Shaken, and Stirred: Long-Range Dependencies Enable Robust Outlier Detection with PixelCNN++

Barath Mohan Umapathi[1], **Kushal Chauhan**[2], **Pradeep Shenoy**[2] and **Devarajan Sridharan**[3,4,*]
[1]Department of Physics, Indian Institute of Science
[2]Google Research
[3]Center for Neuroscience, Indian Institute of Science
[4]Computer Science and Automation, Indian Institute of Science
barathu@iisc.ac.in, {kushalchauhan, shenoypradeep}@google.com, sridhar@iisc.ac.in

Abstract

Reliable outlier detection is critical for real-world deployment of deep learning models. Although extensively studied, likelihoods produced by deep generative models have been largely dismissed as being impractical for outlier detection. First, deep generative model likelihoods are readily biased by low-level input statistics. Second, many recent solutions for correcting these biases are computationally expensive, or do not generalize well to complex, natural datasets. Here, we explore outlier detection with a state-of-the-art deep autoregressive model: PixelCNN++. We show that biases in PixelCNN++ likelihoods arise primarily from predictions based on local dependencies. We propose two families of bijective transformations – "stirring" and "shaking" – which ameliorate low-level biases and isolate the contribution of long-range dependencies to PixelCNN++ likelihoods. These transformations are inexpensive and readily computed at evaluation time. We test our approaches extensively with five grayscale and six natural image datasets and show that they achieve or exceed state-of-the-art outlier detection, particularly on datasets with complex, natural images. We also show that our solutions work well with other types of generative models (generative flows and variational autoencoders) and that their efficacy is governed by each model's reliance on local dependencies. In sum, lightweight remedies suffice to achieve robust outlier detection on image data with deep generative models.

1 Introduction

Deep discriminative models confidently misclassify test samples far removed from their training distributions [Hendrycks and Gimpel, 2017]. By contrast, deep generative models (DGMs) offer a potentially promising approach for identifying outliers. DGMs model the likelihood distribution of the in-distribution (ID) training data and should, in principle, assign lower likelihoods to unfamiliar, out-of-distribution (OOD) samples. In practice, however, DGMs can assign higher likelihoods to OOD samples because of biases arising from low-level image statistics [Ren *et al.*, 2019; Nalisnick *et al.*, 2019a; Choi *et al.*, 2018; Xiao *et al.*, 2020].

In our work[1], we analyze biases in likelihoods produced by a state-of-the-art DGM: PixelCNNs [Van Den Oord *et al.*, 2016]. PixelCNNs are a type of deep autoregressive model that computes the likelihood of an image as a factorized product of the conditional likelihood of its sub-pixels [Van Den Oord *et al.*, 2016]. The likelihood for each pixel is modeled based on the context of preceding sub-pixels using convolutional networks (Fig. 1a-b). We examine the more recent PixelCNN++ model [Salimans *et al.*, 2017]. Despite its ability to produce accurate reconstructions and generate realistic samples, PixelCNN++'s likelihoods are also readily biased and unreliable for outlier detection [Ren *et al.*, 2019; Serrà *et al.*, 2020; Nalisnick *et al.*, 2019a].

We investigate the origin of the biases in PixelCNN++ likelihoods and propose effective solutions for correcting for these biases. Our contributions are as follows:

- We show that biases in PixelCNN++ likelihoods arise primarily from an over-reliance of the model on local dependencies in the image.
- We devise two efficient solutions based on readily computed bijective transformations of the input samples. These enable correcting for biases arising from local dependencies and pinpointing the component of the likelihood arising from long-range dependencies.
- The solutions we propose are computationally inexpensive to implement. Moreover, they can be applied *post hoc* during evaluation time and do not require retraining the model or training multiple (background) models.
- We evaluate our solutions extensively using 11 inlier (5 grayscale and 6 natural image) datasets and 15 evaluation (7 grayscale and 8 natural image) datasets and show that they match or exceed state-of-the-art outlier detection performance [Ren *et al.*, 2019; Serrà *et al.*, 2020].

[1]Code associated with this paper is available at: https://github.com/coglabiisc/googleresearch/tree/main/pixelcnn_ood

The full version of this paper, including Appendices, is available at: https://arxiv.org/abs/2208.13579

* Corresponding author

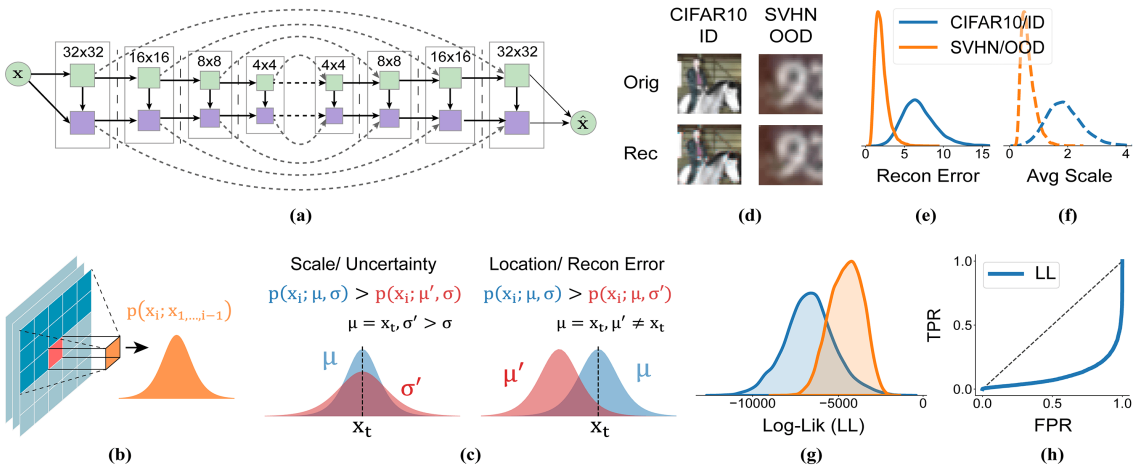


Figure 1: **Biases with PixelCNN++ likelihoods.** **(a)** PixelCNN++ model architecture. We use one more hierarchy, including a nested set of 4×4 convolutional layers, compared to the original PixelCNN++ model. Green and purple blocks: vertical and horizontal stacks, respectively. Solid arrows: convolutional connections; curved, dashed arrows: short-cut connections. **(b)** PixelCNN++ models the likelihood of the current pixel x_i based on the preceding rows and columns of pixels. **(c)** (Left) Larger the scale of the predicted logistic, the higher the uncertainty, and the lower the likelihood for the target pixel $\mathbf{x}_t$. (Right) The higher the deviation between the mode of the predicted logistic and $\mathbf{x}_t$, the higher the reconstruction error, and the lower the likelihood for the target pixel. **(d)** PixelCNN++ model trained on the CIFAR10 datasets reconstructs both CIFAR10/ID and SVHN/OOD samples well. **(e-f)** PixelCNN++ reconstruction error (e) and average predicted scale (f) distributions for CIFAR10/ID (blue) and SVHN/OOD (orange) test samples. Surprisingly, SVHN/OOD samples have lower reconstruction error and scale as compared to CIFAR10/ID. **(g)** PixelCNN++ log likelihood distributions. SVHN/OOD samples get higher log-likelihoods than CIFAR10/ID samples. **(h)** ROC curve for outlier detection using PixelCNN++ likelihoods for CIFAR10/ID and SVHN/OOD. The curve is bowed downwards, indicating anomalously higher likelihoods for OOD than ID samples.

2 Related Work

While considerable past work addresses outlier detection in supervised settings [Lakshminarayanan *et al.*, 2017; Liang *et al.*, 2018], we focus here exclusively on the unsupervised setting where no labels are available. Unlike previous deep, one-class classification approaches [Andrews *et al.*, 2016; Ruff *et al.*, 2018], we do not employ class label information either for training or validation.

Perhaps the most relevant approach for our study is the Input Complexity (IC) metric of [Serrà *et al.*, 2020]. This study characterized an identity relationship between the negative log-likelihood and image complexity, revealing a major source of bias. An elegant outlier detection score was then formulated by simply subtracting image complexity from the negative log-likelihood, with complexity being quantified as the compression length based on standard compressors (e.g., JPEG, PNG, or FLIF). In a later section, we analyze the IC metric and showcase key failure cases that violate the assumptions underlying this approach.

A second, relevant study [Ren *et al.*, 2019] showed that a greater number of zeros in test image backgrounds biased PixelCNN++ likelihoods toward higher values. This study proposed training an additional background model with noise-corrupted images to compute a likelihood ratio that factored out the contribution of background information to the likelihood. In addition to being computationally expensive, due to the requirement of training multiple models, this metric did not perform well with our (simpler) PixelCNN++ model architecture, as we show subsequently.

A third, related study [Bergman and Hoshen, 2020] proposed an open set detection method, GOAD, that computes an anomaly score based on random affine transformations. Yet, unlike GOAD our methods involve only bijective transformations that isolate long-range dependencies in the likelihood ratio (see Sections. 3.4 and 3.5). Moreover, GOAD requires the transformations to be applied both during training and evaluation time whereas our approach requires transformations to be applied during evaluation time alone. Our method can, therefore, be applied to likelihoods generated by pre-trained generative models also.

Similarly, other approaches involving generative ensembles (e.g., WAIC, [Choi *et al.*, 2018]) or principled statistical tests (e.g., typicality, [Nalisnick *et al.*, 2019b]) are either computationally expensive or do not perform well with singleton test samples, unlike our approach.

A few studies have examined outlier detection with other classes of generative models like variational autoencoders (VAEs) or flow models. For example, [Xiao *et al.*, 2020] developed a "Likelihood Regret" score for robust outlier detection with VAEs. Yet, this score is expensive to compute because an optimization must be performed by retraining the VAE's encoder for each sample at test time. Similarly, [Chauhan *et al.*, 2022] developed an efficient correction for biases in VAE visible distributions (e.g., Bernoulli) for robust outlier detection. Both of these approaches cannot be readily extended to PixelCNN++ models. Moreover, [Kirichenko *et al.*, 2020] showed that simple modifications to the architecture of normalizing flows could enable learning semantic features, thereby ameliorating low-level biases. Yet, their modification is specific to flow models and involves fully retraining

the modified model. On the other hand, our approach works with a fully trained model directly at evaluation time.

Additionally, our approach resembles standard data augmentation methods, albeit superficially. For instance, [Yun *et al.*, 2019] augment data by cutting and pasting image patches among training images, enabling efficient network regularization in supervised or weakly-supervised settings. Our approach, on the other hand, uses bijective transformations to isolate long-range dependencies in an unsupervised setting.

3 De-Biasing PixelCNN++ Likelihoods

3.1 What Factors Contribute to the Bias in PixelCNN++ Likelihoods?

To analyze the bias in PixelCNN++ likelihoods, we first analyzed the factors contributing to it. To model pixel likelihoods, PixelCNN++ employs a categorical distribution approximated by a discretized mixture of logistics [Salimans *et al.*, 2017]. Variations in logistic likelihoods can be readily attributed to two sources. One source involves the location parameters (modes) of the underlying mixture of logistics: the model's best guess of the target pixel value. Accurate prediction of the target pixel value yields a higher log-likelihood (Fig. 1c, left). A second source involves the scale parameters (variances): the model's uncertainty with predicting the target pixel. A lower scale parameter (greater certainty) for a mode coinciding with the target pixel value yields higher log-likelihoods (Fig. 1c, right). Thus, more accurate predictions (lower reconstruction error) and more confident (less uncertain) predictions of the correct target pixel value both contribute to higher PixelCNN++ likelihoods.

We analyzed these two factors for a PixelCNN++ model trained with CIFAR10 images (ID) and tested with SVHN images (OOD) (see Fig. 1d for reconstructions). The PixelCNN++ model trained on CIFAR-10 (ID) images, surprisingly reconstructs SVHN images (OOD) both with lower error (Fig. 1d and 1e, blue/SVHN vs. orange/CIFAR-10) and with higher confidence (lower average scale, Fig. 1f, blue/SVHN vs. orange/CIFAR-10). Paradoxically, these two factors yielded "higher likelihoods" for SVHN/OOD than for CIFAR-10/ID samples (Fig. 1g-h). Similar results were also observed with grayscale data (FMNIST/ID vs MNIST/OOD). We investigated the reasons behind these trends.

3.2 Can Global Complexity Fully Explain the Bias in PixelCNN++ Likelihoods?

Are biases in PixelCNN++ likelihoods fully explained by differences in overall image complexity [Serrà *et al.*, 2020]? Pixel values in less complex images are easier to model because of the stronger spatial correlations among adjacent pixels. Therefore, the comparatively lower complexity of images in datasets like MNIST or SVHN may permit more accurate and more confident predictions of sub-pixel values in these datasets, thereby inflating their likelihoods. Based on this logic, [Serrà *et al.*, 2020] showed that PixelCNN++ negative log likelihoods exhibit a near-identity relationship with image complexity. They proposed an elegant OOD score that involved simply subtracting the complexity estimate ($L(\mathbf{x})$) from the negative log-likelihood (NLL) to account for this "complexity bias".

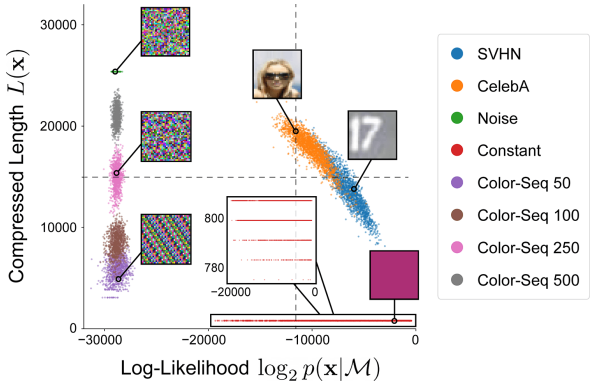


Figure 2: **Compressed lengths have a many-to-one relationship with PixelCNN++ likelihood.** Compressed length, measured with a PNG compressor, plotted against log-likelihood generated by a PixelCNN++ trained on CIFAR-10 for different OOD datasets (different colors, see text). A wide range of compressed lengths occurs for the same model likelihood (dashed vertical line) and vice-versa (dashed vertical line).

To further explore this assumption, we plotted normalized compressed lengths ($L(\mathbf{x})$) using a PNG compressor against log-likelihoods ($\log p(\mathbf{x})$) computed with a PixelCNN++ model trained with the CIFAR10 dataset. Although we observed a strong negative correlation between compressed lengths and log-likelihoods for OOD data comprised of natural images, this relationship was violated for other kinds of data. Specifically, "constant" images – in which all pixels were of a uniform color – revealed a nearly flat relationship (Fig. 2, red points): compressed lengths were virtually identical even as log-likelihoods varied over several orders of magnitude ($\log p(\mathbf{x})$ = ∼-20000 to 0) depending on the image color (Fig. 2, inset). By contrast, images with repeating color sequence patterns across pixels showed the opposite trend: compressed lengths varied over two orders of magnitude ($L(\mathbf{x})$ = ∼250 to 25000) without a substantial change in the log-likelihood (Fig. 2, purple, brown, pink, and gray points). In other words, widely different log-likelihoods occurred for images with identical compressed lengths (Fig. 2, dashed horizontal line). Conversely, images with similar log-likelihoods exhibited widely different compressed lengths (Fig. 2, dashed vertical line).

Similar results were observed with other types of compressors (e.g., JPEG, FLIF) and also when considering the "best" compressor (minimum compression length, $L(\mathbf{x})$= min($L_1(\mathbf{x})$, $L_2(\mathbf{x})$, ...)). In other words, the assumption on which the IC metric is predicated – that the compression length provides a reliable estimate of the negative log-likelihood under an unbiased, universal model – appears to not hold true across all types of images.

3.3 Do Local Dependencies Contribute to the Bias in PixelCNN++ Likelihoods?

In addition to global complexity, could local dependencies across pixels bias PixelCNN++ likelihoods? We explored

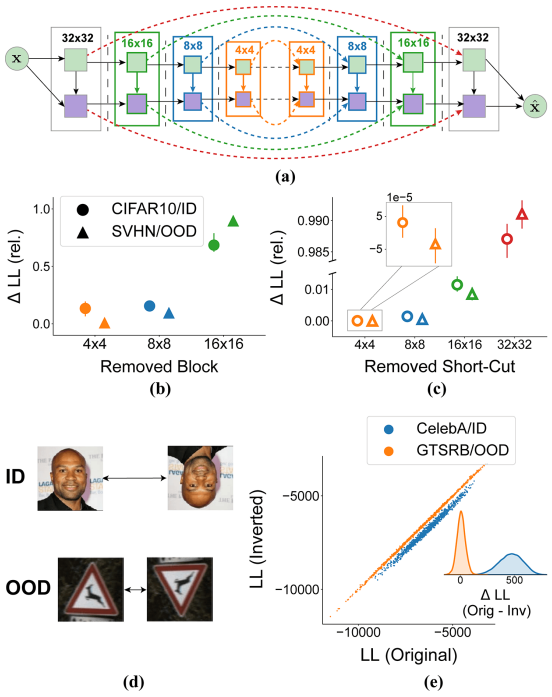


Figure 3: **PixelCNN++ relies heavily on local dependencies for prediction. (a)** A PixelCNN++ model was trained on CIFAR10/ID and tested on SVHN/OOD. Ablation experiments were performed by removing the outermost (local dependency) or innermost (long-range dependency) blocks. Blocks of matching sizes (e.g., 8×8) were ablated jointly (matching colors). In turn, additional ablations were performed by removing each set of short-cut connections. **(b)** Effect of ablating convolutional blocks on model likelihood. Orange, blue, and green: contribution of the 4×4, 8×8, and 16×16 blocks, respectively. **(c)** Effect of ablating short-cut connections on model likelihood. Orange, blue, green, and red: contribution of the 4×4, 8×8, 16×16, and 32×32 short-cut connections, respectively. **(d)**. A PixelCNN++ model was trained on CelebA/ID and tested on GTSRB/OOD. We expect log-likelihoods of original and inverted images to be more similar for GTSRB/OOD samples than for CelebA/ID samples. **(e)** Log-likelihoods of the inverted images plotted against their original counterparts for CelebA/ID and GTSRB/OOD. (Inset) Difference between log-likelihoods of original and inverted samples (ΔLL) is higher for CelebA/ID than for GTSRB/OOD.

the hypothesis that PixelCNN++ leverages local dependencies to generate accurate and confident predictions. For example, both SVHN and MNIST images typically comprise digits with relatively simple features embedded in a fairly uniform background. By acquiring knowledge of simple local features like edges and contours, the model can accurately predict subpixel values in adjacent pixels. In fact, this could happen even for OOD images, using knowledge of the local neighborhood, without learning the long-range structure of the data.

We tested this hypothesis using simple ablations to a standard PixelCNN++ model (Fig. 3a). In the model, the innermost parts of the network capture long-range dependencies over longer spatial scales – a direct consequence of using strided convolutions over progressive layers [Salimans *et al.*, 2017]. In our PixelCNN++ model, in addition to the 8×8 and 16×16 CNN layers in the standard model, we introduced another sequence of 4×4 layers in the innermost part of the network (Fig. 3a, orange block).

We tested the effect of progressively removing nested hierarchies of the innermost 4×4, 8×8, and 16×16 convolutional layers on model likelihoods. Precisely in line with our hypothesis, removing the 4×4 layers (Fig. 3b, orange triangle) produced virtually no change in the log-likelihoods (LLs) for OOD data (SVHN OOD vs CIFAR-10/ID). Yet, ablating the 8×8 (Fig. 3b, blue triangle) or 16×16 (Fig. 3b, green triangle) innermost blocks produced substantially greater reductions in likelihoods. In other words, the model relied primarily on local dependencies when making predictions with OOD data. By contrast, for ID test data, even just removing the 4×4 block (Fig. 3b, orange circle) produced a noticeable change in the likelihoods, indicating that the model had learned to exploit long-range dependencies when making predictions with ID data. These progressive changes in likelihoods were accompanied by progressively higher reconstruction errors and more uncertain predictions in the model. Removing the short-cut connections, at different levels, in turn, yielded similar results, with the largest reduction in likelihoods occurring when the outermost layer (32×32) connections (Fig. 3c, red symbols) were removed. As before, ablating the innermost layers had a more significant effect on ID likelihoods than OOD data (Fig. 3c, circles vs triangles).

In sum, the PixelCNN++ model relied strongly on local dependencies for accurate and confident predictions. Yet, removing network components that captured long-range dependencies (innermost layers) produced larger changes in likelihoods for ID than for OOD samples. We sought to exploit these differences for efficient outlier detection.

3.4 Isolating the Contribution of Long-Range Dependencies to PixelCNN++ Likelihoods

Given the strong bias in PixelCNN++ likelihoods induced by local dependencies, we asked whether removing contributions of local dependencies or isolating contributions of long-range dependencies would de-bias PixelCNN++ likelihoods. To this end, we explored simple transformations to the input images that would preserve local dependencies but systematically perturb long-range dependencies. We hypothesized that perturbing long-range (but not local) dependencies would produce a stronger degradation of the likelihood for ID data as compared to OOD data.

To illustrate this idea, we explore a transformation by "inversion". Because the PixelCNN++ model has a specific order of scanning and predicting pixels in the image (upper left to lower right), upon inversion, characteristic image features fall into a context unfamiliar to the model. For example, inverting an image of a face positions the eyes below the nose, and the nose below the mouth (Fig. 3d, top). A PixelCNN++ model trained, for example, on a dataset with face images (e.g., CelebA) would then predict pixels in an inverted face image less accurately and with higher uncertainty than pixels in a standard, upright face. As a result, the model would yield lower likelihoods for inverted CelebA images rather than upright images. We hypothesized that this would not occur for OOD images (e.g., GTSRB). In this case, the model is un-

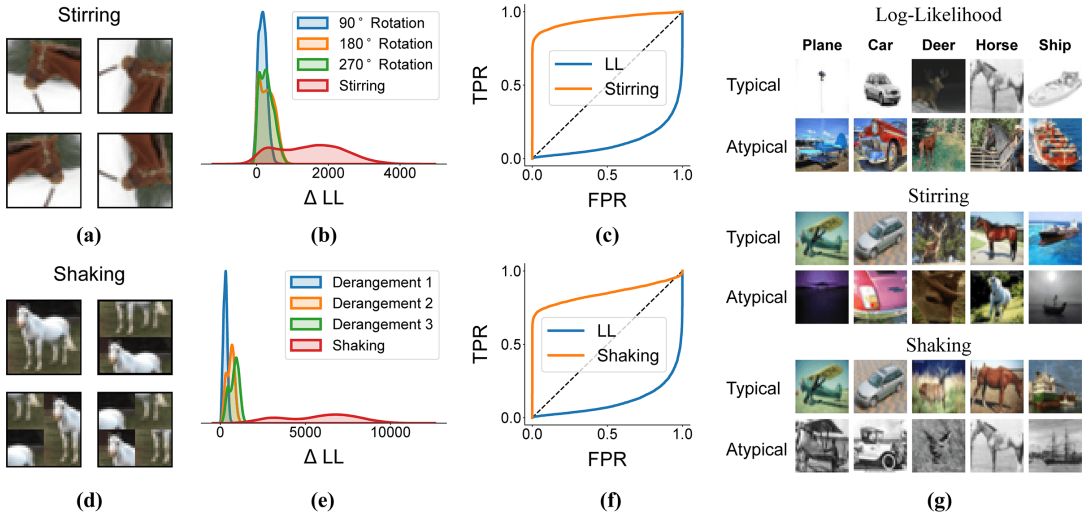


Figure 4: **Robust outlier detection with "stirring" and "shaking". (a)** Examples of "stirring" transformations. **(b)** Change in PixelCNN++ LL after "stirring", (red) as compared with rotations, applied individually (CIFAR10/ID). **(c)** ROC curve for outlier detection for CIFAR10/ID vs SVHN/OOD case using vanilla log-likelihood (blue) and "stirring" (orange). **(d)** Examples of "shaking" transformations **(e, f)** Same as in (b) and (c), but for "shaking". **(g)** Specific sub-category of CIFAR10/ID images (columns) that were assigned among the highest and lowest vanilla log-likelihoods $\log p(\mathbf{x})$ (top 2 rows), following "stirring" (middle 2 rows) or following "shaking" (bottom 2 rows).

likely to rely on long-range predictions for either the upright or the inverted GTSRB images because both categories of images are equally unfamiliar (Fig. 3d, bottom). Thus, the model should yield equivalent likelihoods for both inverted and upright GTSRB images.

We tested and confirmed this hypothesis with the PixelCNN++ model trained on the CelebA dataset. The model yielded systematically lower likelihoods for inverted faces than for upright faces in the ID data (Fig. 3e, blue points). In contrast, the model yielded virtually identical likelihoods for OOD (GTSRB) images (Fig. 3e, orange points). Therefore, one solution for outlier detection is to simply subtract the log-likelihood of the original image from that of the perturbed (inverted) image to isolate the contribution arising from long-range dependencies. With this reasoning, we propose an "outlier detection score" as follows:

$$\log p_{\mathrm{LR}}(\mathbf{x}) = \log p_\theta(\mathbf{x}) - \log p_\theta(\mathbf{x}')$$

where θ represents the PixelCNN++ model parameters, $\mathbf{x}$ represents the test sample (image, in this case), $\mathbf{x}'$ represents the same test sample after a perturbation that preserves local dependencies but disrupts long-range dependencies, $\log p_\theta(\mathbf{x})$ represents the log-likelihood of sample yielded by the PixelCNN++ model and $\log p_{\mathrm{LR}}$ represents a component of the log-likelihood that depends primarily on long-range dependencies in the model. This formulation can also be construed as a log-likelihood ratio between the original and perturbed samples, assuming factorizable contributions of local and long-range dependencies to the overall likelihood. We expect to observe a larger $\log p_{\mathrm{LR}}$ for ID data than OOD data.

Central to the success of this approach is identifying transformations that preserve local dependencies while disrupting long-range ones. We identify and explore two families of transformations that we call "stirring" and "shaking".

3.5 Bijective Transformations for Robust Outlier Detection with PixelCNN++

Stirring. We extend the inversion solution by incorporating a family of 7 geometric transformations, including 3 rotations of the original image (by 90°, 180° or 270°), lateral inversion (mirror reflection about the vertical midline), and 3 rotations of the reflected image (again, by 90°, 180° or 270°). The $\log p_{\mathrm{LR}}$ is summed across all 7 transformations to yield the final outlier detection score. We term the collection of these transformations as "stirring" (Fig. 4a). Because the individual transformations contribute additively, "stirring" produced a larger change in likelihoods for the perturbed images compared to the upright images (Fig. 4b, red density) than individual rotations (Fig. 4b, blue, orange, and green densities). "Stirring" may, thus, enable robust outlier detection for images with distinct axes of symmetry.

Shaking. We consider a second class of bijective transformations that involve dividing the images into patches and shuffling these patches randomly. We consider three ways to achieve this: i) splitting the image in half along the horizontal midline, ii) splitting the image in half along the vertical midline, and iii) splitting the images into four quarters along the horizontal and vertical midlines (Fig. 4d). These permit a total of 9 unique derangements – random permutations in which no patch is located in its original position. $\log p_{\mathrm{LR}}$ is summed across all 9 derangements to yield the final outlier detection score. We term the collection of these derangements as "shaking" (Fig 4d). Again, "shaking" produced a larger change in likelihoods for the perturbed images compared to the upright images than individual derangements (Fig. 4e).

Conditional correction. Our outlier detection score is a ratio of the logarithm of two probability densities: $p_\theta(\mathbf{x})$ and $p_\theta(\mathbf{x}')$. When $p_\theta(\mathbf{x})$ is a very small numerical value, the like-

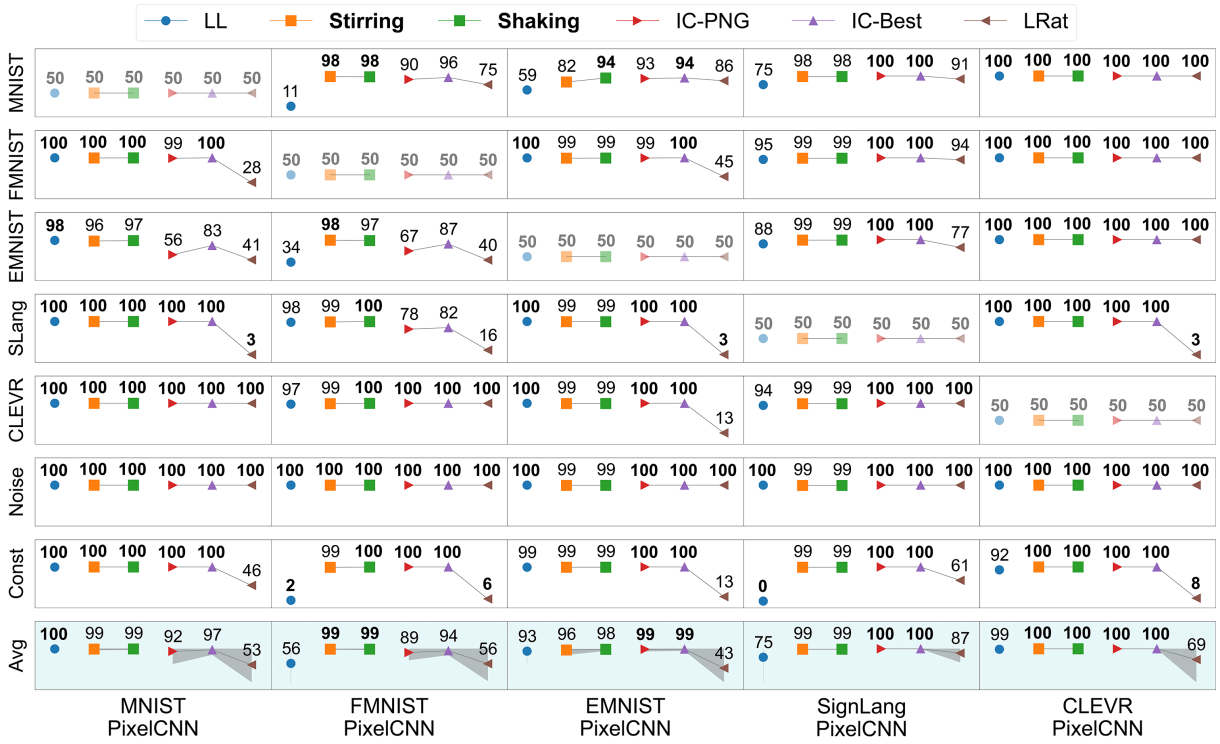


Figure 5: **O**utlier detection performance: Grayscale data. Outlier detection AUROC values for PixelCNNs trained with grayscale image datasets (ID, columns) and tested with other grayscale datasets (OOD, rows). Last row: average AUROC; gray shading: range of AUROC values. Blue: log-likelihood (LL), uncorrected; Orange: "Stirred" LL; Green: "Shaken" LL; Red: Input Complexity (PNG); Purple: Input Complexity (Best); Brown: Likelihood Ratio. Numbers in bold: best performance.

lihood of the perturbed sample $\mathbf{x}'$, $p_\theta(\mathbf{x}')$ would also be comparably small numerically. In this case, it is likely, that the estimation of $\log p_{\text{LR}}$ would be noisy and far from accurate. To avoid such noisy estimates, we adopt a pre-filtering strategy based on identifying outliers with the model log-likelihood alone, following which "stirring" and "shaking" corrections are applied. We perform ablation experiments to estimate the contribution of this conditional correction.

4 Experiments

We trained PixelCNN++ models on each of five grayscale image datasets: MNIST, FashionMNIST, EMNIST Letters, Sign Language MNIST, and CLEVR [Deng, 2012; Xiao *et al.*, 2017; Cohen *et al.*, 2017; Johnson *et al.*, 2017; Sign Language MNIST, 2017]. Each of these models was tested against six OOD datasets, including the other four datasets and noise and constant images. Similarly, we trained PixelCNN++ models on each of six natural image datasets - SVHN, CelebA, CompCars, GTSRB, CIFAR10, and LSUN (classroom) [Netzer *et al.*, 2011; Liu *et al.*, 2015; Yang *et al.*, 2015; Stallkamp *et al.*, 2011; Krizhevsky, 2009; Yu *et al.*, 2015]. Each model was tested against seven OOD datasets, including noise and constant images.

We report the area under the ROC curve (AUROC) between the respective test sets of the ID and OOD datasets in a 7×5 grid for grayscale and an 8×6 grid for natural image data. All results reported include the conditional correction.

4.1 Outlier Detection Performance with "Stirring" and "Shaking"

For illustration, we compare outlier detection performance of $\log p_{\text{LR}}$ with "stirring" with the vanilla log-likelihood. We obtained state-of-the-art AUROCs (∼95%) for the particularly problematic case of CIFAR10 ID versus SVHN OOD (Fig. 4c). Moreover, typical exemplars were assigned among the highest $\log p_{\text{LR}}$ (Fig. 4g, middle), whereas this was not the case using the vanilla log-likelihoods (Fig. 4g, top).

This superlative performance was observed across all other comparisons comprising both grayscale and natural image datasets. In almost all cases, "stirring" (Fig. 5 and Fig. 6, orange symbols) outperformed vanilla LL (Fig. 5 and Fig. 6, blue symbols). "Stirring" achieved a performance near-ceiling in most of the grayscale cases. In the challenging cases of FMNIST/ID, CIFAR10/ID, and LSUN/ID, "stirring" achieved AUROCs of 95 and above. Overall, with "stirring", we saw an average AUROC improvement of ∼14% for grayscale images and ∼57% for natural images.

We also obtained similar results with $\log p_{\text{LR}}$ computed from "shaken" images. Again, "shaking" improved outlier detection for the challenging case of CIFAR10/ID versus SVHN/OOD (Fig. 4f). In general, outlier detection with "shaking" improved across both grayscale and natural images (Fig. 5 and Fig. 6, green symbols), as compared to that with vanilla log-likelihoods. Overall, outlier detection performance improved, on average, by ∼14% for grayscale images

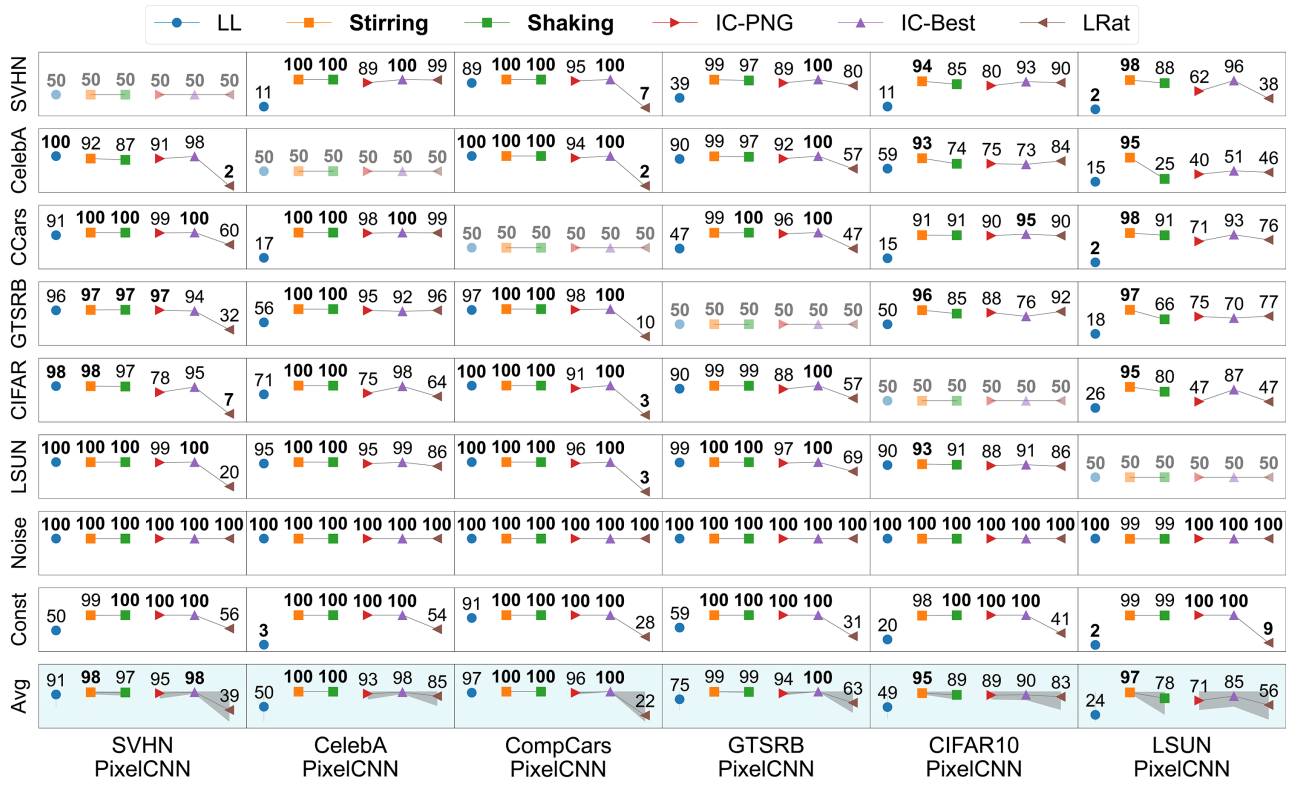


Figure 6: **Outlier detection performance: Natural image data.** Same as in Figure 5 but for natural image datasets.

and ∼47% for natural images, indicating marginally worse improvements than with "shaking". Interestingly, with "stirring", color information played a major role in determining atypical exemplars (Fig. 4g, lower).

4.2 Comparison with Competing Methods

We compare our results with two state-of-the-art methods – Likelihood Ratios [Ren *et al.*, 2019], and Input Complexity [Serrà *et al.*, 2020] – the two most relevant competing approaches for state-of-the-art outlier detection with PixelCNN++ (see section 2).

Our methods comfortably outperformed likelihood ratios (Fig. 5 and Fig. 6, filled red symbols) in all cases. "Stirring" performed ∼48% better on average for grayscale images and ∼80% better for natural images than likelihood ratios. Similarly, "shaking" performed ∼49% better, on average, for grayscale images and ∼69% better for natural images than likelihood ratios. The OOD detection numbers that we report for likelihood ratios are poorer than those reported by [Ren *et al.*, 2019], who used a more complex model architecture; these results suggest that the success of the likelihood ratio metric is architecture dependent.

Our metrics also outperformed or performed comparably with Input Complexity computed using the PNG compressor (Fig. 5 and Fig. 6; IC-PNG, brown symbols) or using the Best compressor (minimum compressed length; IC-Best, purple symbols). "Stirring" performed ∼6% (2%) better, on average, for grayscale images and ∼14% (5%) better for natural images than IC-PNG (IC-Best). Similarly, "shaking" performed ∼7% (2%) better, on average, for grayscale images and ∼7% (-2%) better for natural images than IC-PNG (IC-Best). Surprisingly, IC-Best did not perform well with specific datasets, like LSUN/ID or CIFAR10/ID, on which our metrics, especially those based on "stirring", performed exceedingly well.

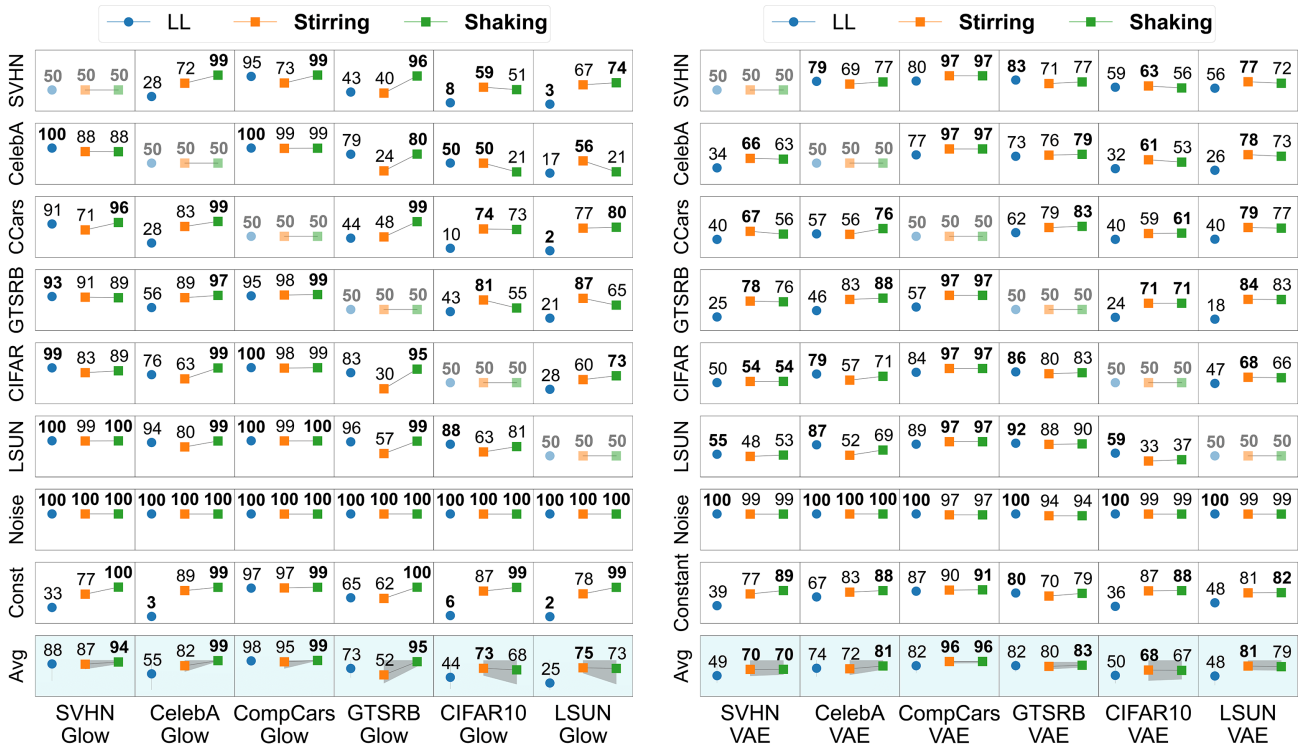
4.3 Time and Space Complexity

We also compared the time and space complexity of our methods with competing methods. We measured time complexity as the average per sample inference time for computing the OOD detection score. We quantified space complexity as the peak memory usage (mebibytes/MiB) during inference. Both metrics were computed with all the natural image datasets. Table. 1 shows these metrics for "stirring" and "shaking" alongside those for competing methods. Our methods do not require training additional background models and are competitive with state-of-the-art in terms of time and space complexity.

4.4 Results with Other Generative Models

Generative models come in at least three major flavors [Kingma and Dhariwal, 2018]: i) Autoregressive models, ii) Variational Autoencoders, and iii) Flow-based models.

While we have shown that our approaches work well with PixelCNN++, a model of the first category, we tested whether these approaches work with the other two models also: Generative flows [Kingma and Dhariwal, 2018] and Variational Autoencoders [Kingma and Welling, 2014].



(a) (b)

Figure 7: **Outlier detection performance (AUROC) with Glow and VAE. (a)** Outlier detection AUROC values for Glow models trained on natural image datasets. Conventions are the same as in Figure. 6. **(b)** Outlier detection AUROC values for VAEs trained on natural image datasets. Conventions are the same as in Figure. 6.

Method	Time / Sample (ms)	Peak Mem Usage (MiB)
Stirring	14.07	3142.76
Shaking	18.02	3289.24
LRat	7.95 + 2580.10*	3634.00
IC	4.69	3120.80

(*additional training time for background model)

Table 1: **Time and Space Complexity.** Time complexity (inference time per sample) and space complexity (peak memory usage during inference) for "stirring", "shaking", likelihood ratios (LRat) [Ren *et al.*, 2019], and Input Complexity (IC) [Serrà *et al.*, 2020].

Generative Flows

Flow-based models are a popular class of DGMs, as they enable computing exact log-likelihoods; this makes it a ready choice for OOD detection tasks. Yet, these models were also shown to assign higher likelihoods to OOD than ID images [Nalisnick *et al.*, 2019a].

We quantified OOD detection performance with natural image datasets using Glow log-likelihoods. These vanilla log-likelihoods (Fig. 7a, blue symbols) failed in several OOD detection cases. Like with PixelCNN++, the clearest failure cases occurred with CIFAR10/ID and LSUN/ID.

We then applied "shaking" and "stirring" to the Glow model likelihoods. In addition, we employed the conditional correction as specified in section 3.5 except for the following modification: because the training log-likelihoods were not normally distributed, we used the 99.5th percentile of the training data as a cutoff, instead of the 3-MAD criterion.

Both "stirring" (Fig. 7a, orange symbols) and "shaking" (Fig. 7a, green symbols) generally improved OOD detection performance. "Stirring" performed ∼17% better, on average, than vanilla log-likelihoods. Similarly, "shaking" performed ∼38% better, on average, than vanilla log-likelihoods.

Variational AutoEncoder

Variational Autoencoders [Kingma and Welling, 2014] are another class of DGMs that enable estimating sample likelihoods using variational inference, which renders these relevant for OOD detection tasks. VAEs are also considered unreliable for OOD detection tasks [Ren *et al.*, 2019; Nalisnick *et al.*, 2019a; Choi *et al.*, 2018; Xiao *et al.*, 2020; Chauhan *et al.*, 2022].

As before, we quantified OOD detection performance with vanilla log-likelihoods (Fig. 7b, blue symbols); several failure cases occurred. Again, we applied "shaking" and "stirring", as with the Glow models, except we also applied contrast stretching and a bias correction for the reconstruction error, following [Chauhan *et al.*, 2022]. These additional steps were necessary as "stirring" and "shaking" are not designed to rectify biases in the reconstruction error arising from the VAE visible distribution (see [Chauhan *et al.*, 2022] for details).

With this approach, OOD detection performance im-

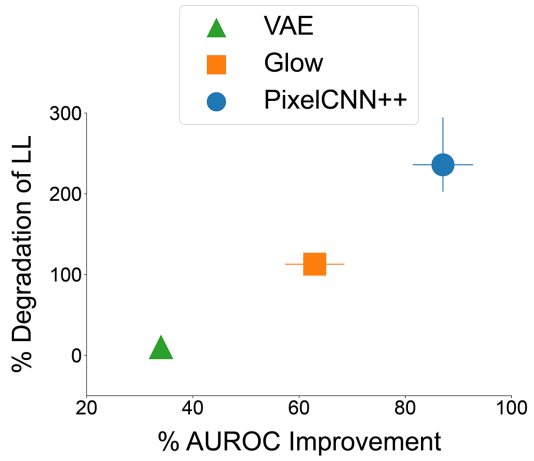


Figure 8: **Efficacy of "stirring" and "shaking" across different deep generative model classes.** Percentage improvement in AUROC (average of "stirring" and "shaking") over vanilla log-likelihood (x-axis) plotted against the % degradation in sample log-likelihood after perturbating a local neighborhood surrounding each pixel (CIFAR10/ID; see text for details). Error bars: range of performance improvements across "stirring" and "shaking" (x-axis) and 25th and 75th percentiles of log-likelihood degradation (y-axis). In some cases, the error bars are smaller than the symbol sizes.

proved. "Stirring" (Fig. 7b, orange symbols) performed ∼21% better, on average, than vanilla log-likelihoods. Similarly, "shaking" (Fig. 7b, green symbols) performed ∼24% better, on average, than vanilla log-likelihoods.

Analysis of Differential Efficacy across DGMs

Comparing improvements across DGM classes, we observed that our approaches were most effective with PixelCNN++ models, but comparatively less effective with Glow, and least effective with VAEs (Fig. 8, x-axis). We hypothesize that the efficacy of our approaches was higher for models that relied more on local dependencies for likelihood estimation.

To test this hypothesis, we replaced a local neighborhood (3x3 patch) surrounding a given pixel, with uniform random values (0-255); the replacement was performed once per pixel for each test image for the CIFAR10/ID dataset. With this perturbation, we quantified the degradation in sample log-likelihood, as a percentage relative to its original value. The degradation was greatest for PixelCNN++, followed by Glow and VAE in that order (Fig. 8, y-axis), reflecting the extent to which each model relied on local dependencies for predicting pixel values (VAE<Glow<PixelCNN++). In other words, the efficacy of our approaches was indeed governed by how much each model relied on local dependencies for assigning sample likelihoods.

5 Discussion

We developed simple, lightweight, and intuitive methods for state-of-the-art image outlier detection with PixelCNN++. Although we motivated our approaches with this particular class of deep generative model, we showed that they generalize to other classes of generative models, like Glow and VAEs, also. In addition, our methods may be relevant for outlier detection with other types of temporal data (e.g., speech), for which "stirring" and "shaking" can be readily performed, either by reversing the data in time or by chunking and shuffling the sequence.

Across all 72 ID/OOD comparisons with PixelCNN++, "stirring" performed noticeably worse than ceiling performance only with EMNIST/ID versus MNIST/OOD (AUROC of ∼82%), a not unreasonable failure given the feature similarity of MNIST and EMNIST samples. In fact, IC exhibited confusion for these two datasets also (MNIST/ID versus EMNIST/OOD, IC-Best AUROC ∼83%). On the other hand, while "shaking" performed well with most datasets, improvements were less impressive than those with "stirring", especially for natural ID image datasets like CIFAR10 and LSUN.

Our approaches could also be relevant for outlier detection with more recent autoregressive models. To our knowledge, such models (e.g., autoregressive transformers [Cao *et al.*, 2021]) have not been widely used for unsupervised outlier detection. Moreover, deep diffusion models require non-standard approximations for maximum likelihood training [Song *et al.*, 2021], precluding their widespread use in OOD detection. Future work will test whether biases due to local dependencies persist in these recent models also.

More generally, our results gainsay widespread claims in recent literature – that likelihoods from deep generative models are unreliable for outlier detection [Nalisnick *et al.*, 2019a; Ren *et al.*, 2019; Serrà *et al.*, 2020]. Rather, PixelCNN++ models are exquisitely sensitive to long-range dependencies in their training data, and isolating these dependencies, with simple geometric transformations, suffices to achieve robust outlier detection.

Acknowledgements

This work was conducted as part of a sponsored Google Research project. The authors wish to thank Dr. Manish Gupta for helpful discussions during the course of the study.

References

[Andrews *et al.*, 2016] Jerone Andrews, Edward Morton, and Lewis Griffin. Detecting anomalous data using auto-encoders. *International Journal of Machine Learning and Computing*, 6:21, 01 2016.

[Bergman and Hoshen, 2020] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2020.

[Cao *et al.*, 2021] Chenjie Cao, Yuxin Hong, Xiang Li, Chengrong Wang, Chengming Xu, Yanwei Fu, and Xiangyang Xue. The image local autoregressive transformer. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 18433–18445. Curran Associates, Inc., 2021.

[Chauhan *et al.*, 2022] Kushal Chauhan, Barath Mohan U, Pradeep Shenoy, Manish Gupta, and Devarajan Sridharan. Robust outlier detection by de-biasing vae likeli-

hoods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9881–9890, June 2022.

[Choi *et al.*, 2018] Hyunsun Choi, Eric Jang, and Alexander A. Alemi. WAIC, but Why? Generative Ensembles for Robust Anomaly Detection. *arXiv e-prints*, page arXiv:1810.01392, October 2018.

[Cohen *et al.*, 2017] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

[Deng, 2012] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[Hendrycks and Gimpel, 2017] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

[Johnson *et al.*, 2017] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910, 2017.

[Kingma and Dhariwal, 2018] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, 2014.

[Kirichenko *et al.*, 2020] Polina Kirichenko, Pavel Izmailov, and Andrew G Wilson. Why normalizing flows fail to detect out-of-distribution data. In *Advances in neural information processing systems*, volume 33, pages 20578–20589, 2020.

[Krizhevsky, 2009] Alex Krizhevsky. Cifar-10. https://www.cs.toronto.edu/~kriz/cifar.html, 2009. Accessed: 2023-05-29.

[Lakshminarayanan *et al.*, 2017] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[Liang *et al.*, 2018] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*, 2018.

[Liu *et al.*, 2015] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *2015 IEEE International Conference on Computer Vision, ICCV 2015*, pages 3730–3738, 2015.

[Nalisnick *et al.*, 2019a] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Görür, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[Nalisnick *et al.*, 2019b] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *arXiv preprint arXiv:1906.02994*, 2019.

[Netzer *et al.*, 2011] 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*, 2011.

[Ren *et al.*, 2019] Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[Ruff *et al.*, 2018] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402, 10–15 Jul 2018.

[Salimans *et al.*, 2017] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*, 2017.

[Serrà *et al.*, 2020] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F. Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *8th International Conference on Learning Representations, ICLR 2020, Conference Track Proceedings*, 2020.

[Sign Language MNIST, 2017] Sign Language MNIST. Sign language mnist. https://www.kaggle.com/datasets/datamunge/sign-language-mnist, 2017. Accessed: 2023-05-29.

[Song *et al.*, 2021] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 1415–1428. Curran Associates, Inc., 2021.

[Stallkamp *et al.*, 2011] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German Traffic

Sign Recognition Benchmark: A multi-class classification competition. In *IEEE International Joint Conference on Neural Networks*, pages 1453–1460, 2011.

[Van Den Oord *et al.*, 2016] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1747–1756, 2016.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Xiao *et al.*, 2020] Zhisheng Xiao, Qing Yan, and Yali Amit. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. In *Advances in Neural Information Processing Systems*, volume 33, pages 20685–20696, 2020.

[Yang *et al.*, 2015] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3973–3981, 2015.

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

[Yun *et al.*, 2019] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.