Do Deep Generative Models Know What They Don't Know?*

Eric Nalisnick[†‡], **Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, Balaji Lakshminarayanan**[†]
DeepMind

1 Introduction

We investigate if modern deep generative models can be used for anomaly detection. Imagine that we wish to protect a classifier from inputs unlike the training data. One approach is to train a density model $p(\boldsymbol{x}; \boldsymbol{\theta})$ (with $\boldsymbol{\theta}$ denoting the parameters) to approximate the true distribution of training inputs $p^*(\boldsymbol{x})$ and refuse to make a prediction for any $\boldsymbol{x}$ that has a sufficiently low density under $p(\boldsymbol{x}; \boldsymbol{\theta})$. This idea has been proposed by various papers (Bishop, 1994) and as recently as in the panel discussion at Advances in Approximate Bayesian Inference (AABI) 2017 (Blei et al., 2017). We find a conspicuous point of failure: when trained on CIFAR-10 (Krizhevsky & Hinton, 2009), VAEs, autoregressive models, and flow-based generative models all assign a higher likelihood to SVHN (Netzer et al., 2011) than to the training data. We find this observation to be quite problematic and unintuitive since SVHN's digit images are so visually distinct from the dogs, horses, trucks, boats, etc. found in CIFAR-10.

We go on to study this CIFAR-10 vs SVHN phenomenon in flow-based models in particular since they allow for exact marginal density calculations. To the best of our knowledge, we are the first[4] to report these unintuitive findings for a variety of deep generative models. Moreover, our experiments with flow-based models isolate some crucial experimental variables such as the effect of constant-volume vs non-volume-preserving transformations. Lastly, our analysis provides some simple but general expressions for quantifying the gap in the model density between two data sets. We close the paper by urging more study of the out-of-training-distribution properties of deep generative models, as understanding their behaviour in this setting is crucial for their deployment to the real world.

2 Motivating Observations

Given the impressive advances of deep generative models, we sought to test their ability to quantify when an input comes from a different distribution than that of the training set. For the experiment, we trained the same Glow architecture described in Kingma & Dhariwal (2018)—except small enough that it could fit on one GPU[5]—on CIFAR-10. We then calculated the *log-likelihood* (higher value is better) and *bits-per-dimension* (BPD, lower value is better)[6] of the test split of SVHN, since it is a popular data set and of the same dimensionality ($32 \times 32 \times 3$)—expecting the model to give a lower probability to SVHN because it was not trained on it.

The table of Figure 1a reports the BPD of the training data (CIFAR10-Train), the in-distribution test data (CIFAR10-Test), and the out-of-distribution data (SVHN-Test). Here we see a peculiar result: the SVHN BPD is one bit *lower* than that of both in-distribution data sets. We observed a BPD of 2.39 for SVHN vs 3.39 for CIFAR10-Train vs 3.46 for CIFAR10-Test. Figure 1b shows a histogram of the log-likelihood for the three data sets. Clearly, the SVHN examples (red bars) have higher likelihood across the board, and the result is therefore not caused by a few outliers. We observed this phenomenon when training on CIFAR-10 and testing on SVHN, but not the other way around so this

*This work is a condensed version of Nalisnick et al. (2018).

[†]Corresponding authors: e.nalisnick@eng.cam.ac.uk and balajiln@google.com.

[‡]Work done during an internship at DeepMind.

[4]The concurrent work of Choi & Jang (2018) reports the same phenomenon for invertible generative models.

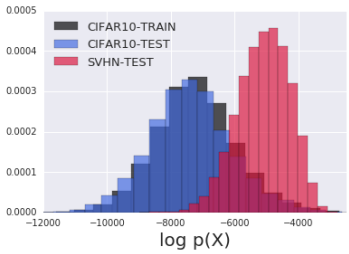
[5]Specifically, we use 3 levels of 8 steps (1×1 convolution followed by an affine coupling layer). Kingma & Dhariwal (2018) use 3 levels of 32 steps. Although we use a smaller model, it still produces good samples (Appendix Figure 9) and competitive bits per dimension for CIFAR-10: 3.46 for ours vs 3.35 for theirs.

[6]See (Theis et al., 2016, Section 3.1) for the definitions of log-likelihood and bits-per-dimension.

Third workshop on Bayesian Deep Learning (NeurIPS 2018), Montréal, Canada.

Data Set	Avg. Bits Per Dimension
Glow Trained on CIFAR10	
CIFAR10-Train	3.386
CIFAR10-Test	3.464
SVHN-Test	**2.389**
Glow Trained on SVHN	
SVHN-Test	2.057

(a) Glow: Bits per dimension

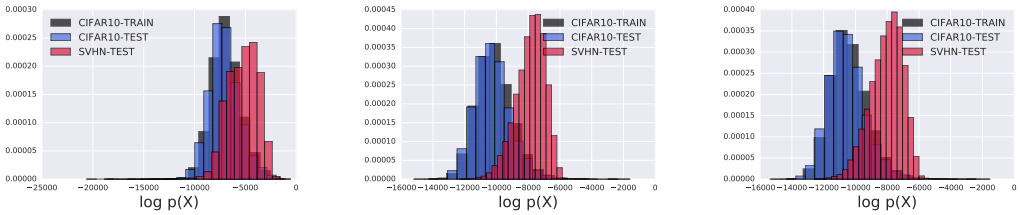


(b) Train on CIFAR10, Test on SVHN

Figure 1: *Testing Out-of-Distribution.* Log-likelihood (expressed in bits per dimension) calculated from Glow (Kingma & Dhariwal, 2018) on SVHN, CIFAR-10.

phenomenon is not symmetric. See Figure 4 in Appendix A for results when training on SVHN and testing on CIFAR-10.

We next tested if the phenomenon occurs for the other common deep generative models: PixelCNN and VAE. We do not include GANs in the comparison since evaluating their likelihood is an open problem. Figure 2 reports the same histograms as above for these models, showing the distribution of $\log p(\boldsymbol{x})$ evaluations for CIFAR-10's train (black) and test (blue) splits and SVHN's test (red) split. Since in all plots the red bars are shifted to the right much as they were before—albeit to varying degrees, with perhaps PixelCNN having the smallest gap—we see that, indeed, this inability to detect inputs unlike the training data persists for these other model classes. SVHN images continue to have higher likelihood than CIFAR-10 training images.



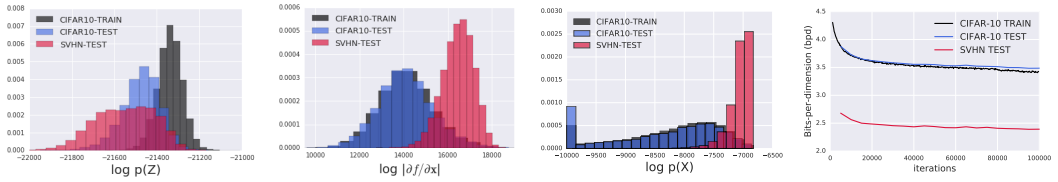
(a) PixelCNN (b) VAE with RNVP as encoder (c) VAE conv-categorical likelihood

Figure 2: *Train on CIFAR10, Test on SVHN*: Log-likelihood calculated from PixelCNN and VAEs. VAE models described in Rosca et al. (2018).

3 Digging Deeper into the Flow-Based Model

While we observed the CIFAR-10 vs SVHN phenomenon for the PixelCNN, VAE, and Glow, we now narrow our investigation to just the class of invertible generative models. The rationale is that they allow for better experimental control as, firstly, they can compute exact marginal likelihoods, unlike the VAE, and secondly, the transforms used in flow-based models have Jacobian constraints that simplify the second order analysis we present later.

To further examine this curious phenomenon, we inspect the change-of-variables objective itself, investigating if one or both terms give the out-of-distribution data a higher value. We report the constituent $\log p(\boldsymbol{z})$ and $\log |\partial \boldsymbol{f}_{\boldsymbol{\phi}}/\partial \boldsymbol{x}|$ terms for NVP-Glow in Figure 3, showing histograms for $\log p(\boldsymbol{z})$ in subfigure (a) and for $\log |\partial \boldsymbol{f}_{\boldsymbol{\phi}}/\partial \boldsymbol{x}|$ in subfigure (b). We see that $p(\boldsymbol{z})$ behaves mostly



(a) CIFAR10: $\log p(\boldsymbol{z})$ (b) CIFAR10: Volume (c) CV-GLOW (d) Likelihood vs. Time

Figure 3: *Decomposing the Likelihood of NVP-Glow.* The histograms show Glow's log-likelihood decomposed into contributions from the $\boldsymbol{z}$-distribution and volume element.

as expected. The red bars are clearly shifted to the left, representing lower likelihoods under the latent distribution. The volume element seems to cause SVHN's higher likelihood. Subfigure (b) shows that all of the SVHN log-volume evaluations (red) are conspicuously shifted to the right—to higher values—when compared to CIFAR-10's (blue and black). Since SVHN's $p(\boldsymbol{z})$ evaluations are only slightly less than CIFAR-10's, the volume term dominates, resulting in SVHN having a higher likelihood.

To experimentally control for the effect of the volume term, we trained Glow with constant-volume (CV) transformations. We modify the affine layers to use only translation operations (Dinh et al., 2015) but keep the 1×1 convolutions as is. The log-determinant-Jacobian is then $HW \sum_k \log |\boldsymbol{U}_k|$, where $|\boldsymbol{U}_k|$ is the determinant of the convolutional weights $\boldsymbol{U}_k$ for the kth flow. This makes the volume element constant across all inputs $\boldsymbol{x}$, allowing us to isolate its effect. Figure 3c shows the results for this model, which we term *CV-Glow* (constant-volume Glow). Subfigure (c) shows a histogram of the $\log p(\boldsymbol{x})$ evaluations, just as shown before in Figure 1b, and we see that SVHN (red) still achieves a higher likelihood (lower BPD) than the CIFAR-10 training set. Subfigure (d) shows the SVHN vs CIFAR-10 BPD over the course of training CV-Glow. Notice that there is no cross-over point in the curves. We also tested if using an ensemble of five Glows would mitigate the effect. It did not: see Appendix E for the results.

Second Order Analysis Lastly, we aim to provide a more direct analysis of when another distribution might have higher likelihood than the one used for training. Consider two distributions: the training distribution $\boldsymbol{x} \sim p^*$ and some dissimilar distribution $\boldsymbol{x} \sim q$ also with support on $\mathcal{X}$. For a given generative model $p(\boldsymbol{x}; \boldsymbol{\theta})$, the adversarial distribution q will have a higher likelihood than the training data's if $\mathbb{E}_q[\log p(\boldsymbol{x}; \boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x}; \boldsymbol{\theta})] > 0$. This expression is hard to analyze directly so we perform a second-order expansion of the log-likelihood around an interior point $\boldsymbol{x}_0$. Applying a second-order expansion to both likelihoods, taking expectations, and canceling the common terms, we have:

$$\mathbb{E}_q[\log p(\boldsymbol{x}; \boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x}; \boldsymbol{\theta})] \approx \nabla_{\boldsymbol{x}_0} \log p(x_0; \boldsymbol{\theta})^T (\mathbb{E}_q[\boldsymbol{x}] - \mathbb{E}_{p^*}[\boldsymbol{x}]) + \frac{1}{2} \operatorname{Tr}\{\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0; \boldsymbol{\theta})(\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\} \quad (1)$$

where $\boldsymbol{\Sigma} = \mathbb{E}\left[(\boldsymbol{x} - \boldsymbol{x}_0)(\boldsymbol{x} - \boldsymbol{x}_0)^T\right]$, the covariance matrix, and $\operatorname{Tr}\{\cdot\}$ is the trace operation. Since the expansion is accurate only locally around $\boldsymbol{x}_0$, we next assume that $\mathbb{E}_q[\boldsymbol{x}] = \mathbb{E}_{p^*}[\boldsymbol{x}] = \boldsymbol{x}_0$. While this at first glance may seem like a strong assumption, it is not too removed from practice since data is usually centered before being fed to the model. For SVHN and CIFAR-10 in particular, we find this assumption to hold (Appendix Figure 8).

We use the expression in Equation 1 to analyze the behavior of CV-Glow on CIFAR-10 vs SVHN, seeing if the difference in likelihoods can be explained by the model curvature and data's second moment. The second derivative terms simplify considerably for CV-Glow with a spherical latent density. Given a $C \times C$ kernel $\boldsymbol{U}_k$, with k indexing the flow and C the number of input channels, the derivatives are $\partial f_{h,w,c} / \partial x_{h,w,c} = \prod_k \sum_{j=1}^{C} u_{k,c,j}$, with h and w indexing the spatial height and width and j the columns of the kth flow's 1×1 convolutional kernel. The second derivative is then $\partial^2 f_{h,w,c} / \partial x^2_{h,w,c} = 0$. Plugging in the second derivative of the Gaussian's log density—a common choice for the latent distribution in flow models, following (Dinh et al., 2015, 2017; Kingma & Dhariwal, 2018)—and the empirical variances, we have:

$$\mathbb{E}_{\text{SVHN}}[\log p(\boldsymbol{x}; \boldsymbol{\theta})] - \mathbb{E}_{\text{CIFAR10}}[\log p(\boldsymbol{x}; \boldsymbol{\theta})] \approx \frac{1}{2\sigma^2_{\boldsymbol{\psi}}} \left[\alpha_1^2 \cdot 12.3 + \alpha_2^2 \cdot 6.5 + \alpha_3^2 \cdot 14.5\right] \geq 0 \quad (2)$$

where $\alpha_c = \prod_{k=1}^{K} \sum_{j=1}^{C} u_{k,c,j}$ and $\sigma^2_{\boldsymbol{\psi}}$ is the variance of the latent distribution. The full derivation is in Appendix F. We know the final expression is greater than or equal to zero since all $\alpha_c^2 \geq 0$. Thus, the second-order expression we derived does indeed predict we should see a higher likelihood for SVHN than for CIFAR-10. Moreover, we leave the CV-Glow's parameters as constants to emphasize the expression is non-negative *for any* parameter setting of the CV-Glow model. This supports our observation that the gap remained relatively constant over the course of training (Figure 3c d). Furthermore, the $\partial^2 \log p(\boldsymbol{z}; \boldsymbol{\psi}) / \partial z^2$ term would be negative for any log-concave density function, meaning that changing the latent density to Laplace or logistic would not change the result.

Discussion Our conclusion is that SVHN simply "sits inside of" CIFAR-10—roughly same mean, smaller variance—resulting in its higher likelihood. Thus, we urge caution when using deep generative models with out-of-training-distribution inputs as we have shown that comparing likelihoods alone

cannot identify the training set or inputs like it. We cannot conclude that there is any pathology innate to deep generative models. It could be a problem that plagues any generative model, no matter how high its capacity. In turn, we must then temper the enthusiasm with which we preach the benefits of generative models until their sensitivity to out-of-distribution inputs is better understood.

References

Christopher M Bishop. Novelty Detection and Neural Network Validation. *IEE Proceedings-Vision, Image and Signal processing*, 141(4):217–222, 1994.

Christopher M Bishop. Training with Noise is Equivalent to Tikhonov Regularization. *Neural Computation*, 7(1):108–116, 1995.

David Blei, Katherine Heller, Tim Salimans, Max Welling, and Zoubin Ghahramani. Panel Discussion. *Advances in Approximate Bayesian Inference*, December 2017. URL https://youtu.be/x1UByHT60mQ?t=46m2s. NIPS Workshop.

Hyunsun Choi and Eric Jang. Generative Ensembles for Robust Anomaly Detection. *ArXiv e-Print arXiv:1810.01392*, 2018.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-Linear Independent Components Estimation. *ICLR Workshop Track*, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density Estimation Using Real NVP. In *International Conference on Learning Representations (ICLR)*, 2017.

Federico Girosi, Michael Jones, and Tomaso Poggio. Regularization Theory and Neural Networks Architectures. *Neural Computation*, 7(2):219–269, 1995.

Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do Deep Generative Models Know What They Don't Know? *ArXiv e-Print arXiv:1810.09136*, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive Auto-Encoders: Explicit Invariance During Feature Extraction. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. Distribution matching in variational inference. *arXiv preprint arXiv:1802.06847*, 2018.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A Note on the Evaluation of Generative Models. In *International Conference on Learning Representations (ICLR)*, 2016.

A Results illustrating asymmetric behavior

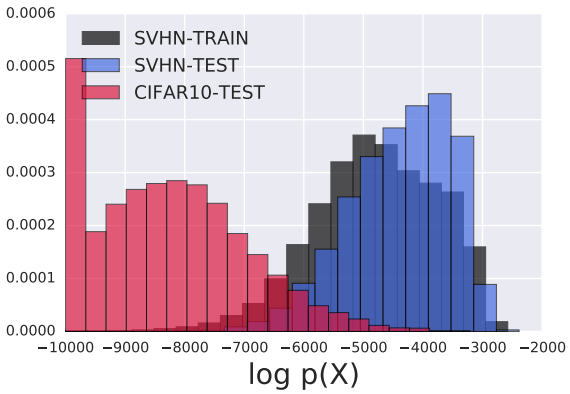


Figure 4: Histogram of Glow log-likelihoods when training on SVHN and testing on CIFAR10. Note that the model trained on SVHN is able to assign lower likelihood to CIFAR10, which illustrates the asymmetry compared to Figure 1b.

B Analyzing the Change-of-Variables Formula as an Optimization Function

Consider the intuition underlying the volume term in the change of variables objective. As we are maximizing the Jacobian's determinant, it means that the model is being encouraged to maximize the $\partial \boldsymbol{f}_j / \partial x_j$ partial derivatives. In other words, the model is rewarded for making the transformation sensitive to small changes in $\boldsymbol{x}$. This behavior starkly contradicts a long history of derivative-based regularization penalties. Dating back at least to (Girosi et al., 1995), *penalizing* the Frobenius norm of a neural network's Jacobian—which upper bounds the volume term[7]—has been shown to improve generalization. This agrees with intuition since we would like the model to be insensitive to small changes in the input, which are likely noise. Moreover, Bishop (1995) showed that training a network under additive Gaussian noise is equivalent to Jacobian regularization, and Rifai et al. (2011) proposed *contractive autoencoders*, which penalize the Jacobian-norm of the encoder. Allowing invertible generative models to maximize the Jacobian term without constraint suggests, at minimum, that these models will not learn robust representations.

Limiting Behavior. We next attempt to quantify the limiting behavior of the log volume element. Let us assume, for the purposes of a general treatment, that the bijection $f_{\boldsymbol{\phi}}$ is an L-Lipschitz function. Both terms in the change-of-variables objective can be bounded as follows:

$$\log p(\boldsymbol{x}; \boldsymbol{\theta}) = \underbrace{\log p_z(f(\boldsymbol{x}; \boldsymbol{\phi}))}_{\mathcal{O}(\max_{\boldsymbol{z}} \log p_z(\boldsymbol{z}))} + \underbrace{\log \left| \frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}} \right|}_{\mathcal{O}(D \log L)} \leq \max_{\boldsymbol{z}} \log p_z(\boldsymbol{z}) + D \log L \tag{3}$$

where L is the Lipschitz constant, D the dimensionality, and $\mathcal{O}(\max_{\boldsymbol{z}} \log p_z(\boldsymbol{z}))$ an expression for the (log) mode of $p(\boldsymbol{z})$. We will make this mode term for concrete for Gaussian distributions below. The bound on the volume term follows from Hadamard's inequality:

$$\log \left| \frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}} \right| \leq \log \prod_{j=1}^{D} \left| \frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}} \boldsymbol{e}_j \right| \leq \log (L \, |\boldsymbol{e}_\cdot|)^D = D \log L$$

where $\boldsymbol{e}_j$ is an eigenvector. While this expression is too general to admit any strong conclusions, we can see from it that the 'peakedness' of the distribution represented by the mode must keep pace with the Lipschitz constant, especially as dimensionality increases, in order for both terms to contribute equally to the objective.

We can further illuminate the connection between L and the concentration of the latent distribution through the following proposition:

[7] It is easy to show the upper bound via Hadamard's inequality: $\det \partial \boldsymbol{f} / \partial \boldsymbol{x} \leq ||\partial \boldsymbol{f} / \partial \boldsymbol{x}||_F$.

Proposition 1. *Assume $x \sim p^*$ is distributed with moments $\mathbb{E}[x] = \mu_x$ and $Var[x] = \sigma_x^2$. Moreover, let $f : \mathcal{X} \mapsto \mathcal{Z}$ be L-Lipschitz and $f(\mu_x) = \mu_z$. We then have the following concentration inequality for some constant δ:*

$$P\left(|f(x) - \mu_z| \geq \delta\right) \leq \frac{L^2\sigma_x^2}{\delta^2}.$$

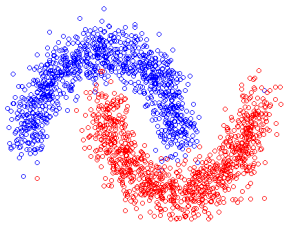
Proof: From the fact that f is L-Lipschitz, we know $|f(x) - \mu_z| \leq L\left|x - f^{-1}(\mu_z)\right|$. Assuming $\mu_x = f^{-1}(\mu_z))$, we can apply Chebyshev's inequality to the RHS: $Pr(L\left|x - f^{-1}(\mu_z)\right| \geq \delta) \leq L^2\sigma_x^2/\delta^2$. Since $L\left|x - f^{-1}(\mu_z)\right| \geq |f(x) - \mu_z|$, we can plug the RHS into the inequality and the bound will continue to hold.

From the inequality we can see that the latent distribution can be made more concentrated by decreasing L and/or the data's variance σ_x^2. Since the latter is fixed, optimization only influences L. Yet, recall that the volume term in the change-of-variables objective *rewards* increasing f's derivatives and thus L. While we have given an upper bound and therefore cannot say that increasing L will necessarily decrease concentration in latent space, it is for certain that leaving L unconstrained does not directly pressure the $f(\boldsymbol{x})$ evaluations to concentrate.

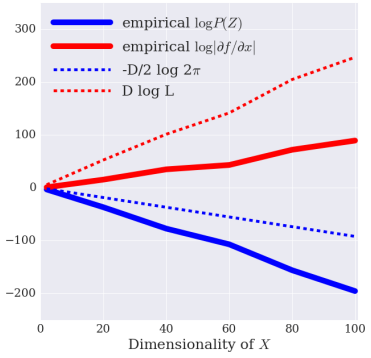
Previous work (Dinh et al., 2015, 2017; Kingma & Dhariwal, 2018) has almost exclusively used a factorized zero-mean Gaussian as the latent distribution, and therefore we examine this case in particular. The log-mode can be expressed as $-D/2 \cdot \log 2\pi\sigma_z^2$, making the likelihood bound

$$\log \mathrm{N}(f(\boldsymbol{x};\boldsymbol{\phi}); \mathbf{0}, \sigma_z^2\mathbb{I}) + \log\left|\frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}}\right| \leq \frac{-D}{2}\log 2\pi\sigma_z^2 + D\log L. \tag{4}$$

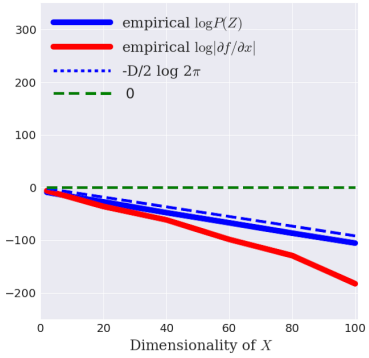
We see that both terms scale with D although in different directions, with the contribution of the z-distribution becoming more negative and the volume term's becoming more positive. We performed a simulation to demonstrate this behavior on the two moons data set, which is shown in Figure 5 (a). We replicated the original two dimensions to create data sets of dimensionality of up to 100. The results are shown in Figure 5 (b). The empirical values of the two terms are shown by the solid lines, and indeed, we see they exhibit the expected diverging behavior as dimensionality increases.



(a) Two Moons Data Set (2D)



(b) Exponential Parametrization



(c) Sigmoid Parametrization

Figure 5: *Limiting Bounds.* We trained an RNVP transformation on two moons data sets—which is shown in (a) for 2 dimensions—of increasing dimensionality, tracking the empirical value of each term against the upper bounds. Subfigure (b) shows Glow with an exp parametrization for the scales and (c) shows Glow with a sigmoid parametrization.

C Glow with Sigmoid Parametrization

Upon reading the open source implementation of Glow,[8] we found that Kingma & Dhariwal (2018) in practice parametrize the scaling factor as $\texttt{sigmoid}(s(\boldsymbol{x}_{d:};\boldsymbol{\phi}_s))$ instead of $\exp\{s(\boldsymbol{x}_{d:};\boldsymbol{\phi}_s)\}$. This choice allows the volume only to decrease and thus results in the volume term being bounded as

[8] https://github.com/openai/glow/blob/master/model.py#L376

(ignoring the convolutional transforms)

$$\log \left| \frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}} \right| = \sum_{f=1}^{F} \sum_{j=1}^{d_f} \log \texttt{sigmoid}(s_{f,j}(\boldsymbol{x}_{d_f:}; \boldsymbol{\phi}_s)) \leq FD \log 1 = 0 \tag{5}$$

where f indexes the flows and d_f the dimensionality at flow f. Interestingly, this parametrization has a fixed upper bound of zero, removing the dependence on D found in Equation 4. We demonstrate the change in behavior introduced by the alternate parametrization via the same two moon simulation. The only difference is that the RNVP transforms use a sigmoid parametrizations for the scaling operation. See Figure 5 (c) for the results: we see that now both change-of-variable terms are oriented downward as dimensionality grows. We conjecture this parametrization helps condition the log-likelihood, limiting the volume term's influence, when training the large models (~ 90 flows) used by Kingma & Dhariwal (2018). However, it does not fix the out-of-distribution over-confidence we report in Section 2.

D Constant and Random Inputs

Data Set	Avg. Bits Per Dimension
Glow Trained on CIFAR10	
Random	15.773
Constant (128)	**0.589**

Figure 6: *Random and constant images.* Log-likelihood (expressed in bits per dimension) of random and constant inputs calculated from NVP-Glow for models trained on CIFAR-10.

E Ensembling Glows

The likelihood function technically measures how likely the parameters are under the data (and not how likely the data is under the model), and perhaps a better quantity would be the posterior predictive distribution $p(\boldsymbol{x}_{test}|\boldsymbol{x}_{train}) = \frac{1}{M}\sum_m p(\boldsymbol{x}_{test}|\boldsymbol{\theta}_m)$ where we draw samples from posterior distribution $\boldsymbol{\theta}_m \sim p(\boldsymbol{\theta}|\boldsymbol{x}_{train})$. Intutitively, it seems that such an integration would be more robust than a single maximum likelihood point estimate. As a crude approximation to Bayesian inference, we tried averaging over ensembles of generative models since Lakshminarayanan et al. (2017) showed that ensembles of discriminative models are robust to out-of-distribution inputs. We compute an "ensemble predictive distribution" as $p(\boldsymbol{x}) = \frac{1}{M}\sum_m p(\boldsymbol{x}; \boldsymbol{\theta}_m)$, where m indexes over models. However, as Figure 7 shows, ensembles did not significantly change the relative difference between in-distribution (CIFAR-10, black and blue) and out-of-distribution (SVHN, red).

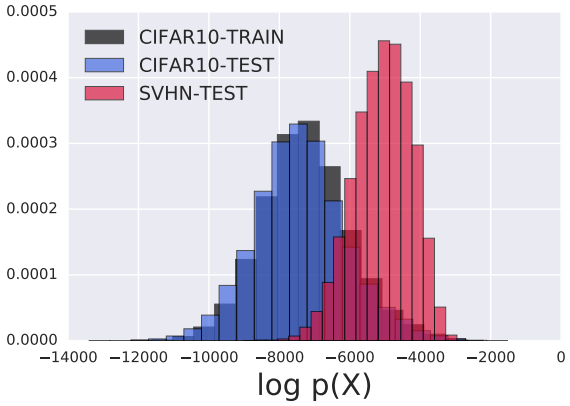


Figure 7: *Ensemble of Glows.* The plot above shows a histogram of log-likelihoods computed using an ensemble of Glow models trained on CIFAR10, tested on SVHN. Ensembles were not found to be robust against this phenomenon.

F Derivation of CV-Glow's Likelihood Difference

We start with Equation 1 and assume equal means:

$$\mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})] \approx \frac{1}{2}\operatorname{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p_z(f(\boldsymbol{x}_0;\boldsymbol{\phi})) + \nabla^2_{\boldsymbol{x}_0}\log\left|\frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}_0}\right|\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}.$$

The volume element for CV-Glow does not depend on $\boldsymbol{x}_0$ and therefore drops from the equation:

$$\nabla^2_{\boldsymbol{x}_0}\log\left|\frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}_0}\right| = \nabla^2_{\boldsymbol{x}_0} HW\sum_k \log|\boldsymbol{U}_k| = 0 \tag{6}$$

where $\boldsymbol{U}_k$ denotes the kth 1×1-convolution's kernel. Moving on to the first term, the log probability under the latent distribution, we have:

$$\begin{aligned}\nabla^2_{\boldsymbol{x}_0}\log p(f(\boldsymbol{x}_0);\boldsymbol{\psi}) &= \nabla^2_{\boldsymbol{x}_0}\left\{\frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}||f(\boldsymbol{x}_0)||_2^2 - \frac{D}{2}\log 2\pi\sigma^2_{\boldsymbol{\psi}}\right\}\\ &= \nabla_{\boldsymbol{x}_0}\left\{\frac{-1}{\sigma^2_{\boldsymbol{\psi}}}\left(\sum_d f_d(\boldsymbol{x}_0)\right)\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)\right\}\\ &= \frac{-1}{\sigma^2_{\boldsymbol{\psi}}}\left[\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)(\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0))^T + \left(\sum_d f_d(\boldsymbol{x}_0)\right)\nabla^2_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)\right].\end{aligned} \tag{7}$$

Since f is comprised of translation operations and 1×1 convolutions, its partial derivatives involve just the latter (as the former are all ones), and therefore we have the partial derivatives:

$$\frac{\partial f_{h,w,c}(\boldsymbol{x}_0)}{\partial x_{h,w,c}} = \prod_{k=1}^{K}\sum_{j=1}^{C_k} u_{k,c,j}, \quad \frac{\partial^2 f_{h,w,c}(\boldsymbol{x}_0)}{\partial x^2_{h,w,c}} = 0 \tag{8}$$

where h and w index the input spatial dimensions, c the input channel dimensions, k the series of flows, and j the column dimensions of the $C_k \times C_k$-sized convolutional kernel $\boldsymbol{U}_k$. The diagonal elements of $\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)(\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0))^T$ are then $(\prod_{k=1}^{K}\sum_{j=1}^{C_k} u_{k,c,j})^2$, and the diagonal element of $\nabla^2_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)$ are all zero.

Then returning to the full equation, for the constant-volume Glow model we have:

$$\begin{aligned}&\frac{1}{2}\operatorname{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p(f(\boldsymbol{x}_0)) + \nabla^2_{\boldsymbol{x}_0}\log\left|\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}_0}\right|\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}\\ &= \frac{1}{2}\operatorname{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p(f(\boldsymbol{x}_0))\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}\\ &= \frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}\operatorname{Tr}\left\{\left[\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)(\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0))^T + \left(\sum_d f_d(\boldsymbol{x}_0)\right)\nabla^2_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}\\ &= \frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}\sum_{l,m}\left\{\left[\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)(\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0))^T + \left(\sum_d f_d(\boldsymbol{x}_0)\right)\nabla^2_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)\right]\odot(\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}_{l,m}.\end{aligned} \tag{9}$$

Lastly, we assume that both $\boldsymbol{\Sigma}_q$ and $\boldsymbol{\Sigma}_{p^*}$ are diagonal and thus the element-wise multiplication with $\nabla^2_{\boldsymbol{x}_0}\log p(f(\boldsymbol{x}_0))$ collects only its diagonal elements:

$$\begin{aligned}&\frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}\sum_{l,m}\left\{\left[\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)(\nabla_{\boldsymbol{x}_0} f(\boldsymbol{x}_0))^T + \left(\sum_d f_d(\boldsymbol{x}_0)\right)\nabla^2_{\boldsymbol{x}_0} f(\boldsymbol{x}_0)\right]\odot(\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}_{l,m}\\ &= \frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}\sum_h^H\sum_w^W\sum_c^C\left(\prod_{k=1}^{K}\sum_{j=1}^{C_k} u_{k,c,j}\right)^2(\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})\\ &= \frac{-1}{2\sigma^2_{\boldsymbol{\psi}}}\sum_c^C\left(\prod_{k=1}^{K}\sum_{j=1}^{C_k} u_{k,c,j}\right)^2\sum_h^H\sum_w^W(\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})\end{aligned} \tag{10}$$

where we arrived at the last line by rearranging the sum to collect the shared channel terms.

G Histogram of data statistics

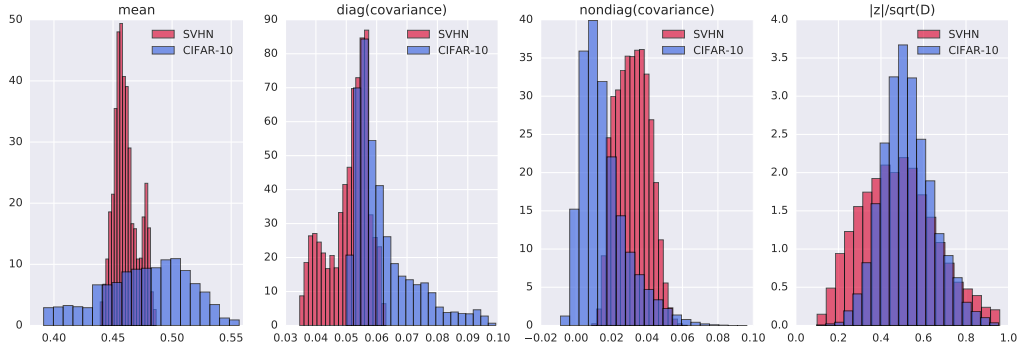


Figure 8: Data statistics: Histogram of per-dimensional mean, per-dimensional variance, non-diagonal elements of covariance matrix, and normalized norm on CIFAR10-SVHN. Note that pixels are converted from 0-255 scale to 0-1 scale by diving by 256.

H Samples from the models

(a) CIFAR-10 samples (b) SVHN samples

Figure 9: *Samples.* Samples from affine-Glow models used for analysis.