Density of States Estimation for Out-of-Distribution Detection: Supplementary Material

A Isotropic Gaussian Densities

Here we work through the simple example given in the main text in detail.

A high-dimensional spherically symmetric Gaussian distribution with mean zero and unit variance in D dimensions has the probability density:

$$p(X=x)\,dx = \frac{1}{(2\pi)^{D/2}} \exp\left(-\frac{x^{\mathsf{T}} x}{2}\right) dx. \tag{5}$$

Transformed for spherical coordinates, this becomes a distribution over the norm of the vectors:

$$p(R=r)\,dr = \frac{2r^{D-1}}{2^{\frac{D}{2}}\Gamma\left(\frac{D}{2}\right)} \exp\left(-\frac{r^2}{2}\right) dr \tag{6}$$

The *energy* of the original distribution is:

$$\begin{aligned} u &\overset{\text{def}}{=} -\log p(X=x) = -\frac{x^{\mathsf{T}} x}{2} - \frac{D}{2}\log(2\pi) \\ &= -\frac{r^2}{2} - \frac{D}{2}\log(2\pi) \end{aligned} \tag{7}$$

The density of states in this case is given by:

$$p(u) = \frac{(2\pi)^{\frac{D}{2}}}{\Gamma\left(\frac{D}{2}\right)} e^{-u} \left(u - \frac{D}{2}\log 2\pi\right)^{\frac{D}{2}-1} \tag{8}$$

B Vulnerability of Likelihoods in Flow-based Models

In many previous works on unsupervised OOD detection [e.g., Nalisnick et al., 2019b, Ren et al., 2019, Choi et al., 2018, Bishop, 1994], it has been taken for granted that the likelihood $q(X|\theta_n)$ (which is usually the optimization target for a deep generative model) should be the most informative statistic either by interpreting it directly as a "likelihood," or by using it as a measurement of typicality. We found in our experiments that tests solely utilizing the likelihood of a deep generative model were often vulnerable to OOD data. Nalisnick et al. [2019b] attributed this to a defect in deep generative models themselves. In this section, we aim to show that this is at least partially due to the methodologies for OOD detection rather than pathologies of generative models themselves.

Let us consider a flow-based model, such as Glow [Kingma and Dhariwal, 2018]. In flow-based models, the log-likelihood is computed as $\log q(X|\theta_n) = T_n^{(\text{latent})}(X) + T_n^{(\text{jac})}(X)$, where $T_n^{(\text{jac})}(X) \overset{\text{def}}{=} \log|\mathsf{J}(X)|$ is the Jacobian of the transformation from X to Z, and $T_n^{(\text{latent})}(X) = q(Z|X,\theta_n)$ is the log-probability of the latent variable Z. Consider the example shown in Figure 5. In this example, we show the two-dimensional distribution of metrics for an in-distribution dataset (blue) and an OOD dataset (red). The two dimensions in this case are $q(Z)$ and $\log|\mathsf{J}|$, which are added together to compute the log-likelihood. From this, it is straightforward that curves of constant likelihood have a slope of -1 in this space.

Consider how different decision rules reject data in this space. If we assume that data with low likelihood were OOD, then our decision rule would be approximately equivalent to that shown in the left panel of Figure 5. If we instead use the typicality test (TT) from Nalisnick et al. [2019b], we observe the result shown in the center panel. Effectively, excluding examples with low log-likelihood determines a half-space for which the data is assumed to be in-distribution. Similarly, TT identifies in-distribution data as the intersection of two half-spaces. However, in both cases, OOD data falls within the region classified as in-distribution. As a result, both metrics do extremely poorly on OOD detection. In contrast, DoSE operates over each dimension of the space separately (or all jointly), and is able to find a more optimal decision boundary.

This behavior is not restricted to flow-based models. In VAEs, the log-evidence is approximated as $q(X) = \mathsf{E}_{Z\sim q(Z|X)}\left[q(X|Z)r(Z)/q(Z|X)\right]$, a nonlinear function of a sum of the cross-entropy between the posterior and the prior, the log-likelihood from the decoder, and the entropy of the encoder. Therefore, models that use only the log-evidence of a VAE as a decision rule can exhibit a similar vulnerability to a flow-based model doing the same.

Furthermore, we observe this phenomenon experimentally. Figure 2 shows a decomposition similar to Figure 5 for a model trained on CelebA, using CIFAR10 as OOD data. We observe that, in this space, the OOD data is projected such that it is nearly perfectly confounded for both $q(X|\theta_n)$ as well as TT. DoSE operates on the granularity of the statistics themselves, and therefore achieves a much better AUROC because it partitions the space using all of the constituent statistics, from which the OOD data is noticeably shifted from the in-distribution data.

C Additional details of experiments

To evaluate the performance of DoSE, we first train a generative model on an in-distribution dataset, and fit density of states estimators to statistics from the

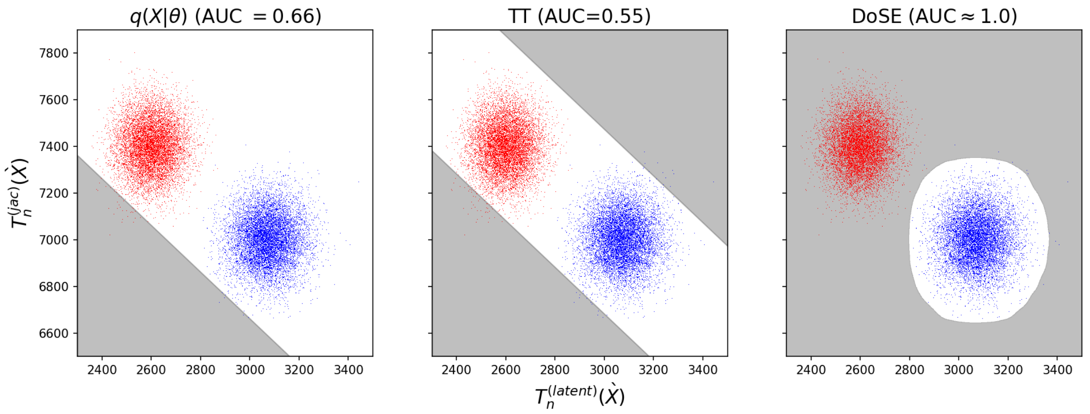

Figure 5: In this toy example, we show the distribution of two statistics, $\log q(Z)$ and $\log |\mathsf{J}|$, returned from a flow-based model on in-distribution data (blue) and OOD data (red). Each panel shows the decision regions produced by different OOD detection techniques operating on these metrics. The left column shows the decision boundaries produced using the log-likelihood. The middle column shows the decision boundaries produced by TT, a typicality test of the log-likelihood. The right column shows the decision boundaries produced by DoSE. In this particular case, the likelihood is the least useful projection over which to attempt to identify OOD data, leading to poor performance of both TT and $q(X|\theta_n)$. DoSE achieves approximately perfect OOD detection in this same setting.

generative model on the training set. Before performing inference, we evaluate the memorization of the model using a random heldout set of 10% of the training examples. When performing inference, we compute the same set of statistics from the generative model on new input data, and calculate the DoSE scores for each example. We measure performance use the AUROC measured using the DoSE scores found from an evaluation set and a specific OOD set. For each trained model, we evaluate the performance against multiple OOD datasets.

Datasets. We use common dataset pairings for the OOD detection task. For our in-distribution datasets, we use MNIST and Fashion MNIST, along with CIFAR10, Street View Housing Numbers (SVHN), and CelebA. These datasets are then paired with the other datasets having the same dimensions, which are taken to be OOD data. Similar to [Choi et al., 2018, DeVries and Taylor, 2018], we also use uniform and Gaussian noise, and horizontally- and vertically-flipped versions of the in-distribution test set as additional OOD datasets. We also use Omniglot for MNIST and Fashion MNIST, and ImageNet-32 and CIFAR100 for CIFAR10, SVHN, and CelebA.

Many of these dataset pairings are "simple," in that likelihood alone would be a reasonable rule to detect OOD data. However, there are several "hard" OOD dataset pairings identified by previous work. FashionMNIST→MNIST and CIFAR10→SVHN were both identified as difficult dataset pairings by Nalisnick et al. [2019a]. Additionally, Nalisnick et al. [2019b] identified CelebA→ CIFAR10/100 and CIFAR10→CIFAR100 to be particularly difficult pairings. The latter is particularly difficult, since both are subsets of the 80 million tiny images dataset [Torralba et al., 2008], but have non-overlapping class labels.

Architectures. Similar to [Choi et al., 2018, Nalisnick et al., 2019b], we train β-VAEs [Higgins et al., 2017] for MNIST and Fashion MNIST, and Glow [Kingma and Dhariwal, 2018] for CIFAR10 and SVHN. For the β-VAE models, our encoder and decoder followed the architecture from Choi et al. [2018]. We use a 2-dimensional latent space, and a trainable mixture of 200 Gaussians for the marginal distribution $r(Z)$. We also considered higher dimensional latent spaces where the model would measure higher log-likelihoods, and found that the DoSE results were similar, but the results from competing techniques worsened substantially. We fix the mean and logit of the first component of $r(Z)$ to improve training stability. For MNIST, we use a Bernoulli distribution for the decoder log-likelihood. For FashionMNIST, we instead used a Logit-Normal distribution, a bijective transformation of the normal distribution to the interval $(0, 1)$ using a sigmoid bijector, since the majority of the spatial variation between pixels in FashionMNIST occurs at values near 0.5, where the Bernoulli distribution struggles to capture variation.

For the Glow models, we replicated the architecture from [Nalisnick et al., 2019b], using 3 spatial hierarchies of 8 steps of the flow. Each step of the flow consists of *actnorm*, an invertible 1×1 convolution, and an affine coupling layer. We use a RealNVP bijector [Dinh et al., 2017] for the coupling layer, which uses a 3-layer convolutional stack with ReLU activations and 400 filters. For stability in training, the last convolutional layer is set to 0 at initialization for each stack, which corresponds to the full Glow network simply producing an identity transformation (with some rearranging of the pixels) at initialization. Between each spatial hierarchy, we remove half of the data to create multiple different levels of spatial variation. Altogether the Glow network constructs a bijective transformation which projects the data X into a latent space Z with the same dimensionality (3072, in these experiments). The full Glow model is then created as a transformed distribution using $\mathcal{N}(0, 1)$ as the base distribution, and the Glow network as the bijector. All experiments were performed using TensorFlow and TensorFlow Probability [Abadi et al., 2015].

Training details Following Kingma and Dhariwal [2018], we train Glow models using the Adamax optimizer [Kingma and Ba, 2014] with a learning rate initialized to 0 and gradually increased to 0.001 over 10 epochs, after which point it is held constant. We trained the models for 250 epochs in total. We optimize the negative log-likelihood $q(X|\theta_n)$ with added L_2-regularization of the weights to reduce memorization in the model. We explored regularization constants of $\lambda = [0., 0.01, 0.05, 0.1, 0.5]$, and determined that $\lambda = [0.05, 0.1]$ limited memorization without also limiting generative model performance.

For VAE models, convergence was much faster, so we train for 50 epochs using a learning rate initialized at 0.0001, and decayed exponentially by half every 10000 training steps. We follow Choi et al. [2018] and use the Adam optimizer to optimize the traditional Evidence Lower Bound (ELBO). We evaluate the ELBO using 16 samples from the posterior distribution. To prevent memorization, we employ two additional procedures: First, we "burn-in" the decoder for one epoch by drawing samples from the prior, and use the decoder to estimate the log-likelihood for each input given the samples. This has the effect of initializing our likelihood to be properly conditioned on the prior, keeping small initial gradients for the encoder early on in training. Second, we employ "reverse beta-annealing" during training. We start with a large value of $\beta = 100$, and we decay its value by a factor of 2.0 every 3 epochs. We found that this causes the posterior to be more effectively anchored to the prior during training, which ultimately results in more informative latent spaces and a more useful sampling distribution (and therefore more reliable outlier detection).

For each dataset, we trained 5 separate models following [Lakshminarayanan et al., 2017, Choi et al., 2018, Nalisnick et al., 2019b]. This allows us to both quantify the variability in performance over separate training runs, as well as to utilize an ensemble of all 5 models in order to produce a stronger and more robust estimator.

Evaluation of performance. Once a model is trained, we construct our DoSE by measuring the value of summary statistics of the model, computed on the elements of the training set. For VAEs, we have an abundance of possibilities:

- KL divergence between the posterior and marginal $T_n^{(\text{rate})}(X) = \mathsf{KL}[q(Z|X, \theta_n), q(Z)]$ (rate)
- Cross-entropy between the posterior and marginal $T_n^{(\text{xent})}(X) = \mathsf{H}[q(Z|X, \theta_n), q(Z)]$
- Entropy of the posterior $T_n^{(\text{ent})}(X) = \mathsf{H}[q(Z|X, \theta_n)]$
- Expected log-likelihood computed over the posterior $T_n^{(\text{dist})}(X) = \mathsf{E}_{q(Z|X,\theta_n)}[q(X|Z, \theta_n)]$ (distortion)
- Estimate of the evidence computed using a 16-sample importance weighted autoencoder (IWAE) given by $T_n^{(\text{iwae})}(X) = q(X|\theta_n) = \mathsf{E}_{q(Z|X)}[q(X|Z, \theta_n)q(Z)/q(Z|X, \theta_n)]$ (log-likelihood, following the terminology of Nalisnick et al. [2019a,b], Choi et al. [2018], Ren et al. [2019])

For Glow models, the number of statistics is more constrained because Glow does not have as many ways to evaluate summaries on the generative model. In this work, we use:

- "Log-likelihood" $T_n^{(\text{like})}(X) = q(X|\theta_n)$, and its two constituents
- Log-probability of the latent variable $T_n^{(\text{latent})}(X) = q(Z(X)|\theta_n)$
- Log of the determinant of the Jacobian between X and Z (i.e., $T_n^{(\text{jacobian})}(X) = \log|\mathsf{J}(X)|$)

For each statistic that we measure in the training set, we compute a Kernel Density Estimate (KDE), using the default implementation in SciPy [Virtanen et al., 2020] to build an individual DoSE. $DoSE_{KDE}$ is then simply the sum over all the DoSE scores for an individual statistic:

$$DoSE_{KDE} = \sum_{j}^{m} KDE_j(x) \tag{9}$$
$$= \sum j^m \frac{1}{nh} \sum_{i}^{n} \phi\left(\frac{T_j(x) - T_j(x_i)}{h}\right)$$

We build $DoSE_{SVM}$ by creating an n-dimensional feature vector of the n metrics for each observation. We first use Principal Components Analysis (PCA) to learn a whitening transformation from the training set to help correct against the wildly different variance observed in different statistics. We then use the transformed space to learn a one-class SVM. Both PCA and the SVM use the default implementations in scikit-learn [Pedregosa et al., 2011].

Before we evaluate the DoSE performance on OOD data we check its memorization. To do this we measure the expected calibration error (ECE) [Guo et al., 2017] of $DoSE_{KDE}$ using a small heldout subset of 10% of the examples from the training set. These examples are in-distribution but never seen during training, and therefore the ECE measures the degree to which the DoSE scores given to new in-distribution data are consistent with the scores given to data seen during training. In our experiments, we found that without some form of intervention, both VAE and Glow models exhibited extreme capacity for memorization, and therefore had high ECE. This inspired our earlier described preventative measures, such as reverse beta-annealing for VAEs, and L_2-regularization for Glow. Using these additional procedures, we found that our memorization scores were typically around 1% for most models.

We evaluate the performance of $DoSE_{KDE}$ and $DoSE_{SVM}$ by computing the scores on the specified OOD datasets, and use these scores to measure the AUROC for OOD detection. We compare our method against four unsupervised baselines: the vanilla likelihood $q(X|\theta_n)$, Watanabe-Akaike information criterion (WAIC) [Choi et al., 2018], the typicality test (TT) using a batch size of 1 (which represents a more realistic application than a larger batch size), [Nalisnick et al., 2019b], and likelihood ratios (LLR) [Ren et al., 2019]. For WAIC, we use Eq. 1 from their paper to compute the scores. For LLR, we train a background model using their method of mutations to perturb the input data. We use a mutation rate of 0.15, which is in the middle of the range of values they found to produce acceptable results. The LLR score is then the difference between the scores from models trained without and with mutations. With the exception of the background models used for LLR, all methods are evaluated on the same models. This provides an apples-to-apples comparison between methods, and highlights the differences between them as a function of the method itself, rather than the underlying model.

D Histograms of Statistics

We show histograms of statistics for VAE on MNIST and FashionMNIST in Figure 6 and Figure 7 respectively. We show histograms for Glow on CIFAR10, SVHN and CelebA in Figures 8,9 and 10 respectively.

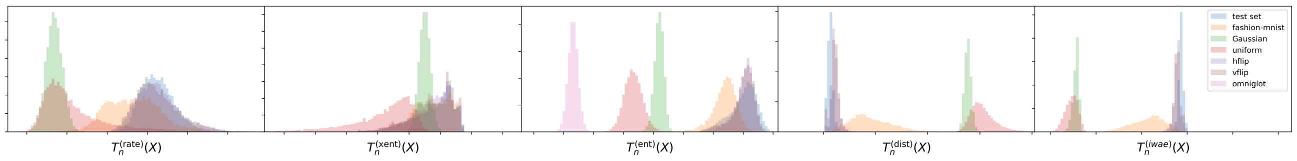

Figure 6: Histograms of 5 different statistics evaluated on a VAE trained on the MNIST dataset. The leftmost column shows the KL divergence between the posterior and the prior. The second column shows the cross-entropy between the posterior and the prior. The third column shows the entropy of the encoder. The fourth shows the distortion (the expected log-likelihood from the decoder). The last column shows the log-evidence, computed using a 16-sample IWAE estimate. For each metric, we show the distribution of that metric observed in the test set, along with multiple different OOD datasets.

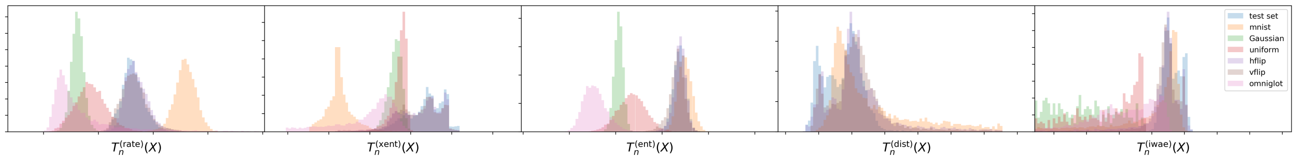

Figure 7: Same as Figure 6, but for a VAE trained on FashionMNIST. Note that while the log-likelihood is a successful OOD detection metric when trained on MNIST, it does not perform similarly when trained on FashionMNIST, often overlapping strongly with various OOD datasets. Other statistics, such as the KL divergence between the posterior and the prior appear to be much more informative in this case.

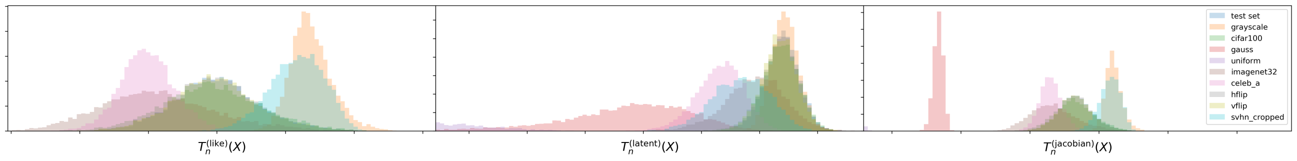

Figure 8: Same as Figure 6, but for a Glow model trained on CIFAR10.

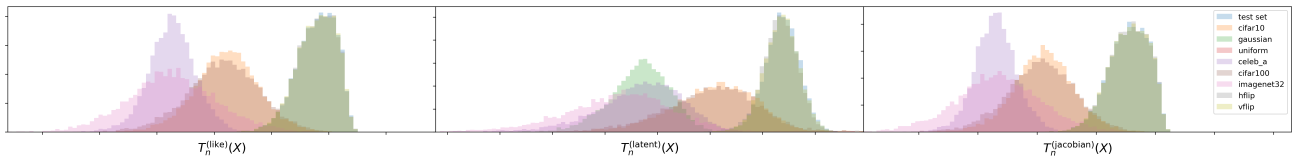

Figure 9: Same as Figure 6, but for a Glow model trained on SVHN.

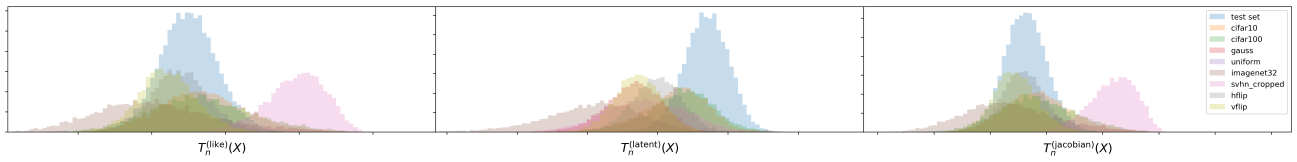

Figure 10: Same as Figure 6, but for a Glow model trained on CelebA.