Semi-supervised novelty detection using ensembles with regularized disagreement – Supplementary material

Alexandru Țifrea[1] **Eric Stavarache**[1] **Fanny Yang**[1]

[1]Department of Computer Science, ETH Zurich, Switzerland

A THEORETICAL STATEMENTS

Definition A.1 ((ϵ, η)-clusterable data set)**.** *We say that a data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ is* (ϵ, η)-clusterable *for fixed $\epsilon > 0$ and $\eta \in [0, 1]$ if there exists a partitioning of it into subsets $\{C_1, ..., C_K\}$, which we call* clusters*, each with their associated unit-norm cluster center c_i, that satisfy the following conditions:*

- $\bigcup_{i=1}^K C_i = \mathcal{D}$ *and* $C_i \cap C_j = \emptyset, \forall i, j \in [K]$*;*
- *all the points in a cluster lie in the ϵ-neighborhood of their corresponding cluster center, i.e. $||x - c_i||_2 \leq \epsilon$ for all $x \in C_i$ and all $i \in [K]$;*
- *a fraction of at least $1 - \eta$ of the points in each cluster C_i have the same label, which we call the* cluster label *and denote $y^*(c_i)$. The remaining points suffer from label noise;*
- *if two cluster C_i and C_j have different labels, then their centers are 2ϵ far from each other, i.e. $||c_i - c_j||_2 \geq 2\epsilon$;*
- *the clusters are balanced i.e. for all $i \in [K], \alpha_1 \frac{n}{K} \leq |C_i| \leq \alpha_2 \frac{n}{K}$, where α_1 and α_2 are two positive constants.*

In our case, for a fixed label $c \in \mathcal{Y}$, we assume that the set $S \cup (U, c)$ is (ϵ, η)-clusterable into K clusters. We further assume that each cluster C_i only includes a few noisy samples from $(U_{\mathrm{ID}}^{\neg c}, c)$, i.e. $\frac{|C_i \cap (U_{\mathrm{ID}}^{\neg c}, c)|}{|C_i|} \leq \eta$ and that for clusters C_i whose cluster label is not c, i.e. $y^*(c_i) \neq c$, it holds that $C_i \cap (U_{\mathrm{OOD}}, c) = \emptyset$.

We define the matrices $C := [c_1, ..., c_K]^T \in \mathbb{R}^{K \times d}$ and $\Sigma := (CC^T) \bigodot \mathbb{E}_g[\phi'(Cg)\phi'(Cg)^T]$, with $g \sim \mathcal{N}(0, I_d)$ and where $\bigodot$ denotes the elementwise product. We use $\|\cdot\|$ and $\lambda_{min}(\cdot)$ to denote the spectral norm and the smallest eigenvalue of a matrix, respectively.

For prediction, we consider a 2-layer neural network model with p hidden units, where $p \gtrsim \frac{K^2\|C\|^4}{\lambda_{min}(\Sigma)^4}$. We can write this model as follows:

$$x \mapsto f(x; W) = v^T \phi(Wx), \tag{1}$$

The first layer weights W are initialized with random values drawn from $\mathcal{N}(0, 1)$, while the last layer weights v have fixed values: half of them are set to $1/p$ and the other half is $-1/p$. We consider activation functions ϕ with bounded first and second order derivatives, i.e. $|\phi'(x)| \leq \Gamma$ and $\phi''(x) \leq \Gamma$. We use the squared loss for training, i.e. $\mathcal{L}(W) = \frac{1}{2}\sum_{i=0}^n (y_i - f(x_i; W))^2$ and take gradient descent steps to find the optimum of the loss function, i.e. $W_{\tau+1} = W_\tau - \eta \nabla \mathcal{L}(W_\tau)$, where the step size is set to $\eta \simeq \frac{K}{n\|C\|^2}$.

In the informal Proposition 3.1 we use the notation $\lambda_C^{\mathrm{NN}} := \lambda_{min}(\Sigma)$. Intuitively, Σ can be seen as a kernel matrix associated with the two-layer neural network, as discussed in Li et al. [2020]. In particular, Σ depends on the choice of the nonlinear activation function of the neural network. For instance, for ReLU activations, $\lambda_{min}(\Sigma)$ can be lower bounded like $\lambda_{min}(\Sigma) \gtrsim \frac{\mu}{K^2}$, where μ is the minimum distance between two clusters.

We can now state the following proposition:

Proposition A.1. *Assume that $\eta \leq \delta/8$ and $\epsilon \leq \alpha\delta\lambda_{min}(\Sigma)^2/K^2$, where δ is a constant such that $\delta \leq \frac{2}{|\mathcal{Y}-1|}$ and α is a constant that depends on Γ. Then it holds with high probability $1 - 3/K^{100} - Ke^{-100d}$ over the initialization of the weights that the neural network trained on $S \cup (U, c)$ perfectly fits S, (U_{ID}^{c}, c) and (U_{OOD}, c), but not $(U_{\text{ID}}^{\neg c}, c)$, after $T = c_4\frac{\|C\|^2}{\lambda_{min}(\Sigma)}$ iterations.*

This result shows that there exists an optimal stopping time at which the neural network predicts the correct label on all ID points and the label c on all the OOD points. As we will see later in the proof, the proposition is derived from a more general result which shows that the early stopped model predicts these labels not only on the points in U but also in an ϵ-neighborhood around cluster centers. Hence, an ERD ensemble can be used to detect holdout OOD samples similar to the ones in U, after being tuned on U. This follows the intuition that classifiers regularized with early stopping are smooth and generalize well.

The clusterable data model is generic enough to include data sets with non-linear decision boundaries. Moreover, notice that the condition in Proposition A.1 is satisfied when $S \cup (U_{\text{ID}}, c)$ is (ϵ, η)-clusterable and (U_{OOD}, c) is ϵ-clusterable and if the cluster centers of (U_{OOD}, c) are at distance at least 2ϵ from the cluster centers of $S \cup (U_{\text{ID}}, c)$. A situation in which these requirements are met is, for instance, when the OOD data comes from novel classes, when all classes (including the unseen ones that are not in the training set) are well separated, with cluster centers at least 2ϵ away in Euclidean distance. In addition, in order to limit the amount of label noise in each cluster, it is necessary that the number of incorrectly labeled samples in $(U_{\text{ID}}^{\neg c}, c)$ is small, relative to the size of S.

In practice, we only need that the decision boundary separating (U_{OOD}, c) from S is easier to learn than the classifier required to interpolate the incorrectly labeled $(U_{\text{ID}}^{\neg c}, c)$, which is often the case, provided that (U_{OOD}, c) is large enough and the OOD samples come from novel classes.

We now provide the proof for Proposition A.1:

Proof. We begin by restating a result from Li et al. [2020]:

Theorem A.1 ([Li et al., 2020])**.** *Let $\mathcal{D} := \{(x_i, y_i)\} \in \mathbb{R}^d \times \mathcal{Y}$ be an (ϵ, η)-clusterable training set, with $\epsilon \leq c_1\delta\lambda_{min}(\Sigma)^2/K^2$ and $\eta \leq \delta/8$, where δ is a constant that satisfies $\delta \leq \frac{2}{|\mathcal{Y}|-1}$. Consider a two-layer neural network as described above, and train it with gradient descent starting from initial weights sampled i.i.d. from $\mathcal{N}(0,1)$. Assume further that the step size is $\eta = c_2\frac{K}{n\|C\|^2}$ and that the number of hidden units p is at least $c_3\frac{K^2\|C\|^4}{\lambda_{min}(\Sigma)^4}$. Under these conditions, it holds with probability at least $1 - 3/K^{100} - Ke^{-100d}$ over the random draws of the initial weights, that after $T = c_4\frac{\|C\|^2}{\lambda_{min}(\Sigma)}$ gradient descent steps, the neural network $x \mapsto f(x; W_T)$ predicts the correct cluster label for all points in the ϵ-neighborhood of the cluster center, namely:*

$$\arg\max_{y\in\mathcal{Y}} |f(x; W_T) - \omega(y)| = y^*(c_i), \text{ for all } x \text{ with } \|x - c_i\|_2 \leq \epsilon \text{ and all clusters } i \in [K], \tag{2}$$

where $\omega : \mathcal{Y} \to \{0,1\}^{|\mathcal{Y}|}$ yields one-hot embeddings of the labels. The constants c_1, c_2, c_3, c_4 depend only on Γ.

Notice that, under the assumptions introduced above, the set $S \cup (U, c)$ is (ϵ, η)-clusterable, since the incorrectly labeled ID points in $(U_{\text{ID}}^{\neg c}, c)$ constitute at most a fraction η of the clusters they belong to. As a consequence, Proposition A.1 follows directly from Theorem A.1.

□

B DISAGREEMENT SCORE FOR NOVELTY DETECTION

As we argue in Section 3, Algorithm 1 produces an ensemble that disagrees on OOD data, and hence, we want to devise a scalar score that reflects this model diversity. Previous works [Lakshminarayanan et al., 2017, Ovadia et al., 2019] first average the softmax predictions of the models in the ensemble and then use the entropy as a metric, i.e.

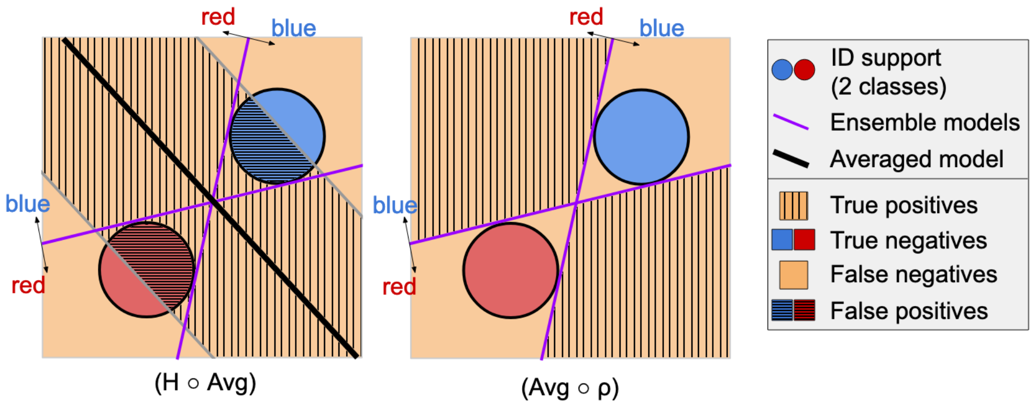


Figure 1: Cartoon illustration showing a diverse ensemble of linear binary classifiers. We compare novelty detection performance for two aggregation scores: $(\mathrm{H} \circ \mathrm{Avg})$ (**Left**) and $(\mathrm{Avg} \circ \rho)$ with $\rho(f_1(x), f_2(x)) = \mathbb{1}_{\mathrm{sgn}(f_1(x)) \neq \mathrm{sgn}(f_2(x))}$ (**Right**). The two metrics achieve similar TPRs, but using $(\mathrm{H} \circ \mathrm{Avg})$ instead of our score, $(\mathrm{Avg} \circ \rho)$, leads to more false positives, since the former simply flags as OOD a band around the averaged model (solid black line) and does not take advantage of the ensemble's diversity.

$(\mathrm{H} \circ \mathrm{Avg})(f_1(x), ..., f_K(x)) := -\sum_{i=1}^{|\mathcal{Y}|}(f(x))_i \log(f(x))_i$ where $f(x) := \frac{1}{K}\sum_{i=1}^{K} f_i(x)$ and $(f(x))_i$ is the i^{th} element of $f(x) \in [0,1]^{|\mathcal{Y}|1}$. We argue later that averaging discards information about the diversity of the models.

Recall that our average pairwise *disagreement* between the outputs of K models in an ensemble reads:[2]

$$(\mathrm{Avg} \circ \rho)(f_1(x), ..., f_K(x)) := \frac{2}{K(K-1)} \sum_{i \neq j} \rho\left(f_i(x), f_j(x)\right), \tag{3}$$

where ρ is a measure of disagreement between the softmax outputs of two predictors, for example the total variation distance $\rho_{\mathrm{TV}}(f_i(x), f_j(x)) = \frac{1}{2}\|f_i(x) - f_j(x)\|_1$ used in our experiments.

We briefly highlight the reason why averaging softmax outputs *first* like in previous works relinquishes all the benefits of having a more diverse ensemble, as opposed to the proposed pairwise score in Equation 3. Recall that varying thresholds yield different true negative and true positive rates (TNR and TPR, respectively) for a given statistic. In the sketch in Figure 1 we show that the score we propose, $(\mathrm{Avg} \circ \rho)$, achieves a higher TNR compared to $(\mathrm{H} \circ \mathrm{Avg})$, for a fixed TPR, which is a common way of evaluating statistical tests. Notice that the detection region for $(\mathrm{H} \circ \mathrm{Avg})$ is always limited to a band around the average model for any threshold value t_0. In order for the $(\mathrm{H} \circ \mathrm{Avg})$ to have large TPR, this band needs to be wide, leading to many false positives. Instead, our disagreement score exploits the diversity of the models to more accurately detect OOD data.

We now provide further quantitative evidence to support the intuition presented in Figure 1. The aggregation metric is tailored to exploit ensemble diversity, which makes it particularly beneficial for ERD. On the other hand, Vanilla Ensembles only rely on the stochasticity of the training process and the random initializations of the weights to produce diverse models, which often leads to classifiers that are strikingly similar as we show in Figure 2 for a few 2D data sets. As a consequence, using our disagreement score $(\mathrm{Avg} \circ \rho)$ for Vanilla Ensembles can sometimes hurt novelty detection performance. To see this, consider the extreme situation in which the models in the ensemble are identical, i.e. $f_1 = f_2$. Then it follows that $(\mathrm{Avg} \circ \rho)(f_1(x), f_2(x)) = 0$, for all test points x and for any function ρ that satisfies the distance axioms.

We note that the disagreement score that we propose takes a form that is similar to previous diversity scores, e.g. Zhang and Zhou [2010], Yu and Aizawa [2019]. In the context of regression, one can measure uncertainty using the variance of the outputs metric previously employed in works such as Gal and Ghahramani [2016]. However, we point out that using the output variance requires that the ensemble is the result of sampling from a random process (e.g. sampling different training data for the models, or sampling different parameters from a posterior). In our framework, we obtain the ensemble by solving a different optimization problem for each of the models by assigning a different label to the unlabeled data.

[1]We abuse notation slightly and denote our disagreement metric as $(\mathrm{Avg} \circ \rho)$ to contrast it with the ensemble entropy metric $(\mathrm{H} \circ \mathrm{Avg})$, which first takes the average of the softmax outputs and only afterwards computes the score.

[2]We abuse notation slightly and denote our disagreement metric as $(\mathrm{Avg} \circ \rho)$ to contrast it with the ensemble entropy metric $(\mathrm{H} \circ \mathrm{Avg})$, which first takes the average of the softmax outputs and only afterwards computes the score.

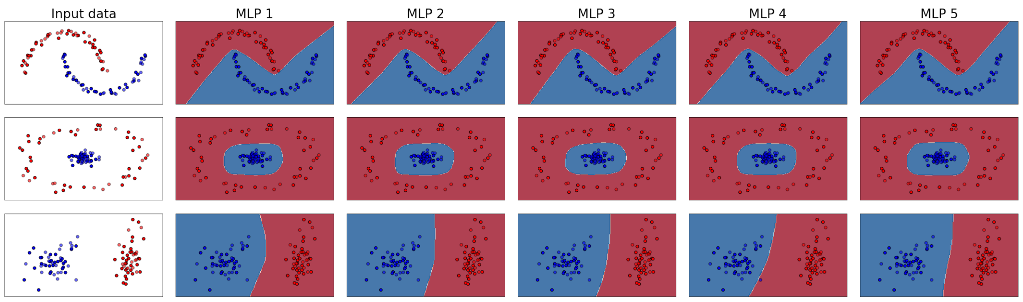


Figure 2: Relying only on the randomness of SGD and of the weight initialization to diversify models is not enough, as it often yields similar classifiers. Each column shows a different predictor trained from random initializations with Adam. All models have the same 1-hidden layer MLP architecture.

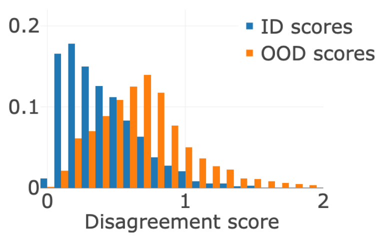
Therefore, despite their similarities, our disagreement score and the output variance are, on a conceptual level, fundamentally different metrics.

Table 1 shows that $(\text{Avg} \circ \rho)$ leads to worse novelty detection performance for Vanilla Ensembles, compared to using the entropy of the average softmax score, $(\text{H} \circ \text{Avg})$, which was proposed in prior work. However, if the ensembles are indeed diverse, as we argue is the case for our method ERD (see Section 3), then there is a clear advantage to using a score that, unlike $(\text{H} \circ \text{Avg})$, takes diversity into account, as shown in Table 1 for 5-model ERD ensembles.

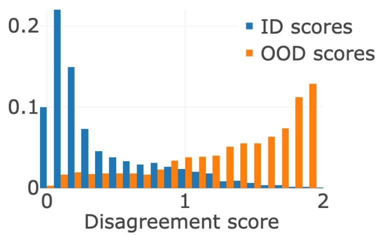
Table 1: The disagreement score that we propose $(\text{Avg} \circ \rho)$ exploits ensemble diversity and benefits in particular ERD ensembles. Novelty detection performance is significantly improved when using $(\text{Avg} \circ \rho)$ compared to the previously proposed $(\text{H} \circ \text{Avg})$ metric. Since Vanilla Ensemble are not diverse enough, a score that relies on model diversity can hurt novelty detection performance. We highlight the AUROC and the TNR@95 obtained with the score function that is ***best for Vanilla Ensemble*** and the **best for ERD**.

ID data	OOD data	Vanilla Ensembles $(\text{H} \circ \text{Avg})$	Vanilla Ensembles $(\text{Avg} \circ \rho)$	ERD $(\text{H} \circ \text{Avg})$	ERD $(\text{Avg} \circ \rho)$
		AUROC ↑ / TNR@95 ↑			
SVHN	CIFAR10	***0.97*** / ***0.88***	0.96 / ***0.89***	0.86 / 0.85	**0.99** / **0.97**
CIFAR10	SVHN	***0.92*** / ***0.78***	0.91 / ***0.78***	0.92 / 0.92	**1.00** / **1.00**
CIFAR100	SVHN	***0.84*** / ***0.48***	0.79 / 0.46	0.36 / 0.35	**1.00** / **1.00**
SVHN[0:4]	SVHN[5:9]	***0.92*** / ***0.69***	0.91 / ***0.69***	**0.94** / **0.66**	**0.94** / **0.66**
CIFAR10[0:4]	CIFAR10[5:9]	***0.80*** / ***0.39***	***0.80*** / ***0.39***	**0.91** / 0.65	**0.91** / **0.66**
CIFAR100[0:49]	CIFAR100[50:99]	***0.78*** / ***0.35***	0.76 / 0.34	0.63 / 0.38	**0.81** / **0.40**
Average		***0.87*** / ***0.60***	0.86 / 0.59	0.77 / 0.64	**0.94** / **0.78**

We highlight once again that other methods that attempt to obtain diverse ensembles, such as MCD, fail to train models with sufficient disagreement, even when they use oracle OOD for hyperparameter tuning (Figure 3a).



(a) Not enough diversity (MCD)



(b) Regularized diversity (ERD)

Figure 3: Distribution of disagreement scores on ID and OOD data for an ensemble that is not diverse enough (**Left**), and an ensemble with regularized disagreement (**Right**). Note that MCD is early-stopped using oracle OOD data. ID=CIFAR10[0:4], OOD=CIFAR10[5:9].

C TAXONOMY OF OOD DETECTION METHODS ACCORDING TO OVERALL OBJECTIVE

We now provide more details regarding the categorization of OOD detection approaches based on the different surrogate objectives that they use in order to detect OOD samples.

Learning the ID marginal P_X. We loosely define OOD samples as all x for which $P_X(x) < \alpha$, for a small constant $\alpha > 0$. Therefore, if we had access to the marginal training distribution P_X, we would have perfect OOD detection. Realistically, however, P_X is unknown, and we need to resort to estimating it. Explicit density estimation with generative models [Akçay et al., 2018, Nalisnick et al., 2019] is inherently difficult in high dimensions. Alternatively, one-class classification [Mũnoz-Marí et al., 2010, Ruff et al., 2020, Sohn et al., 2021] and PU learning approaches [du Plessis et al., 2014, Kiryo et al., 2017] try to directly learn a discriminator between ID and OOD data in the presence of known (e.g. A-UND) or unknown (e.g. SSND) OOD data. However, these methods tend to produce indistinguishable representations for inliers and outliers when the ID distribution consists of many diverse classes.

Learning P_X using label information (ours). Since in a prediction problem, the ID training set has class labels, one can take advantage of that additional information to distinguish points in the support of P_X from OOD data. For instance, Lee et al. [2018], Sastry and Oore [2019] propose to use the intermediate representations of neural networks trained for prediction to detect OOD data. Often, the task is to also simultaneously predict well on ID data, a problem known as open-set recognition [Geng et al., 2021] and tackled by approaches like OpenHybrid [Zhang et al., 2020a].

Learning uncertainty estimates for $P_{Y|X}$. In the prediction setting, calibrated uncertainty estimates error could naturally be used to detect OOD samples. Many uncertainty quantification methods are based on a Bayesian framework [Gal and Ghahramani, 2016, Malinin and Gales, 2018] or calibration improvement [Liang et al., 2018, Hafner et al., 2019]. However, neither of them perform as well as other OOD methods mentioned above [Ovadia et al., 2019].

D EXPERIMENT DETAILS

D.1 BASELINES

In this section we describe in detail the baselines with which we compare our method and describe how we choose their hyperparameters. For all baselines we use the hyperparameters suggested by the authors for the respective data sets (e.g. different hyperparameters for CIFAR10 or ImageNet). For all methods, we use pretrained models provided by the authors. However, we note that for the novel-class settings, pretraining on the entire training set means that the model is exposed to the OOD classes as well, which is undesirable. Therefore, for these settings we pretrain only on the split of the training set that contains the ID classes. Since the classification problem is similar to the original one of training on the entire training set, we use the same hyperparameters that the authors report in the original papers.

Moreover, we point out that even though different methods use different model architectures, that is not inherently unreasonable when the goal is novelty detection, since it is not clear if a complex model is more desirable than a smaller model. For this reason, we use the model architecture recommended by the authors of the baselines and which was used to produce the good results reported in their published works. For Vanilla Ensembles and for ERD we show results for different architectures in Appendix F.8.

- **Vanilla Ensembles** [Lakshminarayanan et al., 2017]: We train an ensemble on the training set according to the true labels. For a test sample, we average the outputs of the softmax probabilities predicted by the models, and use the entropy of the resulting distribution as the score for the hypothesis test described in Section 2.3. We use ensembles of 5 models, with the same architecture and hyperparameters as the ones used for ERD. Hyperparameters are tuned to achieve good validation accuracy.
- **Gram method** [Sastry and Oore, 2019]: The Gram baseline is similar to the Mahalanobis method in that both use the intermediate feature representations obtained with a deep neural network to determine whether a test point is an outlier. However, what sets the Gram method apart is the fact that it does not need any OOD data for training or calibration. We use the pretrained models provided by the authors, or train our own, using the same methodology as described for the Mahalanobis baseline. For OOD detection, we use the code published by the authors. We note that for MLP models, the Gram method is difficult to tune and we could not find a configuration that works well, despite our best efforts and following the suggestions proposed during our communication with the authors.

- **Deep Prior Networks (DPN)** [Malinin and Gales, 2018]: DPN is a Bayesian Method that trains a neural network (Prior Network) to parametrize a Dirichlet distribution over the class probabilities. We train a WideResNet WRN-28-10 for 100 epochs using SGD with momentum 0.9, with an initial learning rate of 0.01, which is decayed by 0.2 at epochs 50, 70, and 90. For MNIST, we use EMINST/Letters as OOD for tuning. For all other settings, we use TinyImages as OOD for tuning.
- **Outlier Exposure** [Hendrycks et al., 2019]: This approach makes a model's softmax predictions close to the uniform distribution on the known outliers, while maintaining a good classification performance on the training distribution. We use the WideResNet architecture (WRN). For fine-tuning, we use the settings recommended by the authors, namely we train for 10 epochs with learning rate 0.001. For training from scratch, we train for 100 epochs with an initial learning rate of 0.1. When the training data set is either CIFAR10/CIFAR100 or ImageNet, we use the default WRN parameters of the author's code, namely 40 layers, 2 widen-factor, droprate 0.3. When the training dataset is SVHN, we use the author's recommended parameters of 16 layers, 4 widen-factor and droprate 0.4. All settings use the cosine annealing learning rate scheduler provided with the author's code, without any modifications. For all settings, we use TinyImages as known OOD data during training. In Section F.6 we show results for known OOD data that is similar to the OOD data used for testing.
- **Mahalanobis** [Lee et al., 2018]: The method pretrains models on the labeled training data. For a test data point, it uses the intermediate representations of each layer as "extracted features". It then performs binary classification using logistic regression using these extracted features. In the original setting, the classification is done on "training" ID vs "training" OOD samples (which are from the same distribution as the test OOD samples). Furthermore, hyperparameter tuning for the optimal amount of noise is performed on validation ID and OOD data. We use the WRN-28-10 architecture, pretrained for 200 epochs. The initial learning rate is 0.1, which is decayed at epochs 60, 120, and 160 by 0.2. We use SGD with momentum 0.9, and the standard weight decay of $5 \cdot 10^{-4}$. The code published for the Mahalanobis method performs a hyperparameter search automatically for each of the data sets.

The following baselines attempt to leverage the unlabeled data that is available in applications such as the one depicted in Figure 1, similar to ERD.

- **Non-negative PU learning (nnPU)** [Kiryo et al., 2017]: The method trains a binary predictor to distinguish between a set of known positives (in our case the ID data) and a set that contains a mixture of positives and negatives (in our case the unlabeled set). To prevent the interpolation of all the unlabeled samples, Kiryo et al. [2017] proposes a regularized objective. It is important to note that most training objectives in the PU learning literature require that the ratio between the positives and negatives in the unlabeled set is known or easy to estimate. For our experiments we always use the exact OOD ratio to train the nnPU baseline. Therefore, we obtain an upper bound on the AUROC/TNR@95. If the ratio is estimated from finite samples, then estimation errors may lead to slightly worse OOD detection performance. We perform a grid search over the learning rate and the threshold that appears in the nnPU regularizer and pick the option with the best validation accuracy measured on a holdout set with only positive samples (in our case, ID data).
- **Maximum Classifier Discrepancy (MCD)** [Yu and Aizawa, 2019]: The MCD method trains two classifiers at the same time, and makes them disagree on the unlabeled data, while maintaining good classification performance. We use the WRN-28-10 architecture as suggested in the paper. We did not change the default parameters which came with the author's code, so weight decay is 10^{-4}, and the optimizer is SGD with momentum 0.9. When available (for CIFAR10 and CIFAR100), we use the pretrained models provided by the authors. For the other training datasets, we use their methodology to generate pretrained models: We train a WRN-28-10 for 200 epochs. The learning rate starts at 0.1 and drops by a factor of 10 at 50% and 75% of the training progress.
- **Mahalanobis-U**: This is a slightly different version of the Mahalanobis baseline, for which we use early-stopped logistic regression to distinguish between the training set and an unlabeled set with ID and OOD samples (instead of discriminating a known OOD set from the inliers). The early stopping iteration is chosen to minimize the classification errors on a validation set that contains only ID data (recall that we do not assume to know which are the OOD samples).

In addition to these approaches that have been introduced in prior work, we also propose a strong novel baseline that that bares some similarity to PU learning and to ERD.

- **Binary classifier** The approach consists in discriminating between the labeled ID training set and the mixed unlabeled set, that contains both ID and OOD data. We use regularization to prevent the trivial solution for which the entire unlabeled set is predicted as OOD. Unlike PU learning, the binary classifier does not require that the OOD ratio in the test distribution is known. The approach is similar to a method described in [Scott and Blanchard, 2008] which also requires that the OOD ratio of the unlabeled set is known. We tune the learning rate and the weight of the unlabeled samples in the training loss by performing a grid search and selecting the configuration with the best validation accuracy, computed on a holdout set containing only ID samples. We note that the binary classifier that appears in Section G in the medical benchmark, is not the same as this baseline. For more details on the binary classifier that appears in the medical data experiments we refer the reader to Cao et al. [2020].

D.2 TRAINING CONFIGURATION FOR ERD

For ERD we always use hyperparameters that give the best validation accuracy when training a model on the ID training set. In other words, we pick hyperparameter values that lead to good ID generalization and do not perform further hyperparameter tuning for the different OOD data sets on which we evaluate our approach. We point out that, if the ID labeled set is known to suffer from class imbalance, subpopulation imbalance or label noise, any training method that addresses these issues can be used instead of standard empirical risk minimization to train our ensemble (e.g. see Li et al. [2020]).

For MNIST and FashionMNIST, we train ensembles of 3-layer MLP models with ReLU activations. Each intermediate layer has 100 neurons. The models are optimized using Adam, with a learning rate of 0.001, for 10 epochs.

For SVHN, CIFAR10/CIFAR100 and ImageNet, we train ensembles of ResNet20 [He et al., 2016]. The models are initialized with weights pretrained for 100 epochs on the labeled training set. We fine-tune each model for 10 epochs using SGD with momentum 0.9, and a learning rate of 0.001. The weights are trained with an ℓ_2 regularization coefficient of $5e-4$. We use a batch size of 128 for all scenarios, unless explicitly stated otherwise. We used the same hyperparameters for all settings.

For pretraining, we perform SGD for 100 epochs and use the same architecture and hyperparameters as described above, with the exception of the learning rate that starts at 0.1, and is multiplied by 0.2 at epochs 50, 70 and 90.

Apart from ERD, which fine-tunes the ensemble models starting from pretrained weights, we also present in the Appendix results for ERD++. This variant of our method trains the models from random initializations, and hence needs more iterations to converge, making it more computationally expensive than ERD. We train all models in the ERD++ ensembles for 100 epochs with a learning rate that starts at 0.1, and is multiplied by 0.2 at epochs 50, 70 and 90. All other hyperparameters are the same as for ERD ensembles.

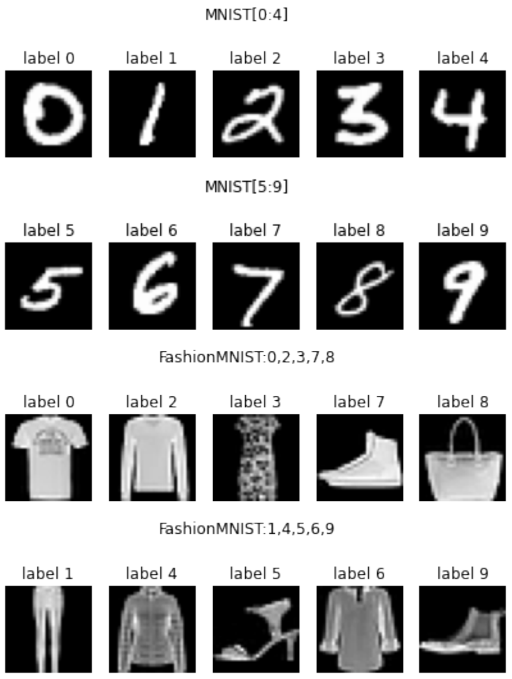
For the medical data sets, we train a Densenet-121 as the authors do in the original paper [Cao et al., 2020]. For ERD++, we do not use random weight initializations, but instead we start with the ImageNet weights provided with Tensorflow. The training configuration is exactly the same as for ResNet20, except that we use a batch size of 32 due to GPU memory restrictions, and for fine tuning we use a constant learning rate of 10^{-5}.

D.3 COMPUTATIONAL CONSIDERATIONS FOR ERD

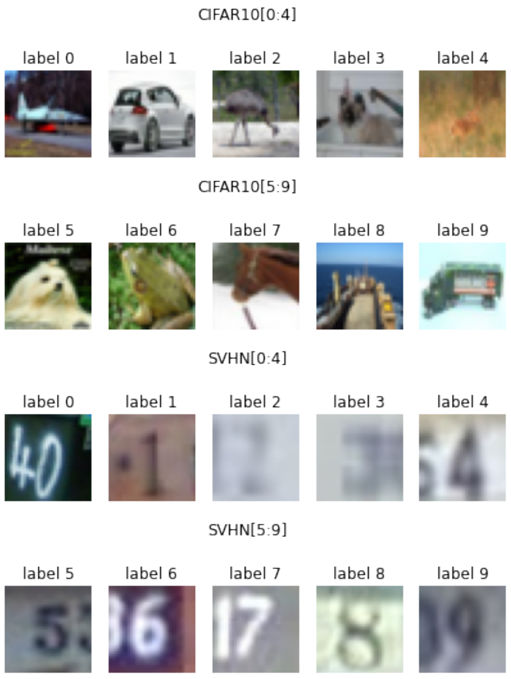
We note that ERD models reach the optimal stopping time within the first 10 epochs on all the data sets that we consider, which amounts to around 6 minutes of training time if the models in the ensemble are fine-tuned in parallel on NVIDIA 1080 Ti GPUs. This is substantially better than the cost of fine-tuning a large ViT transformer model (which takes about 1 hour for 2500 iterations on the same hardware). Moreover, since the loss we use to train the ensemble decouples over the models, it allows for easy parallelization, unlike objectives like MCD where the ensemble models are intertwined.

E ID AND OOD DATA SETS

For evaluation, we use the following image data sets: MNIST [Lecun et al., 1998], Fashion MNIST [Xiao et al., 2017], SVHN [Netzer et al., 2011], CIFAR10 and CIFAR100 [Krizhevsky, 2009].



(a)



(b)

Figure 4: Samples for the settings with novel classes. (a) Data samples for the MNIST/FashionMNIST splits. (b) Data samples for the CIFAR10/SVHN splits.

For the MNIST and FashionMNIST experiments, the training set size is 50K, the validation size is 10K, and the test ID and test OOD sizes are both 10K. For SVHN, CIFAR10 and CIFAR100, the training set size is 40K, the validation size is 10K, and the unlabeled set contains 10K samples: 5K are ID and 5K are OOD. For evaluation, we use a holdout set of 10K examples (half ID, half OOD). When half of the classes are used as ID and the other half as OOD, all the sizes are halved.

F MORE EXPERIMENTS

We now present more experimental results that provide additional insights about the proposed approach. We note that, unless otherwise specified, we use 5-model ERD ensembles in this section.

F.1 EVALUATION ON THE UNLABELED SET

In the main text we describe how one can leverage the unlabeled set U to obtain an novelty detection algorithm that accurately identifies outliers at test time that similar to the ones in U. It is, however, possible to also use our method ERD to flag the OOD samples contained in the same set U used for fine-tuning the ensemble. In Table 2 we show that the novelty detection performance of ERD is similar regardless of whether we use U for evaluation, or a holdout test set T drawn from the same distribution as U.

Table 2: Comparison between the novelty detection performance of ERD when using a holdout test set T for evaluation, or the same unlabeled set U that was used for fine-tuning the models.

ID data	OOD data	ERD (eval on T)	ERD (eval on U)
		AUROC ↑ / TNR@95 ↑	
SVHN	CIFAR10	1.00 / 0.99	1.00 / 0.99
CIFAR10	SVHN	1.00 / 1.00	1.00 / 1.00
CIFAR100	SVHN	1.00 / 1.00	1.00 / 1.00
FMNIST[0,2,3,7,8]	FMNIST[1,4,5,6,9]	0.94 / 0.67	0.94 / 0.67
SVHN[0:4]	SVHN[5:9]	0.95 / 0.74	0.96 / 0.79
CIFAR10[0:4]	CIFAR10[5:9]	0.93 / 0.70	0.93 / 0.69
CIFAR100[0:49]	CIFAR100[50:99]	0.82 / 0.44	0.80 / 0.36
Average		0.95 / 0.79	0.95 / 0.79

F.2 COMPARISON WITH OTHER RELATED WORKS

We compare 5-model ERD ensembles to more OOD detection approaches. For various reasons we did not run these methods ourselves on the data sets for which we evaluate our method in Section 4 (e.g. code not available, unable to replicate published results, poor performance reported by the authors etc). We collected the AUROC numbers presented in Table 3 from the papers that introduce each method. We note that our approach shows an excellent overall performance, being consistently better than or on par with the related works that we consider. While the method of Fort et al. [2021] performs significantly better than all other baselines on CIFAR10/CIFAR100 tasks, we argue in Appendix F.3 that this is primarily due to the convenient choice of the data set used for pretraining the transformer models (i.e. Imagenet21k) which is strikingly similar to the ID and OOD data.

Table 3: AUROC numbers collected from the literature for a number of relevant OOD detection methods. We note that the method of Fort et al. [2021] (†) uses a large scale visual transformer models pretrained on a superset of the OOD data, i.e. ImageNet21k, while the method of Sehwag et al. [2021] (*) uses oracle OOD samples for training from the same data set as test OOD. For the settings with random classes, the numbers are averages over 5 draws and the standard deviation is always strictly smaller than 0.01 for our method.

ID data	OOD data	Fort et al. [2021]†	Zhang et al. [2020a]	Winkens et al. [2020]	Tack et al. [2020]	Sehwag et al. [2021]*	Liu and Abbeel [2020]	Zhang et al. [2020b]	ERD (ours)	ERD++ (ours)
CIFAR10	CIFAR100	98.52	0.95	0.92	0.92	0.93	0.91	-	0.92	0.95
CIFAR100	CIFAR10	96.23	0.85	0.78	-	0.78	-	-	0.91	0.94
SVHN: 6 random classes	SVHN: 4 random classes	-	0.94	-	-	-	-	0.91	0.94	0.94
CIFAR10: 6 random classes	CIFAR10: 4 random classes	-	0.94	-	-	-	-	0.85	0.94	0.97

OpenHybrid [Zhang et al., 2020a] is an open set recognition approach which reports great near OOD detection performance. We note that, despite our best efforts, we did not manage to match in our own experiments the results reported in the paper, even after communicating with the authors and using the code that they have provided. Moreover, we point out that the performance of OpenHybrid seems to deteriorate significantly when the ID data consists of numerous classes, as is the case for CIFAR100.

Furthermore, we note that generative models [Nalisnick et al., 2019, Akçay et al., 2018] and one-class classification approaches [Ruff et al., 2020, Tack et al., 2020, Sohn et al., 2021] showed generally bad performance, in particular on near OOD data. When the ID training set is made up of several diverse classes, it is difficult to represent accurately all the ID data, and only the ID data.

F.3 SHORTCOMINGS OF PRETRAINED VIT MODELS FOR NOVELTY DETECTION

In this section we provide further experimental results pointing to the fact that large pretrained transformer models [Fort et al., 2021] can only detect near OOD samples from certain specific data sets, and do not generalize well more broadly.

Implementation details. We fine-tune visual transformer (ViT) models pretrained on Imagenet21k according the methodology described in Fort et al. [2021]. We report results using the ViT-S-16 architecture (22 million trainable parameters) which we fine-tune for 2500 iterations on labeled ID data. We use the hyperparameters suggested by the authors and always ensure that the prediction accuracy of the fine-tuned model on ID data is in the expected range. The code published by the authors uses three different test statistics to detect OOD data: the maximum softmax probability [Hendrycks and Gimpel, 2017], the vanilla Mahalanobis distance [Lee et al., 2018] and a recently proposed variant of the Mahalanobis approach [Ren et al., 2021]. In Table 4 we present only the metrics obtained with the best-performing test statistic for ViT. We stress that this favors the ViT method significantly, as different test statistics seem to perform better on different data sets. Since test OOD data is unknown, it is not possible to select which test statistic to use a priori, and hence, we use oracle knowledge to give ViT models an unfair advantage.

Experimental results. In Table 4 we compare pretrained visual transformers with 5-model ERD and ERD++ ensembles. Notably, the data sets can be partitioned in two clusters, based on ViT novelty detection performance. On the one hand, if the ID or OOD data comes from CIFAR10 or CIFAR100, ViT models can detect novel-class samples well. Perhaps surprisingly, ViT fails short of detecting OOD data perfectly (i.e. AUROC and TNR@95 of 1) on easy tasks such as CIFAR10 vs SVHN or CIFAR100 vs SVHN, unlike ERD and a number of other baseline approaches.

On the other hand, ViT shows remarkably poor performance on all other data sets, when neither the ID nor the OOD data come from CIFAR10/CIFAR100. This includes some of the novel disease use cases from the medical OOD detection benchmark (see Appendix G for more details about the data sets). This unsatisfactory performance persists even for larger ViT models (we have tried ViT-S-16 and ViT-B-16 architectures), when fine-tuning for more iterations (we have tried both 2500 and 10000 iterations), or when varying hyperparameters such as the learning rate.

Table 4: Pretrained ViT models tend to perform well when the ID and OOD data is semantically similar to (or even included in) the pretraining data, e.g. CIFAR10, CIFAR100 (top part), and their detection performance deteriorates drastically otherwise (bottom part). We compare ViT-S-16 models pretrained on Imagenet21k with 5-model ERD and ERD++ ensembles and ***highlight*** the best method. See Appendix G for more details about the medical data sets.

ID data	OOD data	ViT	ERD	ERD++
		AUROC ↑ / TNR@95 ↑		
SVHN	CIFAR10	0.98 / 0.90	***1.00 / 0.99***	***1.00 / 0.99***
CIFAR10	SVHN	0.99 / 0.98	***1.00 / 1.00***	***1.00 / 1.00***
CIFAR100	SVHN	0.96 / 0.84	***1.00 / 1.00***	***1.00 / 1.00***
CIFAR10[0:4]	CIFAR10[5:9]	***0.97 / 0.83***	0.93 / 0.70	0.96 / 0.79
CIFAR100[0:49]	CIFAR100[50:99]	***0.90 / 0.56***	0.82 / 0.44	0.85 / 0.45
FMNIST[0,2,3,7,8]	FMNIST[1,4,5,6,9]	0.88 / 0.49	0.94 / 0.67	***0.95 / 0.71***
SVHN[0:4]	SVHN[5:9]	0.89 / 0.68	0.95 / 0.74	***0.96 / 0.77***
NIH ID (Xray)	PC (Xray)	0.87 / 0.38	0.94 / 0.63	***0.99 / 0.97***
NIH ID (Xray)	NIH OOD (Xray)	***0.54 / 0.04***	0.46 / 0.04	0.50 / 0.04
PC ID (Xray)	PC UC2 (Xray)	0.89 / 0.61	***0.99*** / 0.99	***0.99 / 1.00***
PC ID (Xray)	PC UC3 (Xray)	0.54 / 0.07	***0.77 / 0.17***	0.72 / 0.08
DRD (fundus)	RIGA (fundus)	0.61 / 0.20	0.91 / 0.73	***1.00 / 0.98***

Intuition for why ViT fails. We conjecture that the novelty detection performance with pretrained ViT models relies heavily on the choice of the pretraining data set. In particular, we hypothesize that, since CIFAR10/CIFAR100 classes are included in the Imagenet21k data set used for pretraining, the models learn features that are useful for distinguishing ID and OOD classes when the ID and/or OOD data comes from CIFAR10/CIFAR100. Hence, this would explain the good performance of pretrained models on the data sets at the top of Table 4. On the other hand, when ID and OOD data is strikingly different from the pretraining data, both ID and OOD samples are projected to the same concentrated region of the representation space, which makes it difficult to detect novel-class points. Moreover, the process of fine-tuning as it is described in Fort et al. [2021] seems to not help to alleviate this problem. This leads to the poor performance observed on the near OOD data sets at the bottom of Table 4.

In conclusion, having a large pretraining data set seems to be beneficial when the OOD data shares many visual and semantic features in common with the pretraining data. However, in real-world applications it is often difficult to collect such large data sets, which makes the applicability of pretrained ViT models limited to only certain specific scenarios.

F.4 OOD DETECTION FOR DATA WITH COVARIATE SHIFT

In this section we evaluate the baselines and the method that we propose on settings in which the OOD data suffers from covariate shift. The goal is to identify all samples that come from the shifted distribution, regardless of how strong the shift is. Notice that mild shifts may be easier to tackle by domain adaptation algorithms, but when the goal is OOD detection they pose a much more difficult challenge.

We want to stress that in practice one may not be interested in identifying *all* samples with distribution shift as OOD, since a classifier may still produce correct predictions on some of them. In contrast, when data suffers from covariate shift we can try to learn predictors that perform well on both the training and the test distribution, and we may use a measure of predictive uncertainty to identify only those test samples on which the classifier cannot make confident predictions. Nevertheless, we use these covariate shift settings as a challenging OOD detection benchmark and show in Table 6 that our method ERD does indeed outperform prior baselines on these difficult settings.

We use as outliers corrupted variants of CIFAR10 and CIFAR100 [Hendrycks and Dietterich, 2019], as well as a scenario where ImageNet [Deng et al., 2009] is used as ID data and ObjectNet [Barbu et al., 2019] as OOD, both resized to 32x32. Figure 5 shows samples from these data sets. The Gram and nnPU baselines do not give satisfactory results on the difficult CIFAR10/CIFAR100 settings in Table 1 and thus we do not consider them for the covariate shift cases. For the SSND methods (e.g. MCD, Mahal-U and ERD/ERD++) we evaluate on the same unlabeled set that is used for training (see the discussion in Section F.1).

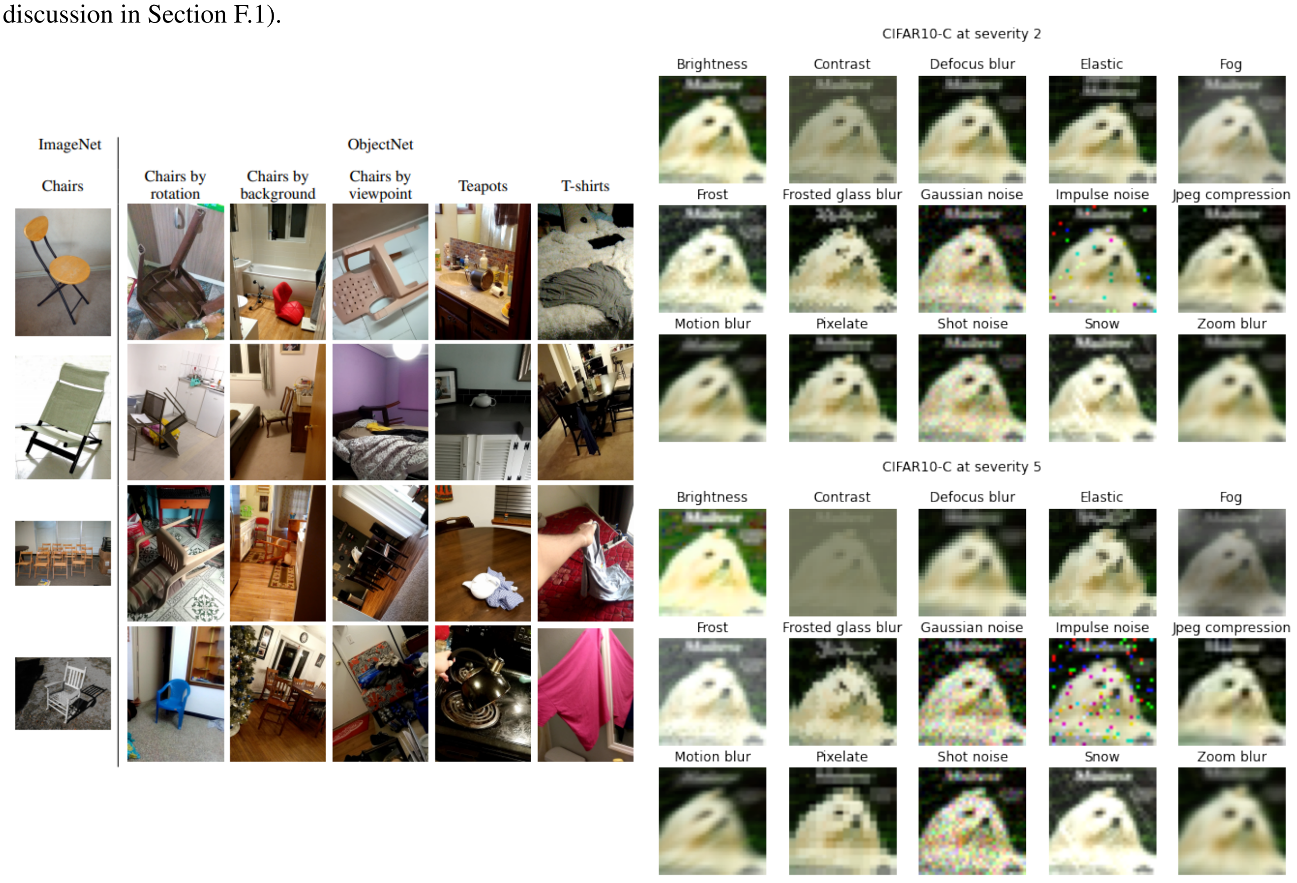


Figure 5: Left: Samples from ObjectNet. Figure from Barbu et al. [2019]. Right: Corrupted samples from CIFAR10-C.

Furthermore, we present results on distinguishing between CIFAR10 [Krizhevsky, 2009] and CIFAR10v2 [Recht et al., 2019], a data set meant to be drawn from the same distribution as CIFAR10 (generated from the Tiny Images collection). In Recht et al. [2019], the authors argue that CIFAR10 and CIFAR10v2 come from very similar distributions. They provide supporting evidence by training a binary classifier to distinguish between them, and observing that the accuracy that is obtained of 52.9% is very close to random.

Our experiments show that the two data sets are actually distinguishable, contrary to what previous work has argued. First, our own binary classifier trained on CIFAR10 vs CIFAR10v2 obtains a test accuracy of 67%, without any hyperparameter tuning. The model we use is a ResNet20 trained for 200 epochs using SGD with momentum 0.9. The learning rate is decayed by 0.2 at epochs 90, 140, 160 and 180. We use 1600 examples from each data set for training, and we validate using 400 examples from each data set.

Table 5: OOD detection performance on CIFAR10 vs CIFAR10v2. We highlight the **best ERD** variant and the ***best baseline***.

ID data	OOD data	Vanilla Ensembles	DPN	OE	Mahal.	MCD	Mahal-U	ERD	ERD++
					AUROC ↑ / TNR@95 ↑				
CIFAR10	CIFAR10v2	***0.64*** / ***0.13***	0.63 / 0.09	***0.64*** / 0.12	0.55 / 0.08	0.58 / 0.10	0.56 / 0.07	0.76 / 0.26	**0.91** / **0.80**

Our OOD detection experiments (presented in Table 5) show that most baselines are able to distinguish between the two data sets, with ERD achieving the highest performance. The methods which require OOD data for tuning (Outlier Exposure and DPN) use CIFAR100.

Table 6: OOD detection performance on data with covariate shift. For ERD and vanilla ensembles, we train 5 ResNet20 models for each setting. The evaluation metrics are computed on the unlabeled set. We highlight the **best ERD** variant and the ***best baseline***.

ID data	OOD data	Vanilla Ensembles	DPN	OE	Mahal.	MCD	Mahal-U	ERD	ERD++
					AUROC ↑ / TNR@95 ↑				
CIFAR10	CIFAR10-C sev 2 (A)	0.68 / 0.20	0.73 / 0.31	0.70 / 0.20	***0.84*** / ***0.53***	0.82 / 0.50	0.75 / 0.38	0.96 / 0.86	**0.99** / **0.95**
CIFAR10	CIFAR10-C sev 2 (W)	0.51 / 0.05	0.47 / 0.03	0.52 / 0.06	***0.58*** / ***0.08***	0.52 / 0.06	0.55 / 0.07	0.68 / 0.19	**0.86** / **0.41**
CIFAR10	CIFAR10-C sev 5 (A)	0.84 / 0.49	0.89 / 0.60	0.86 / 0.54	0.94 / 0.80	***0.95*** / ***0.84***	0.88 / 0.63	**1.00** / 0.99	**1.00** / **1.00**
CIFAR10	CIFAR10-C sev 5 (W)	0.60 / 0.10	0.72 / 0.10	0.63 / 0.11	***0.78*** / ***0.27***	0.60 / 0.08	0.68 / 0.12	0.98 / 0.86	**1.00** / **1.00**
CIFAR100	CIFAR100-C sev 2 (A)	0.68 / 0.20	0.62 / 0.18	0.65 / 0.19	***0.82*** / ***0.48***	0.72 / 0.29	0.67 / 0.22	0.94 / 0.76	**0.97** / **0.86**
CIFAR100	CIFAR100-C sev 2 (W)	0.52 / 0.06	0.32 / 0.03	0.52 / 0.06	***0.55*** / ***0.07***	0.52 / 0.06	0.55 / 0.06	0.71 / 0.19	**0.86** / **0.44**
CIFAR100	CIFAR100-C sev 5 (A)	0.78 / 0.37	0.74 / 0.36	0.76 / 0.37	***0.92*** / ***0.72***	0.91 / 0.65	0.84 / 0.55	0.99 / 0.97	**1.00** / **0.99**
CIFAR100	CIFAR100-C sev 5 (W)	0.64 / 0.14	0.49 / 0.12	0.62 / 0.13	***0.71*** / ***0.19***	0.60 / 0.10	0.63 / 0.13	0.96 / 0.71	**0.98** / **0.89**
Tiny ImageNet	Tiny ObjectNet	0.82 / 0.49	0.70 / 0.32	0.79 / 0.37	0.75 / 0.26	***0.99*** / ***0.98***	0.72 / 0.25	0.98 / 0.88	**0.99** / **0.98**
	Average	0.67 / 0.23	0.63 / 0.23	0.67 / 0.23	***0.76*** / 0.38	0.74 / ***0.39***	0.70 / 0.27	0.91 / 0.71	**0.96** / **0.83**

F.5 RESULTS WITH A SMALLER UNLABELED SET

We now show that our method performs well even when the unlabeled set is significantly smaller. In particular, we show in the table below that ERD maintains a high AUROC and TNR@95 even when only 1,000 unlabeled samples are used for fine-tuning (500 ID and 500 OOD).

Table 7: Experiments with a test set of size 1,000, with an equal number of ID and OOD test samples. For ERD and vanilla ensembles, we train 5 ResNet20 models for each setting. The evaluation metrics are computed on the unlabeled set. We highlight **ERD** and the ***best baseline***.

ID data	OOD data	Vanilla Ensembles	DPN	OE	Mahal.	MCD	Mahal-U	ERD
					AUROC ↑ / TNR@95 ↑			
SVHN	CIFAR10	0.97 / 0.88	***1.00*** / ***1.00***	***1.00*** / ***1.00***	0.99 / 0.98	0.97 / 0.85	0.99 / 0.95	**1.00** / **0.99**
CIFAR10	SVHN	0.92 / 0.78	0.95 / 0.85	0.97 / 0.89	***0.99*** / ***0.96***	1.00 / 0.98	0.99 / 0.96	**1.00** / **1.00**
CIFAR100	SVHN	0.84 / 0.48	0.77 / 0.44	0.82 / 0.50	***0.98*** / ***0.90***	0.97 / 0.73	0.98 / 0.92	**0.99** / **1.00**
SVHN[0:4]	SVHN[5:9]	***0.92*** / 0.69	0.87 / 0.19	0.85 / 0.52	***0.92*** / ***0.71***	0.91 / 0.51	0.91 / 0.63	**0.97** / **0.86**
CIFAR10[0:4]	CIFAR10[5:9]	0.80 / 0.39	***0.82*** / 0.32	***0.82*** / ***0.41***	0.79 / 0.27	0.69 / 0.25	0.64 / 0.13	**0.87** / **0.50**
CIFAR100[0:49]	CIFAR100[50:99]	***0.78*** / ***0.35***	0.70 / 0.26	0.74 / 0.31	0.72 / 0.20	0.70 / 0.26	0.72 / 0.19	**0.79** / **0.38**
CIFAR10	CIFAR10-C sev 2 (A)	0.68 / 0.20	0.73 / 0.31	0.70 / 0.20	***0.84*** / ***0.53***	0.82 / 0.50	0.75 / 0.38	**0.91** / **0.71**
CIFAR10	CIFAR10-C sev 2 (W)	0.51 / 0.05	0.47 / 0.03	0.52 / 0.06	***0.58*** / ***0.08***	0.52 / 0.06	0.55 / 0.07	**0.57** / **0.09**
CIFAR10	CIFAR10-C sev 5 (A)	0.84 / 0.49	0.89 / 0.60	0.86 / 0.54	***0.94*** / ***0.80***	0.95 / 0.84	0.88 / 0.63	**0.99** / **0.95**
CIFAR10	CIFAR10-C sev 5 (W)	0.60 / 0.10	0.72 / 0.10	0.63 / 0.11	***0.78*** / ***0.27***	0.60 / 0.08	0.68 / 0.12	**0.92** / **0.67**
CIFAR100	CIFAR100-C sev 2 (A)	0.68 / 0.20	0.62 / 0.18	0.65 / 0.19	***0.82*** / ***0.48***	0.72 / 0.29	0.67 / 0.22	**0.84** / **0.48**
CIFAR100	CIFAR100-C sev 2 (W)	0.52 / 0.06	0.32 / 0.03	0.52 / 0.06	***0.55*** / ***0.07***	0.52 / 0.06	**0.55** / 0.06	**0.55** / **0.07**
CIFAR100	CIFAR100-C sev 5 (A)	0.78 / 0.37	0.74 / 0.36	0.76 / 0.37	***0.92*** / ***0.72***	0.91 / 0.65	0.84 / 0.55	**0.96** / **0.80**
CIFAR100	CIFAR100-C sev 5 (W)	0.64 / 0.14	0.49 / 0.12	0.62 / 0.13	***0.71*** / ***0.19***	0.60 / 0.10	0.63 / 0.13	**0.81** / **0.25**
	Average	0.75 / 0.37	0.72 / 0.34	0.75 / 0.38	***0.82*** / ***0.51***	0.78 / 0.44	0.77 / 0.42	**0.87** / **0.62**

F.6 MORE RESULTS FOR OUTLIER EXPOSURE

The Outlier Exposure method needs access to a set of OOD samples during training. The numbers we report in the rest of paper for Outlier Exposure are obtained by using the TinyImages data set as the OOD samples that are seen during training. In this section we explore the use of an $\text{OOD}_{\text{train}}$ data set that is more similar to the OOD data observed at test time. This is a much easier setting for the Outlier Exposure method: the closer $\text{OOD}_{\text{train}}$ is to OOD_{test}, the easier it will be for the model tuned on $\text{OOD}_{\text{train}}$ to detect the test OOD samples.

In the table below we focus only on the settings with corruptions. For each corruption type, we use the lower severity corruption as $\text{OOD}_{\text{train}}$ and evaluate on the higher severity data and vice versa. We report for each metric the average taken over all corruptions (A), and the value for the worst-case setting (W).

ID data	OOD data	OE (trained on sev5)	OE (trained on sev2)
		AUROC ↑	
CIFAR10	CIFAR10-C sev 2 (A)	0.89	N/A
CIFAR10	CIFAR10-C sev 2 (W)	0.65	N/A
CIFAR10	CIFAR10-C sev 5 (A)	N/A	0.98
CIFAR10	CIFAR10-C sev 5 (W)	N/A	0.78
CIFAR100	CIFAR100-C sev 2 (A)	0.85	N/A
CIFAR100	CIFAR100-C sev 2 (W)	0.59	N/A
CIFAR100	CIFAR100-C sev 5 (A)	N/A	0.97
CIFAR100	CIFAR100-C sev 5 (W)	N/A	0.67
	Average	0.87	0.98

Table 8: Results for Outlier Exposure, when using the same corruption type, but with a higher/lower severity, as OOD data seen during training.

F.7 RESULTS ON MNIST AND FASHIONMNIST

Table 9: Results on MNIST/FashionMNIST settings. For ERD and vanilla ensembles, we train five 3-hidden layer MLP models for each setting. The evaluation metrics are computed on the unlabeled set. We highlight the **best ERD** variant and the ***best baseline***.

ID data	OOD data	Vanilla Ensembles	DPN	OE	Mahal.	nnPU	MCD	Mahal-U	Bin. Classif.	ERD	ERD++
		AUROC ↑ / TNR@95 ↑									
MNIST	FMNIST	0.81 / 0.01	***1.00*** / ***1.00***	***1.00*** / ***1.00***	***1.00*** / ***1.00***	***1.00*** / ***1.00***	***1.00*** / 0.98	***1.00*** / ***1.00***	1.00 / 1.00	**1.00** / **1.00**	**1.00** / **1.00**
FMNIST	MNIST	0.87 / 0.42	***1.00*** / ***1.00***	0.68 / 0.16	0.99 / 0.97	***1.00*** / ***1.00***	***1.00*** / ***1.00***	0.99 / 0.96	1.00 / 1.00	**1.00** / **1.00**	**1.00** / **1.00**
MNIST[0:4]	MNIST[5:9]	0.94 / 0.72	***0.99*** / 0.97	0.95 / 0.78	***0.99*** / ***0.98***	***0.99*** / 0.97	0.96 / 0.76	***0.99*** / ***0.98***	0.99 / 0.94	**0.99** / 0.96	**0.99** / **0.97**
FMNIST[0,2,3,7,8]	FMNIST[1,4,5,6,9]	0.64 / 0.07	0.77 / 0.15	0.66 / 0.12	0.77 / 0.20	***0.95*** / ***0.71***	0.78 / 0.30	0.82 / 0.39	0.95 / 0.66	**0.94** / 0.67	**0.94** / **0.68**
Average		0.82 / 0.30	0.94 / 0.78	0.82 / 0.51	0.94 / 0.79	***0.98*** / ***0.92***	0.94 / 0.76	0.95 / 0.83	0.98 / 0.90	**0.98** / **0.91**	**0.98** / **0.91**

For FashionMNIST we chose this particular split (i.e. classes 0,2,3,7,8 vs classes 1,4,5,6,9) because the two partitions are more similar to each other. This makes novelty detection more difficult than the 0-4 vs 5-9 split.

F.8 VANILLA AND ERD ENSEMBLES WITH DIFFERENT ARCHITECTURES

In this section we present OOD detection results for 5-model Vanilla and ERD ensembles with different architecture choices, and note that the better performance of our method is maintained across model classes. Moreover, we observe that ERD benefits from employing more complex models, like the WideResNet.

Table 10: Results with three different architectures for Vanilla and ERD ensembles. All ensembles comprise 5 models. For the corruption data sets, we report for each metric the average taken over all corruptions (A), and the value for the worst-case setting (W). The evaluation metrics are computed on the unlabeled set.

		VGG16		ResNet20		WideResNet-28-10	
ID data	OOD data	Vanilla Ensembles	ERD	Vanilla Ensembles	ERD	Vanilla Ensembles	ERD
				AUROC ↑ / TNR@95 ↑			
SVHN	CIFAR10	0.97 / 0.88	0.99 / 0.94	0.97 / 0.88	0.99 / 0.97	0.96 / 0.86	1.00 / 0.99
CIFAR10	SVHN	0.88 / 0.69	1.00 / 1.00	0.92 / 0.78	1.00 / 1.00	0.94 / 0.81	1.00 / 1.00
SVHN[0:4]	SVHN[5:9]	0.89 / 0.60	0.93 / 0.63	0.92 / 0.69	0.94 / 0.66	0.91 / 0.62	0.96 / 0.78
CIFAR10[0:4]	CIFAR10[5:9]	0.74 / 0.29	0.91 / 0.63	0.80 / 0.39	0.91 / 0.66	0.80 / 0.35	0.94 / 0.71
CIFAR10	CIFAR10-C sev 2 (A)	0.66 / 0.17	0.94 / 0.79	0.68 / 0.20	0.96 / 0.86	0.69 / 0.18	0.98 / 0.90
CIFAR10	CIFAR10-C sev 2 (W)	0.51 / 0.05	0.68 / 0.19	0.51 / 0.05	0.68 / 0.19	0.51 / 0.05	0.84 / 0.35
CIFAR10	CIFAR10-C sev 5 (A)	0.80 / 0.41	0.99 / 0.96	0.84 / 0.49	1.00 / 0.99	0.84 / 0.47	1.00 / 1.00
CIFAR10	CIFAR10-C sev 5 (W)	0.58 / 0.10	0.95 / 0.72	0.60 / 0.10	0.98 / 0.86	0.59 / 0.09	0.99 / 0.97
Average		0.75 / 0.40	0.92 / 0.73	0.78 / 0.45	0.93 / 0.77	0.78 / 0.43	0.96 / 0.84

F.9 IMPACT OF THE ENSEMBLE SIZE AND OF THE CHOICE OF ARBITRARY LABEL

In this section we show novelty detection results with our method using a smaller number of models for the ensembles. We notice that the performance is not affected substantially, indicating that the computation cost of our approach could be further reduced by fine-tuning smaller ensembles.

Table 11: Results obtained with smaller ensembles for ERD. The numbers for $K < 5$ are averages over 3 runs, where we use a different set of arbitrary labels for each run to illustrate our method's stability with respect the choice of labels to be assigned to the unlabeled set. We note that the standard deviations are small ($\sigma \leq 0.01$ for the AUROC values and $\sigma \leq 0.08$ for the TNR@95 values).

		K=2		K=3		K=4		K=5	
ID data	OOD data	ERD	ERD++	ERD	ERD++	ERD	ERD++	ERD	ERD++
				AUROC ↑ / TNR@95 ↑					
SVHN	CIFAR10	0.99 / 0.98	0.99 / 0.99	0.99 / 0.98	1.00 / 0.99	0.99 / 0.98	1.00 / 0.99	1.00 / 0.99	1.00 / 0.99
CIFAR10	SVHN	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00
CIFAR100	SVHN	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00
SVHN[0:4]	SVHN[5:9]	0.95 / 0.69	0.94 / 0.68	0.95 / 0.73	0.95 / 0.75	0.96 / 0.76	0.96 / 0.77	0.95 / 0.74	0.96 / 0.77
CIFAR10[0:4]	CIFAR10[5:9]	0.89 / 0.55	0.92 / 0.58	0.89 / 0.57	0.94 / 0.70	0.90 / 0.57	0.95 / 0.73	0.93 / 0.70	0.96 / 0.79
CIFAR100[0:49]	CIFAR100[50:99]	0.81 / 0.40	0.82 / 0.43	0.81 / 0.41	0.84 / 0.44	0.81 / 0.41	0.84 / 0.44	0.82 / 0.44	0.85 / 0.45
Average		0.94 / 0.77	0.95 / 0.78	0.94 / 0.78	0.95 / 0.81	0.94 / 0.79	0.96 / 0.82	0.95 / 0.81	0.96 / 0.83

Impact of the choice of arbitrary labels. Furthermore, we note that in the table we report averages over 3 runs of our method, where for each run we use a different subset of $\mathcal{Y}$ to assign arbitrary labels to the unlabeled data. We do this in order to assess the stability of ERD ensembles to the choice of the arbitrary labels and notice that the novelty detection performance metrics do not vary significantly. Concretely, the standard deviations are consistently below 0.01 for all data sets for the AUROC metric, and below 0.07 for the TNR@95 metric.

F.10 DETECTION PERFORMANCE ON DIFFERENT OOD DATA

In this section we investigate whether the proposed method maintains its good novelty detection performance when the test-time OOD data comes from a different data set compared to the OOD data that is present in the unlabeled set used for fine-tuning. In particular, we are interested if our approach can still identify outliers in situations when they suffer from various corruptions. This scenario can sometimes occur in practice, when machine failure or uncurated data can lead to mild distribution shift.

Concretely, we focus on the difficult near OOD scenarios and take as ID half of the CIFAR10 or CIFAR100 classes, while the other half is OOD. For this experiment, we fine-tune the ERD ensembles using clean OOD data from the other half of CIFAR10 and CIFAR100, respectively. For evaluation, we use clean ID data and corrupted OOD samples from CIFAR10-C and CIFAR100-C, respectively. We give more details on these corrupted data sets in Appendix F.4. We consider corruptions of severity 2 and 5 from all corruptions types.

In Table 12 we show the average AUROC and the worst AUROC over all corruption types for vanilla and ERD ensembles. Note that our approach maintains a similar performance compared to the numbers presented in Table 1 for same test-time OOD data. It is also noteworthy that all the average AUROC values are consistently larger than the baselines in Table 1.

Table 12: Results obtained when evaluating on an OOD data set different from the one used for fine-tuning. All ERD ensembles are tuned on clean ID and OOD data and are evaluated on OOD data with corruptions.

ID data	OOD data in unlabeled set	Test-time OOD data	Vanilla Ensemble	ERD
			AUROC ↑	
CIFAR10[0:4]	CIFAR10[5:9]	CIFAR10[5:9]-C sev 2 (A)	0.82	0.93
CIFAR10[0:4]	CIFAR10[5:9]	CIFAR10[5:9]-C sev 2 (W)	0.77	0.88
CIFAR10[0:4]	CIFAR10[5:9]	CIFAR10[5:9]-C sev 5 (A)	0.85	0.91
CIFAR10[0:4]	CIFAR10[5:9]	CIFAR10[5:9]-C sev 5 (W)	0.79	0.86
CIFAR100[0:49]	CIFAR100[50:99]	CIFAR100[50:99]-C sev 2 (A)	0.78	0.84
CIFAR100[0:49]	CIFAR100[50:99]	CIFAR100[50:99]-C sev 2 (W)	0.75	0.78
CIFAR100[0:49]	CIFAR100[50:99]	CIFAR100[50:99]-C sev 5 (A)	0.77	0.83
CIFAR100[0:49]	CIFAR100[50:99]	CIFAR100[50:99]-C sev 5 (W)	0.63	0.78

G MEDICAL OOD DETECTION BENCHMARK

The medical OOD detection benchmark is organized as follows. There are four training (ID) data sets, from three different domains: two data sets with chest X-rays, one with fundus imaging and one with histology images. For each ID data set, the authors consider three different OOD scenarios:

1. Use case 1: The OOD data set contains images from a completely different domain, similar to our category of easy OOD detection settings.
2. Use case 2: The OOD data set contains images with various corruptions, similar to the hard covariate shift settings that we consider in Section F.4.
3. Use case 3: The OOD data set contains images that come from novel classes, not seen during training.

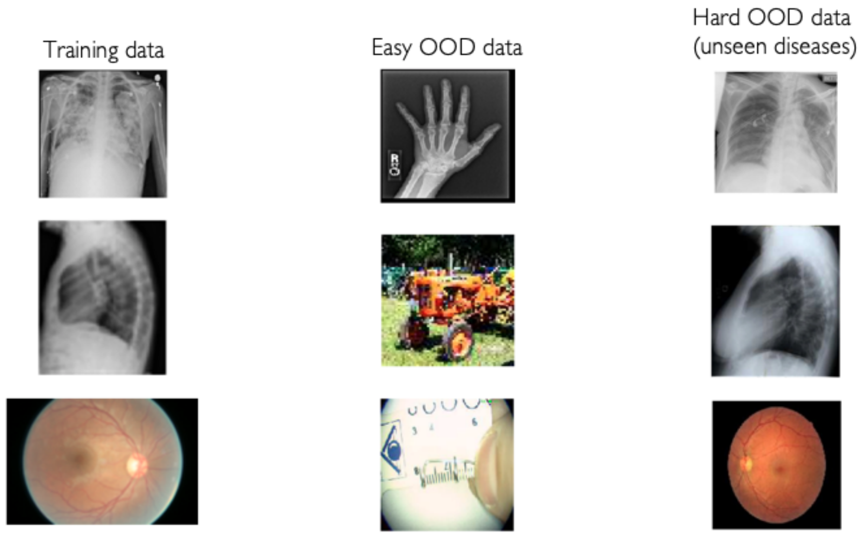


Figure 6: Samples from the medical image benchmark. There are 3 ID data sets containing frontal and lateral chest X-rays and retinal images. Hard OOD samples contain images of diseases that are not present in the training set.

The authors evaluate a number of methods on all these scenarios. The methods can be roughly categorized as follows:

1. Data-only methods: Fully non-parametric approaches like kNN.
2. Classifier-only methods: Methods that use a classifier trained on the training set, e.g. ODIN [Liang et al., 2018], Mahalanobis [Lee et al., 2018]. ERD falls into this category as well.
3. Methods with Auxiliary Models: Methods that use an autoencoder or a generative model, like a Variational Autoencoder or a Generative Adversarial Network. Some of these approaches can be expensive to train and difficult to optimize and tune.

We stress the fact that for most of these methods the authors use (known) OOD data during training. Oftentimes the OOD samples observed during training come from a data set that is very similar to the OOD data used for evaluation. For exact details regarding the data sets and the methods used for the benchmark, we refer the reader to Cao et al. [2020].

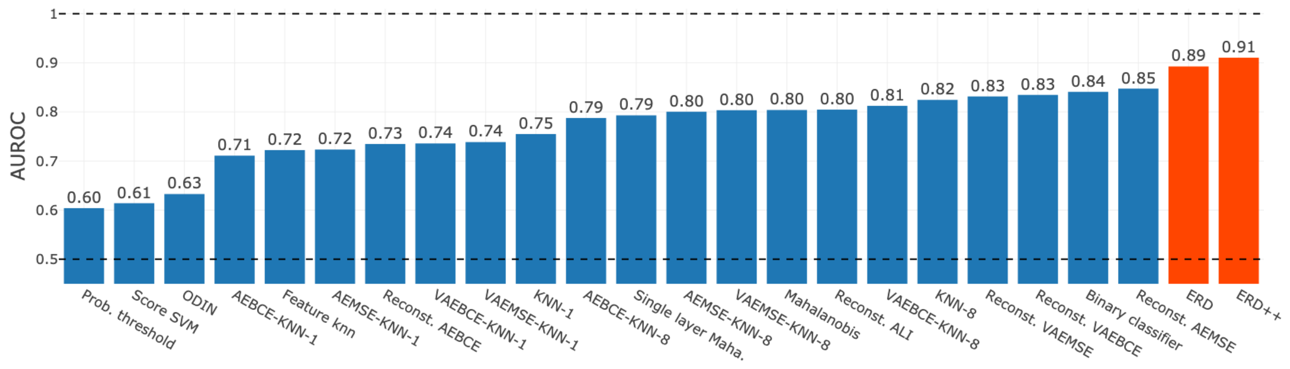


Figure 7: AUROC averaged over all scenarios in the medical OOD detection benchmark [Cao et al., 2020]. The values for all the baselines are computed using code made available by the authors of Cao et al. [2020]. Notably, most of the baselines assume oracle knowledge of OOD data at training time.

In addition, in Figure 8 we present the average taken over only the novel-class settings in the medical benchmark. We observe that the performance of all methods is drastically affected, all of them performing much worse than the average presented in Figure 7. This stark decrease in AUROC and TNR@95 indicates that novelty detection is indeed a challenging task for OOD detection methods even in realistic settings. Nevertheless, 2-model ERD ensembles maintain a better performance than the baselines.

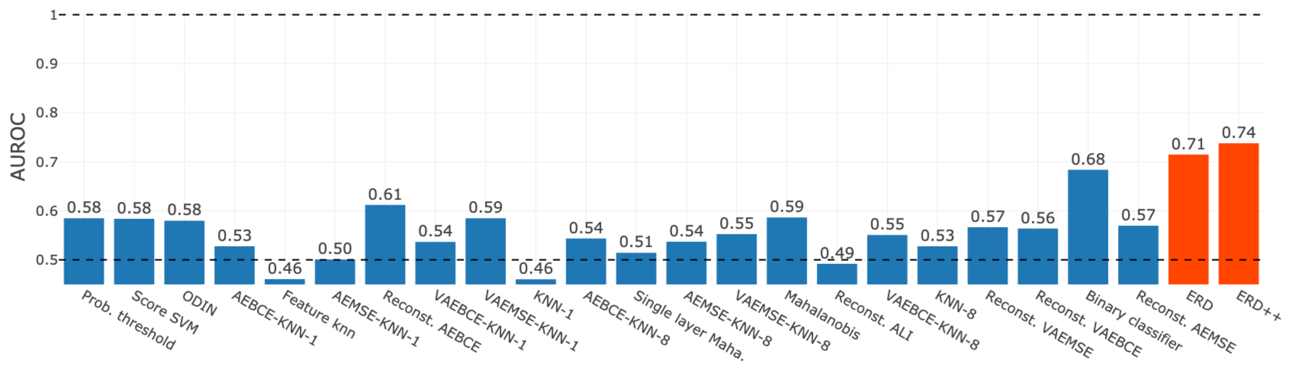


Figure 8: AUROC averaged over the novel-class scenarios in the medical OOD detection benchmark [Cao et al., 2020], i.e. only use case 3.

In Figures 9, 10, 11 we present AUROC and AUPR (Area under the Precision Recall curve) for ERD for each of the training data sets, and each of the use cases. Figure 7 presents averages over all settings that we considered, for all the baseline methods in the benchmark. Notably, ERD performs well consistently across data sets. The baselines are ordered by their average performance on all the settings (see Figure 7).

For all medical benchmarks, the unlabeled set is balanced, with an equal number of ID and OOD samples (subsampling the bigger data set, if necessary). We use the unlabeled set for evaluation.

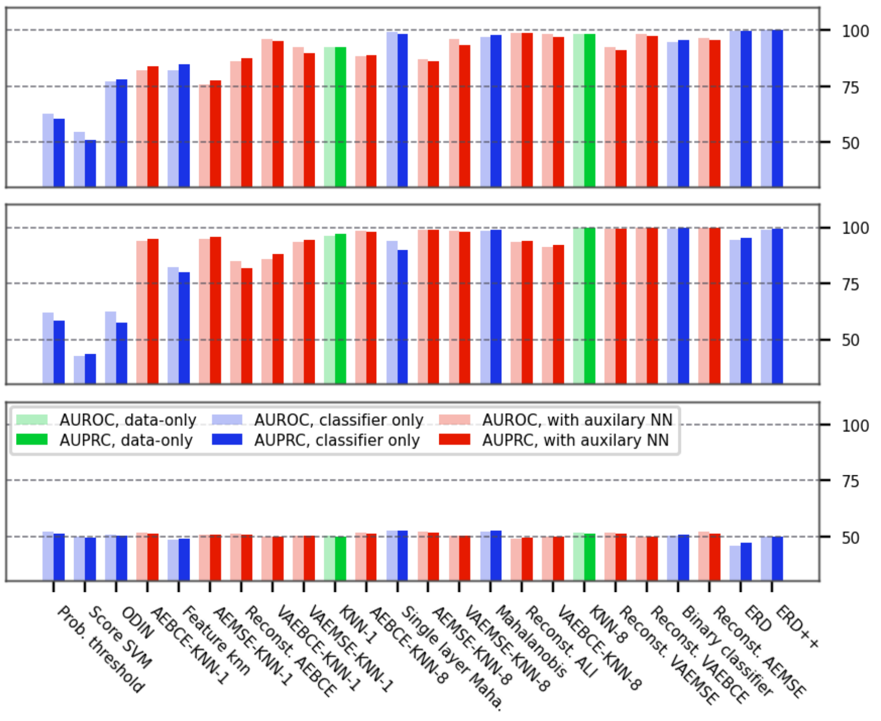


Figure 9: Comparison between ERD and the various baselines on the NIH chest X-ray data set, for use case 1 (top), use case 2 (middle) and use

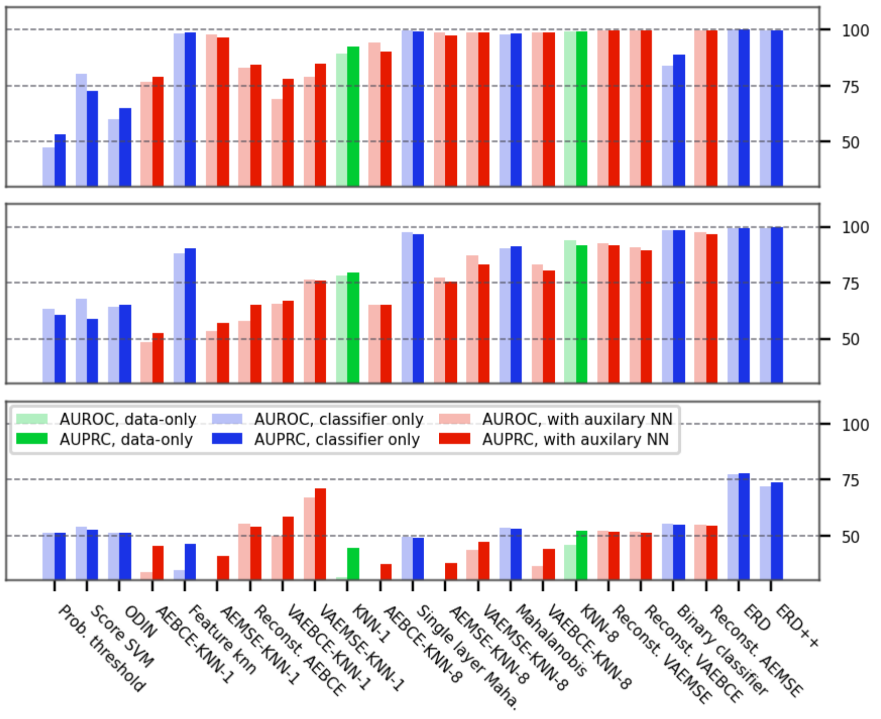


Figure 10: Comparison between ERD and the various baselines on the PC chest X-ray data set, for use case 1 (top), use case 2 (middle) and use case 3 (bottom). Baselines ordered as in Figure 7.

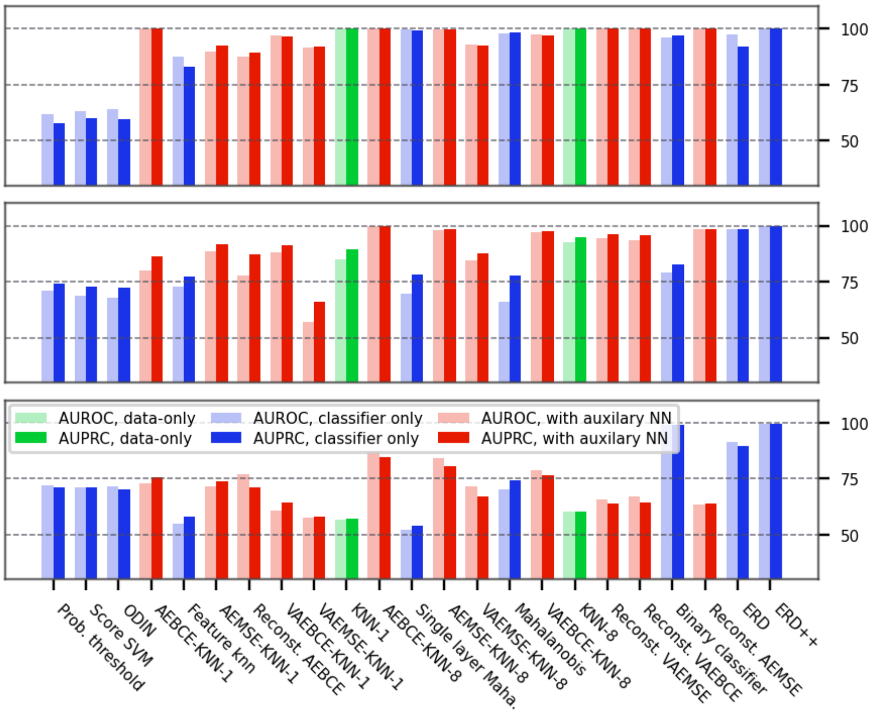
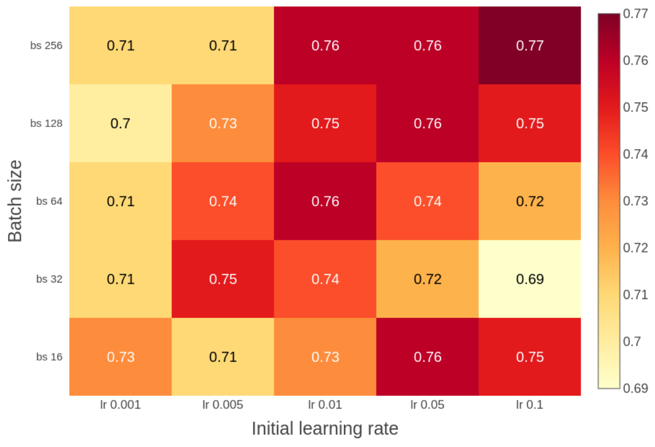


Figure 11: Comparison between ERD and the various baselines on the DRD fundus imaging data set, for use case 1 (top), use case 2 (middle) and use case 3 (bottom). Baselines ordered as in Figure 7.

H EFFECT OF LEARNING RATE AND BATCH SIZE

We show now that ERD ensembles are not too sensitive to the choice of hyperparameters. We illustrate this by varying the learning rate and the batch size, the hyperparameters that we identify as most impactful. As Figure 12 shows, many different configurations lead to similar novelty detection performance.



Batch size	lr 0.001	lr 0.005	lr 0.01	lr 0.05	lr 0.1
bs 256	0.71	0.71	0.76	0.76	0.77
bs 128	0.7	0.73	0.75	0.76	0.75
bs 64	0.71	0.74	0.76	0.74	0.72
bs 32	0.71	0.75	0.74	0.72	0.69
bs 16	0.73	0.71	0.73	0.76	0.75

Figure 12: AUROCs obtained with an ensemble of WRN-28-10 models, as the initial learning rate and the batch size are varied. We used the hardest setting, CIFAR100:0-50 as ID, and CIFAR100:50-100 as OOD.

I ADDITIONAL FIGURE SHOWING THE DEPENDENCE ON THE UNLABELED SET CONFIGURATION

The configuration of the unlabeled set (i.e. the size of the unlabeled set, the ratio of OOD samples in the unlabeled set) influences the performance of our method, as illustrated in Figure 5b. Below, we show that the same trend persists for different data sets too, e.g. when we consider CIFAR10 as ID data and SVHN as OOD data.

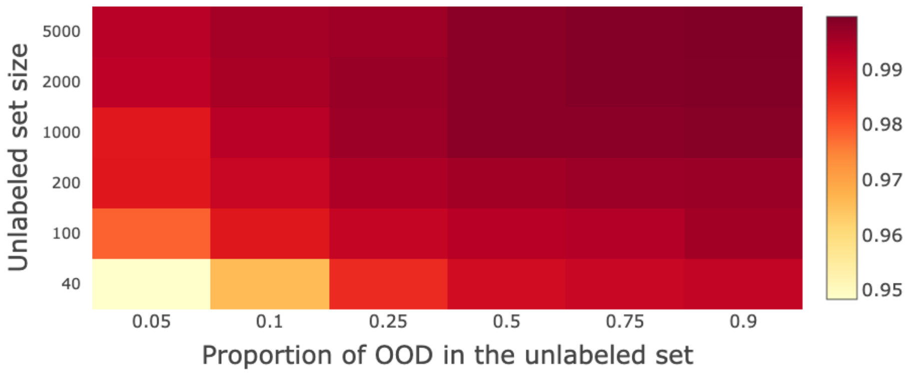


Figure 13: The AUROC of a 3-model ERD ensemble as the number and proportion of ID (CIFAR10) and OOD (SVHN) samples in the unlabeled set are varied.

J LEARNING CURVES FOR OTHER DATA SETS

In addition to Figure 4, we present in this section learning curves for other data sets as well. The trend that persists throughout all figures is that the arbitrary label is learned first on the unlabeled OOD data. Choosing a stopping time before the validation accuracy starts to deteriorate prevents the model from fitting the arbitrary label on unlabeled ID data.

Impact of near OOD data on training ERD ensembles. The learning curves illustrated in Figure 14 provide insight into what happens when the OOD data is similar to the ID training samples and the impact that has on training the proposed method. In particular, notice that for CIFAR10[0-4] vs CIFAR10[5-9] in Figure 14d, the models require more training epochs before reaching an accuracy on unlabeled OOD samples of 100%. The learning of the arbitrary label on the OOD samples is delayed by the fact that the ID and OOD data are similar, and hence, the bias of the correctly labeled training set has a strong effect on the predictions of the models on the OOD inputs. Since we early stop when the validation accuracy starts deteriorating (e.g. at around epoch 8 in Figure 14d), we end up using models that do not interpolate the arbitrary label on the OOD samples. Therefore, the ensemble does not disagree on the entirety of the OOD data in the unlabeled set, which leads to lower novelty detection performance. Importantly, however, our empirical evaluation reveals that the drop in performance for ERD ensembles is substantially smaller than what we observe for other OOD detection methods, even on near OOD data sets.

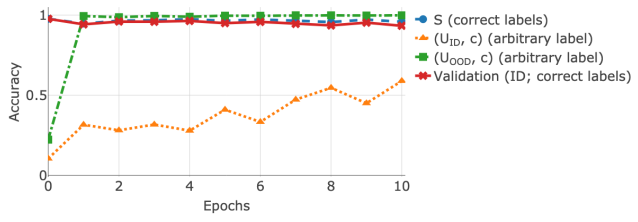
K EVOLUTION OF DISAGREEMENT SCORE DURING FINE-TUNING

In this section we illustrate how the distribution of the disagreement score changes during fine-tuning for ID and OOD data, for a 5-model ERD ensemble. Thus, we can further understand why the performance of the ERD ensembles is impacted by near OOD data.

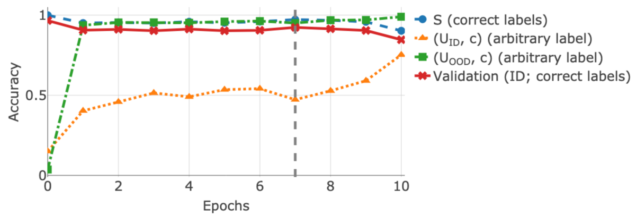
Figure 15 reveals that for far OOD data (the left column) the disagreement scores computed on OOD samples are well separated from the disagreement scores on ID data (note that disagreement on OOD data is so concentrated around the maximum value of 2 that the boxes are essentially reduced to a line segment). On the other hand, for near OOD data (the right column) there is sometimes significant overlap between the disagreement scores on ID and OOD data, which leads to the slightly lower AUROC values that we report in Table 1.

The figures also illustrate how the disagreement on the ID data tends to increase as we fine-tune the ensemble for longer, as a consequence of the models fitting the arbitrary labels on the unlabeled ID samples. Conversely, in most instances one epoch suffices for fitting the arbitrary label on the OOD data.

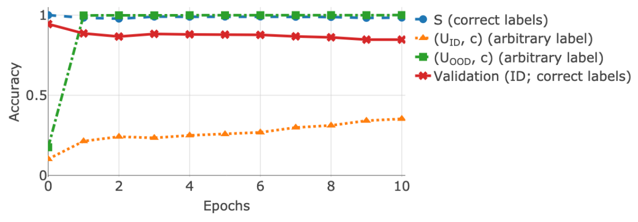
We need to make one important remark: While in the figure we present disagreement scores for the ensemble obtained after each epoch of fine-tuning, we stress that the final ERD ensemble need not be selected among these. In particular, since each



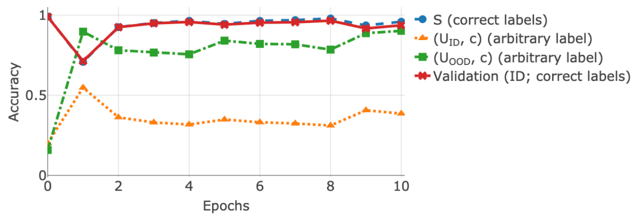
(a) ID = SVHN; OOD = CIFAR10.



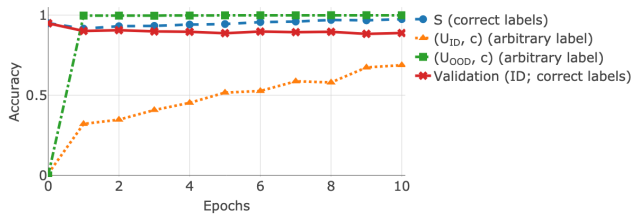
(b) ID = SVHN[0-4]; OOD = SVHN[5-9].



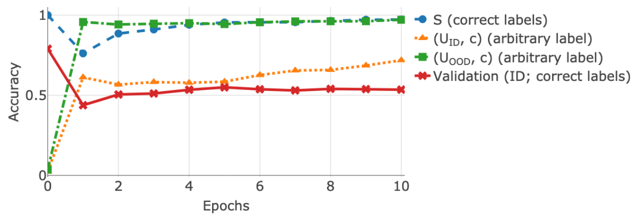
(c) ID = CIFAR10; OOD = SVHN.



(d) ID = CIFAR10[0-4]; OOD = CIFAR10[5-9].



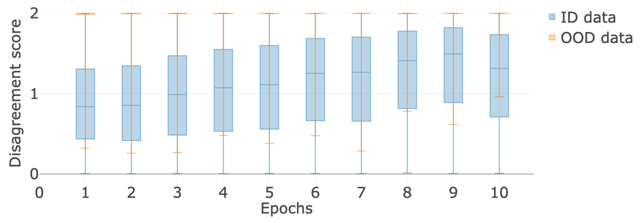
(e) ID = CIFAR100; OOD = SVHN.



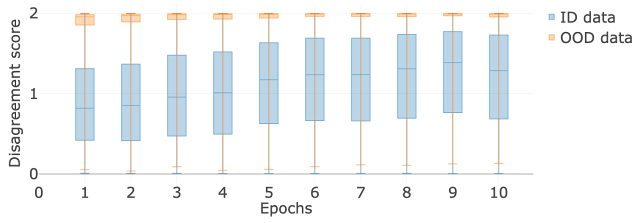
(f) ID = CIFAR100[0-49]; OOD = CIFAR100[50-99].

Figure 14: Accuracy measured while fine-tuning a model pretrained on S (epoch 0 indicates values obtained with the initial pretrained weights). The samples in (U_{OOD}, c) are fit first, while the model reaches high accuracy on (U_{ID}, c) much later. We fine-tune for at least one epoch and then early stop when the validation accuracy starts decreasing.

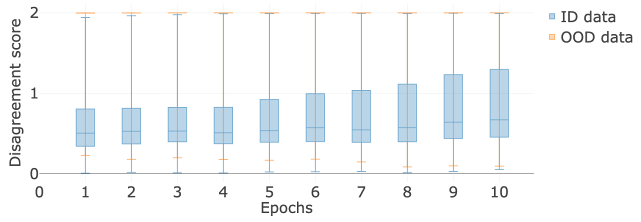
model for ERD is early stopped separately, potentially at a different iteration, it is likely that the ERD ensemble contains models fine-tuned for a different number of iterations. Since we select the ERD ensembles from a strictly larger set, the final ensemble selected by the our proposed approach will be at least as good at distinguishing ID and OOD data as the best ensemble depicted in Figure 15.



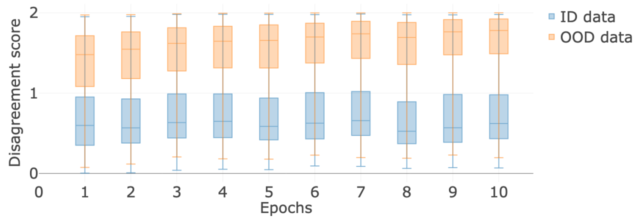
(a) ID = SVHN; OOD = CIFAR10.



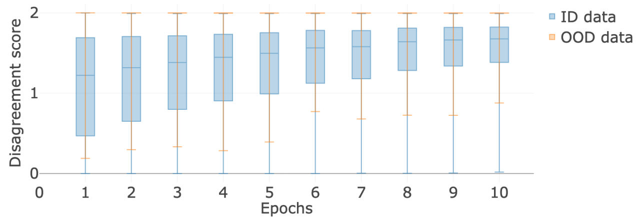
(b) ID = SVHN[0-4]; OOD = SVHN[5-9].



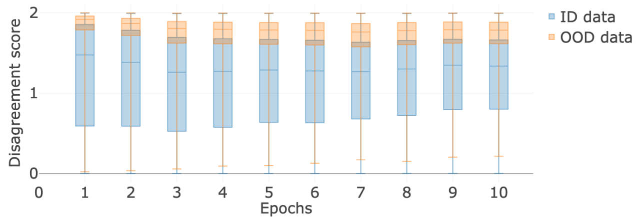
(c) ID = CIFAR10; OOD = SVHN.



(d) ID = CIFAR10[0-4]; OOD = CIFAR10[5-9].



(e) ID = CIFAR100; OOD = SVHN.



(f) ID = CIFAR100[0-49]; OOD = CIFAR100[50-99].

Figure 15: The distribution of the disagreement score measured during fine-tuning on ID and OOD data (blue and orange boxes, respectively). The box indicates the lower and upper quartiles of the distribution, while the middle line represents the median and the whiskers show the extreme values. Notice that the distributions of the scores are easier to distinguish for far OOD data (left column), and tend to overlap more for near OOD settings (right column).

References

Samet Akçay, Amir Atapour-Abarghouei, and T. Breckon. GANomaly: Semi-supervised anomaly detection via adversarial training. In *ACCV*, 2018.

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *NeurIPS*. 2019.

Tianshi Cao, Chin-Wei Huang, David Yu-Tung Hui, and Joseph Paul Cohen. A benchmark of medical out of distribution detection. *arXiv:2007.04250*, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Marthinus C du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In *NeurIPS*, 2014.

Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *arXiv:2106.03004*, 2021.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Chuanxing Geng, Sheng-Jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43, 2021.

Danijar Hafner, Dustin Tran, Timothy P. Lillicrap, Alex Irpan, and James Davidson. Noise contrastive priors for functional uncertainty. In *UAI*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ICLR*, 2017.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019.

Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *NeurIPS*, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*. 2017.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*. 2018.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *AISTATS*, 2020.

Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

Hao Liu and Pieter Abbeel. Hybrid discriminative-generative training via contrastive learning. *arXiv:2007.09070*, 2020.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *NeurIPS*, 2018.

Jordi Mũnoz-Marí, Francesca Bovolo, Luis Gómez-Chova, Lorenzo Bruzzone, and Gustavo Camp-Valls. Semi-supervised one-class support vector machines for classification of remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 48, 2010.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *ICLR*, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *NeurIPS*. 2019.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *ICML*, 2019.

Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. A simple fix to Mahalanobis distance for improving near-OOD detection. *arXiv:2106.09022*, 2021.

Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. In *ICLR*, 2020.

Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with in-distribution examples and gram matrices. *arXiv:1912.12510*, 2019.

Clayton Scott and Gilles Blanchard. Transductive anomaly detection. Technical report, 2008.

Vikash Sehwag, Mung Chiang, and Prateek Mittal. {SSD}: A unified framework for self-supervised outlier detection. In *ICLR*, 2021.

Kihyuk Sohn, Chun-Liang Li, Jinsung Yoon, Minho Jin, and Tomas Pfister. Learning and evaluating representations for deep one-class classification. In *ICLR*, 2021.

Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. CSI: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020.

Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R. Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, Taylan Cemgil, S. M. Ali Eslami, and Olaf Ronneberger. Contrastive training for improved out-of-distribution detection. *arXiv:2007.05566*, 2020.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747*, 2017.

Qing Yu and Kiyoharu Aizawa. Unsupervised out-of-distribution detection by maximum classifier discrepancy. In *ICCV*, 2019.

Hongjie Zhang, Ang Li, Jie Guo, and Yanwen Guo. Hybrid models for open set recognition. In *ECCV*, 2020a.

Min-Ling Zhang and Zhi-Hua Zhou. Exploiting unlabeled data to enhance ensemble diversity. *Data Mining and Knowledge Discovery*, 26, 2010.

Yu-Jie Zhang, Peng Zhao, Lanjihong Ma, and Zhi-Hua Zhou. An unbiased risk estimator for learning with augmented classes. In *NeurIPS*, 2020b.