Variance Regularizing Adversarial Learning

Karan Grewal [*] [1] **R Devon Hjelm** [*] [2 3] **Yoshua Bengio** [2 3 4]

Abstract

We introduce a novel approach for training adversarial models by replacing the discriminator score with a bi-modal Gaussian distribution over the real/fake indicator variables. In order to do this, we train the Gaussian classifier to match the target bi-modal distribution implicitly through meta-adversarial training. We hypothesize that this approach ensures a non-zero gradient to the generator, even in the limit of a perfect classifier. We test our method against standard benchmark image datasets as well as show the classifier output distribution is smooth and has overlap between the real and fake modes.

1. Introduction

Generative adversarial networks (GANs, Goodfellow et al., 2014) are a framework for training a generator of some target (i.e., "real") distribution without explicitly defining a parametric generating distribution or a tractable likelihood function. Training the generator relies on a learning signal from a discriminator, which is optimized on a relatively simple objective to distinguish between generated (i.e., "fake") and real samples. In order to match the true distribution, the generator parameters are optimized to maximize the loss as defined by the discriminator, which by analogy makes the generator and discriminator *adversaries*.

GANs have attained strong recognition as being able to generate high-quality images with sharp / realistic edges (Radford et al., 2015) in comparison to maximum-likelihood estimation (MLE, Dempster et al., 1977)-based methods (e.g., explicit graphical model formulations found in Kingma & Welling, 2013; Salakhutdinov & Hinton, 2009, etc). However, GANs have been shown to suffer from instability in training (Arjovsky & Bottou, 2017), so that successful learning is highly reliant on hyperparameter-tuning and model parameterization. Recent advances in training GANs have attempted to address stability and other issues (e.g., Salimans et al., 2017, by adding label noise), notably by imposing Lipschitz constraints on the discriminator via weight clipping (see Wasserstein GANs (WGAN), Arjovsky et al., 2017) or gradient penalty (Gulrajani et al., 2017).

Imposing Lipshitz constraints can improve stability of GAN training, avoiding the situation where the discriminator is over-optimized to the point that nearly indistinguishable samples can have very different scores. The motivation is to use a "weaker" discrimination metric than the common Kullback–Leibler (KL) or Jensen-Shannon (JSD) divergences, which are arguably poor metrics when the true dataset has support on low-dimensional manifolds (Arjovsky & Bottou, 2017). The Lipschitz constraint then ensures that the compressed representation of the data and generated distributions as defined by the discriminator output are smooth, which should ensure a non-zero learning signal for the generator. This contrasts with traditional GANs and recently introduced least-squares GAN (LSGAN, Mao et al., 2016), where the discriminator is allowed to be arbitrarily powerful and can compress the input to a nearly discretized distribution over the output space.

However, it is unclear how enforcing smoothness in the discriminator affects the overall quality of the generator through optimization, as a weaker discriminator could, in principle, give poor samples good scores (or visa versa). In addition, measuring the Lipschitz constant cannot be done exactly, so imposing smoothness can only be done approximately through auxiliary optimization techniques (such as with weight clipping or gradient penalties).

Rather than penalizing the discriminator for being non-Lipschitz, which is difficult, we take a slightly different approach to learning a smooth discriminator function by training a Gaussian classifier over the real/fake indicator variables. Depending on the overlap between the mixture components, this optimization can produce a "weaker" metric for the generator, which is trained to concentrate on the mode corresponding to the "real" indicator

[*]Equal contribution [1]University of Toronto [2]Montreal Institute for Learning Algorithms [3]University of Montreal [4]CIFAR Fellow. Correspondence to: Karan Grewal <karanraj.grewal@mail.utoronto.ca>, R Devon Hjelm <erroneus@gmail.com>.

variable. In order to train the classifier, we introduce two smaller "meta"-discriminators, each corresponding to the generated and real-data modes (fake/real indicator variables), and each of which use samples drawn from a uni-variate and unit-variance Gaussian as "real" samples. The meta-discriminators are trained using a normal GAN loss, while the classifier is trained to "fool" both meta-discriminators (act like an adversarial generator to both meta-discriminators, simultaneously). We show that this approach ensures overlap between the classifier output modes with a smooth distribution and a non-zero gradient for the generator. We also show that this method trains successfully on a variety of standard image datasets.

2. Methods

Let us consider training a generating function, $G(\mathbf{z})$, where $\mathbf{z}$ is some set of latent variables sampled from a multivariate noisy a-priori (e.g., a multivariate Gaussian) distribution, $p(\mathbf{z})$. Next, define a discriminator, $y(\mathbf{x})$, which takes as input samples from the generator $\mathbf{x} \sim p_g(\mathbf{x})$ or samples from the real empirical distribution, $\mathbf{x} \sim p_d(\mathbf{x})$, where $y(\mathbf{x})$ is a deep neural network with no output nonlinearity.

In the context of generative adversarial learning (a.k.a., GANs), it is typical to define a score function, $D(y(\mathbf{x}))$, and a value function,

$$V(G, y) = \mathbb{E}_{x \sim p_d(\mathbf{x})}[D(y(\mathbf{x}))] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[D^\dagger(y(G(\mathbf{z})))], \tag{1}$$

where $D^\dagger(.)$ is a convex conjugate (see Nowozin et al., 2016). When the discriminator is optimal w.r.t. the value function for a given generator, in other words when $y(\mathbf{x}) = y^\star = \arg\max_y V(G, y)$, it is commonly the case that the value function becomes flat everywhere where the generated and real distributions have support, so that:

$$\mathbb{E}_{x \sim p_d(\mathbf{x})}\left[\frac{\partial D}{\partial y^\star}\frac{\partial y^\star}{\partial \mathbf{x}}(\mathbf{x})\right] = \mathbb{E}_{x \sim p_g(\mathbf{x})}\left[\frac{\partial D^\dagger}{\partial y^\star}\frac{\partial y^\star}{\partial \mathbf{x}}(\mathbf{x})\right] = 0. \tag{2}$$

This pathology occurs despite well-defined score functions, D and $D^\dagger$, as derivatives of the discriminator output, $\partial y/\partial \mathbf{x}$, only needs to be non-zero in a way to ensure that D and $D^\dagger$ are maximized in the regions of real data and generated samples, respectively (for instance, non-zero at the decision boundary of a binary-classifier discriminator).

This partially motivates Wasserstein GANs (WGAN, Arjovsky et al., 2017), which seek to force the derivatives $\partial y/\partial \mathbf{x}$ to be non-zero in the regimes corresponding to data or samples, by imposing a Lipschitz constraint which places an upper bound on $\partial y/\partial \mathbf{x}$. Imposing this upper bound ensures that the optimal discriminator (in this context called a "critic") cannot be too sharp, hence ensuring smoother global discriminator output.

A different view of the above problem is that the output of the optimal discriminator, $y^\star$, has zero *variance* in the regimes of interest. In order to address this, we posit linear classifier whose optimal distribution, $p^\star_D(y)$, given $y^\star$, is a mixture of Gaussian distributions over the real/fake indicator variables, each with set mean and variance. While this does not explicitly forbid sharpness in the classifier output, sharp boundaries are no longer an explicit optimum, as the modes share support.

Let $N_f(y) = \mathcal{N}(\mu_f, 1)$ and $N_r(y) = \mathcal{N}(\mu_r, 1)$ be univariate Gaussian distributions with standard deviation 1 and means μ_f and μ_r respectively, and define the classifier as the mixture:

$$p^\star_C(y) = \frac{1}{2}(N_f(y) + N_r(y)), \tag{3}$$

where we have assumed the priors of the "real" and "fake" labels are both $\frac{1}{2}$. In order to ensure that the classifier follows this distribution, we define two "meta" discriminators, $F(y)$ and $R(y)$, which are trained to maximize the value functions:

$$\begin{aligned} V_f(F, y) =& \mathbb{E}_{y \sim N_f(y)}[\log F(y)] \\ &+ \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - F(y(G(\mathbf{z}))))] \\ V_r(R, y) =& \mathbb{E}_{y \sim N_r(y)}[\log R(y)] \\ &+ \mathbb{E}_{x \sim p_d(\mathbf{x})}[\log(1 - R(y(\mathbf{x})))]. \end{aligned} \tag{4}$$

The meta-discriminators and the classifier are trained by playing two adversarial games with the above value functions simultaneously. While in principle we can have the classifier minimize both value functions, we chose to minimize the proxy losses instead:

$$\begin{aligned} \mathcal{L}_C =& -\mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log F(y(G(\mathbf{z})))] \\ &- \mathbb{E}_{x \sim p_d(\mathbf{x})}[\log(R(y(\mathbf{x})))]. \end{aligned} \tag{5}$$

Finally, the generator is trained to concentrate at the mode corresponding to the real samples on the classifier output. A simple loss function for the generator is the MSE,

$$\mathcal{L}_G = \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[(y(G(\mathbf{z}))) - \mu_r)^2]. \tag{6}$$

In this sense, the generator loss resembles that of least-squares GAN (Mao et al., 2016), where μ_r is the target of the generator. As this technique ensures variance of the classifier output in regions of interest, we call this method variance-regularized adversarial learning (VRAL).

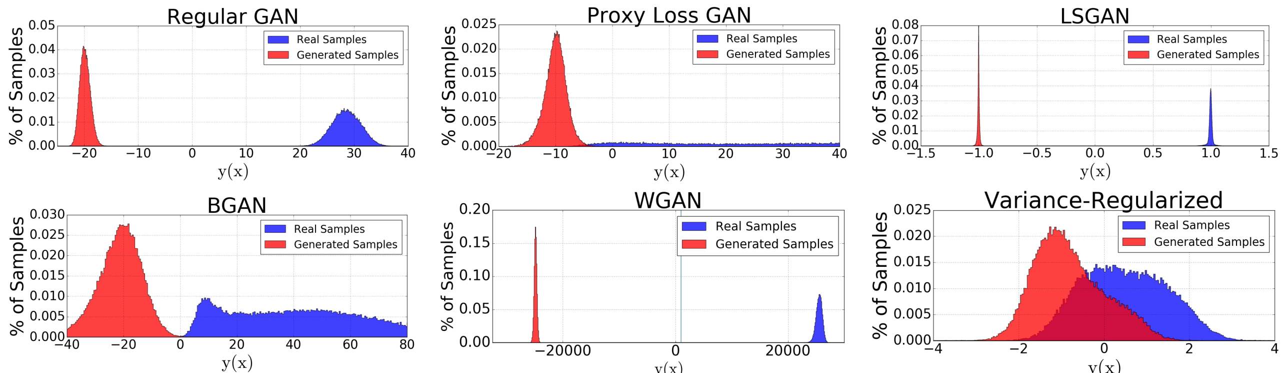


Figure 1. Histograms of the discriminator outputs given real and generated samples using the normal GAN loss, proxy loss, BGAN, LSGAN, WGAN, and VRAL (ours) and updating the discriminator 50 times per generator update on MNIST. Only VRAL shows significant overlap between the two distributions. GAN with the proxy loss and BGAN show highly disburse distributions over real samples and highly peaked distributions over the generated samples. The observed samples for WGAN, LSGAN, BGAN, and GAN with both losses were very poor compared to VRAL (see Appendix).

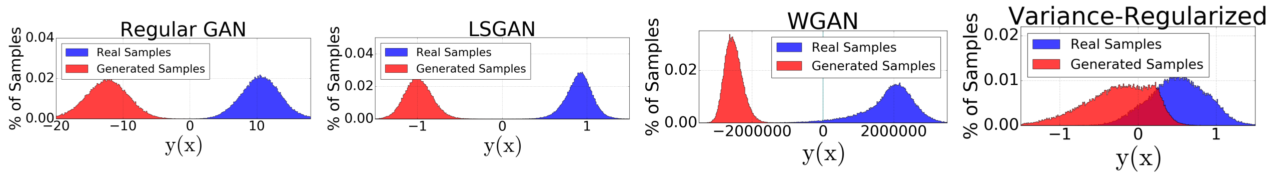


Figure 2. Histograms of the discriminator outputs given real and generated samples for a fixed generator (trained with LSGAN for 5 epochs) and the normal GAN loss, LSGAN, WGAN (with clipping), and VRAL (ours) and were trained for 5 epochs on MNIST. All methods show large flat areas of (zero gradient) between real and generated scores except VRAL. BGAN and GAN with the proxy loss are omitted as they only affect generator learning.

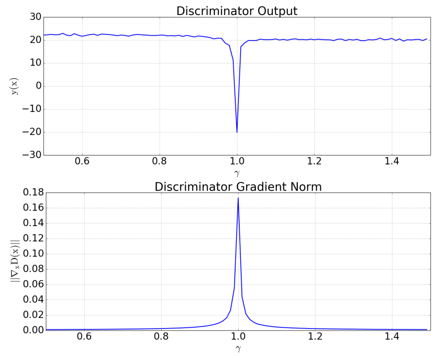
3. Experiments and Results

We now demonstrate the properties of the output distribution of $y(\mathbf{x})$. In order to do this, we first train GANs on the MNIST dataset with a variety of popular methods: normal GANs as defined by Goodfellow et al. (2014), the proxy loss, least-squares GANs (LSGAN, Mao et al., 2016), boundary-seeking GANs (BGAN, Hjelm et al., 2017), Wasserstein GANs with clipping (WGAN, Arjovsky et al., 2017), and our proposed method, variance-regularizing adversarial learning (VRAL). For VRAL, we found setting $\mu_r = 1$ and $\mu_f = 0$ worked well in practice with MNIST. First, we optimize both the generator and discriminator/critic/classifier (which we will call the "discriminator" in all settings) according to the adversarial game defined by the respective value function for each method, updating the discriminator 50 times for every generator update (50:1) for 15 epochs. Next, we train a single generator using the LSGAN objective (with 1:1 updates) for 5 epochs (to ensure it is easy to discriminate), then train a new discriminator for 5 epochs using one of the above methods, keeping the generator fixed. All models were parameterized as DCGANs with batch norm on the generator and discriminator, trained with the same learning rate and optimized using Adam (Kingma & Ba, 2014).

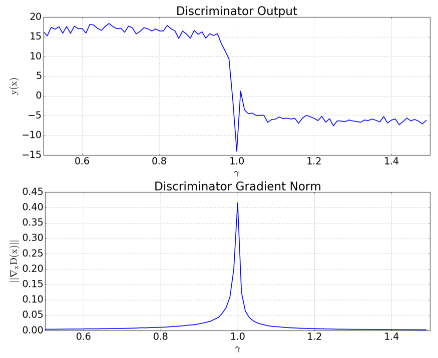
Our results (Figure 1) show that training on the original loss, LSGAN, and WGAN results in peaked output distributions with no overlap, while VRAL shows significant overlap, the latter of which is by design. BGAN and the proxy loss, in contrast, showed some overlap, but highly disburse output distributions given the real samples. Only VRAL showed robustness to these high update ratios (see Appendix for samples). These results are corroborated when we train using the fixed generator (Figure 2).

Next, we trained each of these methods on 1:1 training for 15 epochs, then computed the output, $y(\mathbf{x})$, and the gradient norms, $||\nabla_{\mathbf{x}} y(\mathbf{x})||$, for 64 interpolations between random generated / real sample pairs. Our results (Figure 3) show a smoother decision boundary for VRAL over all other methods.

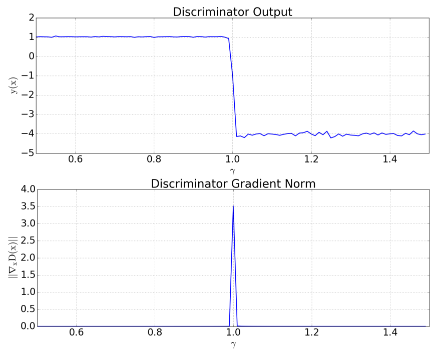
Finally, we train on popular datasets for images: CIFAR-10 (Krizhevsky & Hinton, 2009), CelebA (Chelba et al., 2013), and the large-scale scene understanding challenge (LSUN, Yu et al., 2015) bedroom datasets. These models were trained on standard DCGAN architectures and optimized with Adam as above. We observed that VRAL trained well using the same Gaussian means as above, $\mu_r = 1$ and $\mu_f = 0$, worked well with CIFAR, but $\mu_f = -1$ produced more stable results for CelebA



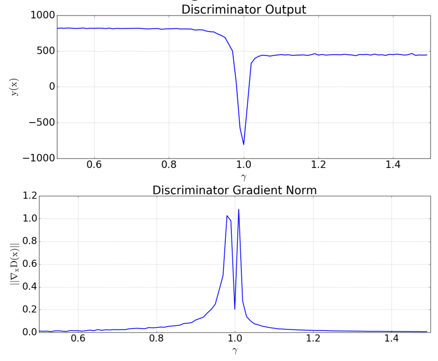
a) Regular GAN



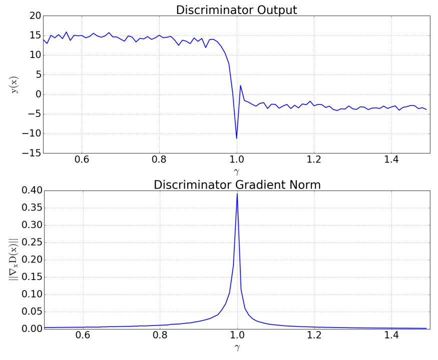
b) Proxy Loss GAN



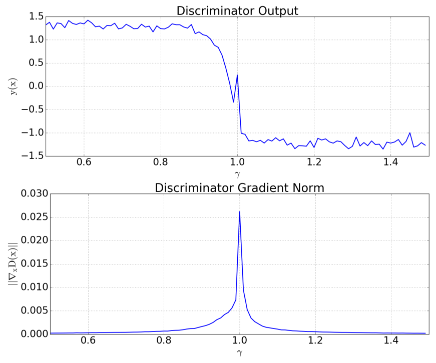
c) LSGAN



d) WGAN

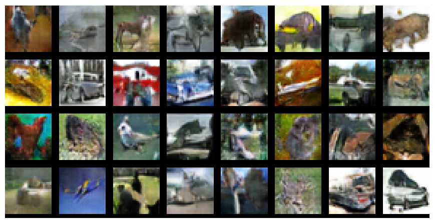


e) BGAN

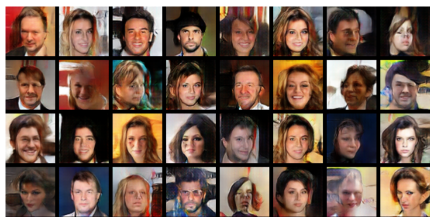


f) VRAL

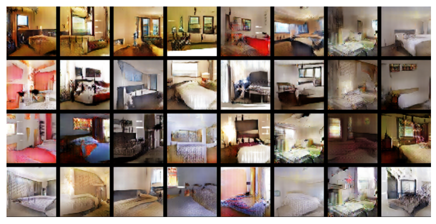
Figure 3. Discriminator output, $y(\mathbf{x})$, and gradient norms, $||\nabla_{\mathbf{x}} y||$, over varying $\hat{\mathbf{x}}(\gamma)$, where $\hat{\mathbf{x}}(\gamma)$ is the interpolation between real and fake samples parameterized as $\hat{\mathbf{x}}(\gamma) = \gamma\mathbf{x} + (1-\gamma)G(\mathbf{z})$, $-0.5 \leq \gamma \leq 1.5$ (averaged over random pairs). $\gamma < 0.5$ was omitted as all models had the same behavior over this range (flat). $\gamma > 1.0$ corresponds to super-unrealistic images, and are only included for demonstration.



a) CIFAR-10



b) CelebA



c) LSUN bedrooms

Figure 4. Samples from CIFAR-10, CelebA, and LSUN trained on VRAL with a DCGAN architecture.

and LSUN (Figure 4).

4. Conclusions

We have rephrased the adversarial game of the generator into that of mode-matching, moving the adversarial learning to train the discriminator to learn a bimodal fixed distribution. While our work shows promise, much additional work is necessary, such as evaluating robustness.

References

Arjovsky, Martin and Bottou, Léon. Towards principled methods for training generative adversarial networks. In *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, volume 2016, 2017.

Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Chelba, Ciprian, Mikolov, Tomas, Schuster, Mike, Ge, Qi, Brants, Thorsten, Koehn, Phillipp, and Robinson, Tony. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

Dempster, Arthur P, Laird, Nan M, and Rubin, Donald B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, and Courville, Aaron. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

Hjelm, R Devon, Jacob, Athul Paul, Che, Tong, Cho, Kyunghyun, and Bengio, Yoshua. Boundary-seeking generative adversarial networks. *arXiv preprint arXiv:1702.08431*, 2017.

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

Kingma, DP and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

Mao, Xudong, Li, Qing, Xie, Haoran, Lau, Raymond YK, Wang, Zhen, and Smolley, Stephen Paul. Least squares generative adversarial networks. *arXiv preprint ArXiv:1611.04076*, 2016.

Nowozin, Sebastian, Cseke, Botond, and Tomioka, Ryota. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pp. 271–279, 2016.

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

Salakhutdinov, Ruslan and Hinton, Geoffrey E. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pp. 448–455, 2009.

Salimans, T, Goodfellow, I, Zaremba, W, Cheung, V, Radford, A, and Chen, X. Improved techniques for training gans. nips, 2016. *Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, W. Zuo, Mind the Class Weight Bias: Weighted Maximum Mean Discrepancy for Unsupervised Domain Adaptation, CVPR*, 2017.

Yu, Fisher, Seff, Ari, Zhang, Yinda, Song, Shuran, Funkhouser, Thomas, and Xiao, Jianxiong. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

A. Robustness to Large Training Ratios

In Figure 5, we show samples from models trained updating the discriminator 50 times for every generator update on the MNIST dataset. Only VRAL (ours) is robust to larger number of discriminator updates in our experiments.

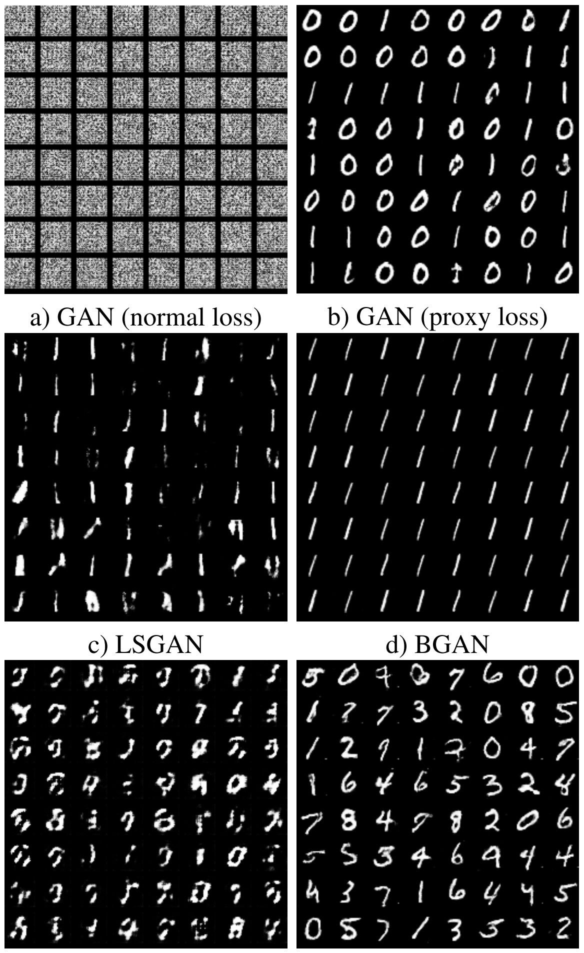


Figure 5. Samples from training on MNIST with a variety of GAN architectures updating the discriminator 50 times for every generator update. Only VRAL (ours) is robust to larger number of discriminator updates in our experiments.