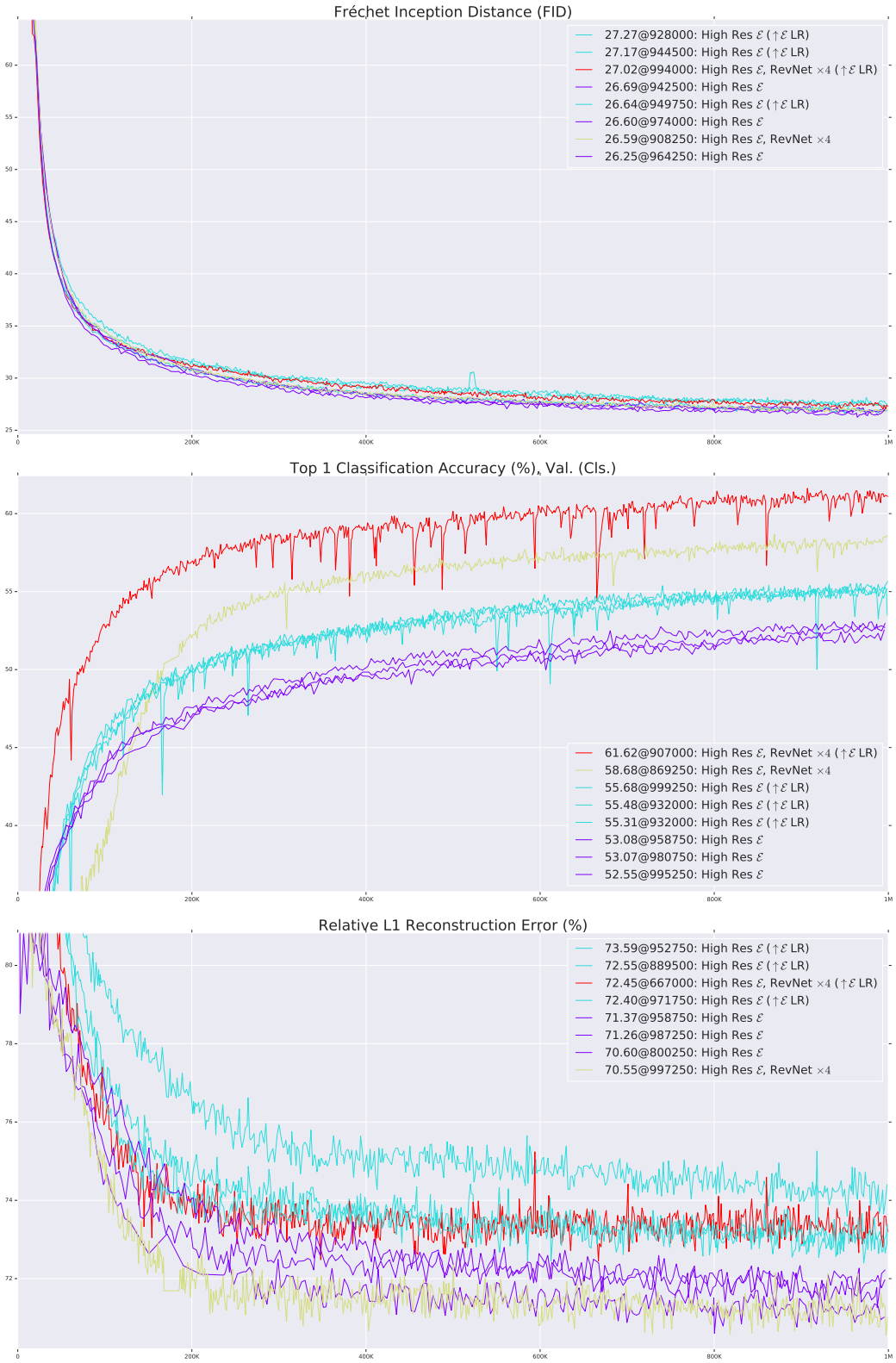


Figure 20: Image generation and representation learning curves showing the effect of decoupling the $\mathcal{E}$ and $\mathcal{G}$ optimizers to train $\mathcal{E}$ with $10\times$ higher learning rate. Legend entries correspond to the following rows in Table 1: *High Res $\mathcal{E}$ (256)*, *ResNet ($\uparrow$ $\mathcal{E}$ LR)*, *RevNet* $\times 4$, and *RevNet* $\times 4$ *($\uparrow$ $\mathcal{E}$ LR)*.