# Simple and Effective Out-of-Distribution Detection via Cosine-based Softmax Loss

SoonCheol Noh DongEon Jeong Jee-Hyong Lee*
Sungkyunkwan University
Suwon, Korea
{yhnuhb27, wjdehddjs, john}@skku.edu

## Abstract

*Deep learning models need to detect out-of-distribution (OOD) data in the inference stage because they are trained to estimate the train distribution and infer the data sampled from the distribution. Many methods have been proposed, but they have some limitations, such as requiring additional data, input processing, or high computational cost. Moreover, most methods have hyperparameters to be set by users, which have a significant impact on the detection rate. We propose a simple and effective OOD detection method by combining the feature norm and the Mahalanobis distance obtained from classification models trained with the cosine-based softmax loss. Our method is practical because it does not use additional data for training, is about three times faster when inferencing than the methods using the input processing, and is easy to apply because it does not have any hyperparameters for OOD detection. We confirm that our method is superior to or at least comparable to state-of-the-art OOD detection methods through the experiments.*

## 1. Introduction

Deep learning models are generally designed to target the closed world. Models are trained based on the assumption that the inputs of the inference stage will be sampled from the same distribution as the train distribution. This means that the model may not make correct predictions if inputs are sampled from other distributions. The model must be able to distinguish whether the input is from the train distribution or not. This problem has been established as the task of out-of-distribution (OOD) detection.

Some researchers tried to detect OOD samples using confidence scores from classification models. Hendrycks and Gimpel simply used the confidence scores, but this approach was not very effective because of the overconfidence of deep learning models [9]. Hendrycks, Mazeika, and Dietterich (2019) and Papadopoulos *et al*. tried to lower the confidence of OOD samples using auxiliary large OOD datasets [10, 21]. However, collecting a massive OOD dataset is highly expensive, and training classification models with such a dataset takes a long time. Liang, Li and Srikant (2018) simply adopted the input processing technique, which added a small perturbation to the input samples to increase the confidence of in-distirubtion (InD) samples, but they assumed OOD samples were given in advance, which is not practical [16].

Some researchers defined measures to detect OOD samples based on input features learned by classification models. Lee *et al*. proposed a measure based on Mahalanobis distance, but they also assumed that OOD samples were given in advance as Liang *et al*. did [15, 16]. Sastry and Oore proposed a measure based on various correlations between features, but their approach was very time-consuming in evaluating the OOD measure [23]. Hsu *et al*. tried to obtain the OOD probability by modifying the network structure, which caused the degradation of classification performances [12].

Additional popular technique is the input processing, which was adopted by most approaches [16, 15, 23]. The input processing is to generate $x'$ by adding a small perturbation to input data $x$ to make the boundary of in-distribution compact. It may be helpful to increase the detection accuracy under certain conditions. However, generating the perturbation takes almost twice of the inference time, so the total inference becomes three times longer. The OOD detection performance is very susceptible to the size of the perturbation, so it should be carefully determined.

Some OOD detection approaches had various hyperparameters, including the perturbation size of input processing. The hyperparameters were usually tuned by using additional datasets to maximize the detection performance. Liang *et al*. [16] and Lee *et al*. [15] tuned them with given OOD samples, and Hsu *et al*. [12] and Sastry and Oore [23] used the validation samples. However, it is hard to find out

*Corresponding Author

hyperparameter values generally applicable to any distributions of OOD samples. When the values are found in a specific dataset, the model is easy to be biased to the dataset.

This paper proposes a simple and effective approach that can detect OOD samples without hyperparameters, additional datasets, and input processing. We detect OOD samples by combining norm and Mahalanobis distances of features. We find that the models trained with the cosine-based softmax loss are more advantageous than the ones trained with the regular softmax loss for OOD detection. We mathematically conjecture that the feature norm from the models with the cosine-based softmax loss can be a measure for the probability of in-distribution, and confirm its effectiveness experimentally. We also develop a robust Mahalanobis distance-based measure for the model with the cosine-based softmax loss. We propose an OOD score function by combining the feature norm and the Mahalanobis distance without any hyperparmaeters.

Our method do not need the input processing because it shows high accuracy in OOD detection without it. So, the additional computation for OOD detection by our method is very small. Feature norms and the Mahalanobis distances can be obtained with tiny additional calculations from classification models. Since the cosine-based softmax loss is generally applicable to any deep models, our OOD detection approach can be easily extended to various structures without a classification accuracy loss. Our method is also practical because it does not use any additional datasets. We confirm the OOD detection performance on various image datasets through experiments. Our method is compared with others using additional datasets, performing the input processing, or tuning hyperparameters to maximize performance. The experiment shows that our approach is superior to or at least comparable to state-of-the-art OOD detection methods.

## 2. Related Work

Previous OOD detection approaches were based on various models, such as classification models [9, 16, 15, 10, 21, 17, 12, 23], generative models [22, 19, 25, 4, 26, 3] and Bayesian models [6, 2, 18].

OOD detection is usually required as a pre-processing of the classification task, which is one of the crucial tasks of deep learning. Due to its effectiveness, many researchers tried simultaneously performing OOD detection and classification in one model. In this section, we cover OOD detection-related approaches based on classification models.

### 2.1. Training with Large OOD Datasets

Hendrycks *et al*. proposed a method to lower the maximum softmax probability (MSP) for OOD samples from classification models [10]. They improved OOD detection performance by adding a loss term which was based on large OOD datasets. The added loss term was designed so that softmax values for OOD samples be closed to a uniform distribution. Papadopoulos *et al*. further improved OOD detection performance by adding another loss term [21]. The loss term prevented classification models from overconfidence. Liu *et al*. proposed an energy-based measure for models trained with large OOD datasets [17].

These methods had the disadvantage of collecting extensive additional datasets to train classification models. The collected OOD dataset was used, assuming it represented all unknown OODs. However, it is virtually impossible to collect data that can represent all OOD samples in real-world situations. In addition, training a model with a large amount of OOD data takes much longer than regular training.

### 2.2. Tuning Hyperparameters Using Known OOD Samples

Some OOD detection approaches had hyperparameters that need to be tuned. Liang *et al*. [16] and Lee *et al*. [15] used specific OOD samples to tune hyperparamters under the assumption that distribution of OOD samples were given in advance. Liang *et al*. [16] tuned the temperature scaling factor and a perturbation size of input processing. Lee *et al*. [15] tuned weights for the feature ensemble and a perturbation size of input processing. However, these approaches are not feasible because it is difficult to know from which distribution OOD samples will occur in a real-world situation.

### 2.3. Tuning Hyperparameters Using In-distribution Samples

Recently, some OOD detection approaches without using OOD samples were proposed because it was unrealistic to use OOD samples. Hsu *et al*. modified the classifying layer to obtain the probability that an input was in-distribution [12]. However, the modified classifying layer made the classification accuracy unstable. They also proposed a strategy that the perturbation size of the input processing was set without OOD samaples, but it was not valid to all datasets. Sastry and Oore [23] proposed measures for OOD detection based on the Gram matrices of features. They stored the maximum and minimum values of the elements of the Gram matrix for each class, layer, power, and index of the matrix. They detected OOD samples using the stored values. However, it required heavy computation and a longer time than other approaches. Those approaches used additional InD validation dataset for tuning hyperparameters: Hsu *et al*. [12] tuned the perturbation size of input processing, and Sastry and Oore [23] tuned normalization factors for each layer.

## 2.4. Using Input Processing

The input processing is an operation that generates $x'$ by adding a small perturbation to an input $x$. Many previous approaches applied the input processing because this process effectively improved OOD detection performance. However, the perturbation size, epsilon, needed to be tuned appropriately. The epsilon has a significant impact on performance, so it should be chosen carefully.

Liang *et al.* [16] and Lee *et al.* [15] chose the epsilon values using given specific OOD samples. The chosen epsilon may not be valid to other OOD samples because it was tuned with a distribution of given OOD samples. Hsu *et al.* [12] proposed a method to set the epsilon value without OOD samples. They tuned the epsilon using the InD validation dataset, but their method did not generally work for all cases. The detection performance decreased in some datasets.

Another disadvantage is that the input processing requires a lot of computation. We need one forward loop and one backward loop to add a perturbation to an input. We need one more forward loop to determine whether the perturbed input is OOD. Since the time taken for the forward loop and the backward loop is almost the same, the total inference time increases by about three times compared to the inference time without the input processing.

## 2.5. Using Self-supervised Learning

Recently, a self-supervised learning technique that can learn a deep learning model without a label is attracting attention. In many studies in the field of OOD detection, self-supervised learning techniques are being tried to improve performance or reduce dependence on data labels. Some approaches build new data by applying transforms such as rotation to the image, and train models by giving the transform applied to the image as a label [7, 11, 27, 1]. Other approaches use contrastive learning techniques to train the model [27, 24]. However, in most cases, the OOD detection performance is not as high as when learning using labels. In addition, since the model is trained only for the purpose of OOD detection or representation learning, an additional method is needed to perform classification.

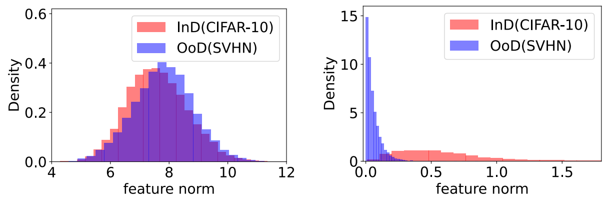
## 3. Proposed Method

A deep learning-based classification model can be decomposed into a feature extractor $f(\cdot)$ and a classifier $g(\cdot)$ which is usually a fully connected layer. When training with the standard softmax loss, a logit is defined as:

$$\text{logit}(x,c) = f(x)\cdot w_c + b_c, \tag{1}$$

and with the cosine-based softmax loss, it is defined as:

$$\text{logit}(x,c) = \frac{f(x)\cdot w_c}{|f(x)|\cdot|w_c|}, \tag{2}$$



(a) Standard softmax loss (b) Cosine-based softmax loss

Figure 1: The feature norm distributions of InD (red) and OOD (blue) data. (a) is from DenseNet3 trained with the standard softmax loss and (b) is from DenseNet3 trained with the cosine-based softmax loss.

where $w_c$ is the connection weights of $g(\cdot)$ for a class $c$. The cosine-based softmax loss is widly used in face recognition field [28, 5, 29].

We introduce two useful measures for OOD detection that can be simply extracted from models trained with cosine-based softmax loss: a feature norm-based measure and a Mahalanobis distance-based measure. We then propose an OOD score by combining these two measures. Let us describe each measure and the score function in detail.

### 3.1. Cosine-based Softmax Loss and Feature Norm

When we train a classification model with the closed-world assumption, the logit value for a class $c$ is considered as being proportional to $P(y=c|x)$. However, in the open world, the logit value should be interpreted as being proportional to $P(y=c|x,D_{in})$ where $D_{in}$ is the event that $x$ is in the train distribution. The probability can be decomposed using $P(y=c,D_{in}|x)$ and $P(D_{in}|x)$ as follows:

$$\text{logit}(x,c) \propto P(y=c|x,D_{in}) = \frac{P(y=c,D_{in}|x)}{P(D_{in}|x)}. \tag{3}$$

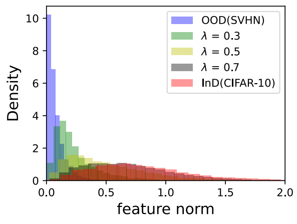
If we use the cosine-based softmax loss, Equations 2 and 3 can be combined as follows:

$$\frac{f(x)\cdot w_c}{|f(x)|\cdot|w_c|} \propto \frac{P(y=c,D_{in}|x)}{P(D_{in}|x)}. \tag{4}$$

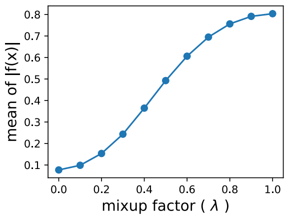
The numerator of the right-hand side includes a class-related term, $c$, and the denominator, $P(D_{in}|x)$, does not. So, we reformulate the left-hand side so that only the numerator has class-related terms as follows:

$$\frac{f(x)\cdot w_c/|w_c|}{|f(x)|} \propto \frac{P(y=c,D_{in}|x)}{P(D_{in}|x)}. \tag{5}$$

Since $w_c$ is the connection weight to $g(\cdot)$, we can say it is class-related, and since $f(x)$ is the feature representation of $x$, we can say it is classifier-independent. Based on this, we rigorously infer that both denominators can be proportional



(a) Distribution of $|f(x)|$



(b) Mean of $|f(x)|$

Figure 2: The distribution of feature norm according to the mixup factor (a), and the mean value of feature norm according to the mixup factor. It can be observed that as OOD samples are more mixed into a sample, the feature norm tends to decrease.

to each, i.e., $|f(x)| \propto P(D_{in}|x)$, because the two terms in Equation 5 are proportional, and their denominators and numerators share class-related characteristics.

To confirm this, we perform some OOD detection experiments. We try to detect SVHN using $|f(x)|$ from DenseNet3 model trained for CIFAR-10 with the cosine-based softmax loss. We compare this to the model using the standard softmax loss as shown in Figure 1. The feature norm distributions of OOD and InD by the standard loss model overlap much, but those of the cosine loss model have a very small overlap between SVHN and CIFAR-10, which says that the cosine loss model can be better to discriminate ODD than the standard model.

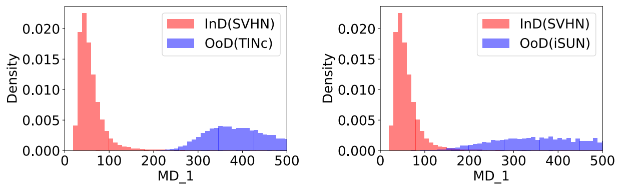
To verify that the length of feature norm decrease as the samples get closer to OOD, we create mixup samples by mixing SVHN and CIFAR-10 images as shown in Equation 6 and observed the feature norm.

$$\text{TestImg} = \lambda \cdot \text{InD} + (1 - \lambda) \cdot \text{OOD} \tag{6}$$

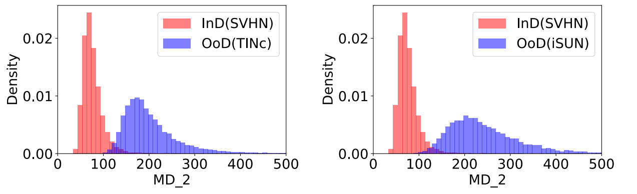
As the mixup factor, $\lambda$ decreases, that is, as the samples get closer to OOD, the feature norm also decreases as shown in Figure 2. The feature norm of an image from the cosine loss model tends to increase as the image get closer to InD samples and decrease closer to OOD samples, which supports our conclusion that $|f(x)|$ from a cosine loss model is proportional to $P(D_{in}|x)$. Also, it supports that the feature norm can be a measure for OOD detection. Additional observations, including CIFAR-100 and SVHN, are presented in the appendix.

## 3.2. Mahalanobis Distance in Models Trained with Cosine-based Softmax Loss

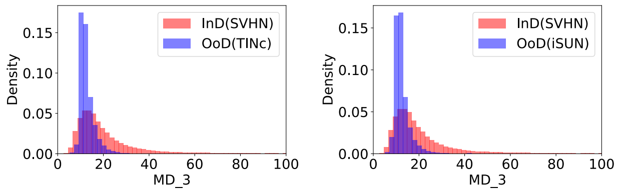
We may assume that features of InD samples obtained from internal layers of classification models follow the Gaussian distribution. We adopt an OOD score defined by



(a) 1st Block (SVHN vs. TINc) (b) 1st Block (SVHN vs. iSUN)



(c) 2nd Block (SVHN vs. TINc) (d) 2nd Block (SVHN vs. iSUN)



(e) 3rd Block (SVHN vs. TINc) (f) 3rd Block (SVHN vs. iSUN)

Figure 3: Distributions of $MD$ scores from each block of DenseNet3 trained with SVHN. The OODs datasets (blue) is TinyImageNet-crop (left) and iSUN (right).

Lee *et al.* [15] based on the Mahalanobis distance:

$$\begin{aligned} &MD_B(x) \\ &= -\max_c - (f_B(x) - \mu_{c,B})^\top \Sigma_B^{-1} (f_B(x) - \mu_{c,B}), \end{aligned} \tag{7}$$

where $\mu_{c,B}$ and $\Sigma_B$ are the mean and the covariance of features from a block $B$. This score is very discriminative for OOD detection as shown in Figure 3. Since the distributions of $MD$ scores of InD and OOD samples have small overlaps, we can detect OOD samples using the scores. However, $MD$ scores from the penultimate layer overlap if the model is trained with cosine-based softmax loss, as shown in Figure 2(e)-(f).

The reason why $MD$ scores from penultimate layer is not discriminative is that the feature from the penultimate layer do not follow a Gaussian distribution. In the model trained with cosine-based softmax loss, the softmax probability with $\sigma_c$, the softmax value for a class $c$, is calculated as follows:

$$\sigma_c\left(g(f(x))\right) = \frac{\exp\left(w_c \cdot f(x)/(|w_c| \cdot |f(x)|)\right)}{\sum_{c'} \exp\left(w_{c'} \cdot f(x)/(|w_{c'}| \cdot |f(x)|)\right)}. \tag{8}$$

| Approach | Auxiliary Dataset | # of Hyperp. | Input Processing |
|---|---|---|---|
| OE [10] | Large OOD dataset | 0 | × |
| ODIN [16] | OOD samples | 2 | ✓ |
| MAHA [15] | OOD samples | 4 | ✓ |
| G-ODIN [12] | InD samples | 1 | ✓ |
| GRAM [23] | InD samples | 100 | × |
| COD (Proposed) | None | 0 | × |

Table 1: OOD detection approach comparison. The number of hyperparameters is counted based on DenseNet3 with 100 layers.

The classification model is trained so that the softmax probability reflects $P(y = c|f(x))$. If the feature distribution from the penultimate layer is assumed to be the multivariate Gaussian distribution for each class, and the class prior is $P(y = c) = \frac{\beta_c}{\sum_{c'} \beta_{c'}}$ where $\beta_c$ is unnormalized prior for class $c$, the probability of class $c$ with $f(x)$ is as follows:

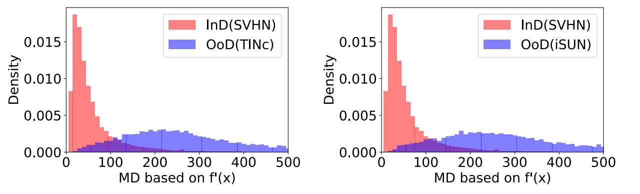
$$\begin{aligned} &P(y = c|f(x)) \\ &= \frac{P(y = c)P(f(x)|y = c)}{\sum\limits_{c'} P(y = c')P(f(x)|y = c')} \\ &= \frac{\exp\left(\mu_c^T \Sigma^{-1} f(x) - \frac{1}{2}\mu_c^T \Sigma^{-1}\mu_c + log\beta_c\right)}{\sum\limits_{c'} \exp\left(\mu_{c'}^T \Sigma^{-1} f(x) - \frac{1}{2}\mu_{c'}^T \Sigma^{-1}\mu_{c'} + log\beta_{c'}\right)}. \end{aligned} \tag{9}$$

That is, if the feature from the penultimate layer follows a Gaussian distribution, $P(y = c|f(x))$ is based on linear expressions. Since a nonlinear expression cannot be transformed in a linear expression, the softmax probability from the cosine loss model (Equation 8) cannot be converted to a similar form of $P(y = c|f(x))$ (Equation 9), which means that the features from the penultimate layer is not expected to follow a Gaussian distribution.

If Equation 8 is rewritten using a normalized feature $f'(x) = f(x)/|f(x)|$, it is as follows:

$$\sigma_c(g(f(x))) = \frac{\exp\left(w_c^\top f'(x)/(|w_c^\top|)\right)}{\sum_{c'} \exp\left(w_{c'}^\top f'(x)/(|w_{c'}^\top|)\right)}. \tag{10}$$

Equation 10 is based on linear expressions of $f'(x)$, from which we conjecture that $f'(x)$ may follow a Gaussian distribution. Since $MD$ scores are useful under the assumption that features follow a Gaussian distribution, we redefine the feature for $MD$ from the penultimate layer as $f'(x) = f(x)/|f(x)|$ instead of $f(x)$. We re-depict the distribution of $MD$ in Figures 2-(e) and (f) using $f'(x)$ as shown in Figure 4. The overlap of the distributions is significantly reduced compared to $MD$ based on $f(x)$, which means that $MD$ based on $f'(x)$ may be better than $f(x)$ for OOD detection.



(a) 3rd Block (SVHN vs TINc) (b) 3rd Block (SVHN vs iSUN)

Figure 4: Distributions of $MD$ scores based on $f'(x)$. The configuration is the same as Figure 3.

We evaluate $MD$ scores for the penultimate layer with the redefined feature, $f'(x) = f(x)/|f(x)|$ and for other layers with $f_B(x)$. Finally, we combine $MD$ scores from each block as Equation 11 by product:

$$MD(x) = \prod_B (MD_B(x)). \tag{11}$$

### 3.3. Proposed OOD Score Function

We define a cosine-based OOD score function, $COD(x)$, with the feature norm and the Mahalanobis distance-based measure, $MD$. Since the feature norm tends to appear large for InD samples and $MD$ tends to be large for OOD samples, we define $COD(x)$ by dividing the feature norm by $MD$ as follows:

$$COD(x) = \frac{|f(x)|}{MD(x)}. \tag{12}$$

$COD(x)$ does not have any hyperparameter to be tuned. Also, no additional dataset is required to use $COD(x)$, and it shows a high discriminative performance without the input processing. $COD(x)$ can be calculated with a very small computation in a forward loop for classification. We compare the previous approaches in the viewpoint of auxiliary datasets, the number of hyperparameters and the input processing in Table 1. Most of the previous methods used auxiliary datasets, and had several to one hundred hyperparameters, and needed the input processing to increase the performance, while our method does not need any of them.

## 4. Experiment

This section shows the effectiveness of our approach through experiments with various image datasets.

### 4.1. Experiment setting

**Datasets**: We use CIFAR-10/100 [14], and SVHN [20] images for InD datasets. We use the training dataset for classification and use the test datasets to evaluate OOD detection. We use TinyImageNet-crop (TINc), TinyImageNet-resize (TINr), LSUN-crop (LSUNc), LSUN-resize (LSUNr), and iSUN as OOD datasets. These datasets have been used in many studies related to OOD detection. Additionally, when CIFAR-10/100 are in-distribution, SVHN test dataset is also used as an OOD dataset, and when SVHN image is in-distribution, CIFAR-10/100 test datasets are also used as OOD datasets.

**Baselines**: We compare our approach to previous OOD detection methods: ODIN [16], MAHA [15], G-ODIN [12], and GRAM [23].

**Networks and Training Details**: We train ResNet34 [8] and DenseNet3 [13] models with the same setting as the baseline approaches. We train ResNet34 for 200 epochs with a batch size of 128 and a weight decay of 0.0005. We train DenseNet3 with 100 layers for 300 epochs with a batch size of 64 and a weight decay of 0.0001. In both training, the optimizer is SGD with a momentum of 0.9, and the learning rate starts with 0.1 and decays by a factor 0.1 at 50% and 75% of the training progress.

When we train models with the cosine-based softmax loss, logit values are between -1 and 1. For expanding the range of logits, we adopt the temperature scaling with a factor of $T$ as in Equation 13. The factor of $T$ needs to be set to 0.1 for SVHN, CIFAR-10 and 0.05 for CIFAR-100 so that the classification performance is maximized. The logit scaled by $T$ is defined as:

$$\text{logit}_T(x) = \text{logit}(x)/T. \tag{13}$$

**Evaluation Metrics**: We evaluate the detection performance using the two metrics that are most widely used for OOD detection. The first metric is the area under the receiver operating characteristic curve (AUROC). It is used to evaluate the average performance considering various thresholds. The second is TNR at TPR $p\%$. We evaluate with $p = 95$ as the metric used in the baseline approaches.

### 4.2. Results

Table 2 shows an overall comparison with DenseNet3. Our approach achieves superior performance in all the datasets. Compared to ODIN, our approach has higher performance in all the cases. Compared to G-ODIN, our approach has higher performance except one cases. Compared to MAHA, our approach has higher performance except some cases.

In some cases, GRAM shows a higher performance than the proposed method, but GRAM is very unstable depending on OOD datasets. For example, GRAM shows a particularly poor performance on LSUNc. It shows a TNR@95 score of 88.4 for CIFAR-10 and 65.6 for CIFAR-100. Our approach is more even performances across all OOD datasets, and achieves the highest average performances. Also, GRAM requires significantly heavy computation. When comparing the wall clock time in the same environment (GTX 2080 Ti GPU with Xeon Silver 4021 CPU * 2), the GRAM takes 0.35 seconds to calculate OOD score for one image, but our approach takes only about 0.018 seconds. Our approach is 19 times faster than GRAM because GRAM evaluates a lot of measures in the forward loop while our approach needs very small additional computation.

The head of Table 2 shows what each approach needs to detect OOD samples. ODIN and MAHA use OOD samples and InD validation samples, and perform the input processing. G-ODIN does not use OOD samples but uses InD validation samples and performs the input processing. GRAM uses InD validation samples to tune hyperparameters. Our approach needs nothing except a training dataset. Our approach is the fastest method because the input processing is not required. Our method is superior to or at least comparable to state-of-the-art OOD detection methods even though it does not require any additional datasets. Our model also shows better performance with ResNet34. The results are presented in the appendix.

## 5. Discussion

### 5.1. Layer Selection for $MD$

Our approach, $COD$, uses $MD$ values from all the blocks of the model. However, it can be more effective if we can choose some layers which are helpful for OOD detection. Useful layers for OOD detection may depend on the model structure, the training dataset, and the OOD dataset. However, OOD samples are not available in advance.

We will discuss how to select useful layers with synthetic OOD samples. We may generate synthetic OOD samples from the training dataset. For example, we may use some training samples rotated by 90 degrees as synthetic OOD samples. Then, we evaluate AUROC scores for each layer, and choose layers of which AUROC scores are greater than a threshold such as 60% of AUROC. In the case of CIFAR10, CIFAR100, and SVHN, it is confirmed that the AUROC score of each $MD$ was over 60%. Therefore, it can be a reasonable threshold to choose layers.

We apply the layer selection to a more realistic dataset, ImageNet-30. In this case, $MD$ from the last layer, $MD$

| OOD | AUROC | | | | | TNR@95 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Use a training dataset | | | | | Use a training dataset | | | | |
| | Use InD validation samples | | | | | Use InD validation samples | | | | |
| | Perform the input processing | | | | | Perform the input processing | | | | |
| | Use OOD samples | | | | (Ours) | Use OOD samples | | | | (Ours) |
| | ODIN | MAHA | G-ODIN | GRAM | COD | ODIN | MAHA | G-ODIN | GRAM | COD |
| InD: CIFAR-10 | | | | | | | | | | |
| SVHN | 95.5 | 98.1 | 98.8 | 99.1 | **99.5** | 86.2 | 90.8 | 94.0 | 96.1 | **98.0** |
| TINc | 97.6 | 95.3 | 98.7 | 99.3 | **99.4** | 87.0 | 84.2 | 93.4 | 96.7 | **97.9** |
| TINr | 98.5 | 98.8 | 99.1 | **99.7** | 99.5 | 92.4 | 95.0 | 95.8 | **98.8** | 97.8 |
| LSUNc | 93.6 | 80.2 | 98.3 | 97.5 | **99.9** | 70.6 | 48.2 | 91.5 | 88.4 | **99.6** |
| LSUNr | 99.2 | 99.3 | 99.4 | **99.9** | 99.6 | 96.2 | 97.2 | 97.6 | **99.5** | 98.9 |
| iSUN | 98.7 | 98.9 | 99.4 | **99.8** | 99.6 | 93.2 | 95.3 | 97.5 | **99.0** | 98.7 |
| MEAN | 97.2 | 95.1 | 99.0 | 99.2 | **99.6** | 87.6 | 85.1 | 95.0 | 96.4 | **98.5** |
| InD: CIFAR-100 | | | | | | | | | | |
| SVHN | 93.8 | 97.2 | 95.9 | **97.3** | 97.0 | 70.6 | 82.5 | 77.0 | **89.3** | 82.4 |
| TINc | 88.3 | 88.8 | 97.6 | 97.7 | **97.9** | 51.0 | 60.1 | 87.8 | 89.0 | **89.3** |
| TINr | 85.2 | 97.4 | 98.6 | **99.0** | 98.6 | 42.6 | 86.6 | 93.3 | **95.7** | 93.1 |
| LSUNc | 91.4 | 81.7 | 95.3 | 91.4 | **98.9** | 57.8 | 42.1 | 75.0 | 65.5 | **95.1** |
| LSUNr | 85.5 | 98.0 | 98.7 | **99.3** | 98.6 | 41.2 | 91.4 | 93.8 | **97.2** | 93.3 |
| iSUN | 84.5 | 97.4 | 98.4 | **99.0** | 98.5 | 37.4 | 87.0 | 92.5 | **95.9** | 92.5 |
| MEAN | 88.1 | 93.4 | 97.4 | 97.3 | **98.3** | 50.1 | 75.0 | 86.6 | 88.8 | **91.0** |
| InD: SVHN | | | | | | | | | | |
| CIFAR-10 | 91.4 | 98.9 | - | 95.5 | **99.0** | 71.7 | **96.8** | - | 80.4 | 96.1 |
| CIFAR-100 | - | - | - | - | **98.9** | - | - | - | - | **95.8** |
| TINc | - | - | - | - | **99.8** | - | - | - | - | **99.7** |
| TINr | 95.1 | **99.9** | - | 99.7 | **99.8** | 84.1 | **99.9** | - | 99.1 | 99.5 |
| LSUNc | - | - | - | - | **99.8** | - | - | - | - | **99.4** |
| LSUNr | 94.5 | **99.9** | - | 99.8 | 99.8 | 81.1 | 99.9 | - | 99.5 | 99.4 |
| iSUN | 94.7 | **99.9** | - | 99.8 | 99.8 | 82.2 | **99.9** | - | 99.4 | 99.5 |
| MEAN | - | - | - | - | **99.6** | - | - | - | - | **98.5** |

Table 2: OOD detection performance (%) with a DenseNet3 classification model. Our performance is averaged over 5 runs. ODIN, MAHA and GRAM are cited from the work by Sastry and Oore [23] and G-ODIN is cited from the work by Hsu *et al*. [12]. In the case of SVHN, all baselines did not present their performances. Dash symbols stand for not-available. The mean rows are the averages of the performances over all OOD datasets. For each case, the best results are in bold.

| ResNet18 InD: ImageNet-30 | Test Acc | OOD Dataset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | DTD | CUB-200 | Caltech | Dogs | Places-365 | Food-101 | Pet | Flowers |
| CSI | 97.0 | 93.7 | 93.4 | 91.9 | 97.7 | 92.5 | 87.0 | 96.9 | 96.0 |
| COD | 97.2 | 94.0 | 92.8 | 91.8 | 95.9 | 93.7 | 80.4 | 96.2 | 95.4 |

Table 3: OOD detection performance comparison with CSI and COD with layer selection.

obtained from the re-defined feature of the last layer, shows an AUROC score of over $80\%$, and the remaining blocks show AUROC scores of about $50\%$. Based on this, we choose only the last layer for $COD$, and perform OOD detection for ImageNet-30. ResNet18 is used for comparison with other approach. The baseline is a contrastive learning-based model, CSI [27], one of SOTA models. CSI is heavier to apply because it use a specific training method for OOD detection such as input transform and contrastive learning. On the other hand, $COD$ is simple and easy to apply because it does not require anything but generating classifiers in the training process. Nevertheless, as shown in Table 3, $COD$ shows comparable performance to CSI.

## 5.2. Classification Accuracy

Some approaches for OOD detection suffered from the degradation of classification performance by change of model structures or loss functions [10, 21, 12]. In this case, two models should be stored aside, one for classification and the other for OOD detection. OOD detection models also need to have high classification performance for efficiency. Table 4 shows that models trained with the cosine-based softmax loss are accurate as much as the standard loss. Our approach can be applied to various model architectures because it only changes the calculation of logits. Also, there is no disadvantage to classification performance.

| model | InD | Standard | Cosine |
|---|---|---|---|
| ResNet34 | CIFAR-10 | 95.25 | 95.34 |
| | CIFAR-100 | 77.70 | 77.78 |
| | SVHN | 96.48 | 96.41 |
| DenseNet3 | CIFAR-10 | 95.14 | 95.08 |
| | CIFAR-100 | 76.43 | 76.87 |
| | SVHN | 96.42 | 96.33 |

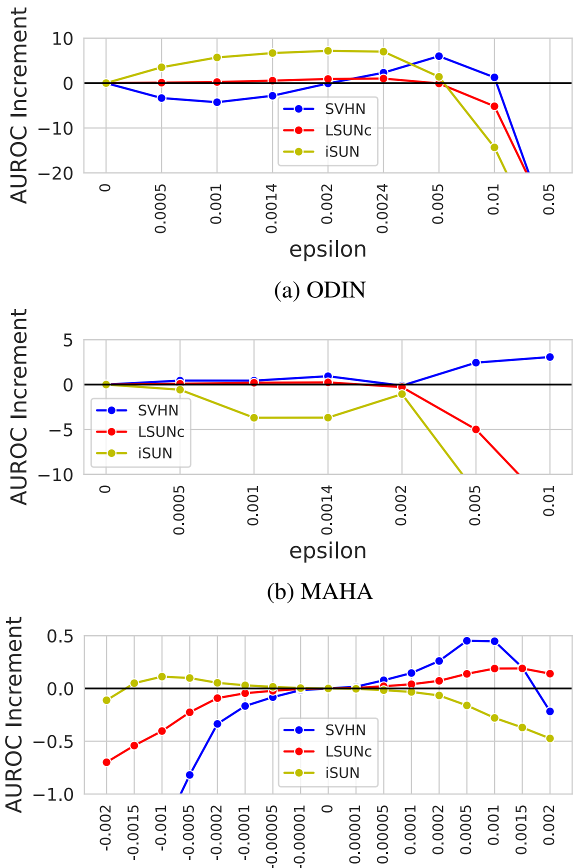
Table 4: Comparison of classification performance

## 5.3. Is the Input Processing Helpful?

The input processing is helpful for performance improvement when the perturbation size is properly chosen. We analyze the performance improvement by the perturbation size of the input processing. We train DenseNet3 with CIFAR-100 and observe the AUROC improvements with different perturbations for SVHN, LSUNc, and iSUN datasets. We perform ODIN and MAHA with pre-trained DenseNet3 with CIFAR-100. Figure 5 shows the improvement by the input processing on the three method.

Each dataset has a different pattern of improvement. For example, our method is improved for SVHN but degraded for iSUN when the epsilon is large. ODIN and MAHA also have similar pattern to our method. This means that it is challenging to find the perturbation size that generally increases the performance regardless of OOD datasets. Hsu *et al.* tried to find the perturbation size without using OOD datasets, but the performance decreased in some cases [12].

The input processing is helpful where the distribution of OOD samples are known. However, in reality, OOD samples can be generated from any distribution. The input processing significantly increases the amount of computation. It is reasonable not to use input processing, which increases



(a) ODIN

(b) MAHA

(c) Proposed Method

Figure 5: OOD detection performance (AUROC) increment by the perturbation size.

computation cost and has a potential to decrease performance.

## 5.4. Fine-tuning with Pre-trained Model

We verify that our approach is also effective to fine-tuned models. We first train a model with the standard softmax, and then fine-tune the last block of the feature extractor and the classifier for only 20 epochs. In the fine-tuning process, all settings except for the learning rate are kept consistent with the previous configuration. The learning rate for fine-tuning starts with 0.5 and decays by a factor 0.1 at 50% and 75% of the training progress.

We can notify that our approach still works for fine-tuned models. The OOD detection performances are comparable to or even higher than those of our original models, as shown in Table 5. The DenseNet3 trained with the standard softmax for SVHN shows a classification accuracy of 96.4%, but its fine-tuned model shows an accuracy of

| InD | OOD | AUROC | TNR95 |
|---|---|---|---|
| CIFAR-10 (Acc: 95.1 $\rightarrow$ 94.1) | SVHN | 99.45 | 98.35 |
| | TINc | 99.53 | 99.61 |
| | TINr | 99.73 | 99.23 |
| | LSUNc | 99.82 | 99.96 |
| | LSUNr | 99.73 | 99.34 |
| | iSUN | 99.72 | 99.27 |
| CIFAR-100 (Acc: 76.4 $\rightarrow$ 75.7) | SVHN | 96.39 | 79.48 |
| | TINc | 97.53 | 88.87 |
| | TINr | 98.03 | 89.37 |
| | LSUNc | 98.72 | 96.81 |
| | LSUNr | 98.16 | 90.91 |
| | iSUN | 97.99 | 88.76 |
| SVHN (Acc: 96.4 $\rightarrow$ 96.1) | CIFAR-10 | 99.17 | 98.18 |
| | CIFAR-100 | 99.22 | 98.33 |
| | TINc | 99.94 | 100.00 |
| | TINr | 99.88 | 99.96 |
| | LSUNc | 99.94 | 100.00 |
| | LSUNr | 99.91 | 100.00 |
| | iSUN | 99.91 | 99.99 |

Table 5: Classification and OOD detection performance of fine-tuned DenseNet3 models. All figures are in %, and are the average results of 5 experiments each.

96.1%. Even though the models are fine-tuned with a limited number of epochs, the classification accuracy and the OOD detection performances remain almost same. The results with ResNet34 are also presented in the appendix.

## 6. Conclusion

We proposed a novel OOD detection method that can be applied to deep neural network-based classifiers. It is simple and fast since it uses only the feature norm and the Mahalanobis distances. Our approach does not use additional data, does not have hyperparameters, and does not perform input processing. Our approach is easy to apply because it only needs to apply the cosine-based softmax loss.

## Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00421, AI Graduate School Support Program (Sungkyunkwan University)), and also supported by the MSIT under the ICT Creative Consilience Program (IITP-2023-2020-0-01821) supervised by the IITP.

## References

[1] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 3

[2] Wenhu Chen, Yilin Shen, Hongxia Jin, and William Wang. A variational dirichlet framework for out-of-distribution detection. *arXiv preprint arXiv:1811.07308*, 2018. 2

[3] Sung-Ik Choi and Sae-Young Chung. Novelty detection via blurring. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 2

[4] Erik Daxberger and José Miguel Hernández-Lobato. Bayesian variational autoencoders for unsupervised out-of-distribution detection. *arXiv preprint arXiv:1912.05651*, 2019. 2

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4690–4699. Computer Vision Foundation / IEEE, 2019. 3

[6] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org, 2016. 2

[7] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9781–9791, 2018. 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. 6

[9] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1, 2

[10] Dan Hendrycks, Mantas Mazeika, and Thomas G. Dietterich. Deep anomaly detection with outlier exposure. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1, 2, 5, 8

[11] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS*

*2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15637–15648, 2019. 3

[12] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized ODIN: detecting out-of-distribution image without learning from out-of-distribution data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10948–10957. IEEE, 2020. 1, 2, 3, 5, 6, 7, 8

[13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017. 6

[14] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009. 6

[15] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7167–7177, 2018. 1, 2, 3, 4, 5, 6

[16] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 1, 2, 3, 5, 6

[17] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. In *Advances in Neural Information Processing Systems*, volume 33, pages 21464–21475. Curran Associates, Inc., 2020. 2

[18] Andrey Malinin and Mark J. F. Gales. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7047–7058, 2018. 2

[19] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *arXiv preprint arXiv:1906.02994*, 5:5, 2019. 2

[20] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 6

[21] Aristotelis-Angelos Papadopoulos, Mohammad Reza Rajati, Nazim Shaikh, and Jiamian Wang. Outlier exposure with confidence control for out-of-distribution detection. *Neurocomputing*, 441:138–150, 2021. 1, 2, 8

[22] Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark A. DePristo, Joshua V. Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14680–14691, 2019. 2

[23] Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with gram matrices. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 8491–8501. PMLR, 2020. 1, 2, 5, 6, 7

[24] Vikash Sehwag, Mung Chiang, and Prateek Mittal. {SSD}: A unified framework for self-supervised outlier detection. In *International Conference on Learning Representations*, 2021. 3

[25] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F. Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 2

[26] Jiaming Song, Yang Song, and Stefano Ermon. Unsupervised out-of-distribution detection with batch normalization. *arXiv preprint arXiv:1910.09115*, 2019. 2

[27] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *Advances in Neural Information Processing Systems*, volume 33, pages 11839–11852. Curran Associates, Inc., 2020. 3, 7

[28] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5265–5274. IEEE Computer Society, 2018. 3

[29] Xiao Zhang, Rui Zhao, Yu Qiao, Xiaogang Wang, and Hongsheng Li. Adacos: Adaptively scaling cosine logits for effectively learning deep face representations. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 10823–10832. Computer Vision Foundation / IEEE, 2019. 3