Exploring Adversarial Training for Out-of-Distribution Detection

Irena Gao
igao@stanford.edu

Ryan Han
ryanhan@stanford.edu

David Yue
davidyue@stanford.edu

1. Introduction

For any classifier to behave reliably in the real world, it is important for the algorithm to recognize when it encounters an unusual example outside of its known input distribution. Unusual examples tend to cause classifiers to fail; worse, misclassifications are still made with high confidence [6, 9]. These **out-of-distribution (OOD)** examples may be close to normal data (*e.g.* blurry samples, adversarially attacked inputs), or they may belong to a completely unknown class that the classifier is not trained to predict (*e.g.* a novel disease). OOD detection is the first step in building classifiers that "fail gracefully" – *i.e.* classifiers able to learn from shifting distributions in deployment – and is seen as critical to AI safety [20, 1].

We formulate this problem as a binary classification problem $\mathcal{B}$ on top of the normal classification task. Given an example x, we say that $\mathcal{B}(x) = 1$ (x is *in-distribution*) if x is drawn from the same distribution as the training set. On the other hand, $\mathcal{B}(x) = 0$ (x is *out-of-distribution*) if x is drawn from a different distribution, such as the space of adversarially-perturbed inputs or a total outlier distribution. OOD examples are distinct from noisy in-examples – for example, Gaussian noise applied to the in-distribution should not be classified as OOD.

We adopt the hypothesis that the in-distribution lies on a low-dimensional surface in high-dimensional feature space, called the **data manifold** (Figure 1). On the data manifold, low-density valleys separate clusters of classes. OOD examples lie off the manifold. An OOD classifier $\mathcal{B}$ should attempt to learn the shape of the manifold boundary to make classifications.

Following [14], we hypothesize that the data manifold can be approximated by a series of class-conditional Gaussian distributions. During training, $\mathcal{B}$ approximates learning the manifold by memorizing Gaussian parameters μ_c, σ_c for each class. At test time, $\mathcal{B}(x) = 1$ if x is close to a class-conditional Gaussian, and $\mathcal{B}(x) = 0$ if x is far from any class-conditional Gaussian, where "close" and "far" are measured by the probability-based **Mahalanobis distance**. Mahalanobis distances are separated into binary decisions using a simple threshold. Because, the effectiveness of this method changes dramatically with the analyzed feature space, supervised learning is used to select the neural network layer and threshold that maximizes $\mathcal{B}$'s accuracy.

This step poses a contradiction. The premise of the OOD problem is that the classifier does not have *a priori* knowledge of what OOD examples it will encounter. Gathering out-examples is both data-intensive and counterproductive – at this point, one might as well retrain the classifier with an explicit "unknown" class [23].

In this project, we explore methods to translate Mahalanobis scores to predictions without the use of out-datasets. Following a suggestion by [14], we hypothesize that **adversarial examples** lie very near but just off the data manifold along particularly uncertain axes [5, 16]. This suggests that training $\mathcal{B}$ using adversarial examples provides a tight picture of the manifold boundary. We examine the effect of training layer selection using various adversarial examples and compare this to (a) only maximizing in-example accuracy and (b) training on a massive out-dataset. Our work suggests that adversarial training, *i.e.* on DeepFool-attacked in-examples, generalizes $\mathcal{B}$ effectively to OOD scenarios.

2. Related Work

The OOD detection problem is closely related to adversarial detection [14, 5, 16, 7] and has applications in class-incremental and active learning [19]. We draw from a mix of this literature.

2.1. Supervised OOD Detection

A baseline approach for OOD Detection would be to look at a deep neural network's softmax scores and classify $\mathcal{B}(x) = 0$ if the scores are low-confidence, *i.e.* beneath a threshold . However, softmax-based neural networks have been shown to be overconfident on OOD examples [6, 9], making this an open problem. Several suggestions to resolve this problem attempt to stratify confidence at the softmax level [15, 10]. This is done by training the classifier on the in-dataset χ_{in} and a second distribution of out-examples χ_{out}.

The downside of these methods is the requirement of the second train-time dataset χ_{out}. The premise of the OOD problem is that the classifier does not have train-time knowledge of what out-distributions it will encounter. Addition-

ally, the choice of out-dataset affects $\mathcal{B}$'s accuracy – $\mathcal{B}$ performs better on out-distributions that are similar to the training out-dataset. Because out-examples so outnumber in-examples, it is unlikely that any one out-dataset can represent the entire OOD space. As a solution, [10] propose standardizing one massive out-dataset to share across classifiers (80 Million Tiny Images [22]). Gathering out-examples appears counterproductive – at this point, one might as well retrain the classifier with an explicit "unknown" class, with good accuracy [23]. Additionally, [3] demonstrate that this outlier exposure method is vulnerable to adversarial attacks. So long as the out-dataset is a known distance from the data manifold boundary, adversarial examples near the boundary will be misclassified.

2.2. The Mahalanobis Method

Thus we advocate for methods that do not rely on a train-time out-dataset. These unsupervised methods focus on modeling the manifold on which in-examples lie. Of particular note is the Mahalanobis Method, proposed by [14].

Let $f_\ell(x)$ be a neural network's features at layer ℓ. Then the authors approximate the data manifold of ℓ using a series of class-conditional Gaussian distributions $f_\ell(x)|_{y=c} \sim \mathcal{N}(\mu_\ell^c, \Sigma_\ell^c)$. Given example x, we can measure in-distribution confidence based on the Mahalanobis distance of x to the closest class-conditional Gaussian. The Mahalanobis confidence M_ℓ is the negative distance Mahalanobis distance.

$$M_\ell(x) = \max_c -(f_\ell(x) - \mu_\ell^c)^T {\Sigma_\ell^c}^{-1} (f_\ell(x) - \mu_\ell^c) \quad (1)$$

During training, $\mathcal{B}$ approximates learning the manifold by memorizing Gaussian parameters $\mu_\ell^c, \Sigma_\ell^c$ for each class in feature space ℓ. At test time,

$$\mathcal{B}(x) = \begin{cases} 1 & \sum_\ell \alpha_\ell M_\ell(x) \geq \tau \\ 0 & \text{o.w.} \end{cases} \quad (2)$$

where τ is some threshold and α is a vector that weighs the Mahalanobis scores of each layer ℓ. Because the effectiveness of score M_ℓ changes dramatically with ℓ, supervised learning with datasets χ_{in}, χ_{out} is used to select the weights α so as to maximize $\mathcal{B}$'s accuracy. The original authors use out-dataset SVHN for in-datasets CIFAR-10, CIFAR-100, and vice versa for in-dataset SVHN. An application of Mahalanobis in [3] uses 80 Million Tiny Images as the out-dataset, as proposed by [10]. We worry that this supervised step raises the same prior concerns about specifying an out-dataset, and [3] show that this step is vulnerable to adversarial attack.

2.3. Adversarial Training

As an alternative to collecting a new out-dataset, the Mahalanobis authors briefly suggest replacing χ_{out} with χ_{in}^A,

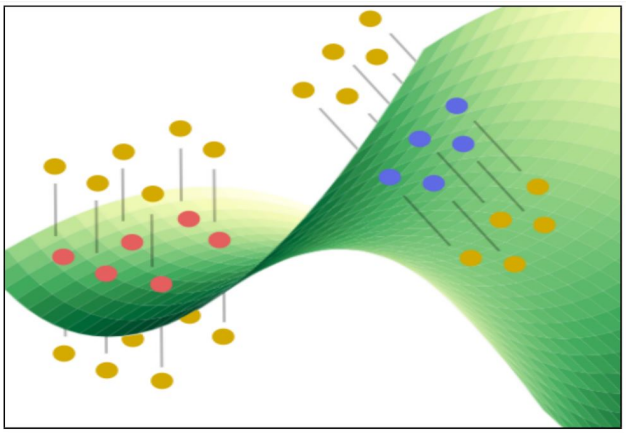

Figure 1. Schematic of the data manifold. In-data (red, blue) lie on a low-dimensional surface in high-dimensional feature space. Classes (red vs. blue) are separated by low-density areas. Adversarial data (yellow) lies just off the data manifold along orthogonal axes. Figure from [23].

or some adversarial perturbation applied to in-dataset χ_{in}. This reduces the data-intensity of training $\mathcal{B}$.

We argue that this choice is also theoretically justified. Work in the adversarial attack literature by [5, 16] suggest that adversarial examples lie very near, but just off, the data manifold along particularly uncertain axes (Figure 1). Thus adversarial examples give a tighter approximation of the manifold boundary than outlier datasets, which are far from the manifold. A study by [4] reinforces the value of adversarial training; they show that training on tight out-examples (in this case, "bad" in-examples produced by a generative model) produces sharper models of the manifold boundary than seeing more in-examples. Adversarial training has been explored in the OOD literature by the authors of ALOE [3], a supervised approach that trains on adversarially-perturbed in-examples, and by [13], who use a Generative Adversarial Network-esque approach that trains $\mathcal{B}$ on increasingly tighter out-examples.

3. Methods

In this project, we explore the effect training the Mahalanobis's method supervised step using different datasets. We implement the Mahalanobis method as detailed in [14]. We run a series of experiments parameterized by an in-dataset χ_{in} and a training set χ_α that supervises the learning of layer weights α. We consider the effect of three setups for χ_α:

- $\chi_\alpha = \chi_{in} \cup \chi_{out}$. In this baseline case, one out-dataset χ_{out} is combined with χ_{in}. α is learned by logistic regression to maximize the true positive rate (TPR).
- $\chi_\alpha = \chi_{in} \cup \chi_{in}^A$. In our experimental case, an adversarial attack is applied to χ_{in}. This perturbed dataset

χ_{in}^A is then treated as an out-dataset. α is learned by logistic regression to maximize the true positive rate (TPR).

We take a moment to clarify our terminology: an OOD example may be either an adversarial example or an example from an out-dataset. An adversarial distribution is derived from the in-distribution, whereas an out-dataset is a disjoint dataset taken from the web.

3.1. Datasets

We evaluate two in-datasets: CIFAR-10 [11] and the Street View House Numbers Dataset (SVHN) [18]. Each in-dataset is paired with five training scenarios: four adversarial attacks (see 3.2) and training on large out-dataset TinyImageNet [21]. Each in-dataset / train-dataset pair is evaluated on two tasks: detecting a batch of four adversarial attacks and detecting a batch of four datasets: CIFAR-10, SVHN, TinyImageNet, and LSUN [24]. In all cases, we deduplicate any shared classes between in- and out-examples as per [9].

3.2. Adversarial Perturbations

We consider four varieties of adversarial perturbations:

1. **Fast Gradient Sign Method** (FGSM) [6], which steps input $x \in \chi_{in}$ in the direction of classifier loss $J(x)$ subject to L_∞ constraint.

$$x^A = x + \epsilon \text{ sign}(\nabla J(x)) \tag{3}$$

2. **Basic Iterative Method** (BIM) [12], which repeats FGSM with intermediate clipping of pixel values to constrain perturbations with an ϵ-ball.

$$x^A := \text{clip}_\epsilon[x^A + \alpha \text{ sign}(\nabla J(x^A))] \tag{4}$$

3. **DeepFool** [17], which iteratively projects x onto the adversarial subspace, which is orthogonal to the class manifold. We refer to the original work for details.

4. **Carlini-Wagner L2 Attack** (CWL-2) [2], which optimizes x^A by jointly penalizing model accuracy and a distance measure for L2 regularization. We refer to the original work for details.

3.3. Classifier

We evaluate the Mahalanobis method on a ResNet-34 deep neural network. In deep residual nets (ResNets), introduced by [8], inputs to blocks of layers are re-added to the block outputs. These shortcut connections improve gradient flow and allow for greater depth, providing state of the art performance. ResNet-34 consists of 34 layers where shortcut connections are added to each pair of convolutional layers. We use a modified implementation of ResNet-34 based on the Torchvision model library.

4. Results

4.1. Evaluation Metrics

We compare each in-dataset / train-dataset / test distribution setup using two metrics: the True Negative Rate at 90% True Positive Rate (TNR @ 90 TPR) and the Area Under the ROC Curve (AUROC). Our results are given in Table 1.

4.2. Trends in Detecting OOD Examples

We evaluated training on four adversarial attacks and one large out-dataset. There is some difference in a given setup's performance between the two in-datasets, CIFAR-10 and SVHN.

When the in-dataset is CIFAR-10, the highest-performing setup trains on DeepFool, which achieves AUROC ≈ 0.8 for detecting both adversarial attacks and out datasets. This is followed by training on the out-dataset TinyImageNet, and then by training on FGSM. While BIM and CWL-2 show good performance detecting adversarial examples (AUROC ≈ 0.7), they fail to detect out-datasets (AUROC ≈ 0.5).

When the in-dataset is SVHN, the highest-performing setup trains on TinyImageNet, which achieves AUROC ≈ 0.86 for detecting adversarial attacks and AUROC ≈ 0.99 for detecting out-datasets. This is followed by FGSM, which achieves similar scores. While DeepFool and CWL-2 attain very good performance detecting adversarial attacks (AUROC ≈ 0.9), they fail to detect out-datasets (AUROC < 0.5).

First, we note that our results are very variable, with some setups resulting in very good performance (AUROC > 0.9) and others in very poor performance (AUROC < 0.5).

Second, we note that this variability follows some trends.

- Training on FGSM and TinyImageNet attain more stable performance between the two detection tasks than training on BIM, DeepFool, and CWL-2.
- Generally, training on BIM, DeepFool, and CWL-2 results in much lower out-dataset accuracy than adversarial detection accuracy.
- DeepFool and CWL-2 consistently perform well in detecting adversarial attacks, especially in the SVHN case.
- The in-distribution also has an effect. Results from SVHN are more stratified than for CIFAR-10: the highs are higher and the lows are lower. We note

In-Distribution	Training Distribution	Test Distribution	Validation on Test Distribution	
			TNR @ 90% TPR	AUROC
CIFAR-10	FGSM	Adversarial Attacks	0.4077	0.6978
		Out Datasets	0.4801	0.6161
	BIM	Adversarial Attacks	0.3645	0.7122
		Out Datasets	0.2987	0.4485
	DeepFool	Adversarial Attacks	0.4008	**0.7807**
		Out Datasets	0.6253	**0.8051**
	CWL-2	Adversarial Attacks	0.3409	0.7167
		Out Datasets	0.3552	0.5808
	TinyImageNet	Adversarial Attacks	**0.4210**	0.7332
		Out Datasets	**0.6407**	0.7246
SVHN	**FGSM**	Adversarial Attacks	0.5856	0.8461
		Out Datasets	**0.9272**	**0.9520**
	BIM	Adversarial Attacks	0.6823	0.8758
		Out Datasets	0.2014	0.2729
	DeepFool	Adversarial Attacks	**0.7436**	**0.9095**
		Out Datasets	0.4405	0.5186
	CWL-2	Adversarial Attacks	**0.7429**	**0.9070**
		Out Datasets	0.3250	0.3876
	TinyImageNet	Adversarial Attacks	0.6206	0.8622
		Out Datasets	**0.9957**	**0.9957**

Table 1. True Negative Rate at 90% True Positive Rate and Area Under the Received Operating Characteristic scores for five training sets (adversarial sets FGSM, BIM, DeepFool, CWL-2, and out-dataset TinyImageNet) for two in-datasets (CIFAR-10 and SVHN). Validation was performed on either Out Datasets (CIFAR-10, SVHN, TinyImageNet, LSUN) or Adversarial Attacks (FGSM, BIM, DeepFool, CWL-2). Exceptionally high scores are bolded, and the best overall training-sets that attain good performance on both adversarial and out-dataset detections are bolded.

that performance overall is stronger on SVHN than CIFAR-10.

4.3. Discussion

We hypothesized that training on adversarial examples would consistently enable high performance on detecting both adversarial examples (which are close to the data manifold) and out-dataset examples (which are far from the manifold). Our results seem to suggest that this is not always true.

First, when there is a large discrepancy between detecting adversarial examples and out-dataset examples, detecting adversarial examples seems to be easier, not harder than detecting out-dataset examples. This does not fit our simple picture of adversarial examples being closer to the manifold boundary and thus harder to detect.

We propose the following explanation, informed by [23]: **our four adversarial attacks perturb in-examples along similar axes, generating an incomplete picture of the manifold boundary**. In this scenario, the axes along which adversarial examples lay would have clear manifold boundaries, but the rest of the manifold would be a massive blind spot. Thus, detecting adversarial attacks along the well-defined axes is easier than detecting examples on other axes.

Under this hypothesis, the inconsistent performance of BIM, DeepFool, and CWL-2 is due to their concentration of attack along one direct axis, whereas FGSM perturbs in-examples more uniformly along different axes.

It also appears that the distance between adversarial examples and examples from TinyImageNet is not too large — training on TinyImageNet gives nearly as good a picture of the boundary as DeepFool in the CIFAR-10 case, and an even better picture in the SVHN case. We do note that the comparable performance betwen training on FGSM and TinyImageNet suggests that FGSM is a good substitute for training on a large out-dataset when out-datasets are not available.

5. Conclusion

In this project, we examined adversarial training within the supervised optimization of the Mahalanobis method for OOD Detection. We hypothesized that training on adversarial examples, which lie very near but off the data manifold, would provide a tight picture of the manifold boundary that generalizes well to OOD detection.

Our results indicate that this statement heavily depends on the type of adversarial attack: some adversarial attacks

(BIM, DeepFool, CWL-2) concentrate points along one axis, whereas other attacks (FGSM) give a more complete picture of the manifold.

We also conclude that training on FGSM gives comparable performance to training on a large out-dataset (*e.g.* Tiny-ImageNet).

Our study is limited in that it raises more theoretical questions than it answers. The variable performance between setups suggests that underlying attack dynamics matter — we cannot make a blanket statement that adversarial training always benefits OOD detection. We include our first attempts to confirm this hypothesis in the Appendix using t-SNE visualizations, though we were not able to make conclusions from this.

For future work, we wonder whether adversarial training could be integrated more effectively earlier in the Mahalanobis method. We imagine a scenario where adversarial distributions are also memorized during training and affect Mahalanobis scores directly, instead of simply being used to optimize layer ensembling.

As a critical problem in AI Safety, OOD detection remains an open concern. We remain interested in this research question and we hope that our work might contribute to this project.

6. Appendix

In Figure 2, we ran a visualization method, t-SNE, to check our intuition that the adversarial attacks may not cover the in-data manifold. However, we were unsure of how to interpret these plots because of confusion on how accurately this very-compressed space relates back to feature space. We were especially confused about what information the number of t-SNE components contained — our visualizations changed dramatically with this number. Though we are not sure that t-SNE is the correct method of validating such a hypothesis due to how much information is lost, we will read more on t-SNE to learn about the number of components.

7. Acknowledgements and Contributions

The authors would like to thank the CS229 course staff for instruction during a difficult quarter. We would also like to acknowledge the codebases that we forked for our work (hyperlinks attached): Deep Mahalanobis Detector, Robust OOD Detection. Finally, we are also grateful for each other and our equal contributions on this team.

References

[1] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[3] Jiefeng Chen, Xi Wu, Yingyu Liang, Somesh Jha, et al. Robust out-of-distribution detection in neural networks. *arXiv preprint arXiv:2003.09711*, 2020.

[4] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Russ R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in neural information processing systems*, pages 6510–6520, 2017.

[5] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[7] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[10] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

[11] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[12] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[13] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *arXiv preprint arXiv:1711.09325*, 2017.

[14] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018.

[15] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[16] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

[17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

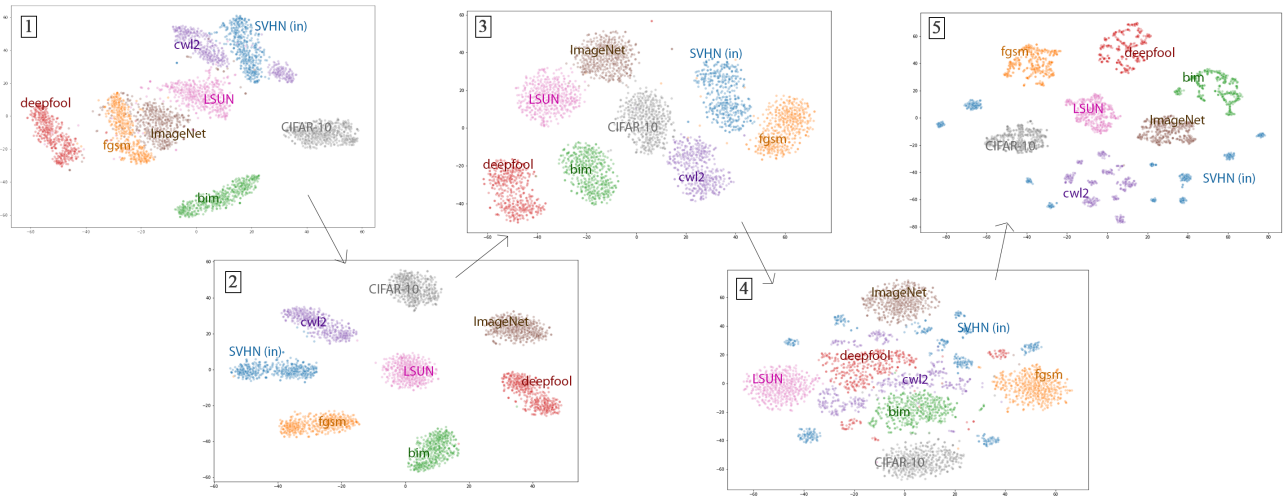

Figure 2. Two-component t-SNE schematics for features after each of five ResNet blocks. ResNet trained on in-dataset SVHN. High-dimensional feature spaces were first reduced to 50 principal components before being fed into t-SNE. 500 random samples from each distribution are plotted. Note the positioning of the in-distribution SVHN (blue) relative to the adversarial distributions FGSM (orange), BIM (green), DeepFool (red), CWL-2 (purple) and to the out-datasets TinyImageNet (brown) and LSUN (pink).

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[19] Dan Pelleg and Andrew W Moore. Active learning for anomaly and rare-category detection. In *Advances in neural information processing systems*, pages 1073–1080, 2005.

[20] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[22] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008.

[23] Sachin Vernekar, Ashish Gaurav, Vahdat Abdelzad, Taylor Denouden, Rick Salay, and Krzysztof Czarnecki. Out-of-distribution detection in classifiers via generation. *arXiv preprint arXiv:1910.04241*, 2019.

[24] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, 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.