Out of Distribution Detection using Flow-based Gaussian Mixture Model

Rohith Mukku [* 1] Gyanesh Gupta [* 1] Aakanksha [* 1]

Abstract

Out-of-distribution (OOD) detection has become increasingly important with the burgeoning real-world applications of machine learning. One of the most popular methods for OOD detection is using Deep Generative Models (DGMs). In this work, we aim to investigate OOD detection using Normalizing Flows (NFs), a class of DGMs that have the capacity to learn exact likelihood distributions, by modeling the in-distribution data and out-distribution data as two different distributions explicitly, which we learn in a semi-supervised fashion. We show that our method performs well on OOD detection for various image datasets and further, compare our results with appropriate baselines. We also show results of our method when there is imbalanced data with abundant in-distribution data and scarce out-distribution data. Additionally, we analyze the performance on different ratios of labeled data and cases when the model sees unseen out-distribution data. The code for this work is available at https://github.com/rohithmukku/ML_Project

1. Introduction

With the ever-increasing integration of machine learning in everyday life, it has become very crucial to make machine learning based systems more reliable (DeBrusk & Wyman, 2020; Auernhammer et al., 2019). There is an implicit assumption that the test data follows the same distribution as the train data. However, it is likely that this might not always be the case in the real world. As such, being able to reliably identify these outliers or out-of-distribution (OOD) samples is key in increasing the robustness of any and all machine learning systems. These can be extremely helpful in cases where the model should not ideally be making any predictions for OOD data - for example, in case of autonomous driving, the model should not guess or perform seemingly random actions upon encountering out-of-distribution data lest accidents happen.

Generative models have gained significant popularity in the OOD detection landscape because of their ability to model the underlying data distribution and the direct use of that for detecting out-distribution samples. Standard methods involve using various statistics of distributions like likelihood (Ren et al., 2019), typical set (Choi et al., 2018) and distance metrics like KLOD (Zhang et al., 2020). However, Lan & Dinh (2020) prove that density may be limited in its use for anomaly detection since a transformation applied to a continuous random variable with strictly positive density can arbitrarily re-rank the density. To strengthen this claim, Zhang et al. (2021) proved that no test statistic guarantees accurate out-of-distribution detection when the in-data and out-data distributions overlap in support.

In case of out-data points not lying in the support of in-data, we do not face the above problems. This suggests the need to find a method to minimize the overlap between the in-data and out-data distribution. In fact, Bitterwolf et al. (2022) suggest one such approach to learn a binary discriminator trained using in-data and (different) out-data, which they trained in a shared fashion in order to improve generalization on unseen data. Izmailov et al. (2019) proposed FlowGMM, a semi-supervised learning-based approach with normalizing flows, demonstrated on a binary classification task by modeling densities in the latent space as a Gaussian mixture, with each mixture component corresponding to a class represented in the labeled data. In this paper, we build upon the ideas presented in the aforementioned works by utilizing normalizing flows for the problem of out-of-distribution detection. We tackle this task as a maximum likelihood-based classification problem by modeling in-distribution and out-distribution data as two separate classes and hypothesize finding an explicit separation between the two in the latent space of a flow network. To this end, we summarize our contributions as follows:

- We investigate the use of FlowGMM for OOD detection using complementary out-distribution datasets.
- We make use of the semi-supervised learning method and additionally, analyze OOD detection performance

[*]Equal contribution [1]Department of Computer Science, New York University. Correspondence to: Rohith Mukku <rm5708@nyu.edu>, Gyanesh Gupta <gg2501@nyu.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

when there is abundant in-distribution data with little out-distribution data.

- We analyze the effect of the ratio of labeled data to unlabeled data on our method.

This paper is organized as follows - we discuss prior work related to OOD detection and the use of normalizing flows in more detail in Section 2. In Section 3, we elaborate upon our approach for OOD detection based on normalizing flows with semi-supervision. Finally, we present and discuss our results in section 4 and highlight some limits and scope for future work in Section 5.

2. Related Work

Various methods have been explored in the OOD detection literature ranging from fairly straightforward thresholding techniques to more complex statistical tests – in both discriminative and generative settings. Hendrycks & Gimpel (2016) introduced the simple idea of Maximum Softmax Probability (MSP), where they tackle OOD detection as a classification problem by taking the maximum across the class probabilities. An extension of this method called ODIN was introduced by Liang et al. (2017) which brought in temperature scaling. It is, however, important to note that the optimal values for these are only identified by assuming prior access to OOD data. Some other works (Xia & Bouganis, 2022; Vyas et al., 2018) have also studied the benefits of ensembling multiple models.

Generative models have made a mark in this problem domain because of their ability to estimate the probability density of the input data where high densities specify in-distribution and low densities specify out-distributions. However, as identified by Nalisnick et al. (2018), deep generative models are susceptible to assigning higher likelihoods to out-distribution samples than to in-distribution. A number of methods have been proposed and investigated in order to mitigate this issue. Kirichenko et al. (2020) provide out-data samples during training so that the model learns to provide the low likelihood to those samples. Choi & Jang (2018) offered a model-independent approach that combines density-based anomaly detection with uncertainty estimation. Bitterwolf et al. (2022) treat OOD as a decision boundary learning problem and learns a combination of a binary discriminator between in- and out-distribution with a standard classifier on the in-distribution. Some other works (Ren et al., 2019; Serrà et al., 2019) study likelihood-ratio-based methods for DGMs. Hendrycks et al. (2018) introduces a modified training strategy that leverages the fact that data is available in abundance by exposing the detectors to out-distribution samples at training time.

Normalizing flows, a particular class of DGMs known for estimating the exact likelihood of distributions, have also been considerably explored for OOD detection. While various methods have investigated the use of normalizing flows in great detail, some with even practical applications (Ciușdel et al., 2022). Kirichenko et al. (2020) have also identified their shortcomings for this problem. We incorporate several suggestions presented by this work in our own in order to best utilize normalizing flows for detecting out-distribution samples.

While a good majority of these works turn to supervised learning for training their models, semi-supervised learning (SSL) based methods are also an exciting alternative. SSL aims to avoid the need for collecting prohibitively expensive labeled training data. Zhou et al. (2021) study semi-supervised OOD detection and present an approach that learns a new representation space in which OOD samples could be separated well by using a novel distance measure in a detection-specific space. Chen et al. (2020) address class distribution mismatch by deriving soft targets that serve as an indicator for OOD detection.

3. Method

In this section, we elaborate on the details of our approach and how we train our models in a semi-supervised manner. We start by expanding on the key components that our method is built on and derives inspiration from. We conclude this section by detailing out the scenario-dependent loss function(s) we use and our data selection strategy for our experiments.

3.1. Normalizing Flows

Normalizing flows are a pleasingly simple approach to generative modeling. They work by transforming a complex distribution, let's say one represented by a raw image, by transforming it to a target standard distribution like a Gaussian and back with the use of invertible transformations, which can also be modeled by neural networks. Since we are using simple transformations on the data distribution, whose inverses are known analytically, the exact likelihood can be found. In contrast, models such as Variational Autoencoders (VAEs) (Kingma & Welling, 2013) approximate a variational distribution and try to minimize the divergence with the data distribution. The network provides both useful inductive biases and a flexible approach to density estimation.

Formally, the mapping learned by a normalizing flow can be expressed as

$$f\colon X \to Z$$

from the input space X to the latent space Z. We can model the data distribution as the following transformation

$$f^{-1}\colon Z \to X$$

applied to a random variable from the latent distribution $z \sim p_z$, which is commonly chosen to be a Gaussian.

The density of the transformed random variable $x = f(z)$ can be computed using the Change of Variables formula and can be mathematically expressed as

$$p_X(x) = p_Z(f(x)) \cdot det\,(J_f(x)) \tag{1}$$

where J is the Jacobian matrix.
The mapping f is implemented as a composition of multiple invertible functions, i.e. $f = f_1 \circ f_2 \circ \cdots \circ f_k$, which are parameterized by a neural network with an architecture that is designed to ensure invertibility and efficient computation of log- determinants, and a set of parameters θ that can be optimized. The model can be trained by maximizing the likelihood in Equation 1 of the training data with respect to the parameters θ.

Normalizing flows also admit controllable latent representations and can be sampled efficiently, unlike auto-regressive models.

Why normalizing flows for OOD detection? As mentioned by Zhang et al. (2021), typical OOD methods tend to fail when supports of in-data distribution and out-data distribution overlap. We aim to learn these exact distributions explicitly to mitigate this drawback and normalizing flows is potentially a good fit for this problem. A great advantage of flows is the simplicity with which they can be trained, i.e., via a simple likelihood maximization method. An accurate bimodal density model should be better than an approximate one for a classification problem, which is the core of the approach that we are using. Furthermore, Kirichenko et al. (2020) show that if we explicitly train flows to distinguish between a pair of datasets, they can assign a large likelihood to one dataset and a low likelihood to the other.

Residual Flows

The main challenge associated with training flows is the efficient computation of determinants of functions under various transformations. Common approaches involve structurally restricting the transformation matrices but they lose expressiveness. Invertible residual networks or I-ResNets by Behrmann et al. (2018) are more expressive but tend to induce bias in their estimation of log density. Residual Flows by Chen et al. (2019) improve on these issues, by trading off between bias and expressiveness, through efficient computation of the Jacobians.

Consider a residual network and its inverse as

$$y = f(x) = x + g(x)$$
$$x = y - g^-1(x)$$

Assuming differentiable and invertible, using Change of Variables, log probability can be defined as

$$log(p(x)) = log(p(f(x)) + \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} tr(J_g(x)^k)$$

Using Skilling-Hutchinson trace estimator which gives $tr(M) = \mathbb{E}[v^T M v]$ where $v \sim \mathcal{N}(0, 1)$, we get

$$log(p(x)) = log(p(f(x)) + \mathbb{E}_v \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} v^t (J_g(x)^k) v$$

To estimate the above infinite series, the authors used the "Russian Roulette estimator", which says

$$\sum_{k=0}^{\infty} \Delta_k = \mathbb{E}_{n \sim P(n)} [\sum_{k=1}^{k=n} \frac{\Delta_k}{P(N \geq k)}]$$

Where $P(N \geq k)$ is the probability of getting atleast k heads in N coin tosses, where each toss has probability $q \sim Bernoulli(q)$. The above can be done in finite time. Our final log probability is

$$log(p(x)) = log(p(f(x)) + \mathbb{E}_{v,n} \sum_{k=1}^{n} \frac{(-1)^{k+1}}{k} \frac{v^t (J_g(x)^k) v}{P(N \geq k)}$$

3.2. FlowGMM

In real-world settings, the amount of labeled data that we have access to can be fairly limited, i.e., it can be very challenging to obtain labels for all data samples. This motivates the need to look into ways that allow for training normalizing flows for classification-like tasks in unsupervised or semi-supervised ways. To this end, we utilize the FlowGMM model introduced by Izmailov et al. (2019) that provides a semi-supervised way of estimating the exact joint likelihood over both labeled and unlabelled data for end-to-end training. This allows us to share structure over labeled and unlabelled data which drives better predictions of class labels on unseen data.

To elaborate upon this, a discrete latent variable y is introduced for the class label $y \in \{1 \dots C\}$ in FlowGMM. The latent space distribution, conditioned on a label k is a Gaussian with mean μ_k and covariance Σ_k:

$$p_Z(z|y = k) = \mathcal{N}(z|\mu_k, \Sigma_k) \tag{2}$$

The labels themselves can have a prior distribution $P(y = k)$, which is usually a Categorical distribution π_k. The marginal distribution of z is then a Gaussian mixture, written as

$$p_Z(z) = \sum_{k=1} \mathcal{N}(z|\mu_k, \Sigma_k)\pi_k$$

Therefore, we can obtain the likelihood for the labeled data using equations 1 and 2 as follows:

$$p_X(x|y = k) = \mathcal{N}(f(x)|\mu_k, \Sigma_k) \cdot \left| det\left(\frac{\partial f}{\partial x}\right)\right|$$

and the likelihood for data with unknown label is $p_X(x) = \sum_k p_X(x|y=k)p(y=k)$. Since we have access to labeled data $\mathcal{D}_l$ as well as unlabeled data $\mathcal{D}_u$ (we form our unlabeled dataset by not considering the labels for a bunch of samples in our labeled data), we can train our model in a semi-supervised way by maximizing the joint likelihood of the labeled and unlabeled data

$$p_X(\mathcal{D}_l, \mathcal{D}_u|\theta) = \prod_{(x_i,y_i)\in\mathcal{D}_l} p_X(x_i, y_i) \prod_{x_j\in\mathcal{D}_u} p_X(x_j) \quad (3)$$

over the parameters θ of the bijective function f, which learns a density model for a Bayes classifier. We can then make predictions for a test point by taking the $argmax$ of the predictive distribution in Equation 4.

$$\begin{aligned} p_X(x|y) &= \frac{p_X(x|y)p(y)}{p(x)} \\ &= \frac{\mathcal{N}(f(x)|\mu_y, \Sigma_y)\pi_y}{\sum_{k=1}^{C}\mathcal{N}(f(x)|\mu_k, \Sigma_k)\pi_k} \end{aligned} \quad (4)$$

3.3. Training Methods

In this subsection, we discuss the details of the loss function and the training strategy used.

Loss function: Consistency Regularization

In all of our experiments, to train our model in a semi-supervised manner, we minimize the negative log-likelihood of both the labeled and unlabeled data in conjunction with a consistency regularization term. This term computes the negative log-likelihood of a random perturbation of the input given the label on another random perturbation, we use the following loss term:

$$\begin{aligned} L_{cons}(x', x'') &= -logp(x'|y'') \\ &= -log\mathcal{N}(f(x')|\mu_{y''}, \Sigma_{y''}) - log\left|det\left(\frac{\partial f}{\partial x'}\right)\right| \end{aligned} \quad (5)$$

where x' and x'' are random perturbations of x and y'' is the label predicted on image x'' by FlowGMM.

This loss term encourages the model to map small perturbations of the same unlabeled inputs to the same components of the Gaussian mixture distribution in the latent space. Thus, the final loss is the weighted sum of the consistency loss specified in Equation 5 and the negative log-likelihood of both labeled and unlabeled data.

Label-Distribution-Aware Margin (LDAM) Loss

Our approach aims at providing just enough information, in the form of out-data distribution examples for a model to learn what is not the in-distribution. One way is to restrict the number of out-data examples. Because our model learns an explicit decision boundary, we could potentially suffer from common problems associated with class imbalance.

To solve this problem, we use Label-Distribution-Aware Margin Loss or LDAM by Cao et al. (2019). The idea is to set the margin of a classifier based on the margin of each class, weighted by the number of points in each class. This implies that the margin will shift towards the class with more examples. Mathematically, we define a generalization error bound, and try to minimize it. For 2 classes,

$$\frac{1}{\gamma_1\sqrt{n_1}} + \frac{1}{\gamma_2\sqrt{n_2}}$$

where γ is the margin and n is the number of examples in each class.

For optimal margin, we shift it by δ, then

$$\frac{1}{\gamma_1\sqrt{n_1}} + \frac{1}{\gamma_2\sqrt{n_2}} \geq \frac{1}{(\gamma_1-\delta)\sqrt{n_1}} + \frac{1}{(\gamma_2+\delta)\sqrt{n_2}}$$

The score is minimized when $\gamma^\star = \frac{C}{n^{-1/4}}$. The $\gamma^\star$ is incorporated in our loss as weights as,

$$\mathcal{L}_{LDAM}((x,y);f) = -log\frac{e^{z_y-\gamma^\star}}{e^{z_y-\gamma^\star} + e^{z_{1-y}}} \quad (6)$$

Data Selection

Since our method involves classifying a data point as either in-distribution or out-distribution, we need to specify the appropriate training data for these modalities. We use a set image datasets for determining the data distribution, i.e. a point is in-distribution if it is derived from a specific dataset like CIFAR and out-distribution otherwise. We could possibly define in-distribution on various levels of granularity. For instance, a dataset of birds and machines would be OOD with a dataset of plants.

Our aim is to ensure that our model correctly classifies in-data images, and at the same time generalizes to any out-data image with minimal amount of data provided to the model. Ideally, we would want to provide a wide range of possible data that could be out-data to improve generalization. But this would be a very difficult task. Naturally, there will be out-data datasets that are "close" to the in-data ones, and these would be harder to classify. We select these datasets, under the assumption that the easier out-data datasets would be classified correctly if the model learns from harder data. Additionally, we also want to control the fraction of labels and out-data our model sees, to understand their effect on our model.

4. Experiments/Results

In this section, we describe the various experiments we have conducted. We select the standard image datasets MNIST

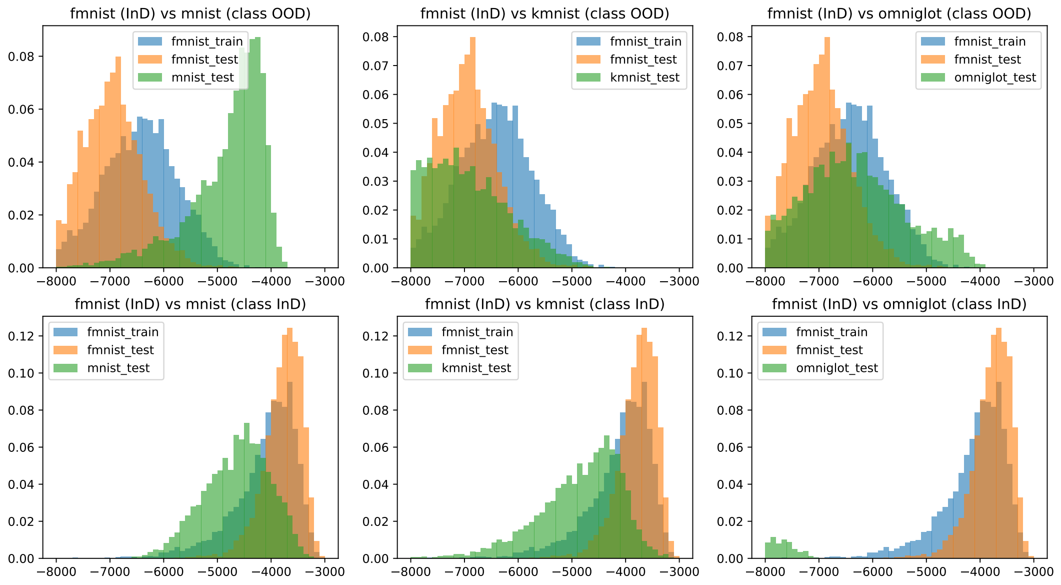

Figure 1. Log densities plot comparison. Left to Right (in-data vs out-data): FMNIST vs MNIST, FMNIST vs KMNIST, FMNIST vs Omniglot. Here top row indicates the plots for $p(x|y_{out})$ and bottom row indicates plots for $p(x|y_{in})$

(Deng, 2012), FashionMNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky, 2009), CIFAR100, SVHN (Netzer et al., 2011) and STL10 (Coates et al., 2011), as in-data, and out-data distributions as given below:

1. MNIST: FashionMNIST, KMNIST, Omniglot
2. FashionMNIST: MNIST, KMNIST, Omniglot
3. CIFAR10: CIFAR100, SVHN, STL10
4. CIFAR100: CIFAR10, SVHN, STL10
5. SVHN: CIFAR10, CIFAR100, STL10

We choose these sets of data because MNIST, FashionMNIST, KMNIST, Omniglot are similar to each other or can be said near OOD datasets. Similarly, CIFAR10, CIFAR100, SVHN, STL10 are near OOD datasets. This makes OOD detection difficult and we aim to investigate our method's performance on these.

Our experiments are as follows:

Experiment 1: We test our method with a held-out fraction of the above-used in-data and out-data which is 10,000 samples of out-data per dataset (total 3 datasets) and 30,000 samples of in-data.

Experiment 2: We test out method with imbalanced data with 30,000 in-data and approximately 2,000 samples of out-data per dataset (total 6,000 samples).

Experiment 3: We analyze the model's performance on different ratios of labeled data. We vary the amount of labeled data ranging from 1% of the dataset to 20% of the dataset. This is done for both the cases: balanced and imbalanced datasets.

Experiment 4: This experiment is a modification of experiment 2, we test our models on a different out-data dataset, which was not used to train the dataset, to demonstrate how our model generalizes to the out-data examples. These results are available in the Github repository.

The model is trained for 20 epochs with training accuracy reaching 96%. We benchmark our model experiments with popular baselines such as MSP and ODIN with OOD metrics such as AUROC (Bradley, 1997), AUPRC, FPR@95%.

Baseline: MSP and ODIN methods are the baselines with ResNet-50 (He et al., 2016) as the model.

4.1. Results

Results for experiment 1 are shown in Table 1 and we can see that our method performs well for cases with MNIST, FashionMNIST, SVHN as in-distribution. In the case of CIFAR10, CIFAR100, STL10; we see that baselines perform better than our method.

We show the results for experiments 2, 3 in the Appendix

Table 1. Baseline vs our method (equal amount of in-data and out-data with 10% of labeled data)

InD	OOD	AUROC			AUPRC			FPR95		
		MSP	ODIN	Ours	MSP	ODIN	Ours	MSP	ODIN	Ours
MNIST	FMNIST	0.95	0.95	**0.96**	0.96	0.97	**0.98**	0.22	**0.16**	0.3
	KMNIST	0.94	**0.95**	0.94	**0.95**	**0.95**	**0.95**	0.31	**0.29**	0.34
	Omniglot	0.78	0.61	**0.96**	0.58	0.96	**0.98**	0.66	0.87	**0.29**
FMNIST	MNIST	0.91	0.92	**0.93**	0.93	0.93	**0.94**	0.12	**0.09**	0.11
	KMNIST	0.81	**0.83**	0.74	0.89	**0.91**	0.88	0.39	**0.31**	0.48
	Omniglot	0.77	0.82	**0.99**	0.77	0.81	**0.99**	0.66	0.5	0.01
CIFAR-10	CIFAR-100	0.54	**0.57**	0.52	0.85	**0.87**	0.59	**0.92**	0.93	0.94
	SVHN	0.84	**0.98**	0.65	0.92	**0.99**	0.81	0.58	**0.07**	0.93
	STL10	0.31	0.15	**0.46**	0.33	0.28	**0.53**	0.98	1.00	**0.95**
CIFAR-100	CIFAR-10	0.45	0.42	**0.52**	0.14	0.13	**0.58**	0.97	0.96	**0.94**
	SVHN	0.8	**0.96**	0.52	0.68	**0.93**	0.76	0.63	**0.14**	0.94
	STL10	0.27	0.12	**0.45**	0.09	0.08	**0.39**	0.99	0.99	**0.95**
SVHN	CIFAR-10	0.95	0.91	**0.96**	0.96	0.95	**0.97**	0.05	0.1	**0.03**
	CIFAR-100	**0.95**	**0.95**	**0.95**	0.95	0.95	**0.96**	**0.04**	0.05	0.05
	STL10	0.98	**0.99**	0.98	**0.99**	**0.99**	**0.99**	0.02	**0.01**	**0.01**

Table 2. in-distribution and different out-distribution data when trained with varying portions of dataset being labeled

InD	OOD	1%	5%	10%	20%
MNIST	FMNIST	0.94	0.97	0.96	0.95
	KMNIST	0.9	0.91	0.94	0.92
	Omniglot	0.95	0.96	0.96	0.93
SVHN	CIFAR10	0.74	0.94	0.96	0.97
	CIFAR100	0.74	0.93	0.95	0.97
	STL10	0.78	0.95	0.98	0.98

section.

We also present the log densities plot as shown in Figure 1. We can see that out-distribution data has higher log density conditioned on out-class ($\log p(x|y_{out})$) and lower log density conditioned on in-class ($\log p(x|y_{in})$).

The results of experiment 3 are shown in Table 2. Model is trained on 30k samples of in-distribution and 30k samples of out-distribution (10k per dataset) with varying portions of data being labeled. We investigate for portions of 1% (600 labeled out of 60k), 5% (3k labeled out of 60k), 10% (6k labeled out of 60k), 20% (12k labeled out of 60k). Table 2 corresponds to SVHN as in-distribution. We see that 10% of the data being labeled gives a reliable performance. This shows the effectiveness of semi-supervised learning and can be useful for scenarios where there isn't much labeled data.

5. Discussion

We have shown that FlowGMM can be a good alternative for OOD detection. We have also analyzed the portion of data needed to be labeled for a good performance. This method can be really helpful in cases where there is a dearth of labeled data. Consistency loss regularization helps classify similar images which in turn is beneficial in learning to distinguish in-distribution images from out-distribution.

We also demonstrated that for SVHN, MNIST, FMNIST as in-data, we are getting good results for any pair of out-data datasets. The test out-data include the ones seen in the training dataset as well as other datasets. We are also able to classify easier examples well, given we trained on the harder examples.

For instance, if our in-data is MNIST, our hard examples would be FMNIST and KMNIST as all three are monochromatic and KMNIST also contains symbols, just like MNIST. The model performs well on Omniglot, which also contain symbols, as well as colored datasets like CIFAR. We observe that our model failed to work with CIFAR as in-data, which is true for all other baselines. One reason, as pointed out by Lan & Dinh was that CIFAR has less sharp edges, given that it contains objects and not numbers.

5.1. Limitations and Future Work

The training method majorly revolves around having unsupervised training with loss consistency regularization along with supervised loss. But there is inherent motivation for the model to push the out-distribution sample towards out-class when there unlabeled samples. A loss function that can help learn the model separate the in-distribution and out-distribution image representations can one way to tackle this problem.

References

Auernhammer, K., Kolagari, R. T., and Zoppelt, M. Attacks on machine learning: Lurking danger for accountability. In *SafeAI@AAAI*, 2019.

Behrmann, J., Grathwohl, W., Chen, R. T. Q., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. 2018. doi: 10.48550/ARXIV.1811.00995. URL https://arxiv.org/abs/1811.00995.

Bitterwolf, J., Meinke, A., Augustin, M., and Hein, M. Breaking down out-of-distribution detection: Many methods based on ood training data estimate a combination of the same core quantities. In *International Conference on Machine Learning*, 2022.

Bradley, A. P. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognit.*, 30:1145–1159, 1997.

Cao, K., Wei, C., Gaidon, A., Aréchiga, N., and Ma, T. Learning imbalanced datasets with label-distribution-aware margin loss. In *Neural Information Processing Systems*, 2019.

Chen, R. T. Q., Behrmann, J., Duvenaud, D., and Jacobsen, J.-H. Residual flows for invertible generative modeling, 2019. URL https://arxiv.org/abs/1906.02735.

Chen, Y., Zhu, X., Li, W., and Gong, S. Semi-supervised learning under class distribution mismatch. In *AAAI Conference on Artificial Intelligence*, 2020.

Choi, H.-J. and Jang, E. Generative ensembles for robust anomaly detection. *ArXiv*, abs/1810.01392, 2018.

Choi, H.-J., Jang, E., and Alemi, A. A. Waic, but why? generative ensembles for robust anomaly detection. *arXiv: Machine Learning*, 2018.

Ciușdel, C. F., Itu, L. M., Cimen, S., Wels, M., Schwemmer, C., Fortner, P., Seitz, S., Andre, F., Buß, S. J., Sharma, P., and Rapaka, S. Normalizing flows for out-of-distribution detection: Application to coronary artery segmentation. *Applied Sciences*, 12(8), 2022. ISSN 2076-3417. doi: 10.3390/app12083839. URL https://www.mdpi.com/2076-3417/12/8/3839.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In Gordon, G., Dunson, D., and Dudík, M. (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/coates11a.html.

DeBrusk, C. and Wyman, O. The risk of machine learning bias (and how to prevent it). 2020.

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

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ArXiv*, abs/1610.02136, 2016.

Hendrycks, D., Mazeika, M., and Dietterich, T. G. Deep anomaly detection with outlier exposure. *ArXiv*, abs/1812.04606, 2018.

Izmailov, P., Kirichenko, P., Finzi, M., and Wilson, A. G. Semi-supervised learning with normalizing flows. *ArXiv*, abs/1912.13025, 2019.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes, 2013. URL https://arxiv.org/abs/1312.6114.

Kirichenko, P., Izmailov, P., and Wilson, A. G. Why normalizing flows fail to detect out-of-distribution data. *ArXiv*, abs/2006.08545, 2020.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Lan, C. L. and Dinh, L. Perfect density models cannot guarantee anomaly detection. *Entropy*, 23, 2020.

Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv: Learning*, 2017.

Nalisnick, E. T., Matsukawa, A., Teh, Y. W., Görür, D., and Lakshminarayanan, B. Do deep generative models know what they don't know? *ArXiv*, abs/1810.09136, 2018.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Reading digits in natural images with unsupervised feature learning. 2011.

Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., DePristo, M. A., Dillon, J. V., and Lakshminarayanan, B. Likelihood ratios for out-of-distribution detection. *ArXiv*, abs/1906.02845, 2019.

Serrà, J., Álvarez, D., Gómez, V., Slizovskaia, O., Núñez, J. F., and Luque, J. Input complexity and out-of-distribution detection with likelihood-based generative models. *ArXiv*, abs/1909.11480, 2019.

Vyas, A., Jammalamadaka, N., Zhu, X., Das, D., Kaul, B., and Willke, T. L. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *European Conference on Computer Vision*, 2018.

Xia, G. and Bouganis, C.-S. On the usefulness of deep ensemble diversity for out-of-distribution detection. *ArXiv*, abs/2207.07517, 2022.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Zhang, L. H., Goldstein, M., and Ranganath, R. Understanding failures in out-of-distribution detection with deep generative models. *Proceedings of machine learning research*, 139:12427–12436, 2021.

Zhang, Y., Liu, W., Chen, Z., Wang, J., Liu, Z., Li, K., Wei, H., and Chen, Z. Out-of-distribution detection with distance guarantee in deep generative models. *ArXiv*, abs/2002.03328, 2020.

Zhou, Z., Guo, L.-Z., Cheng, Z., Li, Y.-F., and Pu, S. Step: Out-of-distribution detection in the presence of limited in-distribution labeled data. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29168–29180. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/f4334c131c781e2a6f0a5e34814c8147-Paper.pdf.

A. Results for Experiments 2, 3

Here, Table 3 compares results when there is balanced dataset and when there is class imbalance,

- **Exp-1:** corresponds to experiment - 1 (equal amount of in-distribution and out-distribution).
- **Exp-2:** corresponds to experiment - 2 (30k samples of in-distribution and 6k total samples of out-distribution).

Table 3. Comparison of performance when the model is trained on equal amount of data and in the other case, equal amounts of data is not available

InD	OOD	AUROC		AUPRC		FPR95	
		Exp-1	Exp-2	Exp-1	Exp-2	Exp-1	Exp-2
MNIST	FMNIST	0.96	0.98	0.98	0.98	0.03	0.01
	KMNIST	0.94	0.94	0.95	0.95	0.34	0.09
	Omniglot	0.96	0.98	0.98	0.99	0.29	0.01
FMNIST	MNIST	0.93	0.97	0.94	0.98	0.11	0.02
	KMNIST	0.74	0.82	0.88	0.86	0.48	0.32
	Omniglot	0.99	0.99	0.99	0.99	0.01	0.01
CIFAR-10	CIFAR-100	0.52	0.52	0.59	0.58	0.94	0.94
	SVHN	0.65	0.62	0.81	0.80	0.93	0.93
	STL10	0.46	0.46	0.53	0.52	0.95	0.95
CIFAR-100	CIFAR-10	0.52	0.53	0.58	0.6	0.94	0.94
	SVHN	0.52	0.64	0.76	0.8	0.94	0.93
	STL10	0.45	0.4	0.5	0.5	0.95	0.96
SVHN	CIFAR-10	0.96	0.96	0.97	0.97	0.03	0.04
	CIFAR-100	0.95	0.95	0.96	0.96	0.05	0.07
	STL10	0.98	0.98	0.99	0.99	0.01	0.01

Additionally, regarding experiment - 3:

Table 4. in-distribution and different out-distribution data when trained with varying portions of dataset being labeled

InD	OOD	1%	5%	10%	20%
FMNIST	MNIST	0.57	0.92	0.93	0.98
	KMNIST	0.6	0.77	0.74	0.85
	Omniglot	0.99	0.99	0.99	0.90
CIFAR10	CIFAR100	0.5	0.5	0.52	0.51
	SVHN	0.47	0.66	0.65	0.64
	STL10	0.47	0.31	0.46	0.48
CIFAR100	CIFAR10	0.51	0.51	0.52	0.52
	SVHN	0.52	0.51	0.52	0.58
	STL10	0.48	0.35	0.45	0.44