Unsupervised Out-of-Distribution Detection by Maximum Classifier Discrepancy

Qing Yu Kiyoharu Aizawa
The University of Tokyo, Japan
{yu, aizawa}@hal.t.u-tokyo.ac.jp

Abstract

Since deep learning models have been implemented in many commercial applications, it is important to detect out-of-distribution (OOD) inputs correctly to maintain the performance of the models, ensure the quality of the collected data, and prevent the applications from being used for other-than-intended purposes. In this work, we propose a two-head deep convolutional neural network (CNN) and maximize the discrepancy between the two classifiers to detect OOD inputs. We train a two-head CNN consisting of one common feature extractor and two classifiers which have different decision boundaries but can classify in-distribution (ID) samples correctly. Unlike previous methods, we also utilize unlabeled data for unsupervised training and we use these unlabeled data to maximize the discrepancy between the decision boundaries of two classifiers to push OOD samples outside the manifold of the in-distribution (ID) samples, which enables us to detect OOD samples that are far from the support of the ID samples. Overall, our approach significantly outperforms other state-of-the-art methods on several OOD detection benchmarks and two cases of real-world simulation.

1. Introduction

After several breakthroughs of deep learning methods, deep neural networks (DNNs) have achieved impressive results and even outperformed humans in fields such as image classification [8], face recognition [17], and natural language processing [5]. Meanwhile, increasingly more commercial applications have implemented DNNs in their systems for solving different tasks with high accuracy to improve the performance of their products.

To achieve stable recognition performance, the inputs of these models should be drawn from the same distribution as the training data that was used to train the model [33]. However, in the real-world, the inputs are uploaded by users, and thus the application can be used in unusual environments or be utilized for other-than-intended purposes, which means that these input samples can be drawn from

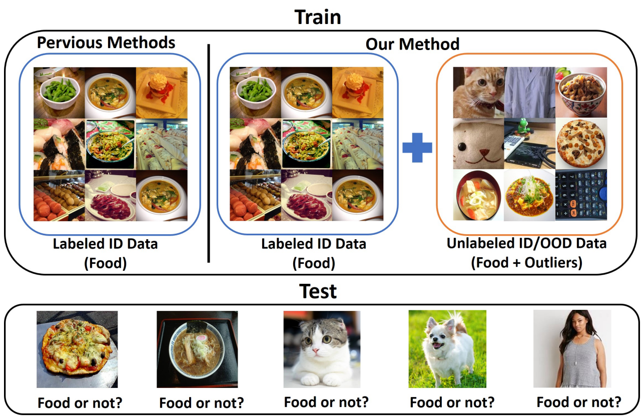


Figure 1: Experimental settings of OOD detection. Our method utilizes both labeled ID data and unlabeled ID/OOD data for training, which is different from previous methods. Please notice that we do not know which semantic class the unlabeled sample belongs to or whether the unlabeled sample is ID or OOD.

different distributions and lead DNNs to provide wrong predictions. Therefore, for these applications, it is important to accurately detect out-of-distribution (OOD) samples.

In this work, we propose a new setting for unsupervised out-of-distribution detection. While previous studies [9, 14, 16, 25, 26] only use labeled ID data to train the neural network under supervision, we also utilize unlabeled data in the training process. Fig. 1 shows our experimental settings of OOD detection for food recognition. Though we do not know the semantic class of the unlabeled sample or whether the unlabeled sample is ID or OOD, we find that these data are helpful for improving the performance of OOD detection and this kind of unlabeled data can be obtained easily in real-world applications.

To utilize these unlabeled data, we also propose a novel out-of-distribution detection method for DNNs. Many OOD detection algorithms attempt to detect OOD samples using the confidence of the classifier [9, 14, 16, 26]. For each input, a confidence score is evaluated based on a pre-trained classifier, then the score is compared to a threshold, and a label is assigned to the input according

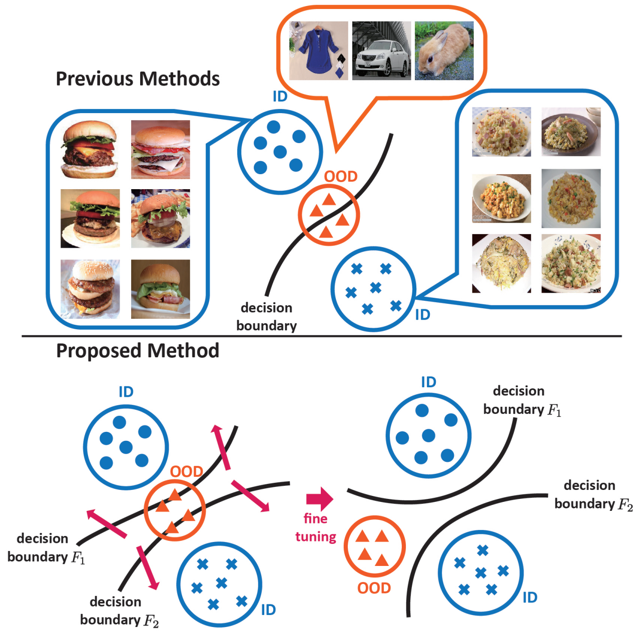


Figure 2: Comparison of previous and the proposed OOD detection methods. **Upper**: Previous methods try to detect OOD samples via the distance from the decision boundary. **Lower**: The proposed method detects OOD samples by the discrepancy between two classifiers.

to whether the confidence score is greater than the threshold. Those samples having lower confidence scores (which means they are closer to the decision boundary) are classified as OOD samples, as shown in the upper part of Fig. 2. In the previous works, they used CIFAR-10/CIFAR-100 as ID and other datasets, TinyImageNet/LSUN/iSUN as OOD. Though there is a small overlap of classes between ID and OOD, we follow the same setting for comparisons.

Although the existing methods are effective on some datasets, they still exhibit poor performance when ID dataset has big class numbers. For example, using CIFAR-100 [13] (a natural image dataset with 100 classes) as the in-distribution (ID) dataset and TinyImageNet [4] (another natural image dataset with 200 classes) as the OOD dataset, which means current methods cannot separate the confidence scores of ID samples and OOD samples well enough.

To overcome this problem, we introduce a two-head deep convolutional neural network (CNN) with one common feature extractor and two separated classifiers as shown in Fig. 3. Since OOD samples are not clearly categorized into classes of ID samples or far from the distribution of ID samples, the two classifiers having different parameters will be confused and output different results. Consequently, as shown in the lower left part of Fig. 2, OOD samples will exist in the gap of the two decision boundaries, which makes it easier to detect OOD samples. To achieve better performance, we further fine-tune the neural network to correctly classify labeled ID samples, and maximize the discrepancy between the decision boundaries of two classifiers, simultaneously. Please note that we do not use labeled OOD samples for training in our method.

We evaluate our method on a diverse set of in- and out-of-distribution dataset pairs. In many settings, our method outperforms other methods by a large margin. The contributions of our paper are summarized as follows:

- We propose a novel experimental setting and a novel training methodology for out-of-distribution detection in neural networks. Our method does not require labeled OOD samples for training and can be easily implemented on any modern neural architecture.
- We propose utilizing the discrepancy between two classifiers to separate in-distribution samples and OOD samples.
- We evaluate our method on state-of-the-art network architectures, such as DenseNet [10] and Wide ResNet (WRN) [32], across several out-of-distribution detection tasks, including not only several OOD detection benchmarks but also real-world simulation datasets.

2. Related Work

Currently, there are several different methods for out-of-distribution detection. A summary of key methods described are shown in Table 1.

As the simplest method, Hendrycks & Gimpel [9] attempted to detect OOD samples depending on the predicted softmax class probability, which is based on the observation that the prediction probability of incorrect and OOD samples tends to be lower than that of the correct samples. However, they also found that some OOD samples still can be classified overconfidently by pre-trained neural networks, which limits the performance of detection.

To improve the effectiveness of Hendrycks & Gimpel's method [9], Lee et al. [14] used modified generative adversarial networks [7], which involves training a generator and a classifier simultaneously. They trained the generator to generate 'boundary' OOD samples that appear to be at the boundary of the given in-distribution data manifold, while the classifier is encouraged to assign these OOD samples uniform class probabilities in order to generate less confident predictions on them.

Liang et al. [16] also proposed an improved solution, which applies temperature scaling and input preprocessing, called Out-of-DIstribution Detector for Neural Networks (ODIN). They found that by scaling the unnormalized outputs (logits) before the final softmax layer by a large constant (temperature scaling), the difference between the largest logit and the remaining logits is larger for ID samples than for OOD samples, which shows that the separation of the softmax scores between ID and OOD samples

Table 1: Summary of recent related methods. Model change is how the method modifies the original classification network. Test complexity equals the required number of passing over the network times the number of the network. Training data is the type of data used for training by each method. AUROC is the area under receiver characteristic curve (detailed in Section 4). Performance is shown for DenseNet trained on CIFAR-100 as ID and TinyImageNet-resized as OOD.

Method	Input pre-processing	Model change	Test complexity	Training data	AUROC
Hendrycks & Gimpel [9]	No	No	1	Labeled ID data	71.6
ODIN [16]	Yes	No	3	Labeled ID data	90.7
ELOC [26]	Yes	Ensemble	15	Labeled ID data	96.3
Proposed	No	Fine tune	2	Labeled ID data & unlabeled data	99.6

is increased. In addition, if they add some small perturbations to the input through the loss gradient, which increases the maximum predicted softmax score, the increases on ID samples are greater than those on OOD samples. Based on these observations, the authors first scaled the logits at a high temperature value to calibrate softmax scores and then pre-processed the input by perturbing it with the loss gradient to further increase the difference between the maximum softmax scores of ID and OOD samples. Their approach outperforms the baseline method [9].

Lee et al. [15] and Quintanilha et al. [20] extracted low- and upper-level features from DNNs to calculate a confidence score for detecting OOD samples. However, these two methods require 1,000 labeled OOD samples to train a logistic regression detector to achieve stable performance.

Some studies [11, 12, 27] utilized the hierarchical relations between labels and trained two classifiers to have different generality (a general classifier and a specific classifier) by using different levels of the label hierarchy. OOD samples can be detected by incongruence between the general classifier and the specific classifier, but the requirement of label hierarchy limits the application of these methods.

There are some other studies on open-set classification [1, 2, 6, 21, 23, 24, 30], which involve tasks very similar to OOD detection. Bendale & Boult [2] proposed a new layer called openMax that can calculate the score for an unknown class by taking the weighted average of all other classes obtained from a Weibull distribution.

The current state-of-the-art method for OOD detection is the ensemble of self-supervised leave-out classifiers proposed by Vyas et al. [26]. They divided the training ID data into K partitions and assign one partition as OOD and the remaining partitions as ID to train K classifiers by a novel loss function, called margin entropy loss, to increase the prediction confidence of ID samples and decrease the prediction confidence of OOD samples. During test time, they used an ensemble of these K classifiers for detecting OOD samples in addition to temperature scaling and input pre-processing proposed in ODIN [16].

Compared to previous studies, our method fine-tunes the neural network by utilizing unlabeled data for unsupervised learning. Our unlabeled data is all or a part of test data.

3. Method

In this section, we present our proposed method for OOD detection. First, we describe the problem statements in Section 3.1. Second, we illustrate the overall concept of our method in Section 3.2. Then, our loss function is explained in Section 3.3 and we detail the actual training procedure of our method in Section 3.4. Finally, we introduce the method used to detect OOD samples at inference time in Section 3.5.

3.1. Problem Statement

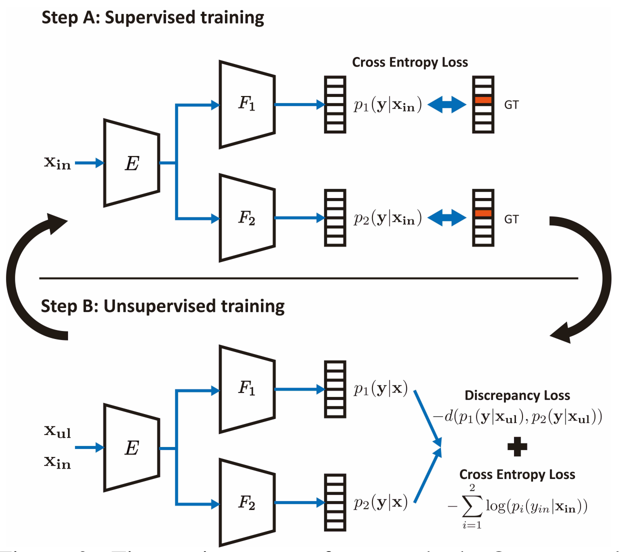
We suppose that an ID image-label pair, $\{\mathbf{x_{in}}, y_{in}\}$, drawn from a set of labeled ID images $\{X_{in}, Y_{in}\}$, is accessible, as well as an unlabeled image, $\mathbf{x_{ul}}$, drawn from unlabeled images X_{ul}. The ID sample $\{\mathbf{x_{in}}, y_{in}\}$ can be classified into K classes, which means $y_{in} \in K$. Please note that $\mathbf{x_{ul}}$ can be either an ID image or an OOD image and $\exists\{\mathbf{x_{ul}}, y_{ul}\}, y_{ul} \notin K$, so we do not know whether this image is from in- or out-of-distribution. Unlike previous methods, we use $\mathbf{x_{ul}}$ for unsupervised training, which is realistic for real-world applications.

The goal of our method is to distinguish whether the image $\mathbf{x_{ul}}$ is from in-distribution or not. For this objective, we have to train the network to predict different softmax class probabilities for ID samples and OOD samples.

3.2. Overall Concept

Hendrycks & Gimpel [9] showed that the prediction probability of OOD samples tends to be lower than the prediction probability of ID samples; thus, OOD samples are closer to the class boundaries and more likely to be misclassified or classified with low confidence by the classifier learned from ID samples (the upper part of Fig. 2).

Based on their findings, we further propose a two-head CNN inspired by [22], consisting of a feature extractor network, E, which takes inputs $\mathbf{x_{in}}$ or $\mathbf{x_{ul}}$, and two classifier networks, F_1 and F_2, which take features from E and classify them into K classes. Classifier networks F_1 and F_2 output a K-dimensional vector of logits; then, the class probabilities can be calculated by applying the softmax function for the vector. The notation $p_1(\mathbf{y}|\mathbf{x})$ and $p_2(\mathbf{y}|\mathbf{x})$ are used to denote the K-dimensional softmax class probabilities for input $\mathbf{x}$ obtained by F_1 and F_2, respectively. Differ-



Step A: Supervised training
Cross Entropy Loss
$\mathbf{x_{in}}$
E
F_1
F_2
$p_1(\mathbf{y}|\mathbf{x_{in}})$
$p_2(\mathbf{y}|\mathbf{x_{in}})$
GT
GT
Step B: Unsupervised training
$\mathbf{x_{ul}}$
$\mathbf{x_{in}}$
$p_1(\mathbf{y}|\mathbf{x})$
$p_2(\mathbf{y}|\mathbf{x})$
Discrepancy Loss
$-d(p_1(\mathbf{y}|\mathbf{x_{ul}}), p_2(\mathbf{y}|\mathbf{x_{ul}}))$
Cross Entropy Loss
$-\sum_{i=1}^{2}\log(p_i(y_{in}|\mathbf{x_{in}}))$

Figure 3: Fine-tuning steps of our method. Our network has one extractor (E) and two classifiers (F_1, F_2). **Step A**: Train the network to classify ID samples correctly under supervision. **Step B**: The classifiers learn to maximize the discrepancy in an unsupervised manner, which helps to detect OOD samples.

ing from [22] which aligns the distributions of two datasets for domain adaptation, we train the network on different loss functions with a different training procedure to detect the difference between the distributions of two datasets.

We found that when the two classifiers (F_1 and F_2) are initialized with random initial parameters and then trained on ID samples supervisedly, they will have different characteristics and classify OOD samples differently (the lower part side of Fig. 2). Fig. 4 shows the disagreement (L1 distance) between the two classifiers' outputs of unlabeled ID (CIFAR-10) and OOD (TinyImageNet-resized and LSUN-resized) samples after training the network on labeled ID samples supervisedly. We can confirm that most OOD samples have larger discrepancy than ID samples in Fig. 4.

By utilizing this characteristic, if we can measure the disagreement between the two classifiers and train the network to maximize this disagreement, the network will push OOD samples outside the manifold of ID samples. Discrepancy, $d(p_1(\mathbf{y}|\mathbf{x}), p_2(\mathbf{y}|\mathbf{x}))$, is introduced to measure the divergence between the two softmax class probabilities for an input. Consequently, we can separate OOD samples and ID samples according to the discrepancy between the two classifiers' outputs.

3.3. Discrepancy Loss

We define the discrepancy loss as the following equation:

$$d(p_1(\mathbf{y}|\mathbf{x}), p_2(\mathbf{y}|\mathbf{x})) = H(p_1(\mathbf{y}|\mathbf{x})) - H(p_2(\mathbf{y}|\mathbf{x})), \quad (1)$$

where $H(\cdot)$ is the entropy over the softmax distribution.

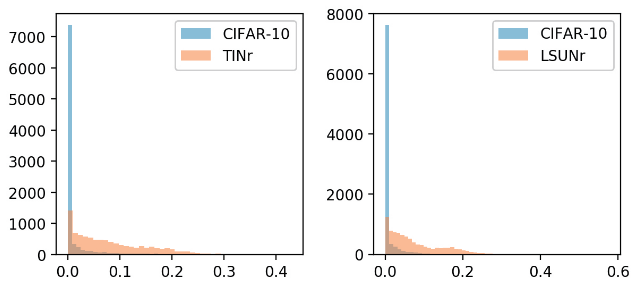


Figure 4: Histogram of the discrepancy (L1 distance) between the two classifiers trained on ID samples.

When the network is trained to maximize this discrepancy term, it maximizes the entropy of F_1's output, which encourages F_1 to predict equal probabilities of all the classes, and minimizes the entropy of F_2's output, which encourages F_2 to predict high probability of one class simultaneously. Since OOD samples are outside the support of the ID samples, the discrepancy between the two classifiers' outputs of OOD samples will be larger. This is demonstrated empirically in Section 4.

3.4. Training Procedure

As the previous discussion in Section 3.2, we need to train our network to classify ID samples correctly and maximize $d(p_1(\mathbf{y}|\mathbf{x}), p_2(\mathbf{y}|\mathbf{x}))$ at the same time. To achieve this, we propose a training procedure consisting of one pre-training step and two repeating fine-tuning steps. Pre-training step uses labeled ID samples $\{X_{in}, Y_{in}\}$ to train the classifier. Then, both $\{X_{in}, Y_{in}\}$ and unlabeled samples X_{ul} are used to train the network for separating ID and OOD samples while keeping correct classification of ID samples in fine-tuning steps. In principle, we use the test data as the unlabeled data. In addition, the unlabeled data can be only a part of the test data. In the ablation studies in Section 4.1.7 , we conduct experiments in the cases of varying sizes and types of unlabeled data.

Pre-training: First, we train the network to learn discriminative features and classify the ID samples correctly under the supervision of labeled ID samples. The network is trained to minimize the cross entropy with the following loss:

$$\mathcal{L}_{sup} = -\frac{1}{|X_{in}|}\sum_{\mathbf{x_{in}}\in X_{in}}\sum_{i=1}^{2}\log(p_i(y_{in}|\mathbf{x_{in}})). \quad (2)$$

Fine-tuning: Once the network converges, we start to fine-tune the network to detect OOD samples by repeating the following two steps at mini-batch level.

- **Step A** First, during the fine-tuning process, we keep training the network to classify the labeled ID samples correctly by supervised learning (Step A in Fig. 3) with Eq. (2) to maintain the manifold of ID samples. This step is helpful to improve the performance of our algorithm.

- **Step B** Then, we train the network to increase the discrepancy in an unsupervised manner in order to make the network detect the OOD samples that do not have the support of the ID samples (Step B in Fig. 3). In this step, we also use labeled ID samples to reshape the support. We add classification loss on the labeled ID samples. The same mini-batch size of labeled and unlabeled samples are utilized to update the model at this step. As a result, we train the network to minimize the following loss:

$$\mathcal{L} = \mathcal{L}_{sup} + \mathcal{L}_{unsup} \tag{3}$$

$$\mathcal{L}_{sup} = -\frac{1}{|X_{in}|} \sum_{\mathbf{x_{in}} \in X_{in}} \sum_{i=1}^{2} \log(p_i(y_{in}|\mathbf{x_{in}})) \tag{4}$$

$$\mathcal{L}_{unsup} = \max\left(m - \frac{\sum_{\mathbf{x_{ul}} \in X_{ul}} d(p_1(\mathbf{y}|\mathbf{x_{ul}}), p_2(\mathbf{y}|\mathbf{x_{ul}}))}{|X_{ul}|}, 0\right). \tag{5}$$

If the average discrepancy of unlabeled samples is greater than the margin m, the unsupervised loss will equal its minimum value, zero; thus, the margin m is helpful for preventing overfitting.

3.5. Inference

At inference time, in order to distinguish between in- and out-of-distribution samples, a straightforward solution would be to use the discrepancy defined in Section 3.3, but this term does not include the discrepancy of each class. We consider the L1 distance between the two classifiers' outputs. When the distance is above a detection threshold δ, we assign the sample as an out-of-distribution sample, denoted by

$$\sum_{i=1}^{K} |p_1(y_i|\mathbf{x}) - p_2(y_i|\mathbf{x})| > \delta. \tag{6}$$

4. Experiments

In this section, we discuss our experimental settings and results. We describe a diverse set of in- and out-of-distribution dataset pairs, neural network architectures and evaluation metrics. We also demonstrate the effectiveness of our method by comparing it against the current state-of-the-art methods, resulting in our method significantly out-performing them. We ran all experiments using PyTorch 0.4.1 [19].

4.1. OOD Detection on benchmarks

As the benchmarks of OOD detection, ODIN [16] and Ensemble of Leave-Out Classifiers (ELOC) [26] introduced several benchmark datasets and evaluation metrics to evaluate the performance of OOD detectors.

4.1.1 Neural Network Architecture

Following [16, 26], we implemented our network based on two state-of-the-art neural network architectures, DenseNet [10] and Wide ResNet (WRN) [32]. We used the modules of DenseNet/Wide ResNet until the *average-pooling* layer just before the last *full-connected* layer as the extractor, and one *full-connected* layer as the classifier.

In the pre-training step proposed in Section 3.4, we used stochastic gradient descent (SGD) to train DenseNet-BC for 300 epochs and Wide ResNet for 200 epochs. The learning rate started at 0.1 and dropped by a factor of 10 at 50% and 75% of the training progress, respectively. After the pre-training step, we further fine-tuned the network in the fine-tuning steps proposed in Section 3.4 for 10 epochs with learning rate 0.1, margin $m = 1.2$ to detect OOD samples.

Furthermore, for fair comparison, we used two classifiers and calculated the average score of these two classifiers as final output in the other methods, ODIN [16] and Ensemble of Leave-Out Classifiers [26], since our method has two classifiers which resulted in a few more parameters.

4.1.2 In-Distribution

CIFAR-10 (contains 10 classes) and CIFAR-100 (contains 100 classes) [13] datasets were used as in-distribution datasets to train deep neural networks for image classification. They both consist of 50,000 images for training and 10,000 images for testing, with the image size of 32×32. The images in the train split were used as X_{in} in our experiment.

4.1.3 Out-of-Distribution

We followed the benchmarks given in [16, 26] and used the OOD datasets below in our experiments:

1. **TinyImageNet (TIN).** The Tiny ImageNet dataset [4] contains 10,000 test images from 200 different classes, which are drawn from the original 1,000 classes of ImageNet [4]. TinyImageNet-crop (TINc) and TinyImageNet-resize (TINr) are constructed by either randomly cropping or downsampling each image to a size of 32×32.
2. **LSUN.** The Large-scale Scene Understanding dataset (LSUN) consists of 10,000 test images from 10 different scene categories.[31]. Similar to TinyImageNet, by randomly cropping and downsampling the LSUN test set, two datasets LSUN-crop (LSUNc) and LSUN-resize (LSUNr) are constructed.
3. **iSUN.** The iSUN is a subset of SUN [28], which is used for gaze tracking, deployed on Amazon Mechanical Turk using a webcam [29]. It contains 8,925 scene images, and all images are downsampled to a size of 32×32.

Table 2: The result of distinguishing in- and out-of-distribution test set data on OOD benchmarks. Our method is compared with ODIN [16] and Ensemble of Leave-Out Classifiers (ELOC) [26]. As described in Section 4.1.1, ODIN [16] and ELOC [26] are modified to have two ensembled *full-connected* layers for fair comparison. ↑ indicates larger value is better, and ↓ indicates lower value is better. All values are percentages.

	OOD dataset	FPR (95% TPR) ↓			Detection Error ↓			AUROC ↑			AUPR In ↑			AUPR Out ↑		
		ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours
Dense-BC CIFAR-10	TINc	3.6	1.5	**0.1**	4.2	3.0	**0.7**	99.2	99.6	**99.9**	99.2	99.6	**100.0**	99.2	99.6	**99.9**
	TINr	10.1	3.2	**1.7**	6.9	4.0	**2.3**	98.2	99.3	**99.6**	98.3	99.3	**99.6**	98.1	99.2	**99.6**
	LSUNc	6.0	3.4	**0.2**	5.3	4.1	**0.7**	98.7	99.3	**99.9**	98.7	99.3	**99.9**	98.6	99.3	**99.9**
	LSUNr	3.5	1.4	**0.4**	4.2	2.7	**1.1**	99.2	99.6	**99.9**	99.3	99.6	**99.9**	99.2	99.6	**99.9**
	iSUN	5.9	-	**0.6**	5.3	-	**1.3**	98.9	-	**99.9**	99.0	-	**99.9**	98.9	-	**99.9**
Dense-BC CIFAR-100	TINc	20.6	8.8	**0.2**	10.2	6.6	**0.7**	96.4	98.3	**99.9**	96.7	98.4	**99.9**	96.1	98.3	**99.9**
	TINr	43.1	20.6	**1.9**	17.2	10.2	**2.0**	90.7	96.2	**99.6**	91.0	96.5	**99.6**	89.8	96.0	**99.7**
	LSUNc	21.9	16.2	**0.3**	10.1	9.3	**0.6**	95.9	97.0	**99.9**	96.4	97.3	**99.9**	96.0	96.8	**99.9**
	LSUNr	43.2	13.1	**0.4**	24.5	7.7	**0.6**	91.0	97.6	**99.9**	91.5	97.9	**99.9**	89.8	97.3	**99.9**
	iSUN	45.4	-	**1.3**	17.2	-	**1.6**	90.5	-	**99.7**	90.9	-	**99.6**	89.1	-	**99.7**
WRN-28-10 CIFAR-10	TINc	16.6	1.5	**0.2**	8.9	3.0	**0.6**	96.9	99.6	**100.0**	97.3	99.6	**100.0**	96.5	99.6	**100.0**
	TINr	6.1	5.5	**0.8**	5.5	5.1	**1.8**	98.8	98.9	**99.7**	98.9	99.0	**99.7**	98.8	98.8	**99.7**
	LSUNc	20.3	1.6	**0.0**	9.6	3.0	**0.2**	96.4	99.6	**100.0**	96.9	99.6	**100.0**	95.8	99.5	**100.0**
	LSUNr	4.6	0.9	**0.4**	4.7	2.5	**1.7**	99.0	99.7	**99.8**	99.1	99.7	**99.8**	99.0	99.7	**99.8**
	iSUN	3.7	-	**0.3**	4.3	-	**1.1**	99.2	-	**99.9**	99.3	-	**99.9**	99.2	-	**99.9**
WRN-28-10 CIFAR-100	TINc	33.3	8.6	**0.6**	13.4	6.3	**1.6**	93.9	98.5	**99.8**	94.6	98.6	**99.7**	92.8	98.4	**99.8**
	TINr	35.8	18.9	**1.6**	15.4	9.1	**2.3**	92.7	96.8	**99.6**	93.2	97.1	**99.5**	92.2	96.4	**99.6**
	LSUNc	34.9	25.1	**0.5**	15.5	10.6	**1.4**	92.7	96.0	**99.8**	93.1	96.5	**99.8**	92.5	95.5	**99.8**
	LSUNr	34.9	12.8	**0.6**	14.9	7.4	**1.4**	93.1	97.6	**99.7**	93.6	97.8	**99.6**	92.8	97.4	**99.8**
	iSUN	34.1	-	**0.9**	14.6	-	**1.4**	93.3	-	**99.6**	93.9	-	**99.5**	92.5	-	**99.7**

For each in-distribution dataset (test split) and each out-of-distribution dataset, 1,000 images (labeled as ID or OOD) were randomly held out for validation, such as parameter tuning and early stopping, while the remaining test images containing unlabeled ID or OOD samples were used as X_{ul} for unsupervised training and evaluation. These datasets are provided as a part of ODIN [16] code release[1].

4.1.4 Evaluation Metrics

We followed the same metrics used by [16, 26] to measure the effectiveness of our method in distinguishing between in- and out-of-distribution samples. TP, TN, FP, FN are used to denote true positives, true negatives, false positives, and false negatives, respectively.

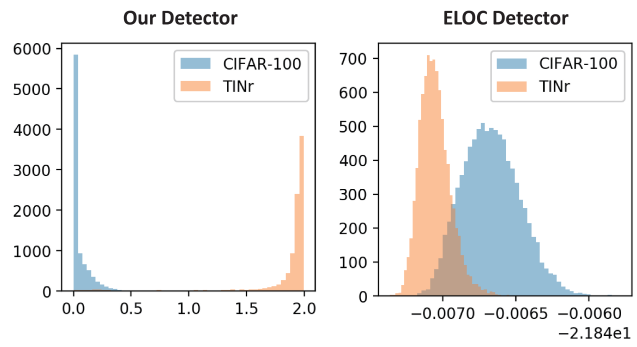
1. **FPR at 95% TPR** shows the false positive rate (FPR) at 95% true positive rate (TPR). True positive rate can be computed by TPR = TP / (TP+FN), while the false positive rate (FPR) can be computed by FPR = FP / (FP+TN).
2. **Detection Error** measures the minimum misclassification probability, which is calculated by the minimum average of false positive rate (FPR) and false negative rate (FNR) over all possible score thresholds.
3. **AUROC** is the Area Under the Receiver Operating Characteristic curve and can be calculated by the area under the FPR against TPR curve.
4. **AUPR In** is the Area Under the Precision-Recall curve and can be calculated by the area under the precision = TP / (TP+FP) against the recall = TP / (TP+FN) curve. For AUPR In, in-distribution images are specified as positive.
5. **AUPR Out** is similar to the metric AUPR-In. The difference is that out-of-distribution images are specified as positive in AUPR Out.

4.1.5 Experimental Results

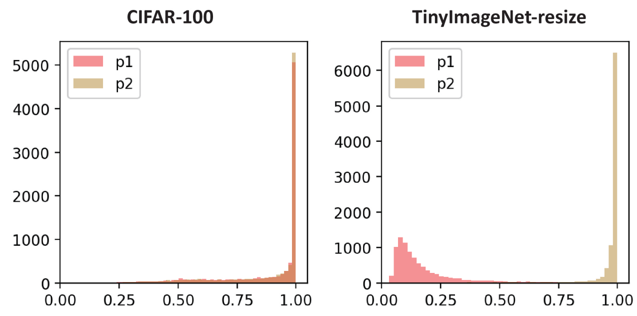
The results are summarized in Table 2, which shows the comparison of our method, ODIN [16] and Ensemble of Leave-Out Classifiers (ELOC) [26] on various benchmarks. In addition, ELOC [26] does not have results for iSUN as an OOD dataset because they use the whole iSUN as a validation dataset. We implemented two ensembled *full-connected* layers in ODIN [16] and ELOC [26], and their performance is almost the same as that of the single classifier (one *full-connected* layer) in their original papers.

Table 2 clearly shows that our approach significantly outperforms other existing methods, including ODIN [16] and ELOC [26] (which is the ensemble of five models), across all neural network architectures on all of the dataset pairs.

[1] `github.com/facebookresearch/odin`



(a) Histogram of ID and OOD detection scores of the proposed method and ELOC [26].



(b) Histogram of the two classifiers' maximum softmax scores after the fine-tuning.

Figure 5: The visualization of the result.

TinyImageNet-resize, LSUN-resize and iSUN, which contain the images with full objects as opposed to the cropped parts of objects, are considered more difficult to detect. Our proposal shows highly accurate results on these more challenging datasets.

Noticeably, our method is very close to distinguishing between in- and out-of-distribution samples perfectly on most dataset pairs. As shown in Fig. 5a, we compared our OOD detector based on DenseNet-BC with ELOC [26] when in-distribution data is CIFAR-100 and out-of-distribution data is TinyImageNet-resize. These figures show that the proposed method has much less overlap between OOD samples and ID samples on all the dataset pairs compared to ELOC [26], indicating that our method separates ID and OOD samples very well.

Another merit of our method is that we can use a simple threshold 1.0 to separate ID and OOD samples as shown in Fig. 5a. On the other hand, it is very difficult to decide an interpretable threshold of that value in ELOC.

We also plot the histogram of the two classifiers' maximum softmax scores of X_{ul} in Fig. 5b. Fig. 5b shows that the distribution of X_{ul}'s p_1 and p_2 differs significantly according to whether the sample is ID (CIFAR-100) or not (TINr) after the fine-tuning. For convenience, we use p_{1k} and p_{2k} to denote the probability output of p_1 and p_2 for class k, respectively. The discrepancy loss makes OOD samples' $\max_k p_{1k}$ close to $1/K$ and $\max_k p_{2k}$ close to 1. On the other hand, ID samples' $\max_k p_{1k}$ are almost the same as $\max_k p_{2k}$ due to the support we added to ID samples in Step A and Eq. (3) in Step B.

Table 3: Results of ablation studies on CIFAR-100 as ID and TinyImageNet-crop as OOD.

#ID in X_{ul}	9k	5k	2k	1k
#OOD in X_{ul}	9k	5k	2k	1k
Detection Error (%)	0.7	0.5	0.9	1.5
ID Discrepancy Loss	0.05	0.06	0.08	0.05
OOD Discrepancy Loss	3.08	3.05	2.84	2.68

#ID in X_{ul}	9k	5k	9k	9k
#OOD in X_{ul}	2k	2k	1k	500
Detection Error (%)	0.3	0.4	1.2	3.8
ID Discrepancy Loss	0.62	0.26	1.05	1.08
OOD Discrepancy Loss	4.03	3.90	3.93	3.40

Table 4: Results of ablation studies on CIFAR-100 as ID and other datasets as OOD.

OOD in X_{ul}	TINc+LSUNc	TINc	LSUNc
OOD for testing	TINc+LSUNc	LSUNc	TINc
Detection Error (%)	0.2	0.5	0.7
ID Discrepancy Loss	0.03	0.05	0.04
OOD Discrepancy Loss	3.53	3.20	2.39

4.1.6 Limitation

Since our method needs to fine-tune the classifier to detect OOD samples which changes the decision boundary, we observed a 5% drop of classification accuracy compared to the original classifier before the fine-tuning. This problem could be solved by using the original classifier to classify ID samples with some runtime increase and it is still much more acceptable than ELOC [26] using an ensemble of five models which needs more runtime and computing resources.

4.1.7 Ablation Studies

Since our approach accesses to unlabeled data X_{ul}, we further analyzed the effects of the following factors:

The size and the data balance of X_{ul}. We used CIFAR-100 as ID and TinyImageNet-crop as OOD and we changed the number of ID and OOD samples in X_{ul} for unsupervised training. The result are summarized in Table 3, which shows that our proposed method works under various X_{ul} settings. Even when 9,000 ID samples and 500 OOD samples are included in X_{ul}, our method still have better performance than [16, 26], which means our method is robust to the size of X_{ul} and the percentage of OOD data in X_{ul}. *Please notice that we used all 9,000 ID samples and 9,000 OOD samples for testing, which means totally unseen samples were included during evaluation.*

The selection of OOD data in X_{ul}. To show the effectiveness of our method, we also tried various pairs of OOD

Table 5: The result of distinguishing in- and out-of-distribution test set data on real-world simulation. Our method is compared with ODIN [16] and ELOC [26]. $\uparrow$ indicates larger value is better, and $\downarrow$ indicates lower value is better. All values are percentages.

ID dataset	OOD dataset	FPR (95% TPR) $\downarrow$			Detection Error $\downarrow$			AUROC $\uparrow$			AUPR In $\uparrow$			AUPR Out $\uparrow$		
		ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours	ODIN	ELOC	Ours
Food (FOOD-101)	Non Food (TINc)	48.2	36.7	**0.1**	22.4	16.5	**0.2**	85.3	91.5	**100.0**	92.2	91.3	**100.0**	76.1	91.9	**100.0**
	Non Food (LSUNc)	30.6	15.9	**0.1**	16.2	10.1	**0.2**	90.9	96.3	**100.0**	95.0	95.7	**100.0**	86.0	96.8	**100.0**
Fashion (DeepFashion)	Non Fashion (TINc)	57.7	6.1	**0.4**	21.3	5.2	**1.1**	86.4	98.7	**99.9**	95.9	98.9	**100.0**	62.5	98.5	**99.7**
	Non Fashion (LSUNc)	35.8	3.6	**0.2**	14.8	4.2	**0.7**	92.9	99.1	**99.9**	98.0	99.3	**100.0**	78.9	99.0	**99.8**

datasets for unsupervised training and evaluation. Table 4 shows that our method still works when multiple datasets are used as OOD or even when OOD dataset used for unsupervised training is different from the OOD dataset for evaluation.

The relationship between the discrepancy loss and the detection error. The mean discrepancy loss in Eq. (1) of ID and OOD samples in test dataset is shown in Table 3 and Table 4, respectively. These results show that the discrepancy loss of ID samples is smaller than OOD samples in all settings. The detection error is lower when the difference between the discrepancy loss of ID and OOD samples is larger, which means ID and OOD samples can be separated by the divergence between the two classifiers' outputs.

4.2. OOD Detection on real-world simulation

Since our goal is to benefit applications in real world, we also evaluated our method in two cases of real-world simulation to demonstrate the effectiveness of our method.

4.2.1 Neural Network Architecture

As previous experiments, we used and pre-trained the same DenseNet-BC [10] as Section 4.1.1. We further fine-tuned the network in the fine-tuning steps proposed in Section 3.4 for 10 epochs with a learning rate of 0.1 and margin $m = 1.2$ to detect OOD samples.

4.2.2 Real-world Simulation Datasets

Considering domain specific applications, we evaluated our method by two simulations of food and fashion applications because there are services focusing on these domains.

For food recognition, we used FOOD-101 [3], which is a real-world food dataset containing the 101 most popular and consistently named dishes collected from `foodspotting.com`. FOOD-101 consists of 750 images per class for training and 250 images per class for testing. The training images of FOOD-101 [3] are not cleaned and contain some amount of noise. We evaluated our method on FOOD-101 [3] as ID and TinyImageNet-crop (TINc)/LSUN-crop (LSUNc) as OOD.

For fashion recognition, we used DeepFashion [18], a large-scale clothes dataset. We used the Category and Attribute Prediction Benchmark dataset of DeepFashion [18], which consists of 289,222 images of clothes and 50 clothing classes. We used DeepFashion [18] as ID and TinyImageNet-crop (TINc)/LSUN-crop (LSUNc) as OOD.

We resized the FOOD-101 [3] and DeepFashion [18] images to 32×32. For FOOD-101 [3], the original train split was used as X_{in}; 1,000 images from the original test split were used for validation and the remaining test images were used as X_{ul}. For DeepFashion [18], the original train split was used as X_{in}; 1,000 images from the original validation split were used for validation and the original test images were used as X_{ul} for unsupervised training and evaluation.

4.2.3 Experimental Results

Table 5 shows the comparison of our method, ODIN [16] and ELOC [26] on real-world simulation datasets. These results clearly show that our architecture significantly outperforms the other existing methods, ODIN [16] and ELOC [26], by a considerable margin on all datasets. Furthermore, our method nearly perfectly detects non-food and non-fashion images.

5. Conclusion

In this paper, we proposed a novel approach for detecting out-of-distribution data samples in neural networks, which utilizes two classifiers to detect OOD samples that are far from the support of the ID samples. Our method does not require labeled OOD samples to train the neural network. We extensively evaluated our method not only on OOD detection benchmarks, but also on real-world simulation datasets. Our method significantly outperformed the current state-of-the-art methods on different DNN architectures across various in and out-of-distribution dataset pairs.

6. Acknowledgements

This work was partially supported by JST CREST JPMJCR1686 and JSPS KAKENHI 18H03254, Japan.

References

[1] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, 2015.

[2] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *CVPR*, 2016.

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *ECCV*, 2014.

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

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Zongyuan Ge, Sergey Demyanov, Zetao Chen, and Rahil Garnavi. Generative openmax for multi-class open set classification. In *BMVC*, 2017.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

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

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

[10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[11] Josef Kittler and Cemre Zor. Delta divergence: A novel decision cognizant measure of classifier incongruence. *IEEE Transactions on Cybernetics*, 2019.

[12] Josef Kittler, Cemre Zor, Ioannis Kaloskampis, Yulia Hicks, and Wenwu Wang. Error sensitivity analysis of delta divergence - a novel measure for classifier incongruence detection. *Pattern Recognition*, 2017.

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

[14] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.

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

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

[17] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017.

[18] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016.

[19] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[20] Igor M. Quintanilha, Roberto de M. E. Filho, José Lezama, Mauricio Delbracio, and Leonardo O. Nunes. Detecting out-of-distribution samples using low-order deep features statistics. In *Submitted to ICLR*, 2019.

[21] Ethan M. Rudd, Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. The extreme value machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[22] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

[23] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[24] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[25] Gabi Shalev, Yossi Adi, and Joseph Keshet. Out-of-distribution detection using multiple semantic label representations. In *NIPS*, 2018.

[26] Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *ECCV*, 2018.

[27] Daphna Weinshall, Alon Zweig, Hynek Hermansky, Stefan Kombrink, Frank W Ohl, Jörn Anemüller, Jörg-Hendrik Bach, Luc Van Gool, Fabian Nater, Tomas Pajdla, et al. Beyond novelty detection: Incongruent events, when general and specific classifiers disagree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[28] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[29] Pingmei Xu, Krista A Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R Kulkarni, and Jianxiong Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *arXiv preprint arXiv:1504.06755*, 2015.

[30] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Classification-reconstruction learning for open-set recognition. In *CVPR*, 2019.

[31] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv: 1506.03365*, 2015.

[32] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[33] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.