Out-Of-Distribution Detection Is Not All You Need

Joris Guérin
Université de Toulouse, LAAS-CNRS
Toulouse, France
joris.guerin@laas.fr

Kevin Delmas
ONERA
Toulouse, France
kevin.delmas@onera.fr

Raul Sena Ferreira
LAAS-CNRS
Toulouse, France
rrsenaferre@laas.fr

Jérémie Guiochet
Université de Toulouse, LAAS-CNRS
Toulouse, France
jeremie.guiochet@laas.fr

Abstract

The usage of deep neural networks in critical systems is limited by our ability to guarantee their correct behavior. Runtime monitors are components aiming to identify unsafe predictions before they can lead to catastrophic consequences. Several recent works on runtime monitoring have focused on out-of-distribution (OOD) detection, i.e., identifying inputs that are different from the training data. In this work, we argue that OOD detection is not a well-suited framework to design efficient runtime monitors and that it is more relevant to evaluate monitors based on their ability to discard incorrect predictions. We discuss the conceptual differences with OOD and conduct extensive experiments on popular datasets to show that: 1. good OOD results can give a false impression of safety, 2. comparison under the OOD setting does not allow identifying the best monitor to detect errors.

1 Introduction

As deep neural networks (DNN) are being used for safety-critical tasks (self-driving cars [1], drones [2]), improving their safety is of utmost importance. This work deals with runtime monitoring, a promising research direction aiming to identify unsafe data encountered during inference. In particular, we focus on the unsupervised setting, i.e., no unsafe data examples available to train the monitor, meaning that one needs to fit a one-class classifier [3] on the DNN training dataset.

Many approaches have been proposed to tackle DNN monitoring. However, in the literature, they are found under different names as they adopt different definitions of "unsafe data instances". The field of Out-Of-Distribution (OOD) detection aims at identifying input data that are far from the training distribution [4; 5; 6; 7]. On the other hand, several works consider directly the problem of identifying DNN errors [8; 9; 10; 11]. In this work, we name this second view Out-of-Model-Scope (OMS) detection. In practice, the approaches to address OOD and OMS detection are not different and follow the same workflow: they use the DNN training dataset to build a one-class classifier (the monitor) to characterize safe data instances and use it to reject unsafe samples. These two paradigms only differ in their objectives, and by extension in how new approaches are evaluated.

In this work, we claim that the goal of a DNN monitor is OMS detection, i.e., identifying errors before they propagate through the system. We argue that OOD was designed as a proxy task for OMS detection, based on the belief that what the DNN knows is equivalent to the information contained in its training dataset. Hence, we first discuss the conceptual differences between OOD and OMS. Then, we conduct experiments to determine whether OOD is a good proxy for OMS, i.e., whether the OMS performance of a monitor can be correctly assessed under the OOD setting.

ML Safety Workshop, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

2 Notations and definitions

Let T be an ML task, defined by an oracle Ω, on a domain $\mathcal{X}$, i.e., $\forall x \in \mathcal{X}$, the ground-truth is $\Omega(x)$. This work mostly discusses classification, but could be extended to other tasks. Let f be a DNN used to solve T, and let m_f be a monitor (one-class classifier) used to reject unsafe predictions of f.

OMS setting We define the scope of f, $\mathcal{D}_f$, to be the set of data instances where f is correct: $D_f = \{x \in \mathcal{X} \mid f(x) = \Omega(x)\}$. Ideally, we want m_f to identify data points that lead to errors of f. Hence, the perfect monitor for f, noted m^*_f, is defined by $\forall x \in \mathcal{X},\ m^*_f(x) = \mathbb{1}_{\neg D_f}(x)$, where $\mathbb{1}_S$ is the indicator function of the set S. We call Out-of-Model-Scope (OMS) detection, the task of designing a monitor that reproduces the behavior of m^*_f. It is defined with respect to a specific model f, as if another DNN f' is used, the model scope will be different ($\mathcal{D}_{f'} \neq \mathcal{D}_f$), and so will the optimal monitor ($m^*_{f'} \neq m^*_f$).

OOD setting In practice, f is trained using a supervised dataset, i.e., a subset of n data points in $\mathcal{X}$ for which the ground truth is known: $D_{\text{train}} = \{(x_i, y_i) \mid \forall i \in \{1, ...n\}\, x_i \in \mathcal{X}, y_i = \Omega(x_i)\}$. A common practice for DNN monitoring is to define an in-distribution domain $\mathcal{D}_{\text{ID}}$, that comprises all data instances drawn from the same distribution as D_{train}. The Out-Of-Distribution (OOD) detection task aims to build a monitor m to identify data instances that do not belong to $\mathcal{D}_{\text{ID}}$, i.e., the perfect OOD monitor (m^*) is defined by $\forall x \in \mathcal{X},\ m^*(x) = \mathbb{1}_{\neg \mathcal{D}_{\text{ID}}}(x)$. The rationale behind OOD detection is that DNNs trained on D_{train} must be good for input data similar to D_{train} ($x \in \mathcal{D}_{\text{ID}}$), but should not perform well on other data ($x \notin \mathcal{D}_{\text{ID}}$). Unlike OMS, the definition of OOD only depends on the task and training dataset, i.e., it is independent of f.

3 Related works

Conceptually, OOD and OMS detectors are not different, i.e., one-class classifiers fitted to D_{train}. In practice, they differ in the way they are evaluated experimentally.

OMS in the literature Several previous works have studied OMS detection, i.e., they developed DNN monitors and assessed their performance based on their ability to detect errors of f. Some rely on monitoring activations of different layers of the DNN [8; 9; 11], while other use model assertion to specify task-specific constraints [10].

OOD in the literature Our definition of OOD is common in the literature. However, the boundaries of $\mathcal{D}_{\text{ID}}$ are fuzzy and there is no clear definition of whether a data point was drawn from the same distribution as D_{train}. To overcome this issue, most works consider that the test split associated with D_{train} belongs to $\mathcal{D}_{\text{ID}}$. Then, different approaches exist to construct OOD sets, and the monitors are evaluated based on their ability to distinguish between the test set and the OOD set. In the literature, three main concepts of "OODness" are used to build OOD datasets:

- **Novelty** A data point $x \in \mathcal{X}$ is OOD if the ground-truth $\Omega(x)$ is not among the predefined classes handled by f, i.e., f cannot be correct. A large body of work was developed and evaluated in this configuration. Several approaches used another dataset, with disjoint label classes, as the OOD set [12; 13; 5; 14]. Another idea to build OOD sets is to remove data samples from certain classes while training f and m_f, and use them as OOD examples to test the monitor [15].
- **Covariate shift** OOD data instances have different characteristics than D_{train} (changes in external conditions, sensor failures, etc.), but with valid ground-truth. Such threats to DNN safety were discussed extensively in [16]. Most works dealing with covariate shift detection have built OOD sets by injecting different kinds of perturbations to test images [6; 7; 17; 18].
- **Adversarial attack** A data sample that was modified to make a DNN fail with high confidence. The difference with covariate shift is in the malicious intent to generate imperceptible perturbations. Several works have tested their monitors for adversarial attack detection [7; 17; 14; 19; 20].

We conclude this section by noting that some works have reported both OOD and OMS results [21; 22]. However, in both of these papers, OOD and OMS are viewed as separate problems. Here, we consider OOD as a proxy for OMS, and we discuss whether both paradigms are useful for the field.

4 Limitations of OOD detection

In this work, we claim that a successful monitor m_f should reject errors of f, and accept other predictions (OMS setting). DNNs are usually good on their training dataset, while their performance decreases when data move away from D_{train} [16]. This fact gave rise to the problem of OOD detection, aiming to reject samples far from the training distribution. Studying the alternate OOD detection problem in place of OMS detection presents several conceptual issues.

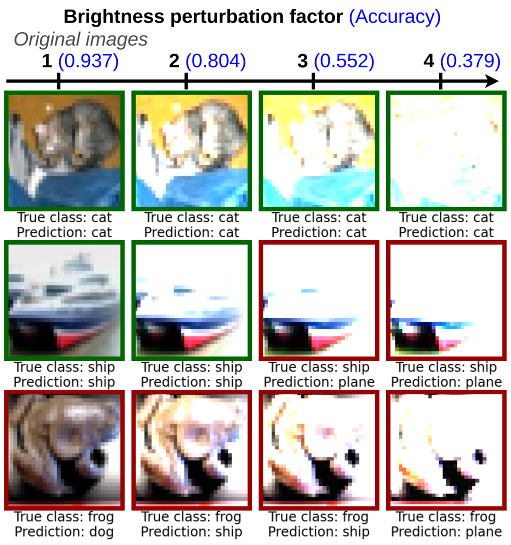

Brightness perturbation factor (Accuracy)
Original images
1 (0.937)
2 (0.804)
3 (0.552)
4 (0.379)
True class: cat Prediction: cat
True class: cat Prediction: cat
True class: cat Prediction: cat
True class: cat Prediction: cat
True class: ship Prediction: ship
True class: ship Prediction: ship
True class: ship Prediction: plane
True class: ship Prediction: plane
True class: frog Prediction: dog
True class: frog Prediction: ship
True class: frog Prediction: ship
True class: frog Prediction: plane

Figure 1: **What is OOD?** Images: CIFAR10 test, predictions: ResNet-34.

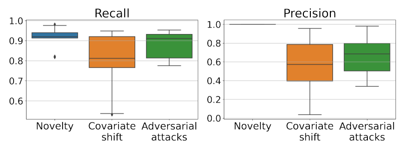
Definition of OOD The first issue with OOD detection is that $\mathcal{D}_{\text{ID}}$ is not well-defined. For example, in Figure 1 it is not clear how to set a threshold on the perturbation factor to define OODness. At a perturbation level of 2, the accuracy of f has dropped about 13%, but it can still produce above 80% of correct predictions. Hence, should the second column be considered OOD? This limitation also applies to adversarial examples, as different attacks lead to different levels of performance drop. For example, if we can generate a few additional misclassifications by changing random pixels, is it an adversarial attack? Regarding novelty detection, the problem is better defined: an image is OOD if its label is not among the output classes handled by f. However, it is not easy to decide whether a highly corrupted image should hold its label: is the top right image of Figure 1 still a cat?

A less ambiguous way to define OOD is to use ideas from the field of DNN calibration [23]: a set of images is OOD if the accuracy of f falls below a fixed threshold on this set. This definition is still threshold dependent, but it describes OODness unambiguously. However, under this definition, OODness is a property of a set, not defined at the individual sample level, which is a different problem than the one studied in most OOD detection works.

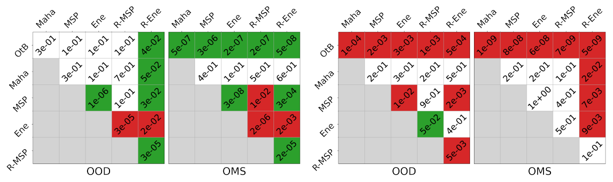
OOD does not always represent OMS Even if we had an unambiguous definition of $\mathcal{D}_{\text{ID}}$, OOD and OMS detection would still be distinct problems. Indeed, $\mathcal{D}_{\text{ID}}$ and $\mathcal{D}_f$ can differ in two ways:

- **Model generalization** The first difference is when input data are OOD but correctly classified by f (top-right image of Figure 1). The limits of $\mathcal{D}_{\text{ID}}$ are often defined by human programmers, independently of the generalization capabilities of f. For such cases, a good OOD monitor will reject valid inputs (false positives), thus decreasing the availability of the model f. For the special case of novelty, this case does not exist as the model cannot be correct on novel data.
- **In-distribution errors** The second difference is when data samples that are similar to D_{train} are misclassified by f (bottom-left image of Figure 1). Such cases can have catastrophic outcomes for safety-critical applications, since if an OOD monitor accepts wrong predictions as long as they are similar to D_{train}, it also let hazardous predictions through the system (false negatives) and decreases its safety. This problem is an issue even when considering novelty detection.

Switching to OMS The above discussion showed that it is hard to come up with an objective definition of what OOD means, and that building good OOD monitors might not be sufficient to ensure the safety of a system using a DNN. The good news is that such a definition is not necessary, as we can simply use OMS as a generic setting for studying DNN monitoring. The OMS paradigm is defined unambiguously and actually corresponds to what we want to achieve, i.e., reject wrong predictions of f. In practice, it is common to define a proxy task to address a more complex generic task of interest. However, when the actual performance at the true task is easy to compute, we should use it to evaluate the performance of new techniques. In our case, computing OMS detection performance is easy as we have access to labeled datasets and we can compute the predictions of f. For these reasons, we argue that DNN monitors should be evaluated with respect to the OMS setting, instead of the ambiguous OOD setting.



(a) Distribution of the OMS recall and precision obtained by the optimal OOD detector m^* across 27 OOD datasets and two DNNs tested in our experiments.



(b) Wilcoxon test to compare recall (left) and precision (right) of monitors across 54 datasets. The p-values are reported. Green: row better than column, Red: column better than row, White: no meaningful difference.

Figure 2: Experimental results.

5 Experimental validation

In this section, we conduct experiments on common OOD datasets from the literature, to see whether analyzing OOD results can lead to misleading conclusions about the safety of the system.

5.1 Experiments description

Our experiments consist of 27 OOD datasets, 2 DNN architectures (54 OOD scenarios), and 6 monitoring approaches. We use three popular image datasets as ID: CIFAR10, CIFAR100, and SVHN [24; 25]. For each ID dataset, the train split is used to fit the monitors, while the test split serves as the ID set for evaluation. Each ID set is combined with 9 distinct OOD sets: three novelty, three covariate shift, and three adversarial attacks. More details about the datasets can be found in Appendix A.

For monitoring, we test two feature-based approaches: Mahalanobis (**Maha**) [14] and Outside-the-Box (**OtB**) [15]. We use the last layer before classification as their data representation. We also use four logit-based approaches: Max Softmax Probability (**MSP**) [21], Energy (**Ene**) [13], and ReAct combined with both MSP (**R-MSP**) and Energy (**R-Ene**) [5]. Except for OtB, the techniques used in our experiments require selecting a rejection threshold on the monitoring scores. This is a complex question that is not studied here. Instead, we consider the optimal F1 threshold, as described by [26].

5.2 OOD can give a false sense of safety

Here, we want to answer the following question: if we manage to develop a perfect OOD detector m^*, can we guarantee that (f, m^*) is safe to use in critical applications?

Let's suppose that we can build m^*, rejecting all OOD samples and accepting all ID samples (precision and recall of 1 at OOD detection). In our experiments, we can simulate the predictions of m^* using the known ID/OOD labels. Then, we want to know how well m^* performs the task of detecting OMS samples. An OMS recall below one means that there exist ID data for which f is erroneous. It is a threat to the safety of the system. An OMS precision below one means that there exist OOD data points for which f is correct. It represents a decrease in the availability of the system. We conduct this experiment for both DNN architectures, across the 27 OOD datasets. The distribution of the OMS performances obtained for the different OOD types is reported in Figure 2a.

We can see that the OMS recall is not close to 1 for most of the experiments conducted. Even for novelty, the median OMS recall of the 18 experiments is below 0.93. In other words, in most cases, more than 7% of the errors of f are not detected by the perfect OOD monitor m^*, which can lead to catastrophic outcomes for safety-critical applications. These results show that, even if perfect OOD detectors are developed, it will not allow us to guarantee the safety of critical systems using ML. Indeed, an OOD recall of 1 does not guarantee the absence of prediction errors. Even worse, reporting very good OOD results can be detrimental as it gives a false sense that the (f, m^*) system is safe.

Regarding OMS precision, we first note that it is always 1 for novelty datasets. This is because f cannot be correct for novel samples, by definition. For other OOD types, we acknowledge an important drop in precision, i.e., an important proportion of correct predictions of f on OOD data.

5.3 OOD can be misleading for comparison

As explained in section 3, monitors are often compared using the OOD setting. Here, we want to verify whether comparing monitors at the OOD task provides relevant insights into OMS performance.

To do this, we compare six monitors from the literature, and for each pair of monitors, we conduct a Wilcoxon signed-rank test across the 54 OOD scenarios to determine whether one is better than the other. The Wilcoxon test is a non-parametric statistical test that can be used to compare pairs of classifiers across multiple datasets [27]. The results obtained are reported in figure 2b.

From these experiments, we can see that the comparison matrices obtained for OOD and OMS look different. For example, OtB is clearly the most conservative approach (highest recall) when looking at the OMS results, but not when considering OOD. On another note, the benefits of using ReAct for monitoring are best seen in the OMS setting (MSP vs. R-MSP, Ene vs. R-Ene).

From these results, it appears that conducting OOD experiments is not a reliable way to compare monitoring approaches regarding their ability to identify errors of f.

6 Conclusion

If we want to use DNN in safety-critical applications, it is paramount to build efficient runtime monitors, which can detect DNN errors from the system before they can have catastrophic consequences. As of today, many research efforts are directed toward solving the problem of out-of-distribution detection, which consists in identifying data samples that were drawn from a different distribution than the DNN training set. In this paper, we discuss the limitations of this setting to enable the safe usage of DNN in critical systems. First, the concept of OOD is not well-defined, which makes it difficult to compare different OOD detection approaches. Second, even a perfect OOD detector can have the undesirable property of discarding valid predictions of the DNN, and even worse, accepting wrong predictions. Extensive experiments conducted on popular OOD detection datasets confirm that these phenomena occur in practical scenarios. Furthermore, we show that the OOD setting cannot be used to compare monitoring approaches accurately, as the best monitor for OOD is not always the best one to detect DNN errors.

On the bright side, adapting current OOD research to these findings will not require drastic changes. The out-of-model-scope (OMS) setting, defined above and discussed in several previous works, is well-defined and perfectly aligned with the DNN monitoring objectives. In addition, most approaches developed for OOD detection can be used for OMS without any modification, and OMS results can be computed straightforwardly for most OOD datasets used in the literature. Hence, the take-home message from this paper is that instead of evaluating new approaches on their ability to detect samples from other data sources, the OOD detection research community should focus on the ability to detect samples leading to erroneous DNN predictions. We also believe that it is a good idea to include OOD samples in OMS evaluation datasets, which is rarely done in OMS papers.

References

[1] A. Stocco, M. Weiss, M. Calzana, and P. Tonella, “Misbehaviour prediction for autonomous driving systems,” in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 2020, pp. 359–371.

[2] J. Guerin, K. Delmas, and J. Guiochet, “Evaluation of runtime monitoring for uav emergency landing,” in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, p. To appear.

[3] S. S. Khan and M. G. Madden, “One-class classification: taxonomy of study and review of techniques,” *The Knowledge Engineering Review*, vol. 29, no. 3, pp. 345–374, 2014.

[4] H. Wang, Z. Li, L. Feng, and W. Zhang, “Vim: Out-of-distribution with virtual-logit matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4921–4930.

[5] Y. Sun, C. Guo, and Y. Li, “React: Out-of-distribution detection with rectified activations,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 144–157, 2021.

[6] C. Schorn and L. Gauerhof, “Facer: A universal framework for detecting anomalous operation of deep neural networks,” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[7] F. Cai and X. Koutsoukos, “Real-time out-of-distribution detection in learning-enabled cyber-physical systems,” in *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2020, pp. 174–183.

[8] F. Granese, M. Romanelli, D. Gorla, C. Palamidessi, and P. Piantanida, “Doctor: A simple method for detecting misclassification errors,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 5669–5681, 2021.

[9] H. Wang, J. Xu, C. Xu, X. Ma, and J. Lu, “Dissector: Input validation for deep learning applications by crossing-layer dissection,” in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. IEEE, 2020, pp. 727–738.

[10] D. Kang, D. Raghavan, P. Bailis, and M. Zaharia, “Model assertions for debugging machine learning,” in *NeurIPS MLSys Workshop*, vol. 3, 2018, p. 10.

[11] C.-H. Cheng, G. Nührenberg, and H. Yasuoka, “Runtime monitoring neuron activation patterns,” in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), Florence, Italy*. IEEE, 2019, pp. 300–303.

[12] S. Liang, Y. Li, and R. Srikant, “Enhancing the reliability of out-of-distribution image detection in neural networks,” in *International Conference on Learning Representations*, 2018.

[13] W. Liu, X. Wang, J. Owens, and Y. Li, “Energy-based out-of-distribution detection,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 464–21 475, 2020.

[14] K. Lee, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” *Advances in neural information processing systems*, vol. 31, 2018.

[15] T. A. Henzinger, A. Lukina, and C. Schilling, “Outside the box: Abstraction-based monitoring of neural networks,” in *24th European Conference on Artificial Intelligence-ECAI 2020*, 2020, pp. 2433–2440.

[16] D. Hendrycks and T. Dietterich, “Benchmarking neural network robustness to common corruptions and perturbations,” in *International Conference on Learning Representations*, 2019.

[17] V. Chen, M.-K. Yoon, and Z. Shao, “Task-aware novelty detection for visual-based deep learning in autonomous systems,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 060–11 066.

[18] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, “Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems,” in *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2018, pp. 132–142.

[19] Y. Kantaros, T. Carpenter, K. Sridhar, Y. Yang, I. Lee, and J. Weimer, “Real-time detectors for digital and physical adversarial inputs to perception systems,” in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, 2021, pp. 67–76.

[20] J. Wang, G. Dong, J. Sun, X. Wang, and P. Zhang, “Adversarial sample detection for deep neural network through model mutation testing,” in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 1245–1256.

[21] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” *Proceedings of International Conference on Learning Representations*, 2017.

[22] R. S. Ferreira, J. Arlat, J. Guiochet, and H. Waeselynck, “Benchmarking safety monitors for image classifiers with machine learning,” in *2021 IEEE 26th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2021, pp. 7–16.

[23] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330.

[24] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[25] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. [Online]. Available: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf

[26] J. Guérin, A. M. de Paula Canuto, and L. M. G. Goncalves, "Robust detection of objects under periodic motion with gaussian process filtering," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2020, pp. 685–692.

[27] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine learning research*, vol. 7, pp. 1–30, 2006.

[28] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[30] Z. Papakipos and J. Bitton, "Augly: Data augmentations for robustness," 2022.

[31] H. Kim, "Torchattacks: A pytorch repository for adversarial attacks," *arXiv preprint arXiv:2010.01950*, 2020.

A Datasets used in our experiments

Our experiments consist of 27 OOD datasets, 2 DNN architectures (54 OOD scenarios), and 6 monitoring approaches.

We use three popular image datasets as ID: CIFAR10, CIFAR100, and SVHN [24; 25]. For each ID dataset, the train split is used to fit the monitors, while the test split serves as the ID set for evaluation. Each ID set is combined with 9 distinct OOD sets:

- Three novelty tasks – For CIFAR10 we use CIFAR100, SVHN, and LSUN [28] to represent novel data. For CIFAR100 we use CIFAR10, SVHN, and LSUN. For SVHN we use CIFAR10, LSUN, and TinyImageNet (subset of ImageNet [29]). With these choices, ID and OOD classes never overlap.
- Three covariate shift tasks – For each ID dataset, we apply three image transformations from the AugLy library [30]: Brightness (factor=3), Blur (radius=2), and Pixelization (ratio=0.5).
- Three adversarial attacks – For each ID dataset, we apply three adversarial attacks from Torchattacks [31], with default parameters: FGSM, DeepFool, and PGD.

For each of these 27 OOD scenarios, we experiment with two DNN architectures: DenseNet and ResNet. We use the pre-trained models from [14].