DICE: A SIMPLE SPARSIFICATION METHOD FOR OUT-OF-DISTRIBUTION DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Detecting out-of-distribution (OOD) inputs is a central challenge for safely deploying machine learning models in the real world. Previous methods commonly rely on an OOD score derived from the overparameterized weight space, while largely overlooking the role of *sparsification*. In this paper, we reveal important insights that reliance on unimportant weights and units can directly attribute to the brittleness of OOD detection. To mitigate the issue, we propose a sparsification-based OOD detection framework termed **DICE**. Our key idea is to rank weights based on a measure of contribution, and selectively use the most salient weights to derive the output for OOD detection. We provide both empirical and theoretical insights, characterizing and explaining the mechanism by which DICE improves OOD detection. By pruning away noisy signals, DICE provably reduces the output variance for OOD data, resulting in a sharper output distribution and stronger separability from ID data. DICE establishes *state-of-the-art* performance, reducing the FPR95 by up to 24.69% compared to the previous best method.

1 INTRODUCTION

Deep neural networks deployed in real-world systems often encounter out-of-distribution (OOD) inputs—unknown samples that the network has not been exposed to during training, and therefore should not be predicted by the model in testing. Being able to estimate and mitigate OOD uncertainty is paramount for safety-critical applications such as medical diagnosis (Wang et al., 2017; Roy et al., 2021) and autonomous driving (Filos et al., 2020). For example, an autonomous vehicle may fail to recognize objects on the road that do not appear in its detection model's training set, potentially leading to a crash. This gives rise to the importance of OOD detection, which allows the learner to express ignorance and take precautions in the presence of OOD data.

The main challenge in OOD detection stems from the fact that modern deep neural networks can easily produce overconfident predictions on OOD inputs, making the separation between ID and OOD data a non-trivial task. The vulnerability of machine learning to OOD data can be hard-wired in high-capacity models used in practice. In particular, modern deep neural networks can overfit observed patterns in the training data (Zhang et al., 2016), and worse, activate features on unfamiliar inputs (Nguyen et al., 2015). To date, existing OOD detection methods commonly derive OOD statistics using overparameterized weights, while largely overlooking the role of *sparsification*.

In this paper, we start by revealing key insights that reliance on unimportant units and weights can directly attribute to the brittleness of OOD detection. Empirically on a network trained with CIFAR-10, we show that an OOD image can activate a non-negligible fraction of units in the penultimate layer (see Figure 1, right). Each point on the horizontal axis corresponds to a single unit. The y-axis measures the unit contribution (*i.e.*, `weight` $\times$ `activation`) to the class output, with the solid line and the shaded area indicating the mean and variance, respectively. Noticeably, for OOD data (gray), we observe a non-negligible fraction of "noisy" units that display high variances of contribution, which is then aggregated to the model's output through summation. As a result, such noisy signals can undesirably manifest in model output—increasing the variance of output distribution and reducing the separability from in-distribution (ID) data.

The above observation naturally inspires a simple and surprisingly effective method, ***Di**rected **S**parisification* (**DICE**), for OOD detection. In particular, we show that utilizing a sparse subset of weights with a significant contribution to the final logit output can better differentiate between

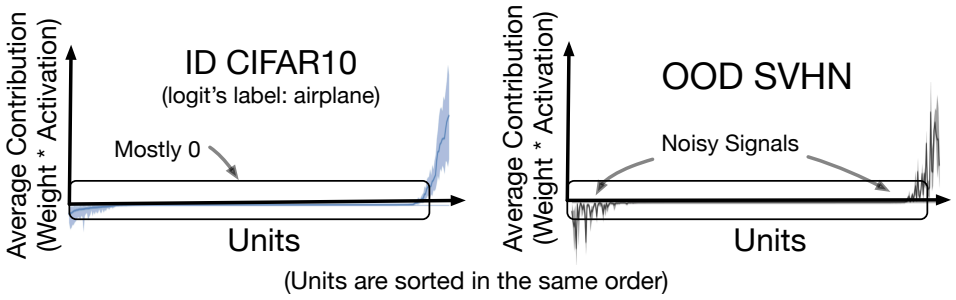

Figure 1: Illustration of unit contribution (*i.e.*, `weight` $\times$ `activation`) to the class output. For class c, the output $f_c(\mathbf{x})$ is the summation of unit contribution from the penultimate feature layer of a neural network. **Units are sorted in the same order**, based on the expectation of ID data's contribution (averaged over many CIFAR-10 samples) on the x-axis. **Shades indicate the variance for each unit**. **Left:** For in-distribution data (CIFAR-10, airplane), only a subset of units contributes to the model output. **Right:** In contrast, out-of-distribution (OOD) data can trigger a non-negligible fraction of units with noisy signals, as indicated by the variances.

ID and OOD data. Conceptually, DICE leverages the observation that a model's prediction for an ID class depends on only a subset of important units (and corresponding weights), as evidenced in Figure 1 (left). To exploit this, our key idea is to rank weights based on the measure of contribution, and selectively use the most significant weights to derive the output for OOD detection. Importantly, DICE can be conveniently used by *post hoc* weight masking on a pre-trained network and therefore can preserve the ID classification accuracy.

We provide both empirical and theoretical insights characterizing and explaining the mechanism by which DICE improves OOD detection. We perform extensive evaluations and establish state-of-the-art performance on a suite of common OOD detection benchmarks, including CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), and a large-scale ImageNet dataset (Deng et al., 2009). DICE reduces the FPR95 by up to 30.13% compared to the counterpart without sparsification. Moreover, we perform ablation using various sparsification techniques and demonstrate the superiority of directed sparsification for OOD detection.

Theoretically, by pruning away noisy signals from unimportant units and weights, DICE *provably reduces the output variance* and results in a sharper output distribution (see Section 6). The sharper distributions lead to a stronger separability between ID and OOD data and overall improved OOD detection performance (*c.f.* Figure 3). Our **key results and contributions** are:

- We introduce DICE, a simple and effective approach for OOD detection utilizing post hoc weight sparsification. We show DICE can generalize effectively to different network architectures, achieving improved OOD detection while preserving the classification accuracy.
- We extensively evaluate DICE on a suite of OOD detection tasks and establish a state-of-the-art performance. Compared to the previous best baseline, DICE reduces the FPR95 by 24.69% on a challenging ImageNet evaluation task.
- We provide empirical ablation and theoretical analysis that improves understanding of a sparsification-based method for OOD detection. Our analysis reveals an important variance reduction effect, which explains the effectiveness of DICE. We hope our insights inspire future research on utilizing weight sparsification for OOD detection.

2 Preliminaries and Analysis

We start by recalling the general setting of the supervised learning problem. We denote by $\mathcal{X} = \mathbb{R}^d$ the input space and $\mathcal{Y} = \{1, 2, ..., C\}$ the output space. A learner is given access to a set of training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ drawn from an unknown joint data distribution P defined on $\mathcal{X} \times \mathcal{Y}$. Furthermore, let $P_{\mathcal{X}}$ denote the marginal probability distribution on $\mathcal{X}$.

Problem statement OOD detection is a binary classification problem to distinguish between in- vs. out-of-distribution data. Given a classifier f learned on in-distribution $P_{\mathcal{X}}$, the goal is to design a

scoring function,

$$g : \mathbf{x} \to \{\text{in}, \text{out}\},$$

that classifies whether a sample $\mathbf{x} \in \mathcal{X}$ is from $P_{\mathcal{X}}$ or not.

Why noisy signals are harmful for OOD detection? A conceptual example We consider a toy example in Figure 2, where the model's output f is the summation of two input variables v_1 and v_2, each representing a unit. For simplicity, we assume v_1 and v_2 are independent. For OOD samples, we assume two inputs are Gaussian distributed noise signals, with $v_1 \sim \mathcal{N}(-1, 1)$ and $v_2 \sim \mathcal{N}(1, 1)$. The output $f = v_1 + v_2$ has a variance of 2 despite a zero mean:

$$f \sim \mathcal{N}(0, 2). \tag{1}$$

Importantly, the variance increases significantly with more units, which is $\sigma_1^2 + \sigma_2^2 + ... + \sigma_m^2$ under m independent variables. The larger variance can result in less separability from ID data's output distribution (see Figure 3), and therefore potentially hinders OOD detection performance.

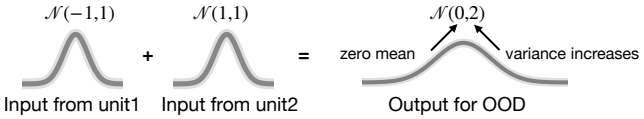

Figure 2: A toy example of summation of two independent Gaussian variables has increased variance.

3 METHOD

Our OOD uncertainty estimation framework with *Directed Sparsification* (DICE) is illustrated in Figure 3. In what follows, we provide a method overview and then describe directed sparsification in detail (Section 3.1). We propose a new OOD detection method with sparsification in Section 3.2.

Method overview As aforementioned, OOD detection performance can suffer from the noisy signals from the high-dimensional inputs. To mitigate this issue, our key idea is to selectively use a subset of important weights to derive the output for OOD detection. By utilizing sparsification, the network prevents adding irrelevant variables to the output. We illustrate our idea in Figure 3. Without DICE (*left*), the final output is a summation of weighted activations across all units, which can have a high variance for OOD data (colored in gray). In contrast, with DICE (*right*), the variance of output can be significantly reduced, which improves separability from ID data. We proceed with describing the mechanism that achieves our novel idea.

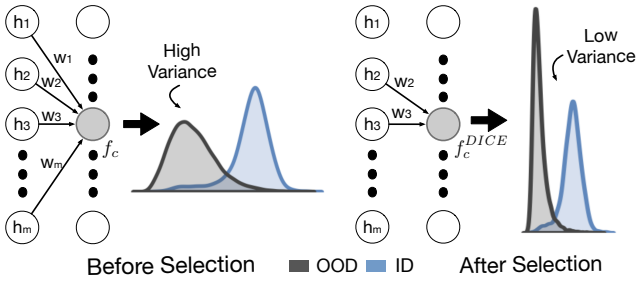

Figure 3: Illustration of out-of-distribution detection using *Directed Sparsification* (**DICE**). We consider a pre-trained neural network, which encodes an input $\mathbf{x}$ to a feature vector $h(\mathbf{x}) \in \mathbb{R}^m$. **Left**: The logit output $f_c(\mathbf{x})$ of class c is a linear combination of activation from *all* units in the preceding layer, weighted by w_i. The full connection results in a high variance for OOD data's output, as depicted in the gray. **Right**: Our proposed approach leverages a selective subset of weights, which effectively reduces the output variance for OOD data, resulting in a sharper score distribution and stronger separability from ID data. The output distributions are based on CIFAR-10 trained network, with ID class label "horse" and SVHN as OOD.

3.1 DIRECTED SPARSIFICATION

We consider a deep feature extractor parameterized by θ, which encodes an input $\mathbf{x} \in \mathbb{R}^d$ to a feature space with dimension m. We denote by $h(\mathbf{x}) \in \mathbb{R}^m$ the feature vector from the penultimate layer of the network. A weight matrix $\mathbf{W} \in \mathbb{R}^{m \times C}$ connects the feature $h(\mathbf{x})$ to the output $f(\mathbf{x})$.

Contribution matrix We perform a *directed sparsification* based on a measure of contribution, and preserve the most important weights in $\mathbf{W}$. To measure the contribution, we define a contribution matrix $\mathbf{V} \in \mathbb{R}^{m \times C}$, where each column $\mathbf{v}_c \in \mathbb{R}^m$ is given by:

$$\mathbf{v}_c = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[\mathbf{w}_c \odot h(\mathbf{x})], \tag{2}$$

where $\odot$ indicates the element-wise multiplication, and $\mathbf{w}_c$ indicates weight vector for class c. Each element in $\mathbf{v}_c \in \mathbb{R}^m$ intuitively measures the corresponding unit's average contribution to class c, estimated on in-distribution data $\mathcal{D}$. A larger value indicates a higher contribution to the output $f_c(\mathbf{x})$ of class c. The vector $\mathbf{v}_c$ is derived for all classes $c \in \{1, 2, ..., C\}$, forming the contribution matrix $\mathbf{V}$. Each element $\mathbf{v}_c^i \in \mathbf{V}$ measures the average contribution (`weight` $\times$ `activation`) from a unit i to the output class $c \in \{1, 2, ..., C\}$.

We can now select the top-k weights based on the k-largest elements in $\mathbf{V}$. In particular, we define a masking matrix $\mathbf{M} \in \mathbb{R}^{m \times C}$, which returns a matrix by setting 1 for entries corresponding to the k largest elements in $\mathbf{V}$ and setting other elements to zeros. The model output under *contribution-directed sparsification* is given by

$$f^{\text{DICE}}(\mathbf{x}; \theta) = (\mathbf{M} \odot \mathbf{W})^\top h(\mathbf{x}) + \mathbf{b}, \tag{3}$$

where $\mathbf{b} \in \mathbb{R}^C$ is the bias vector. The procedure described above essentially accounts for inputs from the most relevant units in the penultimate layer. Importantly, the sparsification can be conveniently imposed by *post hoc* weight masking on a pre-trained network, without changing any parameterizing of the neural network. Therefore one can improve OOD detection while preserving the ID classification accuracy, as we show in Section 5.

Sparsity parameter p To align with the convention in literature, we use the sparsity parameter $p = 1 - \frac{k}{m \cdot C}$ in the remainder paper. A higher p indicates a larger fraction of weights dropped. When $p = 0$, the output becomes equivalent to the original output $f(\mathbf{x}; \theta)$ using dense transformation, where $f(\mathbf{x}; \theta) = \mathbf{W}^\top h(\mathbf{x}) + \mathbf{b}$.

3.2 OOD DETECTION WITH DICE

Our method DICE in Section 3.1 can be flexibly leveraged by the downstream OOD scoring function:

$$g_\lambda(\mathbf{x}) = \begin{cases} \text{in} & S_\theta(\mathbf{x}) \geq \lambda \\ \text{out} & S_\theta(\mathbf{x}) < \lambda \end{cases}, \tag{4}$$

where a thresholding mechanism is exercised to distinguish between ID and OOD during test time. The threshold λ is typically chosen so that a high fraction of ID data (*e.g.,* 95%) is correctly classified. In particular, we derive an energy score using the logit output $f^{\text{DICE}}(\mathbf{x})$ from contribution-directed sparsification. We use energy score since it is easy to compute, hyperparameter-free, directly compatible with DICE (see remark below and Section 6), and achieves strong performance (see Section 4). Note that DICE can also be compatible with the scoring function such as maximum softmax probability (Hendrycks & Gimpel, 2017) (see Appendix F). Specifically, the energy function (Liu et al., 2020) maps the logit outputs $f^{\text{DICE}}(\mathbf{x})$ to a scalar $E_\theta(\mathbf{x}) \in \mathbb{R}$, which is relatively lower for ID data:

$$S_\theta(\mathbf{x}) = -E_\theta(\mathbf{x}) = \log \sum_{c=1}^{C} \exp(f_c^{\text{DICE}}(\mathbf{x})). \tag{5}$$

Note that we negate the sign of energy score to align with the convention that samples with higher scores are classified as ID and vice versa.

Remark Energy score is directly compatible with DICE since `logsumexp` operation is a smooth approximation of maximum logit, *i.e.*, $\log \sum_c e^{f_c(\mathbf{x})} \approx \max_c f_c(\mathbf{x})$. Our theoretical analysis (Section 6) shows that DICE reduces the variance of logit $f_c(\mathbf{x})$. This means that for detection scores such as energy score, the gap between OOD and ID score will be enlarged after applying DICE, which makes thresholding more capable of separating OOD and ID inputs and benefit OOD detection.

Methods	OOD Datasets								Average	
	iNaturalist		SUN		Places		Textures			
	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑
MSP	63.69	87.59	79.98	78.34	81.44	76.76	82.73	74.45	76.96	79.29
ODIN	62.69	89.36	71.67	83.92	76.27	80.67	81.31	76.30	72.99	82.56
Mahalanobis*	96.34	46.33	88.43	65.20	89.75	64.46	52.23	72.10	81.69	62.02
Energy	64.91	88.48	65.33	85.32	73.02	81.37	80.87	75.79	71.03	82.74
G-ODIN§	66.36	84.68	65.93	84.70	65.39	85.40	64.68	80.50	65.59	83.82
DICE (ours)	**33.78**$^{\pm 0.97}$	**93.70**$^{\pm 0.67}$	**35.03**$^{\pm 1.34}$	**93.61**$^{\pm 0.85}$	**50.59**$^{\pm 1.86}$	**89.35**$^{\pm 0.78}$	**44.19**$^{\pm 1.56}$	**88.21**$^{\pm 0.73}$	**40.90**$^{\pm 1.43}$	**91.22**$^{\pm 0.76}$

Table 1: **Main results.** Comparison with competitive *post hoc* out-of-distribution detection methods. All methods are based on a discriminative model trained on ImageNet, without using any auxiliary outlier data. ↑ indicates larger values are better and ↓ indicates smaller values are better. All values are percentages. **Bold** numbers are superior results. § indicates model retraining using a different loss function is required. $\star$ indicates a separate binary classifier needs to be trained for OOD detection. We report standard deviations estimated across 5 independent runs.

4 EXPERIMENTS

In this section, we evaluate our method on a suite of OOD detection tasks. In Section 4.1, we begin with a large-scale image classification network trained on ImageNet (Deng et al., 2009). We continue with the CIFAR benchmarks (Krizhevsky et al., 2009) that are routinely used in literature (Section 4.2).

4.1 EVALUATION ON LARGE-SCALE IMAGENET CLASSIFICATION NETWORKS

Dataset We first evaluate DICE on a large-scale ImageNet classification model. OOD detection for ImageNet model is more challenging due to both a larger feature space ($m = 2,048$) as well as a larger label space ($C = 1,000$). In particular, the large-scale evaluation can be more relevant to real-world applications, where the deployed models often operate on images that have high resolution and contain more class labels. Moreover, as the number of feature dimensions increases, noisy signals may increase accordingly, which can corrupt the output more and make OOD detection more challenging. We use four OOD test datasets (see Appendix I for details). Specifically, the four test OOD datasets are from (subsets of) `Places365` (Zhou et al., 2017), `Textures` (Cimpoi et al., 2014), `iNaturalist` (Horn et al., 2018), and `SUN` (Xiao et al., 2010) with non-overlapping categories *w.r.t* ImageNet. The evaluations span a diverse range of domains including fine-grained images, scene images, and textural images.

Experimental details We use Google BiT-S models (Kolesnikov et al., 2020)[1] for all methods (including baselines). The models are trained on ImageNet-1k, using ResNet-v2 architecture (He et al., 2016) with depth 101. At test time, all images are resized to 480 $\times$ 480. We use the entire training dataset to estimate the contribution matrix and mask $\mathbf{M}$. We use a validation set of `Gaussian noise` images, which are generated by sampling from $\mathcal{N}(0, 1)$ for each pixel location. The optimal p is selected from $\{0.1, 0.3, 0.5, 0.7, 0.9, 0.99\}$, which is 0.9 for all datasets. We use Gaussian noise for its simplicity, easy to generate and use, and the fact that it is a clear OOD w.r.t ID data. We also show in Table 4 using Gaussian can already find the near-optimal one on all three ID datasets considered. The hardware used for experiments is specified in Appendix D.

Comparison with OOD detection baselines In Table 1, we compare DICE with OOD detection methods that are competitive in the literature, where DICE establishes the state-of-the-art performance. For readers' convenience, a brief introduction of baselines and hyperparameters is provided in Appendix B. For a fair comparison, all methods use the model *post hoc*. We report performance for each OOD test dataset, as well as the average of the four. DICE outperforms baselines, including Maximum Softmax Probability (Hendrycks & Gimpel, 2017), ODIN (Liang et al., 2018), Mahalanobis distance (Lee et al., 2018), Generalized ODIN (Hsu et al., 2020), and Energy score (Liu et al., 2020). Noticeably, DICE reduces the FPR95 by **24.69**% compared to the best baseline (Hsu et al., 2020). Note that Generalized ODIN requires model retraining under the DeConf-C loss, and Mahalanobis requires training a separate binary classifier. Moreover, Mahalanobis displays limiting performance with FPR95 40.79% lower than DICE. This is likely due to the increased size of label space makes the class-conditional Gaussian density estimation less viable. In contrast, DICE is easier to use in practice, and can be implemented through simple *post hoc* weight masking.

[1] `https://github.com/google-research/big_transfer`.

4.2 EVALUATION ON COMMON BENCHMARKS

Experimental details We use CIFAR-10 (Krizhevsky et al., 2009), and CIFAR-100 (Krizhevsky et al., 2009) datasets as in-distribution data. We use the standard split with 50,000 training images and 10,000 test images. We evaluate the model on six common OOD benchmark datasets: `Textures` (Cimpoi et al., 2014), `SVHN` (Netzer et al., 2011), `Places365` (Zhou et al., 2017), `LSUN-Crop` (Yu et al., 2015), `LSUN-Resize` (Yu et al., 2015), and `iSUN` (Xu et al., 2015). We use DenseNet-101 architecture (Huang et al., 2017) and train on in-distribution datasets. The feature dimension of the penultimate layer is 342. For both CIFAR-10 and CIFAR-100, the model is trained for 100 epochs with batch size 64, weight decay 0.0001 and momentum 0.9. The start learning rate is 0.1 and decays by a factor of 10 at epochs 50, 75, and 90.

DICE outperforms discriminative-based approaches We show the comparison in Table 2. Due to the space limit, all the numbers reported are averaged over six OOD test datasets described above. The full results for each evaluation dataset are provided in Appendix H. On CIFAR-100, we show that using a sparse connection reduces the average FPR95 by **18.73%** compared to the counterpart without sparsification (Liu et al., 2020). In practice, DICE can be implemented through a simple *post hoc* weight masking, and therefore can improve OOD detection while ensuring the ID classification accuracy. *We show that DICE is effective on a different architecture such as ResNet-101* (results in Supplementary, Section C).

DICE outperforms generative methods In Table 3, we also compare with generative-based methods: Glow (Kingma & Dhariwal, 2018), IGEBM (Du & Mordatch, 2019), JEM (Grathwohl et al., 2019), and input complexity (Serrà et al., 2020). We follow the same evaluations as in prior literature. Using CIFAR-10 as ID data, DICE outperforms the best method (Serrà et al., 2020) by 5.3% on average. It is also important to note that generative models can be prohibitively challenging to train and optimize on large-scale datasets (and therefore are primarily evaluated on CIFAR-10 in literature). DICE does not suffer from this issue and performs competitively on ImageNet, as we show in Section 4.1 above.

Method	SVHN	CIFAR-100	CelebA
Glow	0.64	0.65	0.54
IGEBM	0.43	0.54	0.69
JEM-softmax	0.89	0.87	0.79
JEM-likelihood	0.67	0.67	0.75
Input Complexity	**0.95**	0.74	0.86
DICE (ours)	**0.95**	**0.89**	**0.87**

Table 3: Comparison with generative-based models for OOD detection. Models are trained on CIFAR-10. Values are AUROC.

5 DISCUSSION AND ABLATIONS

Ablation on different pruning methods In this ablation, we explore and contrast OOD detection performance under the most common *post hoc* sparsification methods, and show that our directed sparsification is optimal among all. In particular, we consider a suite of both unit-based and weight-based sparsification strategies that are popular in literature. We consider (1) **unit dropout** (Srivastava et al., 2014) which randomly drops a fraction of units, (2) **unit pruning** (Li et al., 2017) which drops units with the smallest L_2 norm of the corresponding weight vectors, (3) **weight dropout** (Wan et al.,

Method Type	Method	CIFAR-10		CIFAR-100		ImageNet	
		FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑
Non-Sparse	MSP	48.73	92.46	80.13	74.36	76.96	79.29
	ODIN	24.57	93.71	58.14	84.49	72.99	82.56
	Mahalanobis	31.42	89.15	50.14	85.03	81.69	62.06
	Energy	26.55	94.57	68.45	81.19	71.03	82.74
	Generalized ODIN	34.25	90.61	52.87	85.24	65.59	83.82
Sparse	Weight-Droput	62.12	85.12	77.48	71.59	70.99	79.27
	Unit-Droput	86.63	66.40	85.56	61.06	91.66	53.96
	Weight-Pruning	24.25	94.84	59.93	84.22	57.22	87.70
	Unit-Pruning	26.55	94.57	68.48	81.19	79.96	76.07
	DICE (ours)	**20.83**$^{\pm 1.58}$	**95.24**$^{\pm 0.24}$	**49.72**$^{\pm 1.69}$	**87.23**$^{\pm 0.73}$	**40.90**$^{\pm 1.43}$	**91.22**$^{\pm 0.76}$

Table 2: **Ablation results.** Comparison of different OOD detection baselines and sparsification method. All sparsification methods are based on the *same OOD scoring function* (Liu et al., 2020), with sparsity parameter $p = 0.9$. ↑ indicates larger values are better and ↓ indicates smaller values are better. All values are percentages and are averaged over multiple OOD test datasets. **Bold** numbers are superior results. We report standard deviations estimated across 5 independent runs.

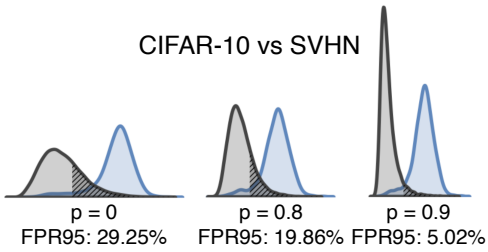

Figure 4: Output distribution for ID (blue) and OOD (gray) under different sparsity parameters p. From left to right, the variances of gray distributions decrease as the sparsity increases, resulting in improved OOD detection performance (measured by FPR95).

Sparsity	FPR95↓	AUROC ↑	OOD STD↓
$p = 0.99$	72.79	75.19	0.1595
$p = 0.9$	50.60	86.89	0.1293
$p = 0.7$	52.74	85.80	0.1247
$p = 0.5$	52.87	85.72	0.1255
$p = 0.3$	52.73	85.68	0.1264
$p = 0.1$	54.02	85.67	0.1366
$p = 0$ (Liu et al., 2020)	68.45	81.19	0.1913

Table 4: Effect of varying sparsity parameter p during inference time. Model is trained on CIFAR-100 using DenseNet101 (Huang et al., 2017). A similar trend is observed for other ID datasets as shown in Appendix A.

2013) which randomly drops weights in the fully connected layer, and (4) **weight pruning** (Han et al., 2015) drops weights with the smallest entries under the L_1 norm. For consistency, we use the same OOD scoring function (*e.g.*, energy) and the same sparsity parameter $p = 0.9$ for all methods.

Our ablation reveals several important insights shown in Table 2. First, in contrasting weight dropout vs. DICE, a salient performance gap of **41.29**% (FPR95) is observed on CIFAR-10 under the same sparsity. This suggests the importance of dropping weights *directedly* rather than *randomly*. Second, DICE outperforms a popular L_1-norm-based pruning method (Han et al., 2015) by up to **10.21**% (FPR95) on CIFAR-100. While it prunes weights with low magnitude, negative weights with large L_1-norm can be kept. Unfortunately, the negative weights can undesirably corrupt the output with noisy signals (as shown in Figure 1). The performance gain of DICE over (Han et al., 2015) attributes to our contribution-directed sparsification, which is better suited for OOD detection. In Appendix G, we show that further tuning p for these methods does not outperform ours.

Effect of sparsity parameter p We now characterize the effect of sparsity parameter p. In Table 4, we summarize the OOD detection performance for DenseNet trained on CIFAR-100, where we vary $p = \{0.1, 0.3, 0.5, 0.7, 0.9, 0.99\}$. Interestingly, we observe the performance improves as with mild sparsity parameter p. This observation confirms that over-parameterization does compromise the OOD detection ability, and DICE can effectively alleviate the problem. When p is too large (*e.g.*, $p = 0.99$), the OOD performance starts to degrade. A similar trend holds on other ID datasets including CIFAR-10 and ImageNet, with full details in Appendix A.

In-distribution classification accuracy DICE can improve OOD detection while maintaining comparable ID classification accuracy. For example, on CIFAR-10, the accuracy is 95.10% (original) vs. 95.12% (DICE, $p = 0.9$). Importantly, once the input image is marked as ID, one can always use the original FC layer, *which is guaranteed to give identical classification accuracy*.

Ablation on unit selection We have shown that choosing a subset of weight parameters (with *top-k* unit contribution) significantly improves the OOD detection performance. In this ablation, we also analyze those "lower contribution units" for OOD detection. Specifically, we consider: (1) *Bottom-k* which only includes k unit contribution with least contribution values, (2) *top+bottom-k* which includes k unit contribution with largest and smallest contribution values, (3) *random-k* which randomly includes k unit contribution and (4) *top-k* which is equivalent to DICE method. In Table 5, we show that DICE outperforms all variants.

Method	CIFAR-10↓	CIFAR-100 ↓	ImageNet ↓
Bottom-k	91.87	99.70	99.10
(Top+Bottom)-k	24.25	59.93	57.22
Random-k	62.12	77.48	70.77
Top-k (**DICE**)	**21.76**	**50.60**	**42.17**

Table 5: Ablation on different strategies of choosing a subset of units. Values are FPR95 (averaged over multiple test datasets).

Effect of variance reduction for output distribution Figure 4 shows that DICE has an interesting variance reduction effect on the output distribution for OOD data, and at the same time preserves the information for the ID data (CIFAR-10, class “horse"). The output distribution without any sparsity ($p = 0$) appears to have a larger variance, resulting in less separability from ID data (see left of Figure 4). In contrast, sparsification with DICE results in a sharper distribution, which benefits OOD detection. In Table 4, we also measure the standard deviation of energy score for OOD data

(normalized by the mean of ID data's OOD scores in each setting). By way of sparsification, DICE can reduce the output variance. Next, we formally characterize this and provide a theoretical explanation.

6 WHY DOES DICE IMPROVE OOD DETECTION?

In this section, we formally explain the mechanism by which reliance on irrelevant units hurts OOD detection and how DICE effectively mitigates the issue. Our analysis highlights that DICE reduces the output variance for both ID and OOD data. Below we provide details.

Setup For a class c, we consider the unit contribution vector $\mathbf{v}$, the element-wise multiplication between the feature vector $\mathbf{h}(\mathbf{x})$ and corresponding weight vector $\mathbf{w}$. We contrast the two outputs with and without sparsity:

$$f_c = \sum_{i=1}^{m} v_i \quad \text{(w.o sparsity)} \tag{6}$$

$$f_c^{\text{DICE}} = \sum_{i \in \text{top units}} v_i \quad \text{(w. sparsity)}, \tag{7}$$

where f_c is the output using the summation of all units' contribution, and f_c^{DICE} takes the input from the top units (ranked based on the average contribution on ID data, see bottom of Figure 5).

DICE reduces the output variance We consider the unit contribution vector for OOD data $\mathbf{v} \in \mathbb{R}^m$, where each element is a *random variable* v_i with mean $\mathbb{E}[v_i] = \mu_i$ and variance $\text{Var}[v_i] = \sigma_i^2$.

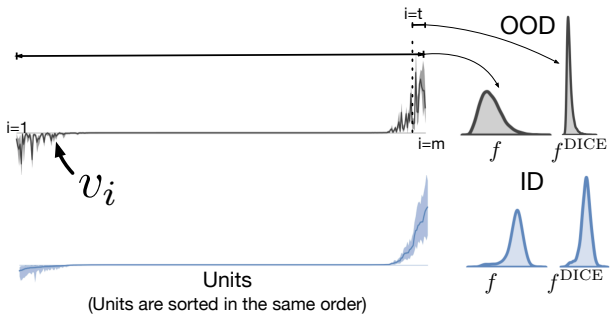

Figure 5: Units in the penultimate layer are sorted based on the average contribution to a CIFAR-10 class ("airplane"). OOD data (SVHN) can trigger a non-negligible fraction of units with noisy signals on the CIFAR-10 trained model. **Units are sorted in the same order, based on the ID contribution.**

For simplicity, we assume each component is independent, but our theory can be extended to correlated variables (see Remark 1). Importantly, indices in $\mathbf{v}$ are sorted based on *the same order* of unit contribution on ID data. In particular, units on the leftmost side generally have negative contributions $u_i < 0$, whereas units on the rightmost side have positive contributions $u_i > 0$. By using units on the rightmost side, we now show the key result that DICE reduces the output variance.

Proposition 1. *Let v_i and v_j be two independent random variables. Denote the summation $r = v_i + v_j$, we have $\mathbb{E}[r] = \mathbb{E}[v_i] + \mathbb{E}[v_j]$ and $\text{Var}[r] = \text{Var}[v_i] + \text{Var}[v_j]$.*

Lemma 2. *Assume there exists top $m - t$ units yielding approximately identical mean $\mathbb{E}[f_c^{DICE}] \approx \mathbb{E}[f_c]$. Then the output variable f_c^{DICE} under sparsification has reduced variance:*

$$\text{Var}[f_c] - \text{Var}[f_c^{DICE}] = \sum_{i=1}^{t} \sigma_i^2$$

Proof. The proof directly follows Proposition 1.

Remark 1 (Extension to correlated variables) We can show in a more general case with correlated variables, the variance reduction is:

$$\text{Var}[f_c] - \text{Var}[f_c^{\text{DICE}}] = \sum_{i=1}^{t} \sigma_i^2 + 2 \sum_{1 \le i < j \le m} \text{Cov}(v_i, v_j) - 2 \sum_{t < i < j \le m} \text{Cov}(v_i, v_j),$$

where $\text{Cov}(\cdot, \cdot)$ is the covariance. Our analysis shows that the covariance matrix primarily consists of 0, which indicates the independence of variables. Moreover, the summation of non-zero entries in the full matrix (i.e., the second term) is greater than that of the submatrix with top units (i.e., the third term), resulting in a larger variance reduction than in Lemma 2. See complete proof in Appendix E.

Sparsity	$p = 0.9$	$p = 0.7$	$p = 0.5$	$p = 0.3$	$p = 0.1$	$p = 0$
$\mathbb{E}_{\text{in}}[f_c^{\text{DICE}}]$ - $\mathbb{E}_{\text{out}}[f_c^{\text{DICE}}]$	7.92	7.28	7.99	8.04	7.36	6.67

Table 6: Difference between the mean of ID's output and OOD's output. Here we use CIFAR-100 as ID data and $\mathbb{E}_{\text{in}}[f_c^{\text{DICE}}]$ - $\mathbb{E}_{\text{out}}[f_c^{\text{DICE}}]$ is averaged over six common OOD benchmark datasets described in Section 4.

Remark 2 (Variance reduction on ID data) Note that we can also show the effect of variance reduction for ID data in a similar way. Importantly, DICE effectively preserves the most important information akin to the ID data, while reducing noisy signals that are harmful for OOD detection.

Remark 3 (Mean of output) We also show in Table 6 the effect of sparsity on the mean of output: $\mathbb{E}_{\text{in}}[f_c^{\text{DICE}}]$ and $\mathbb{E}_{\text{out}}[f_c^{\text{DICE}}]$. We show that DICE maintains similar (or even enlarges) differences in terms of mean as sparsity p increases. Therefore, DICE overall benefits OOD detection due to reduced output variances and increased differences of mean.

7 Related Work

OOD detection for discriminative models The softmax confidence score has become a common baseline for OOD detection (Hendrycks & Gimpel, 2017). Several works attempt to improve the OOD uncertainty estimation using deep ensembles (Lakshminarayanan et al., 2017), ODIN score (Liang et al., 2018), Mahalanobis distance-based confidence score (Lee et al., 2018), generalized ODIN score (Hsu et al., 2020), and the energy score (Liu et al., 2020). However, previous methods primarily derive OOD scores using overparameterzied weights. In contrast, our work is motivated by a novel analysis of unit contribution, and shows that sparsification is a surprisingly effective approach for OOD detection. A separate line of methods uses an auxiliary OOD dataset for model regularization during training (Guenais et al., 2020; Hendrycks et al., 2019; Papadopoulos et al., 2020; Mohseni et al., 2020). However, in many applications, it can be prohibitive to construct an auxiliary outlier dataset. In contrast, our method does not assume the availability of any auxiliary data and can be broadly used for any pre-trained model.

OOD detection with generative models Alternative approaches for detecting OOD inputs resort to generative models that directly estimate in-distribution density (Dinh et al., 2016; Kingma & Dhariwal, 2018; Tabak & Turner, 2013; Van den Oord et al., 2016). However, Nalisnick et al. showed that deep generative models can assign a high likelihood to OOD data. Several methods improve OOD detection using generative models, including likelihood ratios (Ren et al., 2019; Serrà et al., 2020), likelihood regret (Xiao et al., 2020). However, generative models can be prohibitively challenging to train and optimize (Hinz et al., 2019), and the performance can often lag behind the discriminative counterpart (Kirichenko et al., 2020; Wang et al., 2020). In contrast, our method relies on a discriminative classifier, which is much easier to optimize and achieves stronger performance.

Pruning and sparsification A great number of effort has been put towards improving post hoc pruning and training time regularization for deep neural networks (Babaeizadeh et al., 2016; Gomez et al., 2019; Han et al., 2016; 2015; LeCun et al., 1989; Li et al., 2017; Louizos et al., 2018). Our work primarily considers *post hoc* sparsification strategy which operates conveniently on a pre-trained network. The two most popular Bernoulli dropout techniques include unit dropout (Srivastava et al., 2014) and weight dropout (Srivastava et al., 2014). Post hoc pruning strategies truncate weights with low magnitude (Han et al., 2015), or drop units with low weight norms (Li et al., 2017). Orthogonal to existing works, our goal is to improve the OOD detection performance rather than the classification task. DICE first demonstrates that sparsification can be useful for OOD detection. An in-depth discussion and comparison of these methods are presented in Section 5.

8 Conclusion

This paper provides a simple sparsification strategy termed DICE , which ranks weights based on a contribution measure and then uses the most significant weights to derive the output for OOD detection. We provide both empirical and theoretical insights characterizing and explaining the mechanism by which DICE improves OOD detection. By exploiting the most important weights, DICE provably reduces the output variance for OOD data, resulting in a sharper output distribution and stronger separability from ID data. Extensive experiments show DICE can significantly improve the performance of OOD detection for over-parameterized networks. We hope our research can raise more attention to the importance of weight sparsification for OOD detection.

ETHICS STATEMENT

This paper aims to improve the reliability and safety of modern neural networks. Our study can lead to direct benefits and societal impacts when deploying machine learning models in the real world. Our work does not involve any human subjects or violation of legal compliance. We do not anticipate any potentially harmful consequences to our work. Through our study and releasing our code, we hope to raise stronger research and societal awareness towards the problem of out-of-distribution detection in real-world settings.

REPRODUCIBILITY STATEMENT

Authors of the paper recognizes the importance and value of reproducible research. We summarize our efforts below to facilitate reproducible results:

1. **Dataset.** We use publicly available datasets, which are described in detail in *Section 4.1* and *Section 4.2*. The concept list of OOD test datasets is exhaustively specified in *Appendix I*.
2. **Assumption and proof.** The complete proof of our theoretical contribution is provided in *Appendix E*, which supports our theoretical claims made in *Section 6*.
3. **Baselines.** The description and hyperparameters of baseline methods are specified in *Appendix B*.
4. **Model.** Our main results on ImageNet are based on Google's BiT pre-trained model, which has been publicly released `https://github.com/google-research/big_transfer`. Due to the post hoc nature of our method, this allows the research community to reproduce our numbers provided with the same model and evaluation datasets.
5. **Implementation.** The simplicity of the DICE ease the reproducibility, as it only requires a few lines of code modification in the PyTorch model specification. Specifically, one can replace the weight matrix in the penultimate layer of deep networks using the following code:

```
        threshold = numpy.percentile(V, p)
        M = V > threshold
        W_new = W * M
```

6. **Open Source.** The codebase and the dataset will be released for reproducible research.

REFERENCES

Mohammad Babaeizadeh, Paris Smaragdis, and Roy H. Campbell. Noiseout: A simple way to prune neural networks. *CoRR*, abs/1611.06211, 2016.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

Angelos Filos, Panagiotis Tigkas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, pp. 3145–3153. PMLR, 2020.

Aidan N. Gomez, Ivan Zhang, Kevin Swersky, Yarin Gal, and Geoffrey E. Hinton. Learning sparse networks using targeted dropout. *CoRR*, abs/1905.13678, 2019.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2019.

T. Guenais, D. Vamvourellis, Y. Yacoby, F. Doshi-Velez, and W. Pan. Bacoun: Bayesian classifers with out-of-distribution uncertainty. *ICML Workshop on Uncertainty in Deep Learning*, 1:1–24, 2020.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28, pp. 1135–1143, 2015.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations, ICLR*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations*, 2017.

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

Tobias Hinz, Stefan Heinrich, and Stefan Wermter. Generating multiple objects at spatially distinct locations. *arXiv preprint arXiv:1901.00686*, 2019.

Grant Horn, Oisin Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. *The iNaturalist Species Classification and Detection Dataset*. 06 2018.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Rui Huang and Yixuan Li. Towards scaling out-of-distribution detection for large semantic space. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pp. 10215–10224, 2018.

Polina Kirichenko, Pavel Izmailov, and Andrew G Wilson. Why normalizing flows fail to detect out-of-distribution data. *Advances in Neural Information Processing Systems*, 33, 2020.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pp. 6402–6413, 2017.

Yann LeCun, John S Denker, Sara A Solla, Richard E Howard, and Lawrence D Jackel. Optimal brain damage. In *Advances in Neural InformationProcessing Systems*, volume 2, pp. 598–605. Citeseer, 1989.

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*, pp. 7167–7177, 2018.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ICLR*, 2017.

Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l_0 regularization. In *International Conference on Learning Representations*, 2018.

Sina Mohseni, Mandar Pitale, JBS Yadawa, and Zhangyang Wang. Self-supervised learning for generalizable out-of-distribution detection. In *AAAI*, pp. 5216–5223, 2020.

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

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436, 2015.

Aristotelis-Angelos Papadopoulos, Nazim Shaikh, Jiamian Wang, and Mohammad Reza Rajati. Simultaneous classification and out-of-distribution detection using deep neural networks, 2020. URL `https://openreview.net/forum?id=Hyez1CVYvr`.

Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, pp. 14680–14691, 2019.

Abhijit Guha Roy, Jie Ren, Shekoofeh Azizi, Aaron Loh, Vivek Natarajan, Basil Mustafa, Nick Pawlowski, Jan Freyberg, Yuan Liu, Zach Beaver, et al. Does your dermatology classifier know what it doesn't know? detecting the long-tail of unseen conditions. *arXiv preprint arXiv:2104.03829*, 2021.

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 *International Conference on Learning Representations*, 2020.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pp. 4790–4798, 2016.

Li Wan, Matthew D. Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus. Regularization of neural networks using dropconnect. In *ICML*, volume 28, pp. 1058–1066, 2013.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2097–2106, 2017.

Ziyu Wang, Bin Dai, David Wipf, and Jun Zhu. Further analysis of outlier detection with deep generative models. *Advances in Neural Information Processing Systems*, 33, 2020.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pp. 3485–3492. IEEE Computer Society, 2010.

Zhisheng Xiao, Qing Yan, and Yali Amit. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. *Advances in Neural Information Processing Systems*, 33, 2020.

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

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

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2016.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

A EFFECT OF SPARSITY PARAMETER p ON CIFAR-10 AND IMAGENET

We characterize the effect of sparsity parameter p on other ID datasets. In Table 7, we summarize the OOD detection performance for DenseNet trained on CIFAR-10 and ImageNet, where we vary $p = \{0.1, 0.3, 0.5, 0.7, 0.9, 0.99\}$. A similar trend is observed as discussed in the main paper.

Sparsity	**CIFAR-10**		**ImageNet**	
	FPR95 $\downarrow$	**AUROC** $\uparrow$	**FPR95** $\downarrow$	**AUROC** $\uparrow$
$p = 0.99$	57.57	84.29	59.64	83.57
$p = 0.9$	21.76	94.91	41.91	91.10
$p = 0.7$	21.76	94.91	41.88	91.21
$p = 0.5$	21.76	94.91	41.83	91.21
$p = 0.3$	21.75	94.91	41.20	91.22
$p = 0.1$	21.92	94.90	43.97	89.87
$p = 0$ (Liu et al., 2020)	26.55	94.57	71.03	82.74

Table 7: Effect of varying sparsity parameter p on CIFAR-10 and ImageNet. Results are averaged on the test datasets described in Section 4.

B DETAILS OF BASELINES

For the reader's convenience, we summarize in detail a few common techniques for defining OOD scores that measure the degree of ID-ness on a given input. By convention, a higher (lower) score is indicative of being in-distribution (out-of-distribution).

MSP (Hendrycks & Gimpel, 2017) propose to use the maximum softmax score as the OOD score.

ODIN (Liang et al., 2018) Liang et al. improved OOD detection with temperature scaling and input perturbation. In all experiments, we set the temperature scaling parameter $T = 1000$. For ImageNet, we found the input perturbation does not further improve the OOD detection performance and hence we set $\epsilon = 0$. Following the setting in Liang et al. (2018), we set ϵ to be 0.004 for CIFAR-10 and CIFAR-100.

Mahalanobis (Lee et al., 2018) Lee et al. use multivariate Gaussian distributions to model class-conditional distributions of softmax neural classifiers and use Mahalanobis distance-based scores for OOD detection. We use 500 examples randomly selected from ID datasets and an auxiliary tuning dataset to train the logistic regression model and tune the perturbation magnitude ϵ. The tuning dataset consists of adversarial examples generated by FGSM (Goodfellow et al., 2014) with a perturbation size of 0.05. The selected ϵ's are 0.001, 0.0, and 0.0 for ImageNet-1k, CIFAR-10, and CIFAR-100, respectively.

Generalized ODIN (Hsu et al., 2020) Hsu et al. propose a specialized network to learn temperature scaling and a novel strategy to choose perturbation magnitude, in order to replace manually-set hyperparameters. Our training configurations strictly follows the the original paper, where we train the DeConf-C network for 200 epochs without applying the weight decay in the final layer of the Deconf classifier (notated as $h_i(x)$ in Hsu et al. (2020)). The other settings such as learning rate, momentum and training batch size are the same as ours. Note that G-ODIN has slight advantage due to longer training time than ours (100 epochs). We choose the best perturbation magnitude ϵ by maximizing the confidence scores on 1,000 examples randomly selected from ID datasets. The selected ϵ value is 0.02 for all (ImageNet-1k, CIFAR-10, and CIFAR-100).

Energy (Liu et al., 2020) Liu et al. proposed using energy score for OOD detection. The energy function maps the logit outputs to a scalar $E(\mathbf{x}; f) \in \mathbb{R}$, which is relatively lower for ID data. Note that Liu et al. used the *negative energy score* for OOD detection, in order to align with the convention that $S(\mathbf{x})$ is higher (lower) for ID (OOD) data. Energy score does not require hyperparameter tuning.

C PERFORMANCE ON DIFFERENT ARCHITECTURE

In the main paper, we have shown that DICE is competitive comparing to other discriminative-based OOD detection methods on DenseNet. In this section, we show in Table 8 that DICE is also competitive on other network architecture including ResNet-101 (He et al., 2016). For a fair comparison, all the methods use pre-trained networks *post hoc*, without regularizing with additional data. The model is trained on the in-distribution dataset CIFAR-100. All the numbers reported are averaged over six OOD test datasets described in Section 4.2. Our proposed method DICE outperforms baselines.

Method	**FPR95** ↓	**AUROC** ↑	**In-dist acc.** ↑
MSP (Hendrycks & Gimpel, 2017)	81.80	74.74	76.38
ODIN (Liang et al., 2018)	68.82	79.18	76.38
Mahalanobis (Lee et al., 2018)	88.37	67.77	76.38
Generalized-ODIN (Hsu et al., 2020)	76.28	75.24	77.63
Energy score (Liu et al., 2020)	72.38	79.34	76.38
DICE (ours)	**64.55**	**80.55**	76.38

Table 8: **Main comparison results with ResNet-101.** Comparison with competitive *post hoc* out-of-distribution detection methods. All methods are based on a discriminative model trained on **ID data only**, without using any auxiliary outlier data. ↑ indicates larger values are better and ↓ indicates smaller values are better. All values are percentages and are averaged over six OOD test datasets.

D HARDWARE

We conduct all the experiments on NVIDIA GeForce RTX 2080Ti GPUs.

E VARIANCE REDUCTION WITH CORRELATED VARIABLES

Extension of Lemma 2. We can show variance reduction in a more general case with correlated variables. The variance of output f_c without sparsification is:

$$\mathrm{Var}[f_c] = \sum_{i=1}^{m} \sigma_i^2 + 2 \sum_{1 \le i < j \le m} \mathrm{Cov}(v_i, v_j),$$

where $\mathrm{Cov}(\cdot, \cdot)$ is the covariance. The expression states that the variance is the sum of the diagonal of covariance matrix plus two times the sum of its upper triangular elements.

Similarly, the variance of output *with* directed sparsification (by taking the top units) is:

$$\mathrm{Var}[f_c^{\mathrm{DICE}}] = \sum_{i=t+1}^{m} \sigma_i^2 + 2 \sum_{t < i < j \le m} \mathrm{Cov}(v_i, v_j).$$

Therefore, the variance reduction is given by:

$$\mathrm{Var}[f_c] - \mathrm{Var}[f_c^{\mathrm{DICE}}] = \sum_{i=1}^{t} \sigma_i^2 + 2 \sum_{1 \le i < j \le m} \mathrm{Cov}(v_i, v_j) - 2 \sum_{t < i < j \le m} \mathrm{Cov}(v_i, v_j),$$

We show in Fig. 6 that the covariance matrix of unit contribution v primarily consists of elements of 0, which indicates the independence of variables by large. The covariance matrix is estimated on the CIFAR-10 model with DenseNet-101, which is consistent with our main results in Table 1.

Moreover, the summation of non-zero entries in the full matrix (i.e., the second term) is greater than that of the submatrix with top units (i.e., the third term), resulting in a larger variance reduction than in Lemma 2. In the case of OOD data (SVHN), we empirically measure the variance reduction, where $\sum_{i=1}^{t} \sigma_i^2 + 2 \sum_{1 \le i < j \le m} \mathrm{Cov}(v_i, v_j)$ equals to **6.8** and $2 \sum_{t < i < j \le m} \mathrm{Cov}(v_i, v_j)$ equals to **2.2**. Therefore, DICE leads to a significant variance reduction effect.

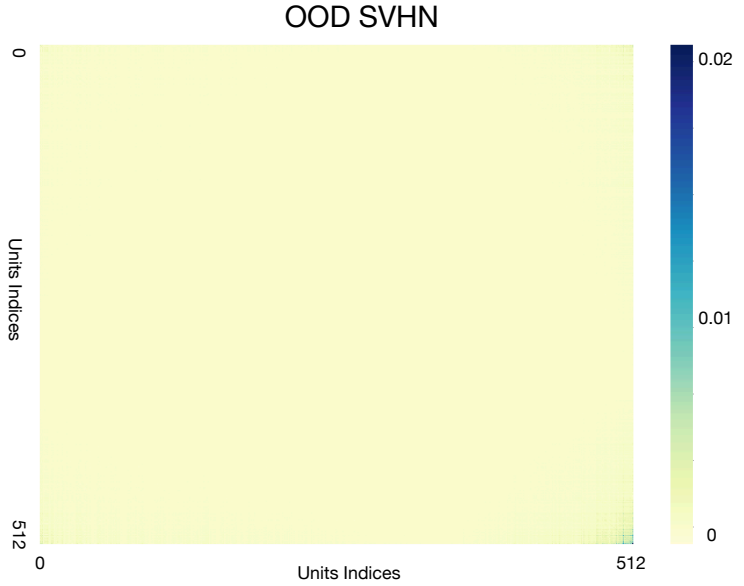

Figure 6: Covariance matrix of unit contribution estimated on the OOD dataset SVHN. Model is trained on ID dataset CIFAR-10. The unit indices are sorted from low to high, based on the expectation value of ID's unit contribution (airplane class, same as in Figure 1). The matrix primarily consists of elements with 0 value.

F Effect of DICE on MSP

Our theory shows the variance reduction effect directly in the logit output space, which is more compatible with the energy score. As a further investigation in Table 9, we find empirically that using DICE for MSP can improve the performance for MSP though it does not yield better performance than our existing results.

Method	SVHN	LSUN-c	LSUN-r	iSUN	Texture	places365	Average
MSP (Hendrycks & Gimpel, 2017)	48.25	33.80	42.37	41.42	63.99	62.57	48.73
DICE+MSP	**45.94**	**24.36**	**35.68**	**34.60**	**62.06**	**59.40**	**43.67**

Table 9: Effect of applying DICE with MSP on DenseNet101 pretrained on CIFAR-10. The number is reported in FPR95.

G Sparsification Methods with the Optimal Spasity Parameter p

In Table 2, we have shown the comparison of all sparsification methods on the same sparsity parameter $p = 0.9$. In Table 10, we perform comparison under the best sparsity p tuned for each method. Note that DICE still outperforms other methods in all test datasets.

Methods	**CIFAR-10**		**CIFAR-100**		**Imagenet**	
	FPR95 ↓	**AUROC** ↑	**FPR95** ↓	**AUROC** ↑	**FPR95** ↓	**AUROC** ↑
Weight-Droput (Wan et al., 2013)	30.98	93.63	69.00	80.12	70.49	80.34
Unit-Droput (Srivastava et al., 2014)	32.00	93.45	70.26	79.67	67.54	83.37
Weight-Pruning (Han et al., 2015)	24.25	94.84	59.93	84.22	50.31	87.70
Unit-Pruning (Li et al., 2017)	26.55	94.57	61.86	84.69	63.81	84.21
DICE (ours)	**20.83**$^{\pm 1.58}$	**95.24**$^{\pm 0.24}$	**49.72**$^{\pm 1.69}$	**87.23**$^{\pm 0.73}$	**40.90**$^{\pm 1.43}$	**91.22**$^{\pm 0.76}$

Table 10: Comparison of different *post hoc* sparsification method. All sparsification methods are based on the *same OOD scoring function* (Liu et al., 2020), with **optimal** sparsity parameters tuned for each method individually. ↑ indicates larger values are better and ↓ indicates smaller values are better. All values are percentages and are averaged over multiple OOD test datasets.

H DETAILED OOD DETECTION PERFORMANCE FOR CIFAR

We report the detailed performance for all six test OOD dataset for models trained on CIFAR10 and CIFAR-100 respectively in Table 11 and Table 12.

I DETAILS ON OOD EVALUATION DATASETS

Following (Huang & Li, 2021), we use the following list of concepts used for OOD test data (Huang & Li, 2021), including iNaturalist (Horn et al., 2018), SUN (Xiao et al., 2010), and Places365 (Zhou et al., 2017). We use the entire dataset from Textures (Cimpoi et al., 2014).

iNaturalist *Coprosma lucida, Cucurbita foetidissima, Mitella diphylla, Selaginella bigelovii, Toxicodendron vernix, Rumex obtusifolius, Ceratophyllum demersum, Streptopus amplexifolius, Portulaca oleracea, Cynodon dactylon, Agave lechuguilla, Pennantia corymbosa, Sapindus saponaria, Prunus serotina, Chondracanthus exasperatus, Sambucus racemosa, Polypodium vulgare, Rhus integrifolia, Woodwardia areolata, Epifagus virginiana, Rubus idaeus, Croton setiger, Mammillaria dioica, Opuntia littoralis, Cercis canadensis, Psidium guajava, Asclepias exaltata, Linaria purpurea, Ferocactus wislizeni, Briza minor, Arbutus menziesii, Corylus americana, Pleopeltis polypodioides, Myoporum laetum, Persea americana, Avena fatua, Blechnum discolor, Physocarpus capitatus, Ungnadia speciosa, Cercocarpus betuloides, Arisaema dracontium, Juniperus californica, Euphorbia prostrata, Leptopteris hymenophylloides, Arum italicum, Raphanus sativus, Myrsine australis, Lupinus stiversii, Pinus echinata, Geum macrophyllum, Ripogonum scandens, Echinocereus triglochidiatus, Cupressus macrocarpa, Ulmus crassifolia, Phormium tenax, Aptenia cordifolia, Osmunda claytoniana, Datura wrightii, Solanum rostratum, Viola adunca, Toxicodendron diversilobum, Viola sororia, Uropappus lindleyi, Veronica chamaedrys, Adenocaulon bicolor, Clintonia uniflora, Cirsium scariosum, Arum maculatum, Taraxacum officinale officinale, Orthilia secunda, Eryngium yuccifolium, Diodia virginiana, Cuscuta gronovii, Sisyrinchium montanum, Lotus corniculatus, Lamium purpureum, Ranunculus repens, Hirschfeldia incana, Phlox divaricata laphamii, Lilium martagon, Clarkia purpurea, Hibiscus moscheutos, Polanisia dodecandra, Fallugia paradoxa, Oenothera rosea, Proboscidea louisianica, Packera glabella, Impatiens parviflora, Glaucium flavum, Cirsium andersonii, Heliopsis helianthoides, Hesperis matronalis, Callirhoe pedata, Crocosmia × crocosmiiflora, Calochortus albus, Nuttallanthus canadensis, Argemone albiflora, Eriogonum fasciculatum, Pyrrhopappus pauciflorus, Zantedeschia aethiopica, Melilotus officinalis, Peritoma arborea, Sisyrinchium bellum, Lobelia siphilitica, Sorghastrum nutans, Typha domingensis, Rubus laciniatus, Dichelostemma congestum, Chimaphila maculata, Echinocactus texensis*

SUN *badlands, bamboo forest, bayou, botanical garden, canal (natural), canal (urban), catacomb, cavern (indoor), corn field, creek, crevasse, desert (sand), desert (vegetation), field (cultivated), field (wild), fishpond, forest (broadleaf), forest (needleleaf), forest path, forest road, hayfield, ice floe, ice shelf, iceberg, islet, marsh, ocean, orchard, pond, rainforest, rice paddy, river, rock arch, sky, snowfield, swamp, tree farm, trench, vineyard, waterfall (block), waterfall (fan), waterfall (plunge), wave, wheat field, herb garden, putting green, ski slope, topiary garden, vegetable garden, formal garden*

Places *badlands, bamboo forest, canal (natural), canal (urban), cornfield, creek, crevasse, desert (sand), desert (vegetation), desert road, field (cultivated), field (wild), field road, forest (broadleaf), forest path, forest road, formal garden, glacier, grotto, hayfield, ice floe, ice shelf, iceberg, igloo, islet, japanese garden, lagoon, lawn, marsh, ocean, orchard, pond, rainforest, rice paddy, river, rock arch, ski slope, sky, snowfield, swamp, swimming hole, topiary garden, tree farm, trench, tundra, underwater (ocean deep), vegetable garden, waterfall, wave, wheat field*

Method Type	Method	SVHN		LSUN-c		LSUN-r		iSUN		Textures		Places365		Average	
		FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑
Non-Sparse	MSP	47.24	93.48	33.57	95.54	42.10	94.51	42.31	94.52	64.15	88.15	63.02	88.57	48.73	92.46
	ODIN	25.29	94.57	4.70	98.86	3.09	99.02	3.98	98.90	57.50	82.38	52.85	88.55	24.57	93.71
	Mahalanobis	6.42	98.31	56.55	86.96	9.14	97.09	9.78	97.25	21.51	92.15	85.14	63.15	31.42	89.15
	Energy	40.61	93.99	3.81	99.15	9.28	98.12	10.07	98.07	56.12	86.43	39.40	91.64	26.55	94.57
	Generalized ODIN	6.68	98.32	17.58	95.09	36.56	92.09	36.44	91.75	35.18	89.24	73.06	77.18	34.25	90.61
Sparse	Unit-Droput	89.16	60.96	72.97	81.33	87.03	68.78	87.29	68.07	88.53	60.10	94.82	59.18	86.63	66.40
	Weight-Droput	81.34	80.03	21.06	96.15	54.70	90.33	58.88	89.80	83.34	73.31	73.42	81.10	62.12	85.12
	Unit-Pruning	40.56	93.99	3.81	99.15	9.28	98.12	10.07	98.07	56.1	86.43	39.47	91.64	26.55	94.57
	Weight-Pruning	28.61	95.40	3.01	99.30	8.58	98.19	9.08	98.16	49.45	88.20	46.78	89.77	24.25	94.84
	DICE (ours)	$25.99^{\pm 5.10}$	$95.90^{\pm 1.08}$	$0.26^{\pm 0.11}$	$99.92^{\pm 0.02}$	$3.91^{\pm 0.56}$	$99.20^{\pm 0.15}$	$4.36^{\pm 0.71}$	$99.14^{\pm 0.15}$	$41.90^{\pm 4.41}$	$88.18^{\pm 1.80}$	$48.59^{\pm 1.53}$	$89.13^{\pm 0.31}$	$20.83^{\pm 1.58}$	$95.24^{\pm 0.24}$

Table 11: Detailed results on six common OOD benchmark datasets: `Textures` (Cimpoi et al., 2014), `SVHN` (Netzer et al., 2011), `Places365` (Zhou et al., 2017), `LSUN-Crop` (Yu et al., 2015), `LSUN-Resize` (Yu et al., 2015), and `iSUN` (Xu et al., 2015). For each ID dataset, we use the same DenseNet pretrained on **CIFAR-10**. ↑ indicates larger values are better and ↓ indicates smaller values are better.

Method Type	Method	SVHN		LSUN-c		LSUN-r		iSUN		Textures		Places365		Average	
		FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑
Non-Sparse	MSP	81.70	75.40	60.49	85.60	85.24	69.18	85.99	70.17	84.79	71.48	82.55	74.31	80.13	74.36
	ODIN	41.35	92.65	10.54	97.93	65.22	84.22	67.05	83.84	82.34	71.48	82.32	76.84	58.14	84.49
	Mahalanobis	22.44	95.67	68.90	86.30	23.07	94.20	31.38	93.21	62.39	79.39	92.66	61.39	50.14	85.03
	Energy	87.46	81.85	14.72	97.43	70.65	80.14	74.54	78.95	84.15	71.03	79.20	77.72	68.45	81.19
	Generalized ODIN	36.74	93.51	43.15	89.55	40.31	92.61	37.41	93.05	64.26	76.72	95.33	65.97	52.87	85.24
Sparse	Unit-Droput	91.43	54.71	56.24	85.25	91.06	57.79	90.88	57.90	89.59	54.57	94.15	56.15	85.56	61.06
	Weight-Droput	92.97	64.39	18.96	95.62	88.67	65.48	87.12	67.82	88.45	64.38	88.69	71.87	77.48	71.59
	Unit-Pruning	87.52	81.83	14.73	97.43	70.62	80.18	74.46	79.00	84.20	71.02	79.32	77.70	68.48	81.19
	Weight-Pruning	77.99	84.14	5.17	99.05	59.42	87.13	61.80	86.09	72.68	73.85	82.53	75.06	59.93	84.22
	DICE (ours)	$54.65^{\pm 4.94}$	$88.84^{\pm 0.39}$	$0.93^{\pm 0.07}$	$99.74^{\pm 0.01}$	$49.40^{\pm 1.99}$	$91.04^{\pm 1.49}$	$48.72^{\pm 1.55}$	$90.08^{\pm 1.36}$	$65.04^{\pm 0.66}$	$76.42^{\pm 0.35}$	$79.58^{\pm 2.34}$	$77.26^{\pm 1.08}$	$49.72^{\pm 1.69}$	$87.23^{\pm 0.73}$

Table 12: Detailed results on six common OOD benchmark datasets: `Textures` (Cimpoi et al., 2014), `SVHN` (Netzer et al., 2011), `Places365` (Zhou et al., 2017), `LSUN-Crop` (Yu et al., 2015), `LSUN-Resize` (Yu et al., 2015), and `iSUN` (Xu et al., 2015). For each ID dataset, we use the same DenseNet pretrained on **CIFAR-100**. ↑ indicates larger values are better and ↓ indicates smaller values are better.