Novelty detection using ensembles with regularized disagreement

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite their excellent performance on in-distribution (ID) data, machine learning-based prediction systems often predict out-of-distribution (OOD) samples incorrectly while indicating high confidence. Instead, they should flag samples that are not similar to the training data, for example when new classes emerge over time. Even though current OOD detection algorithms can successfully distinguish completely different data sets, they fail to reliably identify samples from novel classes. We develop a new ensemble-based procedure that promotes model diversity and exploits regularization to limit disagreement to only OOD samples, using a batch containing an unknown mixture of ID and OOD data. We show that our procedure significantly outperforms state-of-the-art methods, including those that have access, during training, to data that is known to be OOD. We run extensive comparisons of our approach on a variety of novel-class detection scenarios, on standard image data sets such as SVHN/CIFAR-10/CIFAR-100 as well as on new disease detection on medical image data sets.

1 Introduction

Modern machine learning (ML) systems are gaining popularity in many real-world applications, from aiding medical diagnosis [3] to making recommendations for the justice system [1]. Despite achieving great test set performance, many approaches have trouble dealing with out-of-distribution (OOD) data, i.e. test inputs that are unlike the data seen during training. For example, ML models often make incorrect predictions with high confidence when new unseen classes emerge over time (e.g. undiscovered bacteria [40], new diseases [20]), or when data suffers from distribution shift (e.g. corruptions [31], environmental changes [24]).

If the OOD data suffers from covariate shift [44], we can flag the ambiguous samples using uncertainty-based approaches such as like Bayesian methods [34, 12, 32] or Vanilla Ensembles [25]. In contrast, if the OOD data consists of novel classes, then the only reasonable course of action is to identify it as OOD and bring it to the attention of human experts. This scenario is the focus of this paper and we use the terms OOD and novelty detection interchangeably.

In the novel-class setting, the OOD detection problem becomes equivalent to estimating the support of a distribution, a notoriously difficult task in high dimensions [42]. Nonetheless, prior work on OOD detection reports remarkably good detection performance with the true negative rate corresponding to a true positive rate of 95% (i.e. TNR@95) being often larger than 80% (Figure 1 Left). However, these settings are not representative of OOD detection in real scenarios. In particular, they consider two vastly different data sets as in-distribution (ID) and OOD data (such as SVHN vs CIFAR10), and can hence tolerate suboptimal support estimation, since the OOD samples are far away from the ID data. Nevertheless, in real-world applications it is unlikely that the novel data is so easy to distinguish from ID samples. For instance, chest X-rays of a new disease may look quite similar to another

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

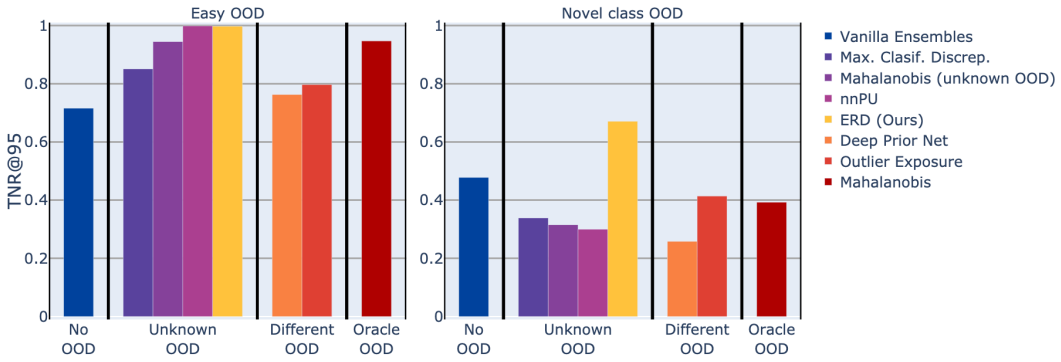


Figure 1: Comparison of OOD detection methods ordered by the amount of information about the OOD distribution that they require. **Left:** On the easy settings usually reported in the literature, many methods achieve near-perfect detection. **Right:** On novel-class settings where ID and OOD data are difficult to distinguish, most baselines reach a significantly lower TNR@95 compared to our method.

pathology; samples from a new class (e.g. images of horses) can resemble data from existing classes (e.g. images of deer). When evaluating state-of-the-art (SOTA) methods on novel-class settings on standard image data sets (e.g. SVHN, CIFAR10), the TNR@95 for the best baseline drops below 40% (see Figure 1 Right).

Apart from a labeled ID training set, SOTA OOD detection methods often use some OOD data for training or calibration. We separate existing approaches into four different levels of access to OOD data: 1) *no OOD* data [25, 41]; 2) an unlabeled set with an unknown mixture of ID and OOD data where OOD samples are not marked (*Unknown OOD*) [43, 10, 30, 52]; 3) known OOD data, but from a different distribution than the test OOD (*Different OOD*) [19, 32]; or 4) known OOD data from the same distribution as test OOD (*Oracle OOD*) [27, 29].

In this work, we adopt the *Unknown OOD* setting and introduce a new method called **Ensembles with Regularized Disagreement (ERD)**. Our algorithm aims to increase the diversity of an ensemble on OOD inputs by performing regularized fine-tuning of a pretrained model on an unlabeled set. With the help of an appropriate ensemble aggregation score, ERD improves the state-of-the-art for novel-class detection, surpassing even approaches that assume oracle knowledge of OOD samples, as illustrated in Figure 1. In summary, our main contributions are as follows:

- We demonstrate that, on realistic novel-class OOD scenarios, SOTA methods achieve a subpar TNR@95 below 40%. We observe this poor performance for all methods, regardless of what kind of access to OOD data they assume.
- We propose a principled method to obtain diverse ensembles by leveraging the unlabeled set and using early stopping regularization. We motivate our algorithm using a theoretical result on the dynamics of gradient descent training under label noise.
- We design a disagreement-based score for ensembles and argue that it successfully exploits model diversity, and thus helps ERD achieve significant improvements compared to SOTA approaches.

2 Problem setting

In this section we formally define the OOD detection problem and motivate the *Unknown OOD* setting that we adopt for our proposed method. Furthermore, we stress the practical significance of this problem in real-world applications.

Problem statement. We consider a labeled data set $S = \{(x_i, y_i)\}_{i=1}^n \sim P$, where $x_i \in \mathcal{X}$ are the covariates and $y_i \in \mathcal{Y}$ are discrete labels. We assume that the labels are obtained as a deterministic function of the covariates, which we denote $y^* : \mathcal{X} \to \mathcal{Y}$. In this paper we focus on detecting samples from novel classes, unseen at training time. Novelty detection aims to identify test samples that have a low probability under the marginal ID distribution P_X, i.e. x should be flagged as OOD if $P_X(x) \leq \alpha$ for some small constant α. In short, we define $\mathcal{X}_{ID} := \{x : P_X(x) > \alpha\}$ as ID points and $\mathcal{X}_{OOD} = \{x : P_X(x) \leq \alpha\}$ the set of OOD points.

Common OOD detection scenarios. If we could learn a model that estimates precisely the level sets of P_X, we would have perfect OOD detection for any $x \notin \mathcal{X}_{ID}$. Unfortunately, when the

input space is high-dimensional and we only have access to limited data, this problem is intractable. For example, OOD detection methods in high-dimensions that only use ID data for training (e.g. generative models [33, 8], Vanilla Ensembles [25]) often have suboptimal performance [21]. In reality, however, we only need to detect outliers that actually appear in a test set, which makes the problem more amenable to statistical methods. Approaches like ODIN [29] or the Mahalanobis method [27] achieve their reported results after training on samples from the same OOD distribution used for evaluation. In contrast to this oracle scenario, methods like Outlier Exposure [19] or Deep Prior Networks [32] train the detection models using a set of known outliers that is different from the test OOD data. However, these methods perform worse when the known OOD data is strikingly different compared to the test data.

OOD detection using unlabeled data. In this work we look at an alternative setting that has been proposed for OOD detection [43, 30, 52] and which assumes that, apart from the ID training data with class labels, we also have access to a batch of unlabeled data U drawn from the same distribution P_{test} as the test data (see Figure 2a Left). This distribution consists of a mixture of ID and OOD data, with proportions determined by $\pi \in [0, 1]$, that is $P_{\text{test}}[x \in \mathcal{X}_{ID}] = 1 - \pi$. The goal is to use the set U to learn to distinguish between ID and OOD data drawn from P_{test}, without explicit knowledge of neither which of the samples in U are OOD, nor of the proportion of outliers π.

Table 1: Related OOD detection methods and the OOD data that they use.

Baseline	Access to OOD
Vanilla Ensemble [25]	No OOD
Gram method [41]	No OOD
Generative [33, 8]	No OOD
nnPU [22]	Unknown OOD
MCD [52]	Unknown OOD
ERD (Ours)	Unknown OOD
Outlier Exposure [19]	Different OOD
DPN [32]	Different OOD
ODIN [29]	Oracle OOD
Mahalanobis [27]	Oracle OOD

The *unknown OOD* setting is relevant for many practical applications that would benefit from better ways to leverage unlabeled data for more effective novel-class detection. Consider, for instance, a medical center that uses an automated system for real-time diagnosis and an offline system which runs once per day, for novelty detection. All the X-rays collected during a day constitute the unlabeled set U that the novelty detection method uses for training. If the detection model identifies any new diseases, then it can also be used without retraining to flag new patients suffering from the novel disease. We note that, in applications where accurate novelty detection is critical, the cost of a delayed detection (e.g. the time needed to collect a batch of unlabeled X-rays) is justified by the substantially better performance of algorithms that leverage unlabeled data.

3 Proposed method

In this section we introduce our proposed algorithm, ERD, and provide a principled justification for the key ingredients that lead to the improved performance of our method.

3.1 The complete ERD procedure

Recall that we have access to both a labeled training set S and an unlabeled set U, which contains both ID and unknown OOD samples. Moreover, we initialize the models of the ensemble using weights pretrained on S.[1]

Algorithm 1: Fine-tuning the ERD ensemble

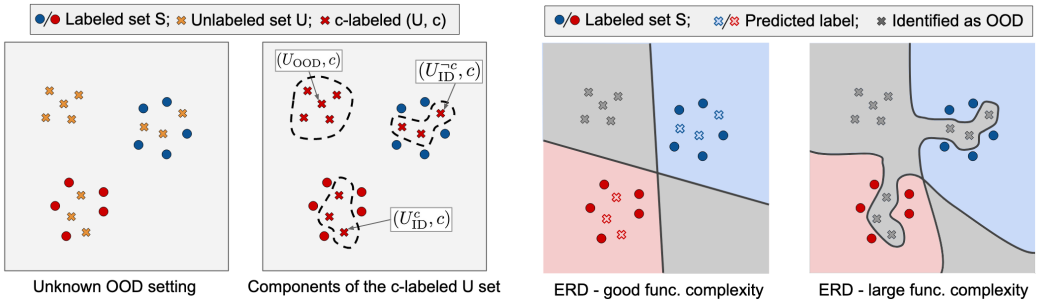
Input : Train set S, Validation set V, Unlabeled set U, Weights W pretrained on S, Ensemble size K
Result: ERD ensemble $\{f_{y_i}\}_{i=1}^K$
Sample K different labels $\{y_1, ..., y_K\}$ from $\mathcal{Y}$
for $c \leftarrow \{y_1, ..., y_K\}$ **do** `// fine-tune` K `models`
  $f_c \leftarrow$ *Initialize*(W)
  $(U, c) \leftarrow \{(x, c) : x \in U\}$
  $f_c \leftarrow$ *FinetuneWithEarlyStopping* $(f_c, S \cup (U, c); V)$
return $\{f_{y_i}\}_{i=1}^K$

Algorithm 2: OOD detection with ERD

Input : Ensemble $\{f_{y_i}\}_{i=1}^K$, Test set T, Threshold t_0, Disagreement metric ρ
Result: O, i.e. the OOD elements of T
$O = \emptyset$
for $x \in T$ **do** `// run hypothesis test`
  if $(Avg \circ \rho)(f_{y_1}, ..., f_{y_K})(x) > t_0$ **then**
    $O \leftarrow O \cup \{x\}$
return O

During training as in Algorithm 1, we begin by assigning an arbitrary label $c \in \mathcal{Y}$, to all the unlabeled samples in U, resulting in the c-labeled set that we denote as $(U, c) := \{(x, c) : x \in U\}$. We

[1] In Section 4 we also present a version of ERD trained from random initializations, i.e. ERD++.



(a) Sketch of the *Unknown OOD* setting.

(b) ERD ensemble for different model complexities.

Figure 2: (a) The unlabeled set (U, c) can be partitioned into (U_{OOD}, c), (U_{ID}^{c}, c), and $(U_{\text{ID}}^{\neg c}, c)$. (b) Restricting model complexity is necessary to prevent from trivially flagging the whole U as OOD. **Left:** Linear classifiers disagree on points in U_{OOD} (gray crosses), but agree to predict the correct label on samples from U_{ID}. **Right:** The models are too complex so they can easily fit the arbitrary label on the entire U.

then fine-tune a classifier f_c on the union $S \cup (U, c)$ of the correctly-labeled training set S, and the unlabeled set (U, c) and we perform early stopping, namely we pick a model at an intermediate epoch, before the accuracy on a holdout ID validation set V starts to decrease. We repeat this procedure to create an ensemble of several classifiers f_c, for different choices of $c \in \mathcal{Y}$. Finally, during test time in Algorithm 2, we use this ensemble to flag as OOD all the points for which an aggregate disagreement measure (described in Section 3.3) surpasses a threshold value t_0.

Intuitively, the models are encouraged to produce different predictions on the OOD samples in U, but early stopping prevents them from fitting the incorrect label c to the ID points. We argue in Section 3.2 that such an optimal stopping time exists and that, indeed, the resulting ensemble disagrees only on the OOD samples in U.

3.2 Theoretical motivation for early stopping regularization

In this section we show how to choose an intermediate checkpoint using the validation set to obtain diverse models that only disagree on OOD samples and not ID samples. This *regularized disagreement* is key to achieving significantly better detection performance on hard OOD tasks than other baselines. We give a rigorous explanation for an ensemble trained from random initializations (i.e. ERD++), but the intuition carries over to ERD fine-tuned from pretrained weights.

Recall that, in our approach, each member of the ensemble tries to fit one label c to the entire unlabeled set U in addition to the correct labels of the ID training set S. Ideally, after training each model with a different label c, we obtain an ensemble of classifiers f_c that disagree only on OOD data. We train the models to fit $S \cup (U, c)$, where we use the notation:

$$(U, c) = (U_{\text{ID}}, c) \cup (U_{\text{OOD}}, c) = \{(x, c) : x \in U_{\text{ID}}\} \cup \{(x, c) : x \in U_{\text{OOD}}\} \tag{1}$$

Moreover, assuming that the labels of the ID data are given by a deterministic function $y^* : \mathcal{X} \to \mathcal{Y}$, we can partition the set (U_{ID}, c) (see Figure 2a) into the subset of samples whose ground truth label differs from c and are thus incorrectly labeled with c, and the subset whose correct label is indeed c:

$$(U_{\text{ID}}^{\neg c}, c) := \{(x, c) : x \in U_{\text{ID}} \text{ with } y^*(x) \neq c\} \tag{2}$$

$$(U_{\text{ID}}^{c}, c) := \{(x, c) : x \in U_{\text{ID}} \text{ with } y^*(x) = c\} \tag{3}$$

We now argue that through regularization we can control the model complexity such that the classifier fits S and all of (U, c), except for $(U_{\text{ID}}^{\neg c}, c)$. The *key intuition* why regularization helps is that it is more difficult to fit the labels c on $(U_{\text{ID}}^{\neg c}, c)$ than on (U_{OOD}, c), since $(U_{\text{ID}}^{\neg c}, c)$ lies closer in covariate space to points in the correctly labeled training set S. Hence, we can exactly fit (U_{OOD}, c) but not $(U_{\text{ID}}^{\neg c}, c)$ if we adequately limit the function complexity (e.g. by choosing a small model class, or through regularization), as illustrated in Figure 2b Left. If the models are too complex (e.g. deep neural networks [54]), then they can even fit the wrong labels on $(U_{\text{ID}}^{\neg c}, c)$ (see Figure 2b Right), causing the models in the ensemble to disagree on the entire unlabeled set U.

We use early stopping regularization, motivated by recent empirical and theoretical works that have found that early stopped neural networks are less vulnerable to label noise in the training data[51, 28].

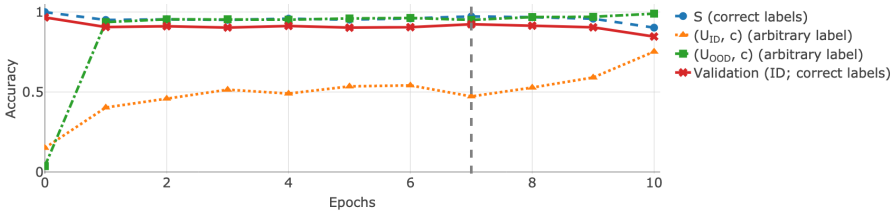


Figure 3: Accuracy measured while fine-tuning a model pretrained on S (epoch 0 indicates values obtained with the initial pretrained weights). The samples in (U_{OOD}, c) are fit first, while the model reaches high accuracy on (U_{ID}, c) much later. We fine-tune for at least one epoch and then early stop when the validation accuracy starts decreasing after 7 epochs (vertical line). The model is trained on SVHN[0:4] as ID and SVHN[5:9] as OOD.

In particular, we show in the following proposition that for a simple neural network trained with gradient descent there exists a stopping time at which all points in (U, c) are fit, except for $(U_{\text{ID}}^{\neg c}, c)$. We say that a labeled set is *clusterable* if points with the same label can be grouped in clusters. Each class may comprise several clusters, but every cluster contains only samples from one class, up to some level of label noise $\rho \in [0, 1]$. We defer the rigorous definition to Appendix A. Note that this model borrowed from [28] is general enough to include data with non-linear decision boundaries. We can now state the following informal proposition:

Proposition 3.1 (informal)**.** *For a fixed label $c \in \mathcal{Y}$ assume that the set $S \cup (U, c)$ is clusterable and each cluster C_i only includes a few noisy samples from $(U_{\text{ID}}^{\neg c}, c)$, namely $\frac{|C_i \cap (U_{\text{ID}}^{\neg c}, c)|}{|C_i|} \leq \rho$. If $\rho \lesssim \frac{1}{|\mathcal{Y}|}$, then it holds with high probability over the initialization of the weights that a two-layer neural network trained on $S \cup (U, c)$ perfectly fits S, (U_{ID}^{c}, c) and (U_{OOD}, c), but not $(U_{\text{ID}}^{\neg c}, c)$, for an appropriately chosen stopping time.*

The precise conditions and statement of this proposition, which is a straight-forward extension of Theorem 2.2 in [28], can be found in Appendix A. Since regularized predictors are smooth, it follows from Proposition 3.1 that at the optimal stopping time the model predicts the label c on OOD samples similar to the ones in the unlabeled set, but the correct, ground truth label on ID data.

To find the best stopping time in practice, we use a validation set of labeled ID points to select an intermediate checkpoint before convergence. As a model starts to fit $(U_{\text{ID}}^{\neg c}, c)$, i.e. the wrongly labeled ID samples in U_{ID}, it also predicts the label c on some validation ID points, leading to a decrease in validation accuracy, as shown in Figure 3. In our experiments, we wait for one epoch to allow for the fine-tuning to have any effect at all, and then pick the iteration with the largest validation accuracy (indicated by the vertical line in the figure).

Naturally, one could use explicit regularization such as dropout or weight decay instead of early stopping, however, running a grid search to select the right hyperparameters can be more computationally expensive than simply using one run of the training process to select the optimal stopping time. Moreover, prior work has shown that performing more gradient descent iterations leads to models with larger complexity [50, 37, 48], and hence, early stopping restricts model complexity.

3.3 Ensemble disagreement for large true negative rates

We now motivate a novel ensemble aggregation technique that we use to detect OOD samples with ERD. Note that we can cast the OOD detection problem as a hypothesis test with null hypothesis $H_0 : x \in \mathcal{X}_{ID}$. Our procedure tests the null hypothesis by using an ensemble-based score: The null hypothesis is *rejected* and we report x as OOD (*positive*) if the score is larger than a threshold t_0 (see Algorithm 2). Ideally, the test should have high power (flag true OOD as OOD) and low false positive rate (avoid flagging true ID as OOD).

As we argue in Section 3.2, 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 [25, 36] first average the softmax predictions of the models in the ensemble and then use the entropy as a metric, i.e. $(\text{H} \circ \text{Avg})(f_1(x), ..., f_K(x)) := -\sum_{i=1}^{|\mathcal{Y}|} (f(x))_i \log(f(x))_i$ where $f(x) := \frac{1}{K} \sum_{i=1}^{K} f_i(x)$ and $(f(x))_i$ is the i^{th} element of $f(x) \in [0, 1]^{|\mathcal{Y}|}$. We argue later that averaging discards information

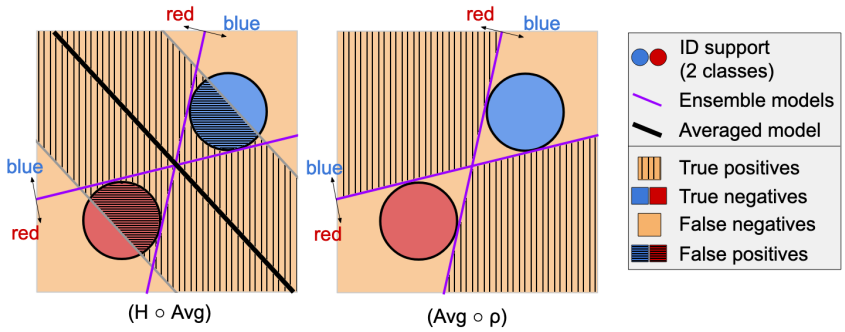


Figure 4: Cartoon illustration showing a diverse ensemble of linear binary classifiers. We compare OOD detection performance for two aggregation scores: $(\mathrm{H} \circ \mathrm{Avg})$ (**Left**) and $(\mathrm{Avg} \circ \rho)$ with $\rho(f_1(x), f_2(x)) = \mathbb{1}_{\mathrm{sgn}(f_1(x)) \neq \mathrm{sgn}(f_2(x))}$ (**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.

about the diversity of the models. Instead, we propose the average pairwise *disagreement* between the outputs of K models in an ensemble:[2]

$$(\mathrm{Avg} \circ \rho)(f_1(x), ..., f_K(x)) := \frac{2}{K(K-1)} \sum_{i \neq j} \rho\left(f_i(x), f_j(x)\right), \tag{4}$$

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 4. 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 4 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. Appendix B provides further quantitative evidence to support the intuition presented in Figure 4.

4 Experimental results

In this section we evaluate the OOD detection performance of ERD for deep neural networks on several image data sets. We find that our approach outperforms all baselines on difficult OOD detection scenarios. In addition, we discuss some of the trade-offs that impact ERD's performance.

4.1 ID vs OOD settings

We report results on two broad types of OOD detection scenarios (see Appendix D for more details):

1. **Easy OOD data (most previous benchmarks):** ID and OOD samples come from strikingly different data sets (e.g. CIFAR10 vs SVHN). These are the settings usually considered in the literature and on which most baselines perform well.

2. **Hard OOD data:** The OOD data consists of "novel" classes the resemble the ID samples: e.g. the first 5 classes of CIFAR10 are ID, the last 5 classes are OOD. The similarities between the ID and the OOD classes make these settings significantly more challenging.

[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.

Table 2: AUROC and TNR@95 for different OOD detection scenarios (the numbers in squared brackets indicate the ID or OOD classes). We highlight the **best ERD** variant and ***best baseline***. The asterisk marks baselines proposed in this paper. nnPU (†) assumes oracle knowledge of the OOD ratio in the unlabeled set.

		Other settings					Unknown OOD					
ID data	OOD data	Vanilla Ensembles	Gram	DPN	OE	Mahal.	nnPU†	MCD	Mahal-U	Bin. Classif. *	ERD *	ERD++ *
		AUROC ↑ / TNR@95 ↑										
SVHN	CIFAR10	0.97 / 0.88	0.97 / 0.86	***1.00*** / ***1.00***	***1.00*** / ***1.00***	0.99 / 0.98	***1.00*** / ***1.00***	0.97 / 0.85	0.99 / 0.95	1.00 / 1.00	**1.00** / **0.99**	**1.00** / **0.99**
CIFAR10	SVHN	0.92 / 0.78	***1.00*** / 0.98	0.95 / 0.85	0.97 / 0.89	0.99 / 0.96	***1.00*** / ***1.00***	***1.00*** / 0.98	0.99 / 0.96	1.00 / 1.00	**1.00** / **1.00**	**1.00** / **1.00**
CIFAR100	SVHN	0.84 / 0.48	0.99 / 0.97	0.77 / 0.44	0.82 / 0.50	0.98 / 0.90	***1.00*** / ***1.00***	0.97 / 0.73	0.98 / 0.92	1.00 / 1.00	**1.00** / **1.00**	**1.00** / **1.00**
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	**0.95** / **0.71**
SVHN [0:4]	SVHN [5:9]	0.92 / 0.69	0.81 / 0.31	0.87 / 0.19	0.85 / 0.52	0.92 / 0.71	***0.96*** / ***0.73***	0.91 / 0.51	0.91 / 0.63	0.81 / 0.40	0.95 / 0.74	**0.96** / **0.77**
CIFAR10 [0:4]	CIFAR10 [5:9]	0.80 / 0.39	0.67 / 0.15	***0.82*** / 0.32	***0.82*** / ***0.41***	0.79 / 0.27	0.61 / 0.11	0.69 / 0.25	0.64 / 0.13	0.85 / 0.43	0.93 / 0.70	**0.96** / **0.79**
CIFAR100 [0:49]	CIFAR100 [50:99]	***0.78*** / ***0.35***	0.71 / 0.16	0.70 / 0.26	0.74 / 0.31	0.72 / 0.20	0.53 / 0.06	0.70 / 0.26	0.72 / 0.19	0.66 / 0.13	0.82 / 0.44	**0.85** / **0.45**
Average		0.84 / 0.52	0.86 / 0.57	0.84 / 0.46	0.84 / 0.54	***0.88*** / 0.60	0.86 / ***0.66***	0.86 / 0.55	0.86 / 0.60	0.89 / 0.66	0.95 / 0.79	**0.96** / **0.82**

Apart from using these canonical data sets, we also compare the performance of our method on more realistic data, namely a recently proposed OOD detection benchmark for medical imaging [6]. This benchmark contains a suite of data sets that cover three categories of difficulty, as detailed in Appendix F.

For all scenarios, we used a labeled training set (e.g. 40K samples for CIFAR10), a validation set with ID samples (e.g. 10K samples for CIFAR10) and an unlabeled test set where half of the samples are ID and the other half are OOD (e.g. 5K ID samples and 5K OOD samples for CIFAR10 vs SVHN). For evaluation, we use a holdout set containing ID and OOD samples in the same proportions as the unlabeled set. In Appendix E.1 we show that ERD can also successfully identify the outliers from the unlabeled set used for fine-tuning. Furthermore, in Appendix E.3 we present results obtained with a smaller unlabeled set of only 1K samples.

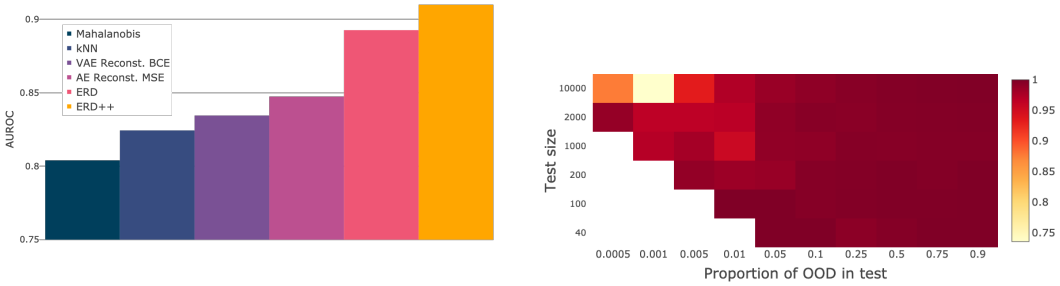
4.2 Baselines

Standard baselines. We compare our method against a wide range of baselines that require different access to OOD data for training, as indicated in Table 1. When it comes to methods that use *no OOD* data for training, the current SOTA on the usual benchmarks is the Gram method [41]. Other approaches that use no OOD data include vanilla ensembles [25], methods that rely on deep generative models [33, 8], which tend to give undesirable results for OOD detection [21], or various Bayesian approaches [12, 5] that are often poorly calibrated on OOD data [36]. Moreover, Outlier Exposure [19] and Deep Prior Networks (DPN) [32] use TinyImages for training as known outliers, irrespective of the OOD set used for evaluation (*Different OOD*). On the other hand, the Mahalanobis baseline [27] is tuned on samples from the same OOD distribution used for evaluation.

Unknown OOD and PU learning. We also compare our method to approaches that assume the same setting, in which an unlabeled set with ID and OOD samples is available. The recently proposed MCD method [52] trains an ensemble of two classifiers with different types of predictive distributions on the unlabeled samples: one model gives high-entropy predictions, while the other has low entropy. Furthermore, positive-unlabeled (PU) learning [10] considers a binary classification setting, in which the labeled data comes from one class (i.e. ID samples, in our case), while the unlabeled set contains a mixture of samples from both classes. Crucially, PU learning methods, like nnPU [22], require oracle knowledge of the ratio of OOD samples in the unlabeled set.

Unknown OOD – new baselines. In addition to these methods, we propose two more baselines that use an unlabeled set. Firstly, we present a version of the Mahalanobis approach (*Mahal-U*) that is calibrated using the unlabeled set. Secondly, since PU learning requires access to the OOD ratio of the unlabeled set, we also consider a less burdensome alternative: a binary classifier trained to separate the training data from the unlabeled set and regularized with early stopping like our method.

Tuning hyperparameters. For all the baselines, we use the default hyperparameters suggested by their authors on the same ID data set. The Gram method requires laborious hyperparameter tuning for multi-layer perceptron (MLP) models, so we do not consider it for the MNIST and FMNIST data



(a) OOD detection performance on medical benchmark (b) Effect of OOD proportion on detection

Figure 5: **Left:** AUROC averaged over all scenarios in the medical OOD detection benchmark. The values for the baselines are computed using the code from [6]. **Right:** The AUROC of ERD as the number and proportion of ID (CIFAR100) and OOD (SVHN) samples in the unlabeled set are varied.

sets.[3] For our method, the binary classifier and nnPU, we pick hyperparameters only to optimize the loss on an ID validation set. We defer the details regarding training the models to Appendix C.

ERD. We present results for two flavors of our method. Firstly, we fine-tune each model in the ensemble with early stopping, starting from weights that are pretrained on the labeled ID set S (ERD). Secondly, we train the models from random initializations (ERD++) to obtain slightly better OOD detection, at the cost of more training iterations. For our method we train ensembles of five MLP models for MNIST and FMNIST and ResNet20 [16] networks for the other settings (results for other architectures are presented in Appendix E). For each model in the ensemble we perform post-hoc early stopping: we train for 10 epochs for ERD (100 epochs for ERD++) and select the iteration with the lowest validation loss.

Evaluation. As in standard hypothesis testing problems, choosing different thresholds for rejecting the null hypothesis leads to different false positive and true positive rates (FPR and TPR, respectively). The ROC curve follows the FPR and the TPR for all possible threshold values and the area under the curve (AUROC; larger values are better) captures the performance of a statistical test without having to select a specific threshold. In addition to the AUROC, we also use the TNR at a TPR of 95% (TNR@95; larger values are better) for evaluation.[4]

4.3 Main results

Table 2 summarizes the main empirical results. On the easy scenarios (top part of the table) most methods achieve near-perfect OOD detection with AUROC close to 1. However, on the novelty detection scenarios (bottom part), ERD has a clear edge over other baselines, even when they are calibrated on *oracle OOD* data, or when they use the true OOD ratio of the unlabeled set, e.g. nnPU. The substantial gap between ERD and other approaches, both in average AUROC and average TNR@95, indicates that our method lends itself well to practical situations when accurate OOD detection is critical. In Appendix E we show that our method successfully identifies OOD samples with mild distribution shift (e.g. corrupted CIFAR10 [18], CIFAR10v2 [38], ObjectNet [2]), which provides further evidence that ERD is well-suited for the most difficult of OOD detection tasks.

For the medical OOD detection benchmark we show in Figure 5a the average AUROC achieved by some representative baselines taken from [6]. Our method improves the average AUROC from 0.85 to 0.91, compared to the best performing baseline. We refer the reader to [6] for precise details on the methods. Appendix F contains more results for the medical settings, as well as additional baselines.

Computation cost. For ERD as few as three epochs of fine-tuning are enough on average to achieve the performance that we report. This amounts to around 2 minutes if the models in the ensemble are fine-tuned in parallel on NVIDIA GeForce GTX 1080 Ti GPUs.

[3] We note that the code provided by the authors does not include the configurations required for MLP models.

[4] In practice, choosing a good rejection threshold is important. A recent work [30] proposes a criterion for setting the threshold that is tailored specifically to a setting with unknown OOD. Alternatively, one can choose the threshold so as to achieve a desired FPR, which we can estimate using a validation set of ID samples.

Varying the unlabeled set. We investigate the impact of the size of the unlabeled set on the OOD detection performance for our method. In addition, we also vary the ratio of OOD samples in the unlabeled set, i.e. $\frac{|U_{\text{OOD}}|}{|U_{\text{ID}}|+|U_{\text{OOD}}|}$. Our findings suggest that there is a broad spectrum of values for which ERD maintains a good performance, as indicated in Figure 5b.

4.4 Limitations of related OOD detection methods

We now discuss some shortcomings of existing OOD detection approaches closely related to ours and indicate how our method attempts to address them. Firstly, vanilla ensembles use only the stochasticity of the training process and the random initialization to obtain diverse models, but this often leads to similar classifiers, that predict the same incorrect label on OOD data [17]. Secondly, in the absence of proper regularization, optimizing the MCD objective leads to models that agree to a similar extent on both ID and OOD data so that one cannot distinguish them from one another (as indicated by low AUROC scores). Furthermore, nnPU does not exploit all the signal in the training set and discards the labels of the ID data.

ERD successfully diversifies an ensemble on OOD data by using the unlabeled set and without requiring additional information about the test distribution (e.g. unlike nnPU which requires the true OOD ratio). We identify the key reasons behind the good performance of our approach to be as follows: 1) utilizing the labels of the ID training data and the complexity of deep neural networks to diversify model outputs on OOD data; 2) choosing an appropriate disagreement score that draws on ensemble diversity; 3) employing early stopping regularization to prevent diversity on ID inputs.

5 Related problems

Predictive uncertainty and Bayesian methods. One of the important appeals of the Bayesian framework is that it directly provides uncertainty estimates together with the predictions. Bayesian methods are particularly useful in the case of covariate shift [44], when predictive uncertainty can be used to decide to abstain [14] on ambiguous samples, while still allowing high-certainty predictions. Approaches like MC-Dropout [12] or Deep Prior Networks [32] attempt to tackle this problem, but the uncertainty estimates they provide are often inaccurate on OOD samples [36]. The same problem has been observed for Bayesian Neural Networks [34, 15, 5], for which sampling efficiently from the posterior over parameters remains challenging for large models [36].

Transductive learning. Transductive learning [46] assumes that the unlabeled test set is available together with a labeled training set and both can be used to select a good predictor. Unlike semi-supervised learning, the transductive framework is only concerned with performing well on the given test set and is not interested in generalization on holdout data. In practice it has been successfully used for problems like zero-shot learning [11, 47]. Transductive OOD detection [43] is equivalent to the scenario that we adopt in this paper if the unlabeled set coincides with the test set used for evaluation (see also Appendix E.1).

Domain adaptation. The OOD detection problem with access to unknown OOD data is reminiscent of unsupervised domain adaptation (UDA) [4, 13, 7], in that both allow access to an unlabeled data set to adjust predictors to a new distribution. However, unlike OOD detection, UDA aims to provide correct predictions on a target distribution with covariate shift [44]. Hence, the UDA problem is ill-posed if the target distribution contains data from novel classes, not present in the source set. In novel-class scenarios, one needs to consider OOD detection instead.

6 Conclusions

Reliable OOD detection is essential in order to deploy classification systems in safety-critical environments. We propose a procedure that results in an ensemble with selective disagreement only on OOD data, by successfully leveraging unlabeled data to fine-tune the models in the ensemble. It outperforms state-of-the-art methods that also have access to a mixture of ID and unknown OOD samples, but also approaches that use known OOD data for training. As future work, we propose an investigation into the influence of the labeling scheme of the unlabeled set on the sample complexity of the method, as well as an analysis of the trade-off governed by the complexity of the model class of the classifiers.

References

[1] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. 2016.

[2] 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 *Advances in Neural Information Processing Systems 32*, pages 9453–9463. 2019.

[3] Emma Beede, Elizabeth Baylor, Fred Hersch, Anna Iurchenko, Lauren Wilcox, Paisan Ruamviboonsuk, and Laura M. Vardoulakis. A human-centered evaluation of a deep learning system deployed in clinics for the detection of diabetic retinopathy. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, page 1–12, 2020.

[4] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 19*, pages 137–144, 2007.

[5] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32th International Conference on Machine Learning*, 2015.

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

[7] Yining Chen, Colin Wei, Ananya Kumar, and Tengyu Ma. Self-training avoids using spurious features under domain shift. *arXiv preprint arXiv:2006.10032*, 2020.

[8] Hyunsun Choi, Eric Jang, and Alexander A. Alemi. WAIC, but why? generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] Marthinus C du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In *Advances in Neural Information Processing Systems 27*, 2014.

[11] Yanwei Fu, Timothy M. Hospedales, Tao Xiang, and Shaogang Gong. Transductive multi-view zero-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37: 2332–2345, 2015.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, 2016.

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, pages 1–35, 2016.

[14] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems 30*, pages 4878–4887. 2017.

[15] Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems 24*, pages 2348–2356. 2011.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[17] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[18] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *Proceedings of the International Conference on Learning Representations*, 2019.

[19] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *Proceedings of the International Conference on Learning Representations*, 2019.

[20] Iason Katsamenis, Eftychios Protopapadakis, Athanasios Voulodimos, Anastasios Doulamis, and Nikolaos Doulamis. Transfer learning for covid-19 pneumonia detection and classification in chest x-ray images. *medRxiv*, 2020.

[21] Polina Kirichenko, Pavel Izmailov, and Andrew G Wilson. Why normalizing flows fail to detect out-of-distribution data. In *Advances in Neural Information Processing Systems 33*, pages 20578–20589, 2020.

[22] Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *Advances in Neural Information Processing Systems 30*, 2017.

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

[24] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. *arXiv preprint arXiv:2002.11361*, 2020.

[25] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6402–6413. Curran Associates, Inc., 2017.

[26] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[27] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems 31*, pages 7167–7177. 2018.

[28] Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. Proceedings of Machine Learning Research, pages 4313–4324, 2020.

[29] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[30] Si Liu, Risheek Garrepalli, Thomas Dietterich, Alan Fern, and Dan Hendrycks. Open category detection with PAC guarantees. pages 3169–3178, 2018.

[31] Alex X. Lu, Amy X. Lu, Wiebke Schormann, David W. Andrews, and Alan M. Moses. The cells out of sample (COOS) dataset and benchmarks for measuring out-of-sample generalization of image classifiers. *arXiv preprint arXiv:1906.07282*, 2019.

[32] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems 32*, page 7047–7058, 2018.

[33] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *Proceedings of the International Conference on Learning Representations*, 2019.

[34] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996.

[35] 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.

[36] 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 *Advances in Neural Information Processing Systems 32*, pages 13991–14002. 2019.

[37] Garvesh Raskutti, Martin J. Wainwright, and Bin Yu. Early stopping and non-parametric regression: An optimal data-dependent stopping rule, 2013.

[38] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

[39] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? *arXiv preprint arXiv:1902.10811*, 2019.

[40] Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems 32*, pages 14707–14718. 2019.

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

[42] Bernhard Scholkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, page 1443–1471, 2001.

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

[44] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

[45] Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1958–1970, 2008.

[46] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[47] Ziyu Wan, Dongdong Chen, Yan Li, Xingguang Yan, Junge Zhang, Yizhou Yu, and Jing Liao. Transductive zero-shot learning with visual structure constraint. In *Advances in Neural Information Processing Systems 32*, pages 9972–9982, 2019.

[48] Yuting Wei, Fanny Yang, and Martin J Wainwright. Early stopping for kernel boosting algorithms: A general analysis with localized complexities. In *Advances in Neural Information Processing Systems 30*, pages 6065–6075. 2017.

[49] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, 2017.

[50] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, pages 289–315, 2007.

[51] Fatih Furkan Yilmaz and Reinhard Heckel. Image recognition from raw labels collected without annotators. *arXiv preprint arXiv:1910.09055*, 2019.

[52] Qing Yu and Kiyoharu Aizawa. Unsupervised out-of-distribution detection by maximum classifier discrepancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[53] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[54] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Checklist (v1.4)

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your approach and contributions? [Yes]
   (c) Did you discuss the potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (data splits, hyperparameters, how they were chosen)? [Yes] See Appendix C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the computational cost of reruning all baselines on all the OOD settings.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster or cloud provider)? [Yes] We included an estimate of the training time of our method.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Note that the Checklist section does not count towards the page limit.

A Theoretical statements

Definition A.1 ((ϵ, ρ)-clusterable data set)**.** *We say that a data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ is* (ϵ, ρ)-clusterable *for fixed $\epsilon > 0$ and $\rho \in [0, 1]$ 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 $1 - \rho$ 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, ρ)-clusterable into K clusters. We further assume that each cluster C_i only includes a few noisy samples from $(U_{\text{ID}}^{\neg c}, c)$, i.e. $\frac{|C_i \cap (U_{\text{ID}}^{\neg c}, c)|}{|C_i|} \leq \rho$ and that for clusters C_i whose cluster label is not c, i.e. $y^*(c_i) \neq c$, it holds that $C_i \cap (U_{\text{OOD}}, c) = \emptyset$.

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:

$$x \mapsto f(x; W) = v^T \phi(Wx), \tag{5}$$

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 $\eta \simeq \frac{K}{n\|C\|^2}$.

We now provide the formal statement of Proposition 3.1:

Proposition A.1. *Assume that $\rho \leq \delta/8$ 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, ρ)-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 [28]:

Theorem A.1 ([28]). *Let $\mathcal{D} := \{(x_i, y_i)\} \in \mathbb{R}^d \times \mathcal{Y}$ be an (ϵ, ρ)-clusterable training set, with $\epsilon \leq c_1 \delta \lambda_{min}(\Sigma)^2 / K^2$ and $\rho \leq \delta/8$, 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:*

$$\arg\max_{y \in \mathcal{Y}} |f(x; W_T) - \omega(y)| = y^*(c_i), \text{ for all } x \text{ with } \|x - c_i\|_2 \leq \epsilon \text{ and all clusters } i \in [K], \quad (6)$$

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, ρ)-clusterable, since the incorrectly labeled ID points in $(U_{\text{ID}}^{\neg c}, c)$ constitute at most a fraction ρ of the clusters they belong to. As a consequence, Proposition A.1 follows directly from Theorem A.1.

□

B Disagreement score for OOD detection

As we outlined in Section 3.3, in this paper we introduce a novel way to aggregate ensemble outputs using a disagreement score. 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 6 for a few 2D data sets. As a consequence, using our disagreement score $(\text{Avg} \circ \rho)$ for Vanilla Ensembles can sometimes hurt OOD 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 $(\text{Avg} \circ \rho)(f_1(x), f_2(x)) = 0$, for all test points x and for any function ρ that satisfies the distance axioms.

Table 3) shows that $(\text{Avg} \circ \rho)$ leads to worse OOD 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 3.2), 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 3.

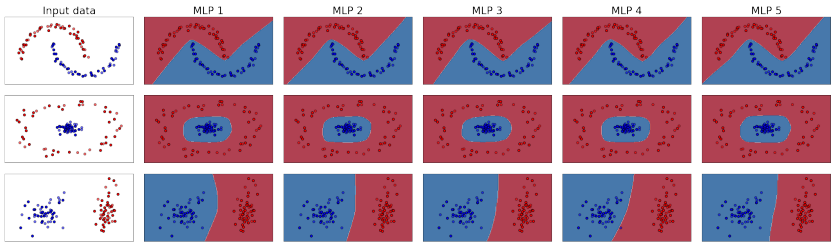


Figure 6: Relying only on the randomness of SGD and of the weight initialization to diversify models is not enough, as it often yields similar classifiers. Each column shows a different predictor trained from random initializations with Adam. All models have the same 1-hidden layer MLP architecture.

Table 3: The disagreement score that we propose $(\text{Avg} \circ \rho)$ exploits ensemble diversity and benefits in particular ERD ensembles. OOD 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 OOD 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**

C Experiment details

C.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 OOD 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 E.6.

- **Vanilla Ensembles** [25]: 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 3.3. 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** [41]: 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)** [32]: 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** [19]: 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) [53]. 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 E.4 we show results for known OOD data that is similar to the OOD data used for testing.
- **Mahalanobis** [27]: 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 assume the same *Unknown OOD* setting as ERD, in which one has access to both a labeled ID training set S and an unlabeled set with an unknown mixture of ID and OOD samples U.

- **Non-negative PU learning (nnPU)** [22]: 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, [22] 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)** [52]: 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 [43] 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 F 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 [6].

C.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.

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 [16]. 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 [6]. 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}.

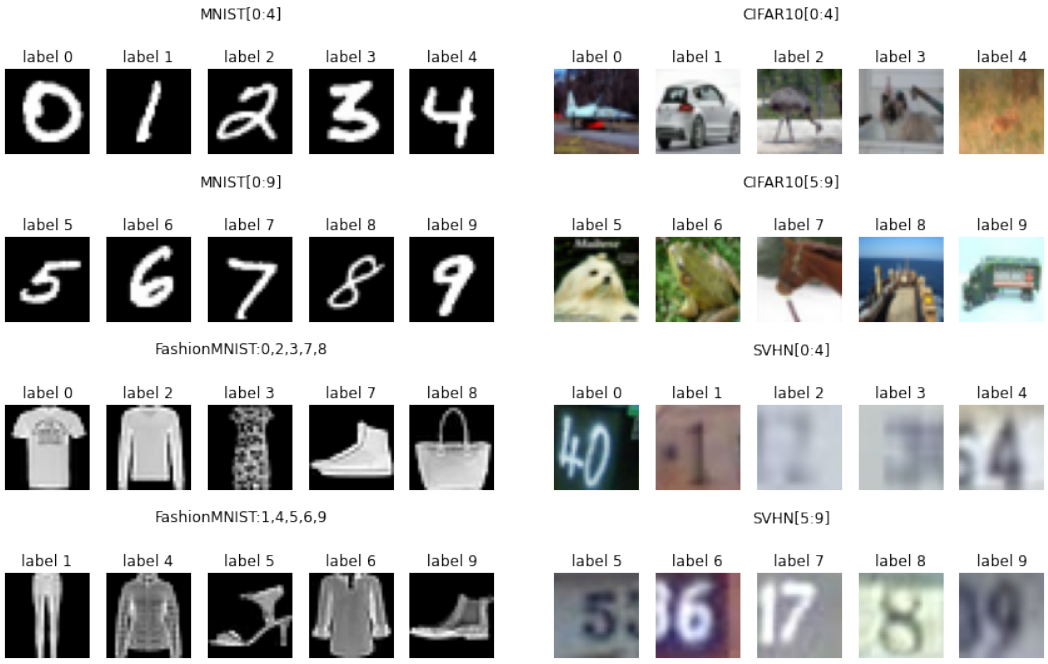
D ID and OOD data sets

D.1 Data sets

For evaluation, we use the following image data sets: MNIST [26], Fashion MNIST [49], SVHN [35], CIFAR10 and CIFAR100 [23].

For the experiments using MNIST and FashionMNIST 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). For the settings that use half of the classes as ID and the other half as OOD, all the sizes are divided by 2.

D.2 Samples for the settings with novel classes



(a) (b)

Figure 7: (a) Data samples for the MNIST/FashionMNIST splits. (b) Data samples for the CIFAR10/SVHN splits.

E More experiments

E.1 Evaluation on the unlabeled set

In the main text we describe how one can leverage the unlabeled set U to obtain an OOD 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 4 we show that the OOD 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 4: Comparison between the OOD 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) AUROC ↑ / TNR@95 ↑	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

E.2 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 [44]. 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 [18], as well as a scenario where ImageNet [9] is used as ID data and ObjectNet [2] as OOD, both resized to 32x32. Figure 8 shows samples from these data sets. The Gram and nnPU baselines do not give satisfactory results on the difficult CIFAR10/CIFAR100 settings in Table 2 and thus we do not consider them for the distribution shift cases. For the *Unknown OOD* methods (i.e. MCD, Mahal-U and ERD/ERD++) we evaluate on the same unlabeled set that is used for training (see the discussion in Section E.1).

Furthermore, we present results on distinguishing between CIFAR10 [23] and CIFAR10v2 [38], a data set meant to be drawn from the same distribution as CIFAR10 (generated from the Tiny Images collection [45]). In [39], 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

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.

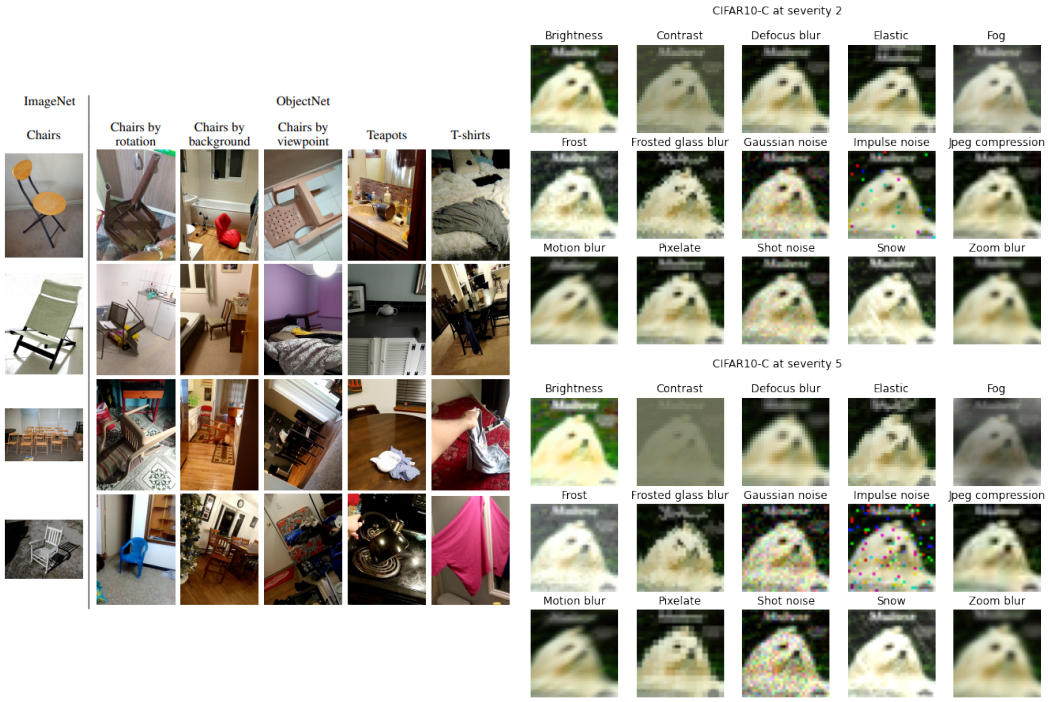


Figure 8: Left: Samples from ImageNet and ObjectNet taken from the original paper by [2]. Right: Data samples for the corrupted CIFAR10-C data set.

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.

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**

E.3 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.

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**

E.4 More results for Outlier Exposure

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

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

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 Table 8 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).

E.5 Results on MNIST and FashionMNIST

Table 9: Results on MNIST/FashionMNIST settings. For ERD and vanilla ensembles, we train 5 3-hidden layer MLP models for each setting. The evaluation metrics are computed on the unlabeled set.

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 OOD detection more difficult than the 0-4 vs 5-9 split.

E.6 Vanilla and ERD Ensembles with different architectures

In this section we present OOD detection results for 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.

ID data	OOD data	VGG16		ResNet20		WideResNet-28-10	
		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 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 E.2.
3. Use case 3: The OOD data set contains images that come from novel classes, not seen during training.

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 [29], Mahalanobis [27]. 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 [6].

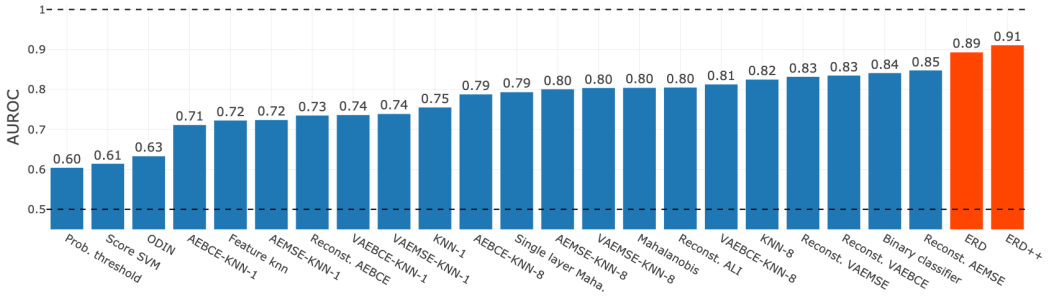


Figure 9: AUROC averaged over all scenarios in the medical OOD detection benchmark [6]. The values for all the baselines are computed using code made available by the authors of [6]. Notably, most of the baselines assume oracle knowledge of OOD data at training time.

In addition, in Figure 10 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 9. 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, our method maintains a better performance than the baselines.

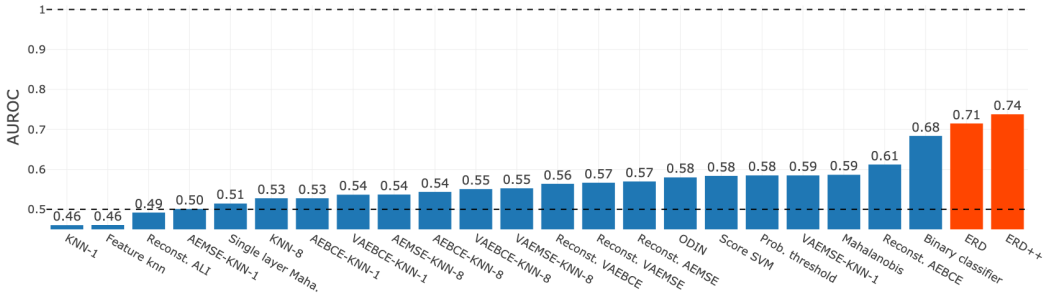


Figure 10: AUROC averaged over the novel-class scenarios in the medical OOD detection benchmark [6], i.e. only use case 3.

In Figures 11, 12, 13 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 9 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 9).

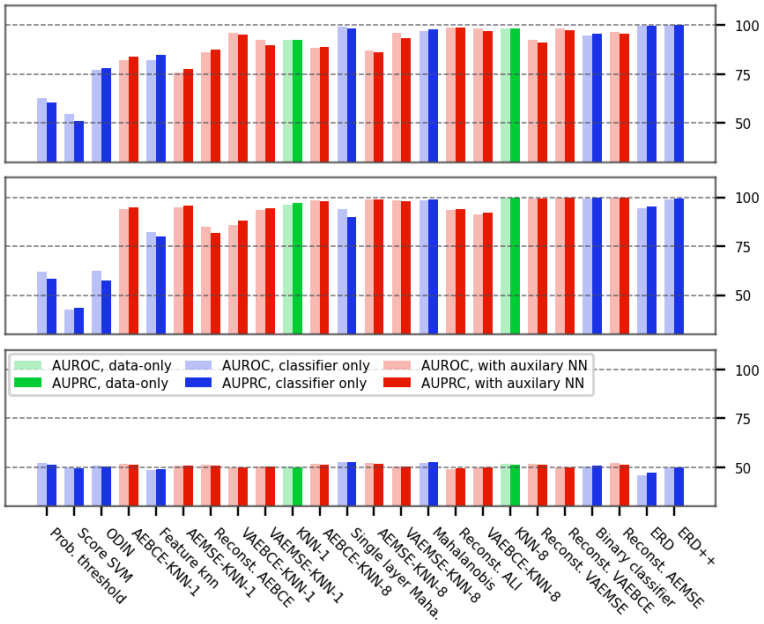


Figure 11: Comparison between ERD and the various baselines on the NIH chest X-ray data set, for use case 1 (top), use case 2 (middle) and use case 3 (bottom). Baselines ordered as in Figure 9.

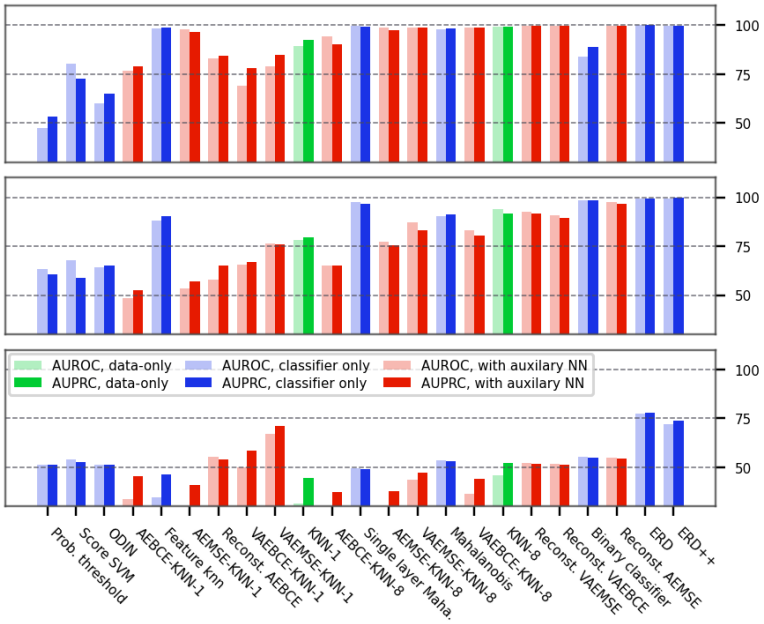


Figure 12: Comparison between ERD and the various baselines on the PC chest X-ray data set, for use case 1 (top), use case 2 (middle) and use case 3 (bottom). Baselines ordered as in Figure 9.

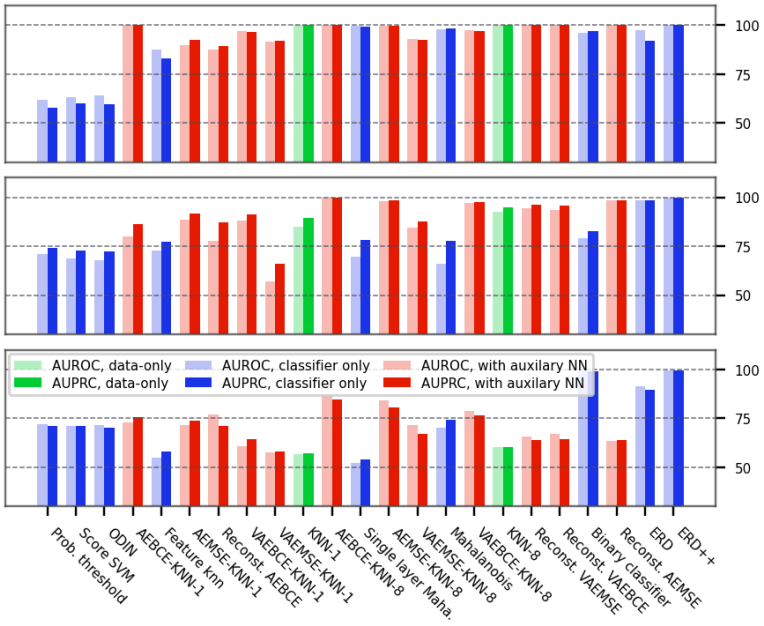


Figure 13: Comparison between ERD and the various baselines on the DRD fundus imaging data set, for use case 1 (top), use case 2 (middle) and use case 3 (bottom). Baselines ordered as in Figure 9.

For all of 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.

G Effect of learning rate and batch size

We show now that our method ERD is 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 14 shows, many different configurations lead to similar OOD detection performance.

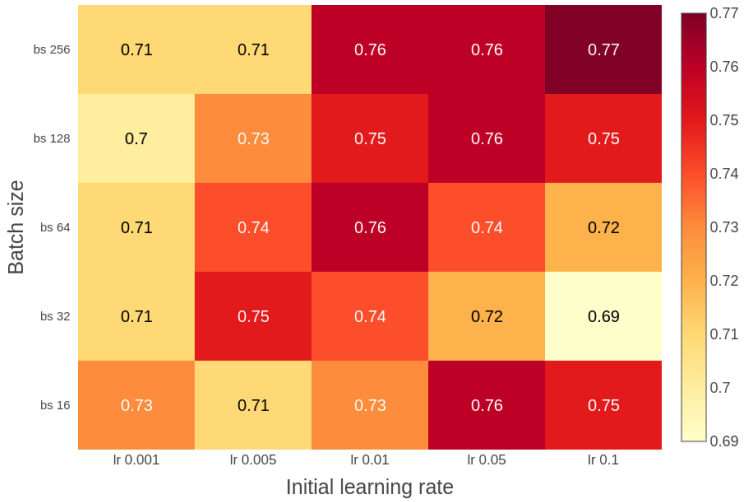


Figure 14: AUROCs obtained with an ensemble of WRN-28-10 models, as the initial learning rate and the batch size are varied. We used the hardest setting, CIFAR100:0-50 as ID, and CIFAR100:50-100 as OOD.