A Benchmark of Medical Out of Distribution Detection

Tianshi Cao [1 2] **Chin-Wei Huang** [3] **David Yu-Tung Hui** [3] **Joseph Paul Cohen** [3]

Abstract

There is a rise in the use of deep learning for automated medical diagnosis, most notably in medical imaging. Such an automated system uses a set of images from a patient to diagnose whether they have a disease. However, systems trained for one particular domain of images cannot be expected to perform accurately on images of a different domain. These images should be filtered out by an Out-of-Distribution Detection (OoDD) method prior to diagnosis. This paper benchmarks popular OoDD methods in three domains of medical imaging: chest x-rays, fundus images, and histology slides. Our experiments show that despite methods yielding good results on some types of out-of-distribution samples, they fail to recognize images close to the training distribution.

1. Introduction

A safe system for medical diagnosis should withhold diagnosis on cases outside its validated expertise. For machine learning (ML) systems, the expertise is defined by the validation score on the distribution of data used during training, as the performance of the system can be validated on samples drawn from the same distribution (as per PAC learning (Valiant, 1984)). This restriction can be translated into the task of *Out-of-Distribution Detection* (OoDD), the goal of which is to distinguish between samples in and out of a desired distribution (abbreviated to *In* and *Out* data). In this case, *In* data is the training distribution of the diagnosis system.

In contrast to natural image analysis, medical image analysis must often deal with orientation invariance (e.g. in cell images), high variance in feature scale (in X-ray images), and locale specific features (e.g. CT) (Razzak et al., 2017). A systematic evaluation of OoDD methods for applications specific to medical image domains remains absent, leaving practitioners blind as to which OoDD methods perform well and under which circumstances. This paper fills this gap by benchmarking many current OoDD methods under four medical image types (frontal and lateral chest X-ray, fundus imaging, and histology). Our empirical studies show that current OoDD methods perform poorly when detecting correctly acquired images that are not represented in the training data. We find a different conclusion on the efficacy of OOD methods in comparison to the prior work of Shafaei et al. (2018), which benchmark the OoDD methods on a suite of natural image datasets.

[*]Equal contribution [1]Vector Institute, Toronto, Canada [2]Department of Computer Science, Univeristy of Toronto, Canada [3]Mila, Université de Montréal. Correspondence to: Joseph Paul Cohen <joseph@josephpcohen.com>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning. Copyright 2020 by the author(s).

2. Defining OoD in Medical Data

Given an *In* distribution dataset, how should we define what constitutes *Out* data? To address this, we identify three distinct out-of-distribution categories:

- **usecase 1** Reject inputs that are unrelated to the evaluation. This includes obviously-wrong inputs from a different domain (e.g. fMRI image in X-ray, cartoon in natural image etc) and less obviously-wrong inputs (e.g. wrist X-ray in chest X-ray).
- **usecase 2** Reject inputs which are incorrectly prepared (e.g. blurry image of chest X-ray, poor contrast, Lateral vs Dorsal position).
- **usecase 3** Reject inputs that are unseen due to a selection bias in the training distribution (e.g. image with an unseen disease).

We justify these usecases by enumerating different types of mistakes or biases that can occur at different stages of the data acquisition. This is visually represented in Figure 1. We construct our experiments to evaluate OoDD methods' performance on each category.

3. Task Formulation

Let us denote a sample of *In* data used to train the underlying ML application as D_{tr}. In this paper, we will assume that the underlying application is to perform classification using a deep neural network. Then, a OoDD method M is trained on a "validation set" $D_{val} = D^{in}_{val} \cup D^{out}_{val}$, a

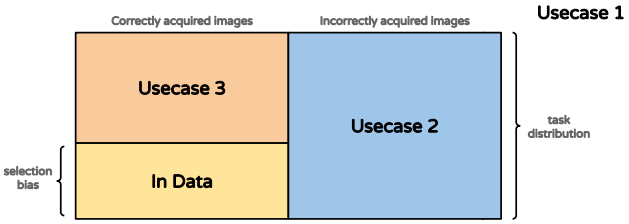

Figure 1. The three usecases shown in relation to each other. The training data is sampled iid from the *In* data distribution.

union of *In* and *Out* samples. M may also use the features learned by the classification network, thereby also making use of D_{tr}. Finally, M is evaluated on the test set $D_{test} = D^{in}_{test} \cup D^{out}_{test}$, also composed of *In* and *Out* samples. Each tuple $(M, D_{tr}, D^{in}_{val}, D^{out}_{val}, D^{in}_{test}, D^{out}_{test})$ constitutes an experiment.

3.1. Methods of OoDD (M)

We consider three classes of OoDD methods. Data-only methods do not rely on any pre-trained models and are learned directly on D_{val}. Classifier-only methods assume access to a downstream classifier trained for classification on *In* data (D_{tr}). Methods with auxiliary models requires pre-training of a neural network that is trained on *In* data through other tasks such as image reconstruction or generative modeling. **Data-only methods** The most simple and easy to implement baseline is k-Nearest-Neighbors (KNN) which only needs to observe the training data. This is performed on images as a baseline for our evaluations. For speed only 1000 samples are used from D_{tr} to calculate neighbor distance. A threshold is determined using samples from D_{val}.

Classifier-only methods Classifier-only methods make use of the downstream classifier for performing OoDD. Compared to data-only methods they require less storage, however their applicability is constrained to cases with classification as downstream tasks. *Probability Threshold* (Hendrycks & Gimpel, 2017) uses a threshold on the prediction confidence of the classifier to perform OoDD. *Score SVM* trains an SVM on the logits of the classifier as features, generalizing probability threshold. *Binary Classifier* trains on the features of the penultimate layer of the classifier. *Feature KNN* uses the same features as the binary classifier, but constructs a KNN classifier in place of logistic regression. *ODIN* (Liang et al., 2017) is a probability threshold method that preprocesses the input by taking a gradient step of the input image to increase the difference between the *In* and *Out* data. *Mahalanobis* (Lee et al., 2018) models the features of a classifier of *In* data as a mixture of Gaussians, preprocesses the data as ODin, and thresholds the likelihood of the feature.

Methods with Auxiliary Models OoDD methods in this section require an auxiliary model trained on *In* data. This results in extra setup time and resources when the downstream classifier is readily available. However, this could also be advantageous when the downstream task is not classification (such as regression) where methods may be difficult to adapt. *Autoencoder Reconstruction* thresholds the reconstruction loss of the autoencoder to achieve OOD detection. Intuitively, the autoencoder is only optimized for reconstructing *In* data, and hence reconstruction quality of *Out* data is expected to be poor due to the bottleneck in the autoencoder. In this work we consider three variants of autoencoders: standard autoencoder (AE) trained with reconstruction loss only, variational autoencoder trained with a variational lower bound (VAE) (Kingma & Welling, 2014), and decoder+encoder trained with an adversarial loss (ALI (Dumoulin et al., 2016), BiGAN (Donahue et al., 2017)). Furthermore, we include two different reconstruction loss functions in the benchmark: mean-squared error (MSE) and binary cross entropy (BCE). Finally, *AE KNN* constructs an KNN classifier on the features output by the encoder.

3.2. *In* Datasets ($D_{tr}, D^{in}_{val}, D^{in}_{test}$)

For D_{tr}, we select from four medical datasets ranging over three modalities of medical imaging. Each dataset has been randomly split three ways for use in D_{tr}, D^{in}_{val}, and D^{in}_{test}. Each dataset also contains a classification task. As most ML applications only deal with one image type (i.e. an medical application wouldn't simultaneous diagnose chest conditions and diabetic retinopathy), we consider each *In* distribution dataset as distinct evaluations and do not consider their combinations. The *In* datasets of each evaluation are:

1. Frontal view chest X-ray images. The task is to predict 10 of the 14 radiologcal findings defined by the **NIH** Chest-X-Ray dataset (Wang et al., 2017). The remaining conditions are held-out for usecase 3.

2. Lateral view chest X-ray images (PC-Lateral). The task is the same as evaluation 1, but the data is from lateral view images in the PADChest (**PC**) dataset (Bustos et al., 2019). Remaining conditions are also held-out for usecase 3.

3. Fundus/retinal (back of the eye) images. The task is to detect diabetic retinopathy in the retina defined by the **DRD** (Diabetic Retinopathy Detection) [1] dataset.

4. H&E stained histology slides of lymph nodes. The task is to predict if image patches contain cancerous tissue defined by the **PCAM** dataset (Veeling et al., 2018).

[1] https://www.kaggle.com/c/diabetic-retinopathy-detection

Domain	Eval	*In* data	Usecase 1 *Out* data	Usecase 2 *Out* data	Usecase 3 *Out* data
Chest X-ray	1	NIH (*In* split)	UC-1 Common, MURA	PC-Lateral, PC-AP, PC-PED, PC-AP-Horizontal	NIH-Cardiomegaly, NIH-Nodule, NIH-Mass, NIH-Pneumothorax
	2	PC-Lateral (*In* split)	UC-1 Common, MURA	PC-AP, PC-PED, PC-AP-Horizontal, PC-PA	PC-Cardiomegaly, PC-Nodule, PC-Mass, PC-Pneumothorax
Fundus Imaging	3	DRD	UC-1 Common	DRIMDB	RIGA
Histology	4	PCAM	UC-1 Common, Malaria	ANHIR, IDC	None

Table 1. Datasets used in evaluations. See Appendix A for more details.

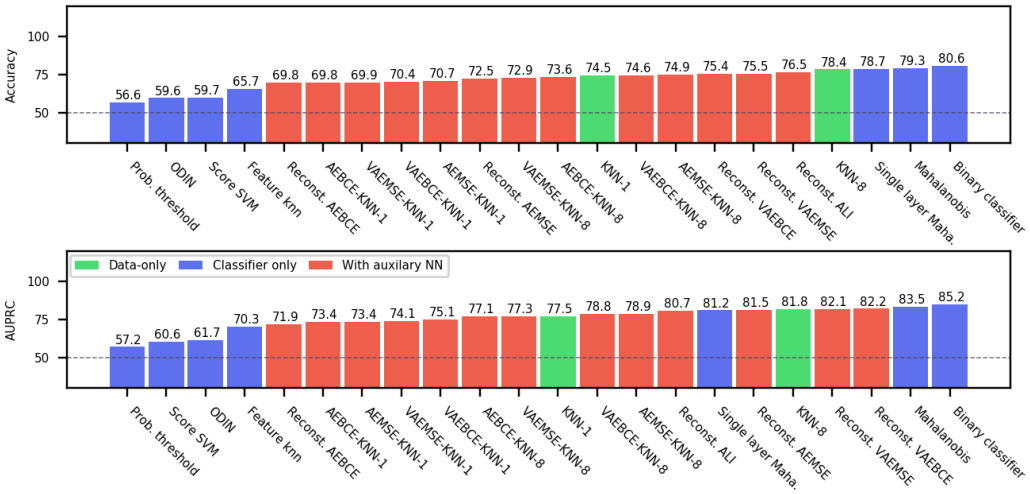

Figure 2. Accuracy and AUPRC of OoDD methods aggregated over all evaluations

3.3. *Out* Datasets (D_{val}^{out} and D_{test}^{out})

We select *Out* datasets according to usecases described in section 2. As users may be independently interested in a particular usecase, we evaluate the OoDD methods per usecase. Clearly, characteristics of each usecase are defined relative to the *In* distribution, hence we may need to select different *Out* datasets for each *In* dataset.

For D_{val}^{out} and D_{test}^{out} under **usecase 1**, we take a combination of natural image and symbols datasets which we call *UC-1 Common*. This is used for every *In* data. For **usecase 2**, we use datasets of the same modality of the *In* distribution, but incorrectly captured. For example, different views (e.g. lateral vs frontal) of the chest area are used as D_{val}^{out} and D_{test}^{out} for evaluations 1 and 2. Finally, for **usecase 3**, we use images of different conditions/diseases as *Out* data. For evaluations 1 and 2, the four held-out conditions are used as usecase 3 *Out* data. We did not include a usecase 3 *Out* dataset for histology slides due to lack of available data. Table 1 summarizes our roster of *In* and *Out* datasets. Each *Out* dataset is split 50/50 for D_{val}^{out} and D_{test}^{out}. Subsampling is used to balance the number of *In* and *Out* samples in D_{val} and D_{test}.

4. Experimental Procedure and Results

In this benchmark, we report the performance of each OoDD method on every evaluation and usecase. We measure the accuracy and Area Under Precision-Recall Curve (AUPRC) on D_{test}, totaling at 11 pairs of performance numbers per method. Since D_{test} is class-balanced, accuracy provides an unbiased representation of type I and type II errors. AUPRC characterizes the separability of *In* and *Out* samples in predicted value (the value that we threshold to obtain classification). Details of experimental setup can be found in Appendix B.

Figure 3 and Appendix figures 4 to 6 show the performance of OoDD methods on the four evaluations. Generally, we observe that our choice of datasets for *In* and *Out* data create a range of simple to hard test cases for OoDD methods. While many methods can solve usecase 1 and usecase 2 adequately in evaluations 1-3, usecase 3 proves difficult for all methods tested. This is reflected in the UMAP visualization of the AE latent spaces (column B of figures 3 to 5), in which we observe that the *In* data points are easily separable from *Out* data in usecases 1 and 2, but well-mixed with *Out* data in usecase 3. It is surprising that no method achieved significantly better accuracy than random in usecase 3 of

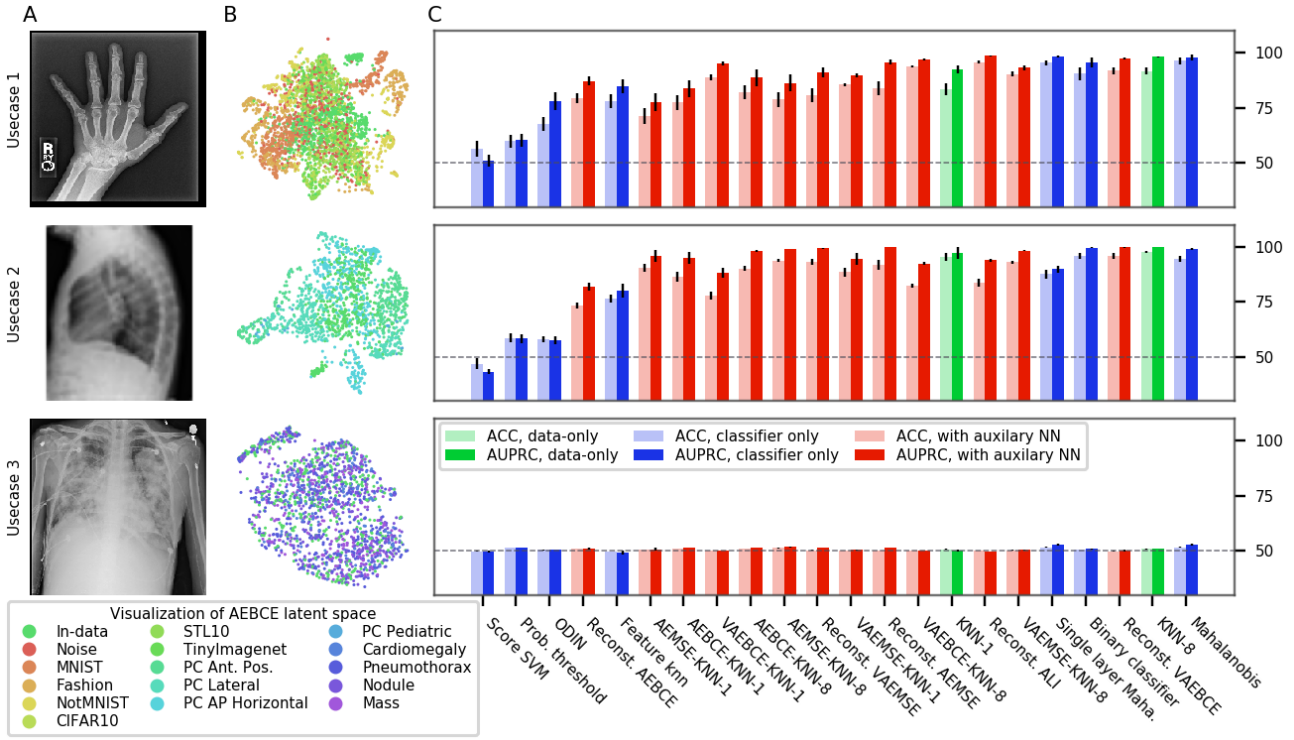

Figure 3. Visualizations and OoDD results on AP view chest-xray (Evaluation 1). Each row of figures correspond to a usecase. Column A shows examples of *Out* data for each usecase (hand x-ray, lateral view chest x-ray, and xray of pneumothorax from top to bottom). Column B shows UMAP visualizations of AE latent space - colors of points represent their respective datasets. Column C plots the accuracy and AUPRC of OoDD methods in each usecase, averaged across all randomized trials. Bars are sorted by accuracy averaged across usecases, and coloured according to method's grouping: green for baseline image space methods, blue for methods based upon the task specific classifier, and red for methods that use an auxilary neural network. Error bars represent 95% confidence interval.

evaluations 1 and 2 across all repeated trials. This illustrates the extreme difficulty of detecting unseen/nouveau diseases, which corroborates the findings of Ren et al. (2019).

Overall Performance Across evaluations, the better performing classifier-only methods are competitive with the methods that use auxiliary models. When performance is aggregated across all evaluations (Figure 2), the best classifier-only methods (Mahalanobis and binary classifier) outperform auxiliary models in accuracy. The performance of binary classifier is surprisingly strong. The performance of 8 nearest neighbor (KNN-8) is surprisingly competitive with the best OoDD methods. This may indicate that knowledge of classification on *In* data does not transfer directly to the task of OoDD.

5. Discussion

Overall, the top three classifier-only methods obtain better accuracy than all methods with auxiliary models except for fundus imaging. Binary classifier has the best accuracy and AUPRC on average, and is simple to implement. Hence, we recommend binary classifier as the default method for OoDD in the domain of medical images. While usecase 1 and 2 are easily solved with non-complicated models, the failure of most models in almost all tasks to significantly solve usecase 3 is consistent with the finding of Ahmed & Courville (2019). This leaves an open door for future research.

6. Conclusion

The methods we find to work best are almost opposite that of Shafaei et al. (2018), who evaluate on natural images instead of medical images, despite using the same code for overlapping methods. We performed an extensive hyperparameter search on all methods and conclude that this discrepancy is due to the specific data and tasks we have defined.

Users of diagnostic tools which employ these OoDD methods should still remain vigilant that images very close to the training distribution yet not in it (and a false negative for usecase 3) could yield unexpected results. In the absence of OoDD methods which have good performance on usecase 3, another approach is to develop methods which will systematically generalize to these examples.

References

Ahmed, F. and Courville, A. Detecting semantic anomalies. In *Association for the Advancement of Artificial Intelligence*, aug 2019. URL http://arxiv.org/abs/1908.04388.

Bustos, A., Pertusa, A., Salinas, J.-M., and de la Iglesia-Vayá, M. PadChest: A large chest x-ray image dataset with multi-label annotated reports. *arXiv preprint*, jan 2019. URL http://arxiv.org/abs/1901.07441.

Donahue, J., Krähenbühl, P., and Darrell, T. Adversarial Feature Learning. In *International Conference on Learning Representations (ICLR)*, 2017. URL http://arxiv.org/abs/1605.09782.

Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., and Courville, A. Adversarially Learned Inference. *International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1606.00704.

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

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely Connected Convolutional Networks. In *Computer Vision and Pattern Recognition*, 2017. URL https://arxiv.org/abs/1608.06993.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6114.

Lee, K., Lee, K., Lee, H., and Shin, J. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. jul 2018. URL http://arxiv.org/abs/1807.03888.

Liang, S., Li, Y., and Srikant, R. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. jun 2017. URL http://arxiv.org/abs/1706.02690.

Razzak, M. I., Naz, S., and Zaib, A. Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps*, pp. 323–350, Nov 2017. ISSN 2212-9413. doi: 10.1007/978-3-319-65981-7_12. URL http://dx.doi.org/10.1007/978-3-319-65981-7_12.

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, 2019.

Shafaei, A., Schmidt, M., and Little, J. J. Does Your Model Know the Digit 6 Is Not a Cat? A Less Biased Evaluation of "Outlier" Detectors. *arxiv*, sep 2018. URL http://arxiv.org/abs/1809.04729.

Valiant, L. G. A theory of the learnable. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 436–445. Association for Computing Machinery, dec 1984. ISBN 0897911334. doi: 10.1145/800057.808710.

Veeling, B. S., Linmans, J., Winkens, J., Cohen, T., and Welling, M. Rotation Equivariant CNNs for Digital Pathology. In *Medical Image Computing & Computer Assisted Intervention (MICCAI)*, jun 2018. URL http://arxiv.org/abs/1806.03962.

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., and Summers, R. M. ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. In *Computer Vision and Pattern Recognition*, 2017. doi: 10.1109/CVPR.2017.369. URL http://arxiv.org/abs/1705.02315.

A. Description of Datasets

The following datasets are used in UC-1 Common:

- **MNIST**[2] 28x28 black and white hand written digits data. Original test split is used in UC-1 Common.
- **notMNIST** [3] Letters A-J in various fonts. Black and white with resolution of 28x28. Test split is used.
- **CIFAR10 and CIFAR100**[4] 32x32 natural images. Original test split used in UC-1 Common.
- **TinyImagenet**[5] 96x96 downsampled subset of ILSVRC2012. Validation split used in UC-1 Common.
- **FashionMNIST**[6] Grayscale 28x28 images of clothes and shoes. Validation split is used in UC-1 Common.
- **STL-10** [7] Natural image dataset of size 96x96. 8000 testing images are used in UC-1 Common.
- **Noise** White noise generated at any desired resolution.

The following medical datasets are used:

- **ANHIR** [8] Automatic Non-rigid Histological Image Registration Challenge. Microscopy images of histopathology tissue samples stained with different dyes. Images of intestine and kidney tissue were used in evaluation 4, usecase 2.
- **DRD** [9] High-resolution retina images with presence of diabetic retinopathy in each image labeled on a scale of 0 to 4. We convert this into a classification task where 0 corresponds to healthy and 1-4 corresponds to unhealthy.
- **DRIMDB** Fundus images of various qualities labeled as good/bad/outlier. We use the images labeled as bad/outlier in evaluation 3, usecase 2.
- **Malaria** [10] Image of cells in blood smear microscopy collected from healthy persons and patients with malaria. Used in evaluation 4 usecase 1.
- **MURA** [11] MUsculoskeletal RAdiographs is a large dataset of skeletal X-rays. We use its validation split in evaluation 1 and 2's usecase 1. Images are grayscale and the square cropped.
- **NIH Chest** [12] This NIH Chest X-ray Dataset is comprised of 112,120 X-ray images with 14 condition labels. The x-rays images are in posterior-anterior view (X-tray traverses back to front).
- **PAD Chest** [13] This is a large scale chest x-ray dataset. It is labeled with 117 radiological findings - we use the subset with correspondence to the 14 condition labels in the NIH Chest dataset. Images are in 5 different views: posterior-anterior (PA), anterior-posterior (AP), lateral, AP horizontal, and pediatric.
- **PCAM** [14] Patch Camelyon dataset is composed of histopathologic scans of lymph node sections. Images are labeled for presence of cancerous tissue.
- **RIGA** Fundus imaging dataset for glaucoma analysis. Images are marked by physicians for regions of disease. We use this dataset for evaluation 3, usecase 3.

[2] http://yann.lecun.com/exdb/mnist/

[3] http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html

[4] https://www.cs.toronto.edu/ kriz/cifar.html

[5] https://tiny-imagenet.herokuapp.com/

[6] https://www.kaggle.com/zalando-research/fashionmnist

[7] https://ai.stanford.edu/ acoates/stl10/

[8] https://anhir.grand-challenge.org/

[9] https://www.kaggle.com/c/diabetic-retinopathy-detection/data

[10] https://lhncbc.nlm.nih.gov/publication/pub9932

[11] https://stanfordmlgroup.github.io/competitions/mura/

[12] https://www.kaggle.com/nih-chest-xrays/data

[13] https://bimcv.cipf.es/bimcv-projects/padchest/

[14] https://github.com/basveeling/pcam

B. Details of Experimental Procedure

B.1. Network training

For classifier models, we use a DenseNet-121 architecture (Huang et al., 2017) with Imagenet pretrained weights. The last layer is re-initialized and the full network is finetuned on D_{tr}. As the NIH and PC-Lateral datasets only contain grayscale images, the pretrained weights of features in the first layer are averaged across channels prior to finetuning.

For all of the autoencoders, we use a 12-layer CNN architecture with a bottleneck dimension of 512 for all evaluations. Due to computational constraints, all images are downsampled to 64×64 when fed to an autoencoder. These AEs are trained from scratch on their respective D_{tr} with MSE loss and BCE loss. We also trained VAEs with the same architectures, except that the bottleneck dimension is doubled to 1024 to allow the code to be split into means and variances.

In addition, we explore the potential benefits of training encoder+decoder using ALI in evaluation 1. We use the same network architecture as proposed in (Dumoulin et al., 2016), with weights pretrained on Imagenet and finetuned on NIH *In* classes. Due to the added complexity of training GANs and the lack of significant improvements in OoDD performance over regular AEs (see §4), we did not train ALI models for the other three evaluations.

In order to gauge training progress and overfitting, we hold out 5% of D_{tr} as validation set. We select the training checkpoint with the lowest error on D_{tr} for use in OoDD methods.

B.2. OoDD Method Training

When training the OoDD methods for usecase 1, three *Out* datasets are randomly selected for D_{val} while the rest is used for D_{test}. For usecases 2 and 3, we enumerate over configurations where each *Out* dataset is used as D_{val} with

the rest as D_{test}. D_{val} and D_{test} are class-balanced by subsampling equal numbers of *In* and *Out* samples. Addtionally, some methods (ODIN and Mahalanobis) require additional hyper-parameter selection. Hence, we further subdivide D_{val} in to a 80% 'training' split and a 20% 'validation' split; methods are trained/optimized on the 'training' split with early-stopping/calibration on the 'validation' split. Hyperparameter sweep is carried out where needed. 10 repeated trials, with re-sampled D_{val} and D_{test}, are performed for each evaluation.

C. Additional Results

Accuracy vs. AUPRC as performance metric There are some tests with accuracy that's much lower than AUPRC. This is caused by the classification threshold calibrated for D_{val} being ill-suited for classification on D_{test}. As AUPRC is computed by scanning all threshold values, it is not effected by the calibration performed on D_{val}. If online recalibration is available, then methods with low accuracy and high AUPRC can be improved more significantly over methods with similar accuracy but lower AUPRC.

Computational Cost We consider computational cost of each method in terms of setup time and run time. The setup time is measured as the wall-clock computation time taken for hyperparameter search and training. For methods with auxiliary models, the training time of auxiliary neural networks are also included in the setup-time. Run time is measured as the per-sample computation time (averaged over fixed batch size) at test time. Figure 7 plots the accuracy of models over their respective setup and run time. All methods can make predictions reasonably fast, allowing for potential online usage. Mahalanobis and its single layer variant take significantly more time to setup and run than other classifier methods. KNN-8 exhibits the best time vs performance trade-off with its low setup time and good performance. However, as it requires the storage of training images for predictions, it may be unsuitable for use on memory constrained platforms (e.g. mobile) or when training data privacy is of concern.

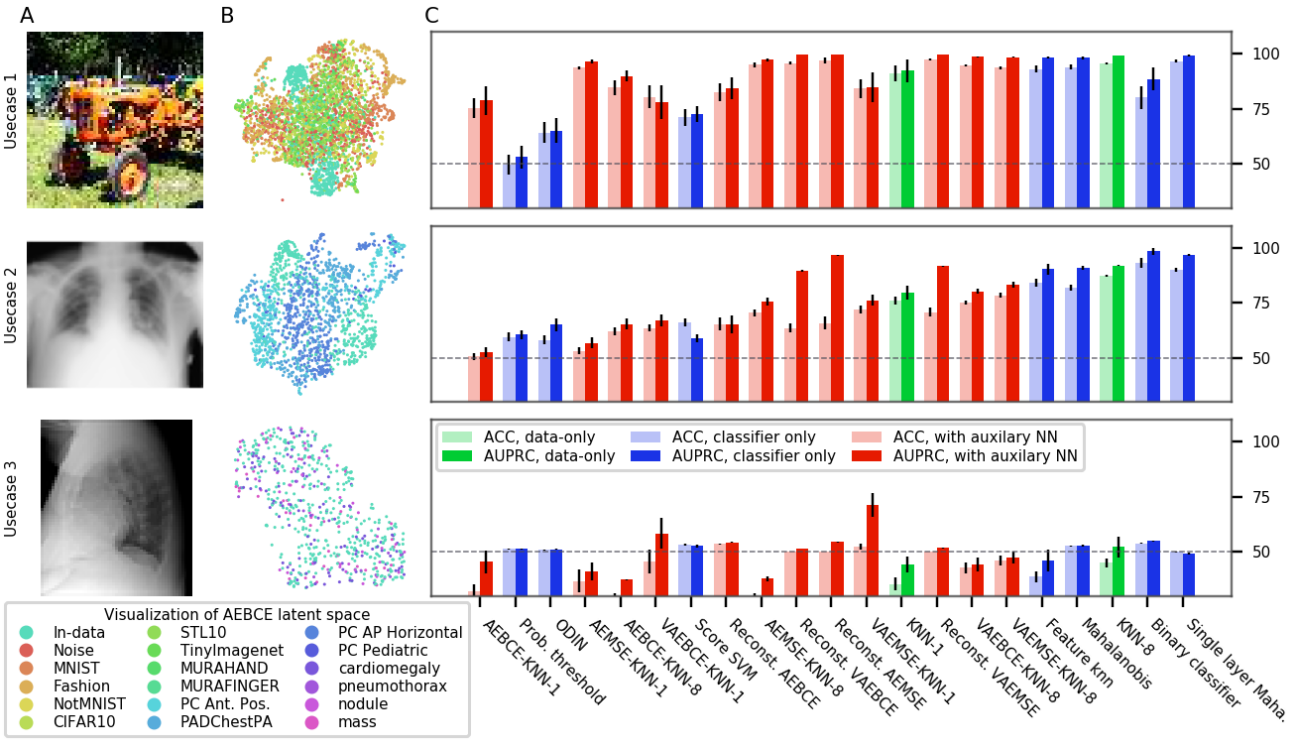

Figure 4. Lateral X-ray imaging

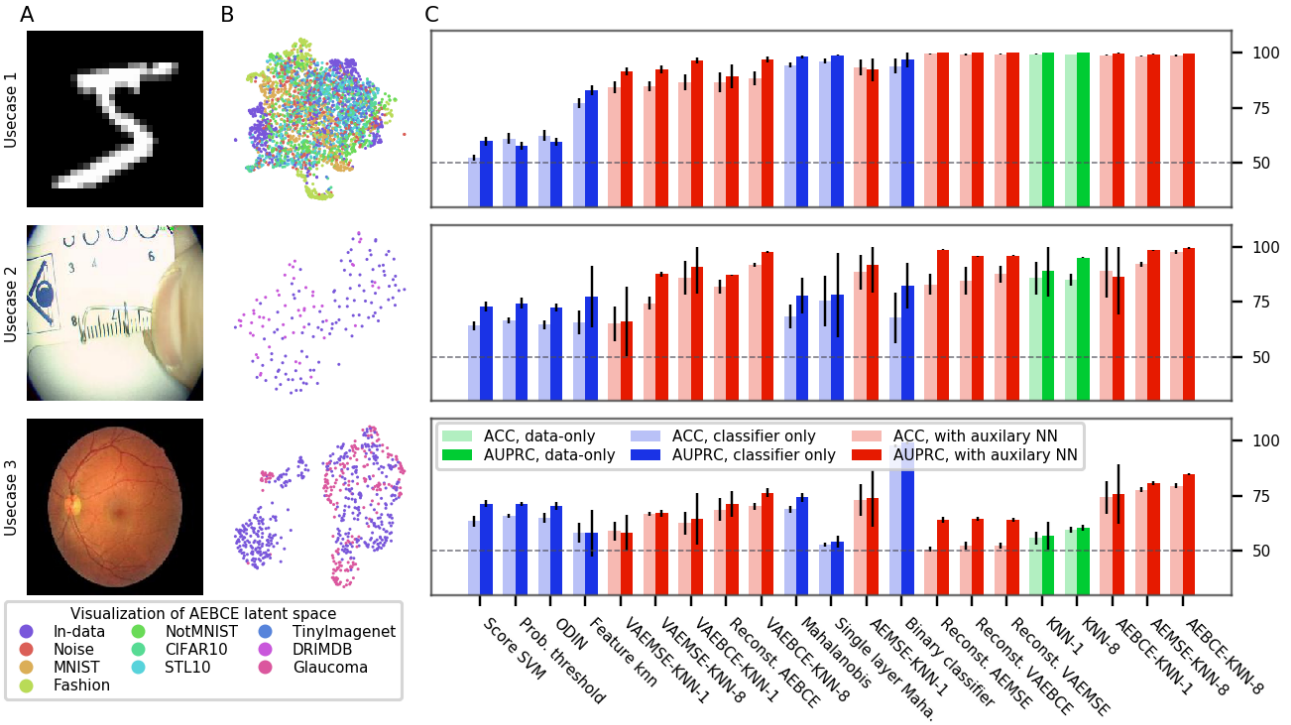

Figure 5. Fundus Imaging (see Figure 3 for description)

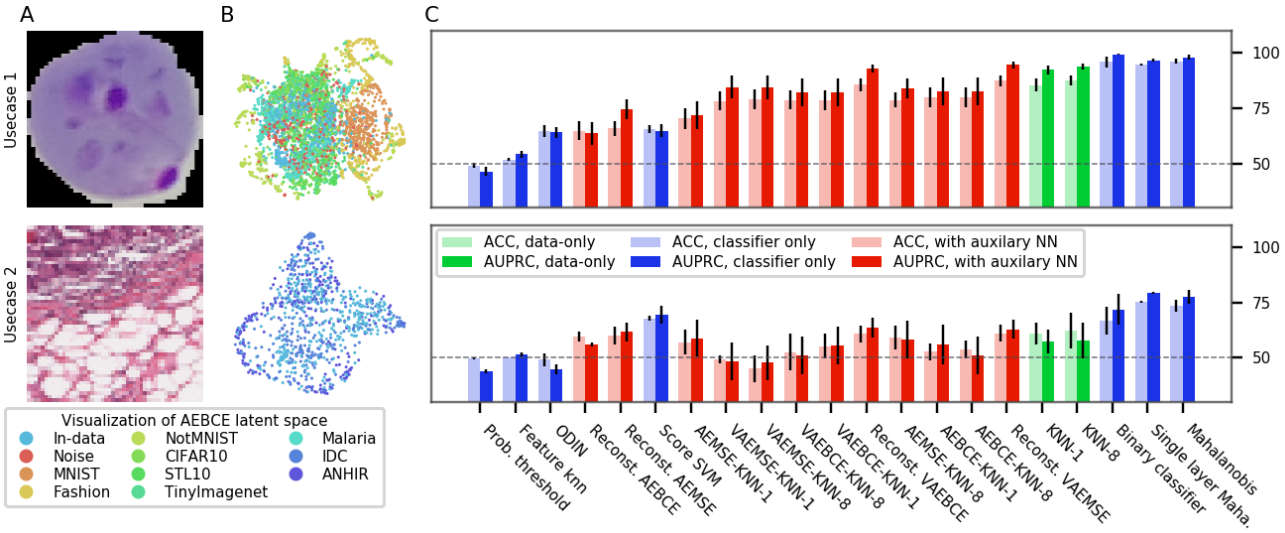

Figure 6. Histology Imaging (see Figure 3 for description)

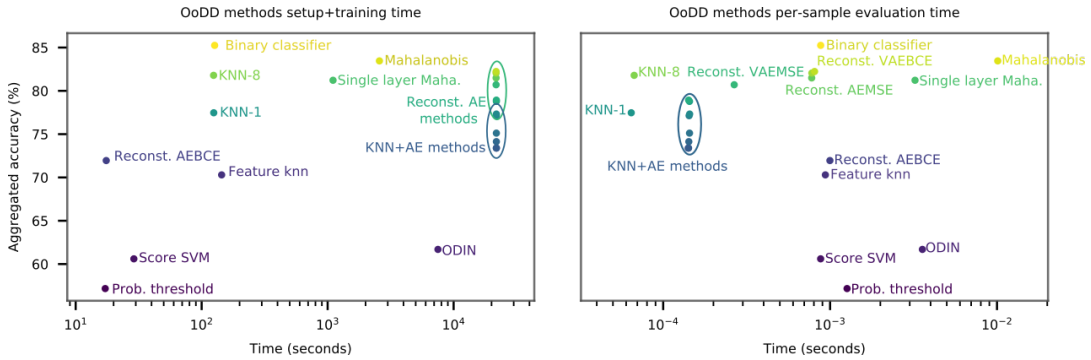

Figure 7. Overall accuracy of methods plotted over total setup time (left) and per-sample run time (right)

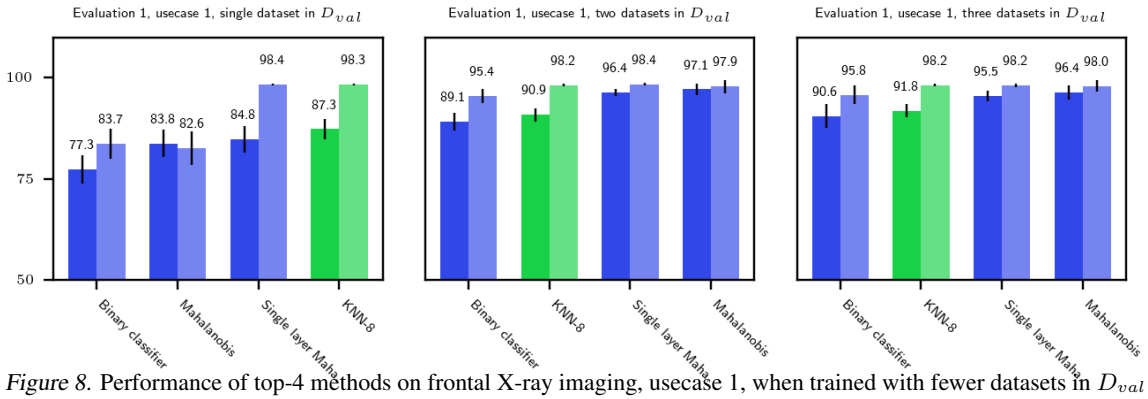

Figure 8. Performance of top-4 methods on frontal X-ray imaging, usecase 1, when trained with fewer datasets in D_{val}