Watermarking for Out-of-distribution Detection

Anonymous Author(s)
Affiliation
Address
email

Abstract

Out-of-distribution (OOD) detection aims to identify OOD data based on representations extracted from well-trained deep models. However, existing methods largely ignore the *reprogramming* property of deep models and thus may not fully unleash their intrinsic strength: *without* modifying parameters of a well-trained deep model, we can reprogram this model for a new purpose via data-level manipulation (e.g., adding a specific feature perturbation). This property motivates us to reprogram a classification model to excel at OOD detection (a new task), and thus we propose a general methodology named *watermarking* in this paper. Specifically, we learn a unified pattern that is superimposed onto features of original data, and the model's detection capability is largely boosted after watermarking. Extensive experiments verify the effectiveness of watermarking, demonstrating the significance of the reprogramming property of deep models in OOD detection.

1 Introduction

Deep learning systems in the open world often encounter *out-of-distribution* (OOD) inputs whose label spaces are disjoint with that of training data, known as *in-distribution* (ID) data. For safety-critical applications, deep models should make reliable predictions for ID data, meanwhile detecting OOD data without prediction [2]. This leads to the OOD detection problem [25, 33, 37, 45], which has attracted intensive attention in reliable machine learning.

Identifying OOD data remains non-trivial since deep models can be overconfident with them [35]. As a promising technique, the *classification-based* OOD detection [52] relies on various scoring functions derived by the well-trained classification models, taking those inputs with small scores as OOD cases. In general, the scoring functions can be defined by logit outputs [15, 29], gradients [19], and embedding features [25, 40]. Without interfering with original models or requiring extra computation, they exploit the inherent capability of models learned from only ID sample. In general, these advantages can be critical in reality, where the cost of re-training is prohibitively high and the acquisition of real OOD data is largely limited [52].

Although promising progress has been achieved, previous works largely ignore the *reprogramming* property [9] of deep models: a well-trained model can be repurposed for a new task by a proper transformation of original inputs (e.g., a universal feature perturbation), without modifying any model parameter. For example, a model pre-trained on ImageNet [7] dataset can be reprogrammed for biomedical images [44] and an acoustic model can excel at time series analysis [51]. The above examples have shown the significance of the reprogramming property for deep models, motivating us the *first* attempt to investigate if it can also help to address OOD detection, *i.e., can we reprogram well-trained deep models for OOD detection (a new task)?*

To this end, we propose a novel method, named **watermarking**, to *reprogram* the well-trained classification model for effective OOD detection. In our context, "reprogramming" is reflected by a *watermark* with its features on the shape of original inputs, which is defined as a static pattern that

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

can be added for the test-time inputs (cf., Figure 1). After watermarking, the pre-defined scoring strategy is expected to be enhanced, with an improved gap of OOD scores between ID data and OOD data during the test (cf., Figure 2).

Actually, it is non-trivial to find the proper watermarks since we do not know much about the test-time OOD inputs in advance. To address the issue, we propose a learning framework for effective watermarking. The insight is to make the model produce low scores when it perceives the watermark, except that the pattern of ID inputs is also observed. In this case, the model will have a relatively high score for a watermarked ID input, while the score is low for an OOD case (cf., Figure 2). The reason is that the watermark is not trained to perceive OOD pattern and only the watermark can be observed.

Figure 1: Watermarking on CIFAR-10 [23] with free energy scoring [29]. The left figure is the learned watermark; the middle figure is an original input; the right figure is the watermarked result.

In our realization, we adopt several representative scoring strategies, devising specified learning objectives and proposing a reliable optimization algorithm. To understand our mechanism, Figure 1 depicts the watermark learned on CIFAR-10 [23] dataset, with the free energy scoring [29] (an advanced classification-based method). As we can see, the centre area of the learned watermark largely preserves the original input pattern, containing the semantic message that guides the detection primitively. By contrast, the edge area of the original input is superimposed by the specific pattern of the watermark, which may *encode* the knowledge once hidden by the model in boosting OOD detection. Overall, watermarking can preserve the meaningful pattern of original inputs in detection with further enhancement learned from the trained model. Therefore, the detection capability of the original model is largely improved after watermarking.

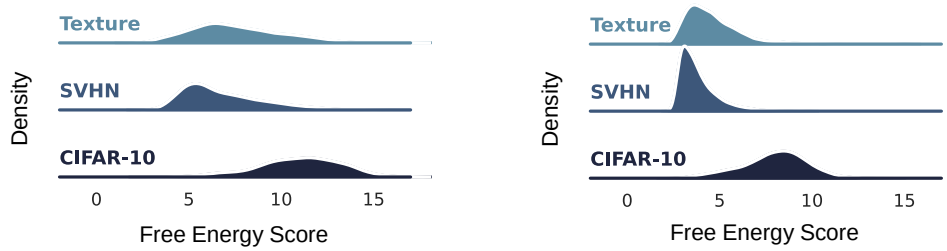
Figure 2 demonstrates the effect of our learned watermark, which is an example with the free-energy scoring. After watermarking, the scoring distributions are much concentrated, and the gap between ID (i.e., CIFAR-10) and OOD (i.e., SVHN [34] and Texture [5] datasets) cases is enlarged notably. We conduct extensive experiments for a wide range of OOD evaluation benchmarks in Section 6 and Appendix C, and the results verify the effectiveness of our proposal.

The success of watermarking takes roots in the following aspects: (1) a well-trained model on classification has the potential to be reprogrammed for OOD detection since they are two closely related tasks; (2) reprogramming has been widely studied, ranging from image classification to time series analysis [7, 44], making our proposal general across various domains; and (3) OOD detection suffers from the lack of knowledge about the real-world OOD distributions. Fortunately, with only data-level manipulation in low dimensions, watermarking can largely mitigate this issue of limited data. Overall, this data-level manipulation is orthogonal to existing methods, and thus provides a new road in OOD detection and can inspire more ways to design OOD detection methods in the future.

2 Related Works

To begin with, we briefly review the related works in OOD detection and model reprogramming. Please refer to Appendix A for the detailed discussion.

OOD Detection discerns ID and OOD data by their gaps regarding specified metrics/scores, and existing methods can mainly be attributed into three categories [52], namely, the *classification-based* methods, the *density-based* methods, and the *distance-based* methods. Specifically, the classification-based methods [15, 19, 29, 40] use representations extracted from the well-trained models in OOD scoring; and the distance-based methods [3, 18, 36, 54] measure the distance of inputs from class centers in the embedding space. Moreover, the density-based methods estimate input density with probabilistic models [25, 37, 41], identifying those OOD data with small likelihood values. Since distance-based and density-based methods may suffer from complexity in computation [25] and difficulty in optimization [56], recently, more researchers focus on developing classification-based methods and have made big progress on benchmark datasets [19, 29].



(a) before watermarking

(b) after watermarking

Figure 2: Experimental results before (a) /after (b) watermarking with CIFAR-10 being the *in-distribution* (ID) dataset, SVHN and Texture being the *out-of-distribution* (OOD) datasets. Data with large (small) OOD scores should be taken as ID (OOD) cases, and a larger distribution gap of scoring between ID and OOD cases ensures a better detection performance. As we can see, after watermarking, the scoring distributions are much concentrated and the gap between ID and OOD cases is enlarged, demonstrating the improved capability of the original model in OOD detection. The horizontal axes are ignored for illustration, and please refer to Figure 4 for a formal manifestation.

Model Reprogramming repurposes the well-trained models for new tasks with only data-level manipulation [9], indicating that deep models are competent for different jobs without changing model parameters. In previous works, the data-level manipulation typically refers to a static padding pattern (different from our proposal) learned for the target task, which is added to the test-time data. The effectiveness of the model reprogramming is verified across image classification [9, 44] and time-series analysis [13, 51]. In this paper, we use the reprogramming property of deep models for effective OOD detection, which has been overseen previously.

3 Preliminary

Let $\mathcal{X} \subset \mathbb{R}^d$ be the input space and $\mathcal{Y} = \{1, \dots, c\}$ be the label space. We consider the ID distribution $D^{\text{ID}}_{\mathcal{X},\mathcal{Y}}$ defined over $\mathcal{X} \times \mathcal{Y}$, the training sample $S_n = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ of size n independently drawn from $D^{\text{ID}}_{\mathcal{X},\mathcal{Y}}$, and a classification model $f : \mathcal{X} \to \mathbb{R}^c$ (with logit outputs) well-trained on S_n.

Based on the model $f(\cdot)$, the goal of the classification-based OOD detection is to design a detection model $g : \mathcal{X} \to \{0, 1\}$ that can distinguish test-time inputs given by the ID distribution $D^{\text{ID}}_{\mathcal{X}}$ from that of the OOD distribution $D^{\text{OOD}}_{\mathcal{X}}$. In general, $D^{\text{OOD}}_{\mathcal{X}}$ is defined as an irrelevant distribution of which the label set has no intersection with $\mathcal{Y}$, and thus should not be predicted by $f(\cdot)$. Overall, with 0 the OOD case and 1 the ID case, a detection model $g(\cdot)$ is defined as

$$g(\boldsymbol{x}; \tau) = \begin{cases} 1 & s(\boldsymbol{x}; f) \geq \tau \\ 0 & s(\boldsymbol{x}; f) < \tau \end{cases}, \tag{1}$$

where $\tau \in \mathbb{R}$ is a threshold and $s : \mathcal{X} \to \mathbb{R}$ is the scoring function defined by $f(\cdot)$, typically with its fixed parameters. Here, we focus on two representative methods in the classification-based OOD detection, namely, the *softmax scoring* and the *free energy scoring*.

Softmax Scoring Function [15] uses the maximum softmax prediction in OOD detection, of which the scoring function $s_{\text{SM}}(\cdot)$ is given by

$$s_{\text{SM}}(\boldsymbol{x}; f) = \max_k \ \texttt{softmax}_k \ f(\boldsymbol{x}), \tag{2}$$

where $\texttt{softmax}_k(\cdot)$ denotes the k-th element of the softmax outputs. In general, with a large (small) $s_{\text{SM}}(\boldsymbol{x}; f)$, the detection model will take the input $\boldsymbol{x}$ as an ID (OOD) case. However, previous works have found that the softmax prediction may be overconfident on OOD inputs [29], which is detrimental for its detection capability.

Free Energy Scoring Function [29] adopts the free energy function for scoring, defined by the logit outputs with a negative `logsumexp` operation, namely,

$$s_{\text{FE}}(\boldsymbol{x}; f) = -\log \sum_k \exp f_k(\boldsymbol{x})/T, \tag{3}$$

where $T > 0$ is the temperature parameter. It aligns with the density of inputs to some extent, and thus is less susceptible to the overconfidence issue mentioned above [29].

4 Watermarking Strategy

This section introduces the key concepts of watermarking for classification-based OOD detection.

Definition. A watermark $\boldsymbol{w} \in \mathbb{R}^d$ is a unified pattern with the exact shape as original inputs. It is added to test-time inputs statically, and we refer $\boldsymbol{w} + \boldsymbol{x}$ a *watermarked input* for $\forall \boldsymbol{x} \in \mathcal{X}$. In expectation, regarding the specified scoring function $s(\cdot)$, our watermarking should make the model excel at OOD detection for watermarked data.

Learning Strategy. Given the scoring function $s(\cdot)$, it is challenging to devise the exact watermark pattern by predefined rules. Therefore, for the proper watermarks in OOD detection, we need to devise learning objectives regarding watermarks, which consider both ID and OOD cases.

We generally have no information about the OOD distribution $D_{\mathcal{X},\mathcal{Y}}^{\text{OOD}}$, while we hope the model to produce low scores for unseen OOD inputs after watermarking. For this challenge, we adopt the following training strategy—the model should produce high scores if both the watermark and ID data are observed, while the score should be low if the model perceives only the watermark. From the lens of our model, the scores should remain low if a watermarked OOD input is given since the learning procedure does not see any OOD data during training, and only the watermark can be observed.

Benefits of Watermarking. Watermarking directly reprograms the model to make an adaptation to our specified task for scoring, such that the detection capability of the original model is largely improved. By contrast, previous methods typically adapt to their specified tasks by only the threshold τ as in Eq. (1). However, it requires the trade-off between false positive (ID) and false negative (OOD) rates when densities of scoring are non-separable (cf., Figure 2(a)).

Further, watermarking enjoys the benefits of previous classification-based methods in that we do not modify the original training procedure in classification, making our proposal easy to be deployed in real-world systems. Although the watermark also should be learned, the parameter space is in low dimension, and the learning procedure could be conducted *post-hoc* after the systems are deployed.

Comparison with Existing Works. In OOD detection, this paper is a first attempt in using the reprogramming property of deep models, leading to an effective learning framework named watermarking. At first glance, our methodology is seemingly similar to ODIN [28], which also conducts data-level perturbation for OOD detection. However, their instance-specified perturbation relies on extra backward-forward iterations during the test, which is not required in our method. Further, ODIN is designed for the softmax scoring, but our proposal is much general in OOD detection.

5 Realization of Watermarking Strategy

In this section, we discuss our learning framework of watermarking in detail.

Learning Objectives. As mentioned above, we need to consider the ID and OOD situations separately, with the associated objective functions denoted by $\ell^{\text{IN}}(\cdot)$ and $\ell^{\text{OUT}}(\cdot)$. For the ID case, the ID training sample is required, where we make the high scores for their watermarked counterparts. By contrast, since we typically lack knowledge about the test-time OOD data, only the watermark is used here, and we expect the model to produce the score as low as possible when perceiving the watermark.

Further, since only the watermark is adopted for training in the OOD case, the learned watermark is pretty sensitive regarding the detection model, i.e., the model may return different predictions when facing small perturbations. Thus, it hardly generalizes to real-world OOD cases, and the watermarked OOD inputs may not guarantee the low scores. To this end, the watermark is further perturbed during training. Here we adopt the Gaussian noise, leading to the perturbed watermark of the form,

$$\boldsymbol{\epsilon} + \boldsymbol{w}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_1 \mathbf{I}_d), \tag{4}$$

where $\boldsymbol{\epsilon}$ is the (i.i.d.) Gaussian noise of d-dimension with the $\mathbf{0}$ mean and the standard deviation $\sigma_1 \mathbf{I}_d$. In summary, the overall risk is written as,

$$\mathcal{L}_n(\boldsymbol{w}) = \underbrace{\sum_n \ell^{\text{IN}}(\boldsymbol{x}_i + \boldsymbol{w}, y_i; f)}_{\mathcal{L}_n^{\text{IN}}(\boldsymbol{w})} + \beta \underbrace{\sum_n \ell^{\text{OUT}}(\boldsymbol{\epsilon}_j + \boldsymbol{w}; f)}_{\mathcal{L}_n^{\text{OUT}}(\boldsymbol{w})}, \tag{5}$$

with $\beta \geq 0$ the trade-off parameter, $\mathcal{L}_n^{\text{IN}}(\boldsymbol{w})$ the risk for ID data, and $\mathcal{L}_n^{\text{OUT}}(\boldsymbol{w})$ the risk for OOD data.

Optimization. Data-level optimization remains difficult in deep learning, of which the results may get stuck at suboptimal points [47]. A common approach is to use the signum of first-order gradients, guiding the updating rule of the current watermark via

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \texttt{sign}(\nabla_{\boldsymbol{w}} \mathcal{L}_n(\boldsymbol{w})), \tag{6}$$

where $\texttt{sign}(\cdot)$ denotes the signum function and $\alpha > 0$ is the step size.

Further, for generality and insensibility, we prefer the solution that lies in the neighbourhood having uniformly low loss, i.e., with a smooth loss landscape [21]. Therefore, we adopt the *sharpness-aware minimization* (SAM) [10], an effective optimization framework in the seek of both the low loss value and the smooth loss landscape. Specifically, given the original risk $\mathcal{L}_n(\boldsymbol{w})$, the SAM problem is:

$$\mathcal{L}_n^{\text{SAM}}(w) = \max_{||\boldsymbol{\kappa}||_2 \leq \rho} \underbrace{[\mathcal{L}_n(\boldsymbol{w} + \boldsymbol{\kappa}) - \mathcal{L}_n(\boldsymbol{w})]}_{\text{sharpness}} + \mathcal{L}_n(\boldsymbol{w}) = \max_{||\boldsymbol{\kappa}||_2 \leq \rho} \mathcal{L}_n(\boldsymbol{w} + \boldsymbol{\kappa}) \tag{7}$$

where $\rho \geq 0$ is a constraint. For efficiency, the SAM makes the first-order Taylor expansion w.r.t. $\boldsymbol{\kappa}$ around $\mathbf{0}$, obtaining the approximated solution of the form [1]:

$$\boldsymbol{\kappa} = \rho \texttt{sign}(\nabla_{\boldsymbol{w}} \mathcal{L}_n(\boldsymbol{w})) \frac{|\nabla_{\boldsymbol{w}} \mathcal{L}_n(\boldsymbol{w})|^{q-1}}{(||\nabla_{\boldsymbol{w}} \mathcal{L}_n(\boldsymbol{w})||_q^q)^{1/p}}, \tag{8}$$

where $1/p + 1/q = 1$ and we set $p = q = 2$ for simplicity. Therefore, the final form of the SAM is written as $\mathcal{L}_n(\boldsymbol{w} + \boldsymbol{\kappa})$, with corresponding updating rule of

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \texttt{sign}(\nabla_{\boldsymbol{w}} \mathcal{L}_n(\boldsymbol{w} + \boldsymbol{\kappa})), \tag{9}$$

yielding an efficient optimization algorithm that induces the effective watermark.

The Overall Algorithm. In summary, we describe the overall learning framework. To begin with, the watermark is initialized by the (i.i.d.) Gaussian noise with the $\mathbf{0}$ mean and a small standard deviation $\sigma_2 \mathbf{I}_d$, and the learning procedure consists of three stages for each updating step:

- Negative sampling: a set of noise data given by Eq. (4) is sampled, assuming be of the size m as that of the mini-batch regarding the ID sample;
- Risk calculating: the risk for ID and OOD cases are computed, and the overall risk is given by their sum with a trade-off parameter β as in Eq. (16);
- Watermark updating: the first-order gradient guides the pixel-level update of the watermark, using the signum of gradients and the SAM to make a reliable update as in Eq. (9).

The learned watermark is added to test-time inputs for OOD detection, and the detection model with the pre-defined scoring function is then deployed. Appendix B summarizes our learning framework of watermarking. Moreover, two specifications of watermarking are discussed in the following.

Two Realizations. Here, we focus on two representative methods in OOD detection, namely, the softmax scoring and the free energy scoring. For other representative methods in OOD detection, please refer to Appendix C for their descriptions and the experiments.

Softmax Scoring-based Watermarking. Following [16], we set $\ell_{\text{SM}}^{\text{IN}}(\cdot)$ to be the cross entropy loss and $\ell_{\text{SM}}^{\text{OUT}}(\cdot)$ to be the cross entropy regarding the uniform distribution, namely,

$$\ell_{\text{SM}}^{\text{IN}}(\boldsymbol{x}, y; f) = -\log \texttt{softmax}_y f(\boldsymbol{x}) \;\text{ and }\; \ell_{\text{SM}}^{\text{OUT}}(\boldsymbol{x}; f) = -\sum_k \frac{1}{c} \log \texttt{softmax}_k f(\boldsymbol{x}), \tag{10}$$

[1] With an abuse of notation, we denote the estimated solution in the SAM as $\boldsymbol{\kappa}$ for simplicity.

specifying the learning objectives in Eq. (16) for the softmax scoring-based watermarking.

Free-Energy Scoring-based Watermarking. [29] use a set of learning objectives for model re-training with free energy scoring. However, their `logsumexp` operation originating from the free energy function is difficult for optimization, posing notorious computing issues [31]. To this end, we drop the $\log$ operation and make the overall risk always positive by the following learning objectives:

$$\ell_{\text{FE}}^{\text{IN}}(\boldsymbol{x}; f) = \sum_k \exp f_k(\boldsymbol{x})/T \;\text{ and }\; \ell_{\text{FE}}^{\text{OUT}}(\boldsymbol{x}; f) = \sum_k \exp -f_k(\boldsymbol{x})/T, \tag{11}$$

realizing the learning objectives in Eq. (16) for the free-energy scoring.

6 Experiments

In this section, we conduct extensive experiments for watermarking in OOD detection. Specifically, we demonstrate the effectiveness of our method on a wide range of OOD evaluation benchmarks; we conduct ablation analysis for the important hyper-parameters in our learning framework; and we provide further experiments for an improved interpretation of our proposal.

Baselines results are achieved by our re-run of the publicly available codes. All the methods are realized by Pytorch 1.81 with CUDA 11.1, where we use several machines equipped with GeForce RTX 3090 GPUs and AMD Ryzen Threadripper 3960X Processors.

ID and OOD Datasets. Following previous works [29, 32], we use CIFAR-10 and CIFAR-100 [23] datasets as two ID datasets, with data pre-processing including the horizontal random flip and the normalization. In Appendix C, we also conduct experiments on ImageNet [38] dataset, which has a larger semantic space than that of the CIFAR benchmarks. Furthermore, for the OOD datasets, we adopt several commonly-used benchmarks, including Textures [5], SVHN [34], Places365 [57], LSUN [53], and iSUN [50].

Evaluation Metrics. The performance in OOD detection is measured via three representative metrics, which are all threshold-independent [6]: (1) the false positive rate of OOD sample when true positive rate of ID data is at 95% (FPR95); (2) the *area under the receiver operating characteristic curve* (AUROC), interpreted as the probability that an ID input has a greater score than an OOD one; and (3) the *area under the precision-recall curve* (AUPR), which further adjusts for different base rates.

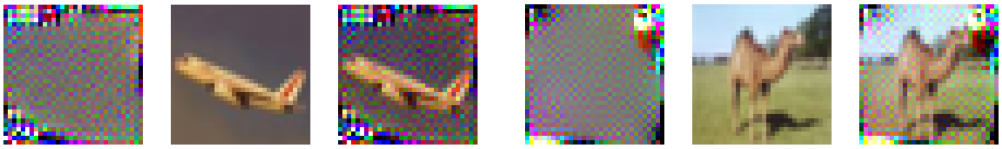
Configuration. Following previous works [29], we employ WideResNet [55] (WRN-40-2) as the backbone model. Specifically, the models are trained for 200 epochs via the stochastic gradient descent, with the batch size 64, the momentum 0.9, and the initial learning rate 0.1. Further, the learning rate is divided by 10 after 100 and 150 epochs.

Hyper-parameters. By default, our method is executed for 50 epochs. The initial learning rate $\alpha = 0.01$ divided by 10 after 25 epochs. σ_2 is fixed to be 0.001 and T is set to be 1 following [29]. Please refer to the Appendix C for the hyper-parameter setups and the selection policy regarding the standard deviation σ_1, the perturbation constraint ρ, and the trade-off parameter β.

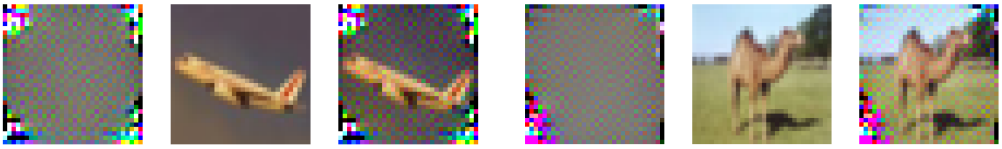
Performance Evaluation. We depict the learned watermarks in Figure 3. As we can see, the centre areas maintain the pattern of original inputs, which are helpful in OOD detection primitively. By contrast, the edge areas of the watermarks distort the original features, superimposed with the pattern that may further boost the capability of the original models in OOD detection.

Then, we demonstrate the improvement of watermarking on CIFAR-10 and CIFAR-100 datasets. For the results of the softmax scoring in Table 1, our watermarking reduces the average FPR95 by $2.99 \sim 12.84$, boosts the average AUROC by $3.09 \sim 3.69$ and the average AUPR by $1.10 \sim 1.18$. Moreover, for each of the considered test-time OOD dataset, there also exist stable improvements after watermarking, except for the comparable results regarding Places365 on CIFAR-100 dataset.

For the results of the free energy scoring in Table 2, the improvement via watermarking is also substantial, with $4.54 \sim 13.98$, $3.43 \sim 4.65$, and $1.13 \sim 1.37$ better performance regarding the average FPR95, AUROC, and AUPR. Overall, Table 1 and Table 2 not only justify the effectiveness of watermarking and also demonstrate the generality of the proposed watermarking. Further, comparing between CIFAR-10 and CIFAR-100 datasets, the improvements of watermarking on CIFAR-10 is much greater than that of CIFAR-100, aligning with the previous observations [20] that a large semantic space can exaggerate the challenge in effective OOD detection.



(a) softmax scoring on CIFAR-10 (b) softmax scoring on CIFAR-100



(c) free energy scoring on CIFAR-10 (d) free energy scoring on CIFAR-100

Figure 3: The learned watermarks (left) and the example images with (middle) and without (right) the watermarks. All the pictures are clamped between 0 and 255 for the purpose of illustration.

Table 1: Softmax scoring with/without watermarking on CIFAR benchmarks. ↓ (↑) indicates smaller (larger) values are preferred.

	FPR95 ↓	AUROC ↑	AUPR ↑
	w/ (w/o) watermark		
CIFAR-10			
iSUN	**43.60** (55.55)	**93.53** (90.14)	**98.67** (97.84)
Places365	**60.75** (62.50)	**87.85** (87.41)	**96.98** (96.94)
Texture	**42.00** (59.30)	**92.83** (88.37)	**98.43** (97.14)
SVHN	**27.25** (49.10)	**96.00** (91.69)	**99.17** (96.54)
LSUN	**40.70** (52.05)	**94.36** (91.50)	**98.86** (98.16)
average	**42.86** (55.70)	**92.91** (89.82)	**98.42** (97.32)
CIFAR-100			
iSUN	**77.85** (83.35)	**79.91** (75.28)	**95.35** (94.00)
Places365	83.25 (**82.20**)	74.28 (**74.40**)	**93.47** (93.44)
Texture	**79.10** (83.80)	**77.14** (72.83)	**94.26** (92.81)
SVHN	**82.95** (85.05)	**76.92** (70.64)	**94.72** (92.61)
LSUN	**76.75** (80.45)	**79.60** (76.25)	**95.27** (94.32)
average	**79.98** (82.97)	**77.57** (73.88)	**94.61** (93.43)

Table 2: Free energy scoring with/without watermarking on CIFAR benchmarks. ↓ (↑) indicates smaller (larger) values are preferred.

	FPR95 ↓	AUROC ↑	AUPR ↑
	w/ (w/o) watermark		
CIFAR-10			
iSUN	**16.30** (32.10)	**96.97** (92.84)	**99.39** (98.33)
Places365	**36.25** (41.45)	**91.87** (89.65)	**97.94** (97.21)
Texture	**32.60** (52.05)	**93.14** (85.43)	**98.08** (95.52)
SVHN	**16.45** (35.25)	**97.11** (90.91)	**99.39** (97.68)
LSUN	**16.85** (27.50)	**96.97** (93.98)	**99.38** (98.59)
average	**23.69** (37.67)	**95.21** (90.56)	**98.83** (97.46)
CIFAR-100			
iSUN	**75.05** (81.80)	**83.07** (79.04)	**96.15** (94.98)
Places365	**80.45** (80.50)	**77.78** (74.99)	**94.45** (93.37)
Texture	**75.15** (80.20)	**79.55** (76.00)	**94.79** (93.53)
SVHN	**82.85** (85.10)	**75.26** (74.20)	**94.18** (93.70)
LSUN	**71.85** (80.45)	**84.01** (78.29)	**96.33** (94.69)
average	**77.07** (81.61)	**79.93** (76.50)	**95.18** (94.05)

Figure 4 illustrates the scoring distributions before (a) and after (b) watermarking on CIFAR-10, where we take the free energy scoring as an example. Due to the space limit, we only consider two test-time OOD datasets, namely, Texture and SVHN. As we can see, after watermarking, the distribution gap between the ID (i.e., CIFAR-10) and the OOD (i.e., Texture and SVHN) cases is enlarged, and thus the detection capability is improved.

Ablation Study. To further interpret our learning framework, we compare the performance of the resultant watermarks regarding various setups of hyper-parameters, focusing on the standard deviation σ_1 in the Gaussian noise and the perturbation constraint ρ in the SAM, which are both critical for our proposal. As a case study, we conduct experiments regarding the softmax scoring on CIFAR-10. For detailed results about the ablation study, please refer to Appendix C.

Table 3 lists the results with various values of σ_1, ranging from 0.00 to 1.00. Note that σ_1 is the standard deviation of the Gaussian noise added to the watermark, with different values indicating various degrees of the perturbation. As we can see, a mild perturbation of the watermark (e.g., $\sigma_1 = 0.4$) can truly lead to improved results in detection, with 10.05, 1.60, and 0.34 improvements regarding the average FPR95, AUROC, and AUPR. It indicates that the learned watermark is sensitive when facing small perturbations, if the Gaussian noise is not applied during training. However, some extreme values (e.g., $\sigma_1 = 1$) may overwhelm the watermark pattern and thus be detrimental. We also observe a trade-off on various test-time OOD datasets for the watermarks learned from different σ_1. In Figure 5, test-time OOD datasets such as Places365 and LSUN-C prefer small values of σ_1, while

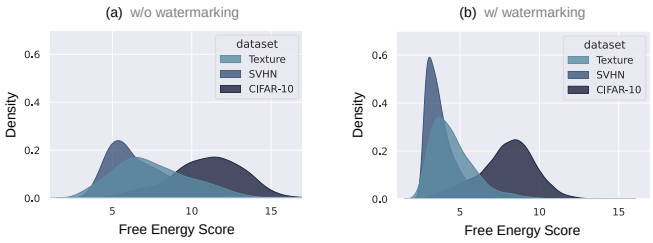


Figure 4: The scoring distributions on CIFAR-10 regarding the free energy scoring. (a) depicts the results before watermarking and (b) is the results after watermarking.

Figure 5: FPR95 on various test-time OOD datasets after watermarking with various σ_1.

Table 3: The average performance of the softmax scoring on CIFAR-10 dataset with various values of the parameter σ_1. The notion $\downarrow$ ($\uparrow$) indicates smaller (larger) values are preferred.

σ_1	FPR95 $\downarrow$	AUROC $\uparrow$	AUPR $\uparrow$
2.00	42.11	92.01	98.24
1.80	42.84	91.89	98.22
1.60	41.41	92.14	98.25
1.40	42.20	91.82	98.20
1.20	41.98	91.91	98.20
1.00	42.76	91.95	98.23
0.80	43.38	91.89	98.21
0.60	39.16	92.89	98.41
0.40	**38.66**	**93.03**	**98.45**
0.20	43.88	92.80	98.42
0.00	48.71	91.43	98.11

Table 4: The average performance of the softmax scoring on CIFAR-10 dataset with various values of the parameter ρ. The notion $\downarrow$ ($\uparrow$) indicates smaller (larger) values are preferred.

ρ	FPR95 $\downarrow$	AUROC $\uparrow$	AUPR $\uparrow$
5.00	40.11	92.48	98.34
4.50	41.30	92.39	98.32
4.00	41.21	92.39	98.33
3.50	**38.65**	**92.55**	**98.34**
3.00	41.01	92.55	98.35
2.50	39.66	92.61	98.35
2.00	38.95	92.98	98.47
1.50	43.89	92.35	98.33
1.00	40.47	93.08	98.49
0.50	44.51	92.35	98.30
0.00	49.91	91.46	98.08

iSUN and LSUN-R much prefer that of the large ones. It may indicate that the Gaussian perturbation is not optimal for the stability purpose, requiring our study in the future.

In Table 4, we summarize the experimental results given by various values of ρ for the SAM. Overall, a large value of ρ indicates that a wide range in the solution's neighbours should be smooth, and thus the stability of the result is expected to be improved. However, such a solution may consume too much capacity of the watermark, misleading the learning procedure to some unsatisfactory results. On the other side, our watermarking truly benefits from the SAM with a mild choice of the hyper-parameter. Specifically, comparing with the results without the SAM (i.e., $\rho = 0.00$), the detection capability of the watermark with a suitable ρ (i.e., $\rho = 3.5$) is largely improved, with 11.26, 1.09, and 0.26 better results regarding the average FPR95, AUROC, and AUPR.

Impact on test accuracy. Now, we study the impact of watermarking on the classification accuracy in Table 5, comparing with the results before watermarking. As we can see, watermarking has a negative impact on the test accuracy, dropping from 94.84% to 91.85% and 93.49% on CIFAR-10; and from 75.96% to 72.03% and 74.08% on CIFAR-100. Further, after watermarking, the classification accuracy with the free energy scoring is much better than that of the softmax scoring, with only 1.35% to 1.88% decrease in classification accuracy. Therefore, we suggest using the free energy scoring in watermarking as a default setup, which leads to better detection capability and largely preserves the original capability in classification.

Transferability of watermarking. Further, we explore the transferability of the watermarks learned with different scoring functions, e.g., we study the effect of the watermark learned with the softmax scoring when it is deployed with the free energy scoring. The results are summarized in Table 6. Note that, a "learn" with FE and "score" with SM indicates that the watermark is learned with the softmax scoring and tested regarding the free energy scoring. As we can see, the watermarks learned with the free energy scoring can be reused for the softmax scoring (even with better results), while the reverse typically leads a deterioration. It may indicate that our learning objectives in Eq. (11) are general and effective for both the softmax scoring and the free energy scoring. However, we do not observe any transferability between different datasets. For example, for the softmax scoring, when

Table 5: Test accuracy before/after watermarking. w/o denotes the benchmark without watermarking, SE (FE) denotes the softmax (free energy) scoring with watermarking.

dataset	w/o	SM	FE	dataset	w/o	SM	FE
CIFAR-10	**94.84%**	91.85%	93.49%	CIFAR-100	**75.96%**	72.03%	74.08%

Table 6: Transferability of watermarking across scoring functions. SM denotes softmax scoring and FE denotes free energy scoring.

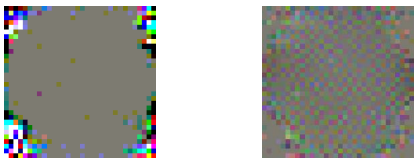
learn	score	FPR95 ↓	AUROC ↑	AUPR ↑
CIFAR-10				
SM	SM	42.86	92.91	98.42
FE	SM	**40.19**	**94.83**	**98.99**
FE	FE	**23.69**	**95.21**	**98.83**
SM	FE	28.22	94.70	98.81
CIFAR-100				
SM	SM	79.98	**77.57**	**94.61**
FE	SM	**77.07**	76.27	94.02
FE	FE	**77.07**	**79.93**	**95.18**
SM	FE	78.48	79.24	94.56

Table 7: The average performance of the free energy scoring on CIFAR-10 with masking. ↓ (↑) indicates smaller (larger) values are preferred.

	FPR95 ↓	AUROC ↑	AUPR ↑
w/ watermark	**23.69**	**95.21**	**98.83**
w/o watermark	37.67	90.56	97.46
χ_1			
0.10	30.46	94.41	98.68
1.00	37.72	93.00	98.52
10.0	37.66	91.87	97.94
χ_2			
0.10	42.06	89.88	97.38
1.00	51.28	87.54	96.86
10.0	36.72	93.08	98.39

the watermark learned on CIFAR-10 is adopted for CIFAR-100, there is a drop of performance with 51.93 (from 42.86 to 94.79), 35.29 (from 92.91 to 57.62) and 12.00 (98.42 to 86.42) regarding the average FPR95, AUROC, and AUPR.

Different areas in the watermark. After watermarking, the edge area of the image is overwhelmed by the watermark's pattern, while the centre part is not much affected. However, it does not mean that the centre area is not important. Instead, under the premise of maintaining the original features, the centre area also encodes useful information in OOD detection. Table 7 is a verification of this conclusion on CIFAR-10 dataset with the free energy scoring, where we employ various masks in only preserving the watermark's features with their absolute values larger (smaller) than a threshold χ_1 (χ_2). As we can see, even if only a small portion of the watermark is masked (e.g., $\chi_1 = 0.10$ or $\chi_2 = 10.0$), there is a large drop in performance, even lower than the results without any watermarking. It indicates that both areas of the watermark contribute, and the overall watermark works as a whole for effective OOD detection.



(a) (b)

Figure 6: An illustration of the masked watermarks for the free energy scoring on CIFAR-10, with (a) $\chi_1 = 1$ and (b) $\chi_2 = 1$.

7 Conclusion

This paper demonstrates that the model's inherent capability in OOD detection can be largely improved with only data-level manipulation, where we propose a general and effective methodology named watermarking. Overall, we learn a discriminative pattern that could be superimposed onto original inputs, such that the OOD scores become much separable between ID and OOD situations. Our results indicate a promising direction in OOD detection that warrants our exploration in the future. Further investigation should focus on the stability in optimization and general learning objectives regarding advanced OOD detection methods.

References

[1] J. Bitterwolf, A. Meinke, and M. Hein. Certifiably adversarially robust detection of out-of-distribution data. In *NeurIPS*, 2020.

[2] S. Bulusu, B. Kailkhura, B. Li, P. Varshney, and D. Song. Anomalous instance detection in deep learning: A survey. Technical report, Lawrence Livermore National Lab., 2020.

[3] X. Chen, X. Lan, F. Sun, and N. Zheng. A boundary based out-of-distribution classifier for generalized zero-shot learning. In *ECCV*, 2020.

[4] S. Choi and S. Chung. Novelty detection via blurring. In *ICLR*, 2020.

[5] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

[6] J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *ICML*, 2006.

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[8] T. DeVries and G. W. Taylor. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865*, 2018.

[9] G. F. Elsayed, I. Goodfellow, and J. Sohl-Dickstein. Adversarial reprogramming of neural networks. In *International Conference on Learning Representations, ICLR*, 2019.

[10] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021.

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[12] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[13] K. Hambardzumyan, H. Khachatrian, and J. May. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*, 2021.

[14] M. Hein, M. Andriushchenko, and J. Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, 2019.

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

[16] D. Hendrycks, M. Mazeika, and T. G. Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019.

[17] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020.

[18] H. Huang, Z. Li, L. Wang, S. Chen, B. Dong, and X. Zhou. Feature space singularity for out-of-distribution detection. *arXiv preprint arXiv:2011.14654*, 2020.

[19] R. Huang, A. Geng, and Y. Li. On the importance of gradients for detecting distributional shifts in the wild. In *NeurIPS*, 2021.

[20] R. Huang and Y. Li. MOS: towards scaling out-of-distribution detection for large semantic space. In *CVPR*, 2021.

[21] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017.

[22] P. Kirichenko, P. Izmailov, and A. G. Wilson. Why normalizing flows fail to detect out-of-distribution data. In *NeurIPS*, 2020.

[23] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Technical Report TR-2009, University of Toronto*, 2009.

[24] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[25] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.

[26] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu. Invisible backdoor attack with sample-specific triggers. In *ICCV*, 2021.

[27] Y. Li, B. Wu, Y. Jiang, Z. Li, and S.-T. Xia. Backdoor learning: A survey. *arXiv preprint arXiv:2007.08745*, 2020.

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

[29] W. Liu, X. Wang, J. D. Owens, and Y. Li. Energy-based out-of-distribution detection. In *NeurIPS*, 2020.

[30] Y. Liu, X. Ma, J. Bailey, and F. Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *ECCV*, 2020.

[31] T. P. Mann. Numerically stable hidden markov model implementation. *An HMM Scaling Tutorial*, pages 1–8, 2006.

[32] P. Morteza and Y. Li. Provable guarantees for understanding out-of-distribution detection. In *AAAI*, 2022.

[33] E. T. Nalisnick, A. Matsukawa, Y. W. Teh, D. Görür, and B. Lakshminarayanan. Do deep generative models know what they don't know? In *ICLR*, 2019.

[34] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[35] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

[36] J. Ren, S. Fort, J. Liu, A. G. Roy, S. Padhy, and B. Lakshminarayanan. A simple fix to mahalanobis distance for improving near-ood detection. *arXiv preprint arXiv:2106.09022*, 2021.

[37] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NeurIPS*, 2019.

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

[39] A. Saha, A. Subramanya, and H. Pirsiavash. Hidden trigger backdoor attacks. In *AAAI*, 2020.

[40] C. S. Sastry and S. Oore. Detecting out-of-distribution examples with gram matrices. In *ICML*, 2020.

[41] J. Serrà, D. Álvarez, V. Gómez, O. Slizovskaia, J. F. Núñez, and J. Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *ICLR*, 2020.

[42] Y. Sun, C. Guo, and Y. Li. React: Out-of-distribution detection with rectified activations. In *NeurIPS*, 2021.

[43] S. Thulasidasan, G. Chennupati, J. A. Bilmes, T. Bhattacharya, and S. Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *NeurIPS*, 2019.

[44] Y. Tsai, P. Chen, and T. Ho. Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources. In *ICML*, 2020.

[45] S. Vernekar, A. Gaurav, V. Abdelzad, T. Denouden, R. Salay, and K. Czarnecki. Out-of-distribution detection in classifiers via generation. *arXiv preprint arXiv:1910.04241*, 2019.

[46] H. Wang, W. Liu, A. Bocchieri, and Y. Li. Can multi-label classification networks know what they don't know? In *NeurIPS*, 2021.

[47] Q. Wang, F. Liu, B. Han, T. Liu, C. Gong, G. Niu, M. Zhou, and M. Sugiyama. Probabilistic margins for instance reweighting in adversarial training. In *NeurIPS*, 2021.

[48] Y. Wang, B. Li, T. Che, K. Zhou, Z. Liu, and D. Li. Energy-based open-world uncertainty modeling for confidence calibration. In *ICCV*, 2021.

[49] Z. Xiao, Q. Yan, and Y. Amit. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. In *NeurIPS*, 2020.

[50] P. Xu, K. A. Ehinger, Y. Zhang, A. Finkelstein, S. R. Kulkarni, and J. Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *arXiv preprint arXiv:1504.06755*, 2015.

[51] C. H. Yang, Y. Tsai, and P. Chen. Voice2series: Reprogramming acoustic models for time series classification. In *Proceedings of the International Conference on Machine Learning, ICML*, 2021.

[52] J. Yang, K. Zhou, Y. Li, and Z. Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.

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

[54] A. Zaeemzadeh, N. Bisagno, Z. Sambugaro, N. Conci, N. Rahnavard, and M. Shah. Out-of-distribution detection using union of 1-dimensional subspaces. In *CVPR*, 2021.

[55] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016.

[56] L. H. Zhang, M. Goldstein, and R. Ranganath. Understanding failures in out-of-distribution detection with deep generative models. In *ICML*, 2021.

[57] B. Zhou, À. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018.

Checklist

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 work? [Yes]
   (c) Did you discuss any 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? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
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 (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (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]
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? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We use only standard datasets
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We use only standard datasets
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]