Preventing Arbitrarily High Confidence on Far-Away Data in Point-Estimated Discriminative Neural Networks

Ahmad Rashid[1,4] **Serena Hacker**[2] **Guojun Zhang**[3]

Agustinus Kristiadi[4] **Pascal Poupart**[1,4]

University of Waterloo[1] University of Toronto[2] Huawei Noah's Ark Lab[3] Vector Institute[4]

Abstract

Discriminatively trained, deterministic neural networks are the *de facto* choice for classification problems. However, even though they achieve state-of-the-art results on in-domain test sets, they tend to be overconfident on out-of-distribution (OOD) data. For instance, ReLU networks—a popular class of neural network architectures—have been shown to almost always yield high confidence predictions when the test data are far away from the training set, even when they are trained with OOD data. We overcome this problem by adding a term to the output of the neural network that corresponds to the logit of an extra class, that we design to dominate the logits of the original classes as we move away from the training data. This technique *provably* prevents arbitrarily high confidence on far-away test data while maintaining a simple discriminative point-estimate training. Evaluation on various benchmarks demonstrates strong performance against competitive baselines on both far-away and realistic OOD data.

1 INTRODUCTION

Machine learning has made substantial progress over the last decade with the help of a strong deep learning toolkit, larger data sets, better optimization algorithms, faster and cheaper computation, and a vibrant research community. As machine learning systems continue to be deployed in safety-critical applications, important questions around their robustness and uncertainty quantification continue to be asked. A common expectation in uncertainty quantification is to assign high confidence to test cases close to the training data and low confidence to test cases that are out-of-distribution (OOD).

Recent advances in machine learning are in part due to deep neural networks (DNNs), which are powerful function approximators. However, DNN classifiers tend to be overconfident for both in-domain examples (Guo et al., 2017) and data that is far away from the training examples (Nguyen et al., 2015). Hein et al. (2019) showed that the ubiquitous ReLU Networks almost always exhibit high confidence on samples that are far away from the training data.

A number of methods have been proposed to deal with the overconfidence issue in DNNs. Calibration methods attempt to solve overconfidence of neural network classifiers by various methods including smoothing the softmax distribution (Guo et al., 2017; Gupta et al., 2020; Kull et al., 2019), regularization (Müller et al., 2019; Thulasidasan et al., 2019) and adding additional constraints to the loss function (Kumar et al., 2018; Lin et al., 2017). These methods, however, do not resolve overconfidence issues around OOD data (Minderer et al., 2021). Other methods, both Bayesian (Blundell et al., 2015; Gal and Ghahramani, 2016; Kristiadi et al., 2020) and non-Bayesian (Lakshminarayanan et al., 2017; Mukhoti et al., 2023) have improved OOD detection while training only with the in-domain data distribution.

State-of-the-art methods for OOD detection are typically trained with additional OOD training data with the goal for the classifier to output either high "None" class probability (Zhang and LeCun, 2017; Kristiadi et al., 2022b) or uniform confidence (Hendrycks et al., 2018), in the presence of OOD samples. Hein et al. (2019) showed that there is no guarantee that OOD data would be predicted as the "None" class. More-

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

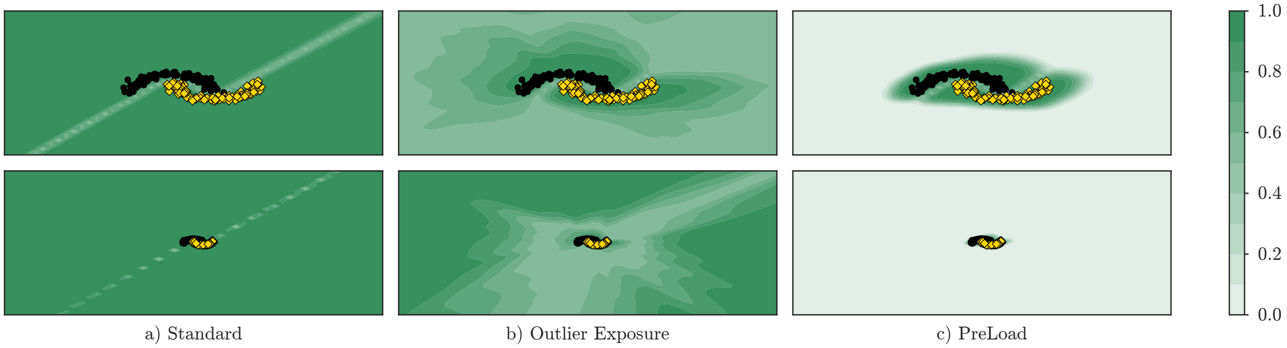

Figure 1: An illustrative example of the confidence of different methods trained on a synthetic binary classification dataset. The shades of green display the confidence of each algorithm with a darker shade signifying a higher confidence. The bottom row gives a zoomed-out view.

over, we will demonstrate that these methods still exhibit high confidence when the test points are far away from the data.

One way of overcoming the problem of arbitrarily high confidence on far-away data is to incorporate generative modeling, either as a *posthoc* method (Lee et al., 2018; Mukhoti et al., 2023) or as a prior on the data (Meinke and Hein, 2020), into a neural network. The former assumes that the neural network embedding can be approximated with a Gaussian distribution. However, on realistic, OOD data, we will demonstrate that these methods are not competitive with the state-of-the-art. The latter assumes a generative model over the data which is a harder problem than the underlying discriminative modeling.

Finally, while Bayesian neural networks (Louizos and Welling; Kristiadi et al., 2022a) have also been used to overcome this issue, they are not guaranteed to obtain the optimal confidences on far-away OOD test data (Kristiadi et al., 2020). While a more sophisticated remedy exists for this (Kristiadi et al., 2021), they are specifically constructed to only fix the far-away high confidence, and their detection performance on 'nearby' OOD data are more of an afterthought.

In this work, we present our method, called **Pr**oducing Larg**er** **Lo**gits **a**way from **D**ata, or **PreLoad**, which fulfills the following desiderata: (i) it must maintain the simplicity of the standard discriminative training procedure for DNNs (unlike generative- and Bayesian-based methods), (ii) it must provably be less confident on inputs far away from the training data, and (iii) it must perform well on realistic OOD examples (e.g. CIFAR-10 vs. CIFAR-100).

We accomplish this by training an extra class, such that under an OOD input, this extra logit is larger than the logits of the other classes as we move farther away from the training data. This construction provably helps PreLoad almost always predict far-away data as OOD. Furthermore, the extra class is trained on an auxiliary, OOD dataset, which helps it detect realistic, nearby OOD examples well.

Figure 1 illustrates the confidence level of PreLoad as we move away from the training data, compared to a standard-trained neural network and a discriminative OOD training approach called Outlier Exposure (OE, Hendrycks et al., 2018). Standard neural networks with a softmax output layer exhibit high confidence as we move away from the decision boundary. OE's confidence initially decreases away from the data, but it becomes high far away as we zoom out. In contrast, PreLoad is confident when close to the data and uncertain when away from it.

2 PRELIMINARIES

We define a neural network as a function $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^k$ with $(x, \theta) \mapsto f_\theta(x)$, where $\mathbb{R}^n$ is the input space, $\mathbb{R}^k$ the output space, and $\mathbb{R}^p$ the parameter space. Let $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^m$ be a training dataset. The standard way of training a neural network is by finding optimal parameters θ^* such that $\theta^* = \arg\min_\theta \sum_{i=1}^m \ell(f_\theta(x_i), y_i)$ for some loss function ℓ such as the cross-entropy loss for classification.

One of the most widely used neural network architectures is a ReLU network. We use the term ReLU networks for feedforward neural networks with piecewise affine activation functions, such as the ReLU or leaky ReLU activation, and a linear output layer. ReLU networks can be written as continuous piecewise affine functions (Arora et al., 2018; Hein et al., 2019).

Definition 1. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called piecewise affine if there exists a set of polytopes $\{Q_r\}_{r=1}^M$ such that their union is $\mathbb{R}^n$ and f is affine in each polytope (Arora et al., 2018; Hein et al., 2019).

Piecewise affine functions include networks with fully connected layers, convolution layers, residual layers, skip connections, average pooling, and max pooling. We will rely on the neural network being a continuous piecewise affine function to prove that our algorithm prevents arbitrarily high confidence on far-away data.

Consider a classification problem where x is the input and $y \in \{1, \cdots, k\}$ denotes the target class. A neural network with a linear output layer in conjunction with the softmax link function can be used to compute the probability $P(y|x)$. More precisely, consider the following decomposition of the neural network $f_\theta(x) = WG_\psi(x)$ where $W \in \mathbb{R}^{k\times d}$ is the weight matrix for the last layer, $G_\psi(x) \in \mathbb{R}^d$ is the neural network embedding and $\theta = \{\psi, W\}$. Each row of f corresponds to the logit $z_c(x)$ of class c:

$$z_c(x) = w_c^\top G_\psi(x) + b_c. \tag{1}$$

Then the last layer computes class probabilities via a softmax such that:

$$P(y = c|x) = \frac{\exp(w_c^\top G_\psi(x) + b_c)}{\sum_{c'=1}^{k} \exp(w_{c'}^\top G_\psi(x) + b_{c'})} \tag{2}$$

where $w_c \in \mathbb{R}^d$ and $b_c \in \mathbb{R}$ are the parameters of the last layer associated with class $c \in \{1, \cdots, k\}$.

Generally, learning $P(y|x)$ is referred to as discriminative modeling. Generative models, such as GANs (Goodfellow et al., 2014) and VAEs (Kingma and Welling, 2013) learn the distribution of the data $P(x)$. Meanwhile, class-conditional generative models (Mukhoti et al., 2023) learn $P(x|y)$.

2.1 Arbitrarily High Confidence on Far-Away Data

Arbitrarily high confidence on far-away data i.e. data which is far away from the training set (Hein et al., 2019), can be formalized as observing that the probability of some class approaches 1 in the limit of moving infinitely far from the training data.

Definition 2. A model exhibits far-away arbitrarily high confidence if there exists $x \in \mathbb{R}^n$ and $c \in \{1, \cdots, k\}$ such that

$$\lim_{t\to\infty} P(y = c|tx) = 1. \tag{3}$$

Hein et al. (2019) showed that piecewise affine networks (including ReLU networks) with a linear last layer almost always exhibit arbitrarily high confidence far away from the training data.

3 METHODOLOGY

Consider a neural network, $f_{\psi,W}$, trained on a k-class classification problem such that the logit z_c is defined according to (1) and $P(y|x)$ is computed according to (2). Arbitrarily high confidence arises when the logit of one class becomes infinitely higher than the logits of the other classes:

Lemma 3. *Let $P(y|x)$ be a classifier defined in (2) and let $x \in \mathbb{R}^n$. If the classifier exhibits arbitrarily high confidence on far-away inputs (i.e., $\lim_{t\to\infty} P(y|tx) = 1$), then there must exist $c \in \{1, \dots, k\}$ such that $\lim_{t\to\infty} z_c(tx) - z_{c'}(tx) = +\infty$ for all $c' \neq c$.*

Proof. From (2), if $\lim_{t\to\infty} P(y = c|tx) = 1$, then we have:

$$\lim_{t\to\infty} \frac{\exp(z_c(tx))}{\sum_{c'} \exp(z_{c'}(tx))} = 1.$$

Therefore,

$$\begin{aligned}
&\lim_{t\to\infty} \frac{\exp(z_c(tx))}{\sum_{c'} \exp(z_{c'}(tx))} = 1\\
\implies &\lim_{t\to\infty} \frac{\sum_{c'} \exp(z_{c'}(tx))}{\exp(z_c(tx))} = 1\\
\implies &\lim_{t\to\infty} 1 + \sum_{c'\neq c} \frac{\exp(z_{c'}(tx))}{\exp(z_c(tx))} = 1\\
\implies &\lim_{t\to\infty} \sum_{c'\neq c} \exp(z_{c'}(tx) - z_c(tx)) = 0\\
\implies &\lim_{t\to\infty} \exp(z_{c'}(tx) - z_c(tx)) = 0 \quad \forall c' \neq c.
\end{aligned}$$

Thus, we can conclude that $\lim_{t\to\infty} z_c(tx) - z_{c'}(tx) = +\infty$ for all $c' \neq c$. □

An immediate consequence of the above lemma is that networks with normalization such as layernorm do not suffer from far-away arbitrarily high confidence since the layers that follow layernorm (including the logits) will remain bounded. Note that networks with batchnorm may still exhibit far-away arbitrarily high confidence since batchnorm ensures that the logits of the training set are bounded, but not necessarily the logits of the test set, which may include OOD data that could be arbitrarily far.

Our solution consists of creating an additional, $(k+1)$-st class such that the confidence in the original classes vanishes far away from the training data while the confidence in the $(k+1)$-st class becomes arbitrarily high. Note that this is desirable since the extra class represents the OOD class. Based on Lemma 3, we achieve this by making sure that the corresponding logit z_{k+1} is infinitely higher than the logits $z_{c\in\{1,\dots,k\}}$ of the other classes far away from the training data. More precisely, let

$$z_{k+1}(x) = w_{k+1}^\top G_\psi(x)^2 + b_{k+1} \tag{4}$$

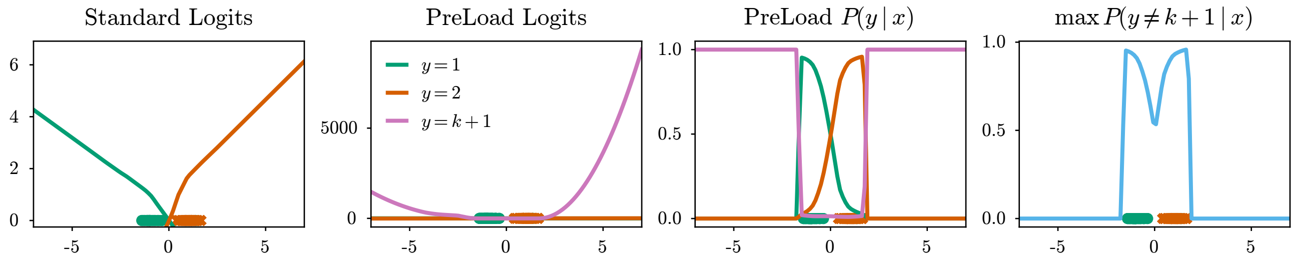

Figure 2: Effect of training with an OOD class with our method on a 1-D binary classification problem. Standard logits keep on growing when away from the data. We implement the OOD class such that the logits grow much faster for the OOD class compared to the in-domain class. This 'fixes' the probabilities and the confidence away from the dataset. Note that the range of y values is larger on the second plot.

where the weights $w_{k+1} \in \mathbb{R}^d_{>0}$ are restricted within the positive orthant and $G_\psi(x)^2$ is the component-wise square of the network embedding $G_\psi(x)$.

Then, given these logits, classification is performed as follows:

$$P(y = c|x) = \frac{1}{A(x)} \exp(w_c^\top G_\psi(x) + b_c) \qquad (5)$$

for $c \in \{1, \ldots, k\}$, and

$$P(y = k+1|x) = \frac{1}{A(x)} \exp(w_{k+1}^\top G_\psi(x)^2 + b_{k+1}), \quad (6)$$

where

$$\begin{aligned} A(x) := \sum_{c'=1}^{k} & \exp(w_{c'}^\top G_\psi(x) + b_c) \\ & + \exp(w_{k+1}^\top G_\psi(x)^2 + b_{k+1}) \end{aligned} \qquad (7)$$

is the softmax's denominator.

Intuitively, as we move away from the training data the magnitude of $G_\psi(x)$ may also increase, which may result in some class c dominating with arbitrarily high confidence (Hein et al., 2019). However, by using $G_\psi(x)^2$ in the logit of the $(k+1)$-st class we make sure that it grows faster than other logits and therefore, eventually dominates. In Theorem 4, we prove that any classification network augmented with such a construction never exhibits far-away arbitrarily high confidence in classes $\{1, ..., k\}$. Note that the theorem holds if the exponent of the term $G_\psi(x)$ in (4) is replaced with another even integer greater than 2.

Theorem 4. *Let G_ψ be any neural network embedding used for classification according to (5) and (6). Let w_c and b_c be finite weights and biases in the penultimate classification layer for each class c. Let $tx_* \in \mathbb{R}^n$ be a test input with magnitude regulated by t. Then $\lim_{t\to\infty} P(y = c|tx_*) < 1$ for all $c \neq k+1$.*

Proof. Based on Lemma 3, arbitrarily high confidence (i.e., $\lim_{t\to\infty} P(y = c|tx) = 1$) arises when there is a c such that $\lim_{t\to\infty} z_c(tx) - z_{c'}(tx) = \infty \ \forall c' \neq c$. We prove by contradiction that this cannot happen once we introduce the extra class with its logit as defined in (4). Consider two cases:

1. Suppose that there exists a class $c \neq k+1$ such that $\lim_{t\to\infty} z_c(tx) = \infty$ and for all $c' \neq c$, $\lim_{t\to\infty} z_c(tx) - z_{c'}(tx) = \infty$. Since $z_c(tx) = w_c^\top G_\psi(x) + b_c$, and the weights and biases are finite, then $\lim_{t\to\infty} z_c(tx) = \infty$ implies that $\lim_{t\to\infty} \|G_\psi(tx)\| = \infty$. Since $z_{k+1}(tx) = w_{k+1}^\top G_\psi(x)^2 + b_{k+1}$ where $w_{k+1} \in \mathbb{R}^d_{>0}$, i.e. each component of w_{k+1} is positive, and $G_\psi(x)^2$ is always component-wise positive, then $\lim_{t\to\infty} z_{k+1}(tx) = \infty$ and $\lim_{t\to\infty} z_{k+1}(tx) > \lim_{t\to\infty} z_c(tx)$, which contradicts the assumption that $\lim_{t\to\infty} z_c(tx) - z_{k+1}(tx) = \infty$.

2. Suppose that there exists a class $c \neq k+1$ such that $\lim_{t\to\infty} z_c(tx) < \infty$ and for all $c' \neq c$, $\lim_{t\to\infty} z_c(tx) - z_{c'}(tx) = \infty$. Since $z_{k+1} = w_{k+1}^\top G_\psi(x)^2 + b_{k+1}$, $w_{k+1} \in \mathbb{R}^d_{>0}$, $G_\psi(x)^2 > 0$ and $b_{k+1} < \infty$, then $\lim_{t\to\infty} z_{k+1}(tx) > -\infty$, which contradicts the assumption that $\lim_{t\to\infty} z_c(tx) - z_{k+1}(tx) = \infty$.

Altogether, they imply the desired result. □

The above theorem guarantees that arbitrarily high confidence will not occur for any neural network with an extra class that we propose. In addition, we show a stronger result in Theorem 6 for ReLU classification networks. As we move far away from the training data, we show that the confidence in the original classes (i.e., $c \in \{1, ..., k\}$) will be dominated by the extra class. To prove this, we first recall an important lemma from Hein et al. (2019) about ReLU networks.

Lemma 5 (Hein et al., 2019)**.** *Let $\{Q_r\}_{r=1}^R$ with $\mathbb{R}^n = \cup_{r=1}^R Q_r$ be a set of linear regions associated with a ReLU network $G_\psi : \mathbb{R}^n \to \mathbb{R}^d$. For any $x \in \mathbb{R}^n$ there exists an $\alpha \in \mathbb{R}_{>0}$ and $r \in \{1, \cdots, R\}$ such that for all $t \geq \alpha$, we have $tx \in Q_r$.* □

This lemma tells us that as we move far away from the data region via scaling an input $x \in \mathbb{R}^n$ with a nonnegative scalar, at some point we can represent the ReLU network with just an affine function. It follows that in this case, increasing the scaling factor makes the magnitude of the network's output larger. We will use this fact in our main theoretical result.

Theorem 6. *Let $G_\psi(x)$ be a ReLU network embedding used for classification according to (5) and (6) with a piecewise affine representation $G_\psi|_{Q_r}(x) = V_\psi^r x + a_\psi^r$ on the linear regions $\{Q_r\}_{r=1}^R$, where $V_\psi^r \in \mathbb{R}^{d\times n}$ and $a_\psi^r \in \mathbb{R}^d$. Suppose $V_\psi^r x$ does not contain identical rows for all $r = 1, \ldots, R$. Then for almost any input $x_* \in \mathbb{R}^n$, we have $\lim_{t\to\infty} \arg\max_{c=1,\ldots,k+1} P(y = c|tx_*) = k + 1$.*

Proof. First, since the coefficients $w_{k+1} \in \mathbb{R}_{>0}^d$ are constrained to be component-wise positive, the logit $w_{k+1}^\top G_\psi(x_*)^2$ of the additional $(k+1)$-st class is always positive. Second, by Lemma 5, there exists $\alpha > 0$ s.t. for all $t \geq \alpha$, the ReLU network is represented by a single affine function $G_\psi(tx_*) = V_\psi t x_* + a_\psi$. Therefore, as $t \to \infty$, the norm $\|G_\psi(tx_*)^2\|$ of $G_\psi(tx_*)^2 = (tV_\psi x_* + a_\psi)^2$ also tends to infinity (recall that we use the notation $(\cdot)^2$ on vectors as component-wise square).

Now, notice that we can write $P(y = k+1|x = tx_*)$ as:

$$\frac{e^{w_{k+1}^\top G_\psi(tx_*)^2+b_{k+1}}}{\sum_{c'=1}^k e^{w_{c'}^\top G_\psi(tx_*)+b_{c'}} + e^{w_{k+1}^\top G_\psi(tx_*)^2+b_{k+1}}} \tag{8}$$

which is equal to:

$$\frac{1}{1+\sum_{c'=1}^k e^{w_{c'}^\top G_\psi(tx_*)+b_{c'}-w_{k+1}^\top G_\psi(tx_*)^2-b_{k+1}}} \tag{9}$$

Recall that $\lim_{t\to\infty} \|G_\psi(tx_*)\| \to \infty$. Moreover, $\|G_\psi(tx_*)^2\|$ grows even faster. So, as $t \to \infty$ we can see from the expression above that $P(y = k+1|x = tx_*) = \frac{1}{1+k\exp(-\infty)} = 1$. This immediately implies that the class $k+1$ achieves the maximum softmax probability since probability vectors sum to one. Moreover, the in-distribution classes $\{1, \ldots, k\}$ have the probabilty zero. □

Figure 2 shows the effect of our method on the prediction and confidence of a neural network classifier on a one-dimensional binary classification toy dataset. We can observe that the standard logits keep on increasing as we move away from the data. Therein we train an extra class using uniform noise and observe that as we move away from the training data the logits of the extra class dominate. This 'fixes' the neural network prediction and confidence away from the training data as far-away inputs are predicted as the extra class.

Algorithm 1 PreLoad Algorithm

```
Input:
    Training Set $\mathcal{D}_{\text{in}} := \{(x_i \in \mathbb{R}^n, y_i \in \{1, \cdots, k\})\}$
    OOD Training Set $\mathcal{D}_{\text{ood}} := \{(x'_i \in \mathbb{R}^n)\}$
    Neural network $f_\theta$ with $\theta = \{\psi, W\}$, number of iterations $T$, learning rate $\eta$
for $i \leftarrow 1$ to $T$ do
    Sample a mini-batch S from $\mathcal{D}_{in} : S = \{x_i, y_i\}_{i=1}^m$
    Sample a mini-batch S' from $\mathcal{D}_{ood} : S' = \{x'_i\}_{i=1}^m$
    Compute the objective function $\mathcal{R}$ such that
    $\mathcal{R}(f_{\psi,W}(x)) = \mathbb{E}_S \, \mathcal{L}_{CE}(f_{\psi,w_{k\in\{1,\cdots,K\}}}(x_i), y_i)$
        $+\lambda \mathbb{E}_{S'} \mathcal{L}_{CE}(f_{\psi,w_{k+1}}(x'_i), k+1)$
    Update the parameters $\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{R}(f_{\psi,W}(x))$
end for
Predict OOD: is_ood = $(\arg\max_c P(y = c|x_*) == k+1)$ for test sample $x_*$
```

In order to train the extra class we rely on an auxiliary OOD dataset like previous methods (Hendrycks et al., 2018; Meinke and Hein, 2020). Such methods tend to demonstrate strong performance on OOD detection on standard benchmarks. Our overall training objective is as follows:

$$\begin{aligned}\mathcal{R}(f_{\psi,W}(x)) &= \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{in}}} \mathcal{L}_{\text{CE}}(f_{\psi,w_{k\in\{1,\cdots,K\}}}(x), y) \\ &+ \lambda \mathbb{E}_{(x')\sim\mathcal{D}_{\text{ood}}} \mathcal{L}_{\text{CE}}(f_{\psi,w_{k+1}}(x'), k+1).\end{aligned}$$

Here $\mathcal{L}_{CE}$ is the cross-entropy loss and λ controls the relative weight between the loss on in-domain and OOD training inputs. Algorithm 1 shows the training procedure for PreLoad. Note that we have the option of either training our method from scratch or fine-tuning after a neural network has been trained on in-domain data, similar to Hendrycks et al. (2018).

4 RELATED WORKS

Gaussian Assumption. Some recent works on OOD detection have assumed that the embedding, $G_\psi(x)$, produced by the penultimate layer of a neural network is Gaussian, and have built algorithms based on this. Lee et al. (2018) propose fitting class conditional Gaussians on $G(x)$ such that $p(G_\theta(x)|y = c) = \mathcal{N}(G_\theta(x|\mu_c, \Sigma)$ where Σ is a diagonal covariance matrix. The mean and covariance are empirically estimated from the class-wise neural network embeddings of the training data. The method computes a confidence score, for a test sample x_i, based on the Mahalanobis distance from the class-conditional Gaussians. Mukhoti et al. (2023) go further by fitting a Gaussian Mixture Model (GMM) on $G_\psi(x)$. Thereafter, they use the GMM density as the OOD score for a test sample x_i.

Even though these methods are deterministic and prevent arbitrarily high confidence on far-away data, they assume that $G_\psi(x)$ follows a Gaussian or mixture of Gaussian distribution. Moreover, they need to adjust a confidence threshold for each dataset and require an additional step beyond standard discriminative training to fit the Gaussian or mixture of Gaussians.

OOD training. Zhang and LeCun (2017) presented the concept of a "None" class or an additional class for a supervised learning problem, which, is trained on unlabeled data for regularization of a DNN to improve generalization. Kristiadi et al. (2022b) adapted this method to OOD detection such that they train an additional output of a neural network to predict a "None" class. The linear layer weights corresponding to the "None" class, w_{k+1} are trained on an additional OOD data set which is carefully selected to remove any overlap with the training set. Even though using an OOD set may not be ideal, Kristiadi et al. (2022b) demonstrated that these methods show state-of-the-art performance. Hein et al. (2019) showed that theoretically, "None" class methods are prone to arbitrarily high confidence on far-away data.

Outlier Exposure (OE, Hendrycks et al., 2018) also relies on OOD data, but, instead of learning an extra class, trains the class probabilities, $P(y|x)$ to output a uniform distribution when the data is OOD. Distributional-agnostic Outlier Exposure (DOE, Wang et al., 2023), is a variant of OE, which, uses model perturbation to generate "worst-case" OOD data and applies the OE algorithm on these data. We will demonstrate in the results that these methods also fail in the presence of far-away data. Meinke and Hein (2020) present an algorithm that models a joint probability distribution, $P(x, y)$ over both the in-distribution and OOD data. Using this, they jointly train a neural network that models the predictive distribution and two GMMs that model the generative distribution for in-domain and OOD data. Similar to OE, the neural network is trained to output uniform probabilities for OOD data. This algorithm has some provable guarantees on far-away OOD detection and reaches close to the performance of OE on standard benchmarks. Our algorithm, however, can achieve that without any generative modeling on either in-domain or OOD data.

An alternative to relying on the softmax for confidence is using an energy function. Liu et al. (2020) propose a fine-tuning algorithm that combines an energy-based loss function with the standard cross-entropy loss. This additional loss uses two additional margin hyper-parameters, $\{m_{in}, m_{ood}\}$, and penalizes in-domain samples which produce energy higher than m_{in} and OOD samples which produce energy lower than m_{ood}. They only rely on discriminative training but do not prevent arbitrarily high confidence on far-away data, which, we will demonstrate in the results.

5 EXPERIMENTS

We evaluate our algorithm, PreLoad,[1] in three ways. First, we evaluate on synthetic far-away data to validate our theoretical results. Then, we evaluate on standard benchmarks which measure OOD detection performance on realistic data. Finally, we evaluate the calibration of our model under dataset shift.

Datasets. Our in-domain datasets include MNIST (LeCun et al., 1998), Fashion MNIST (FMNIST) (Xiao et al., 2017), SVHN (Netzer et al., 2011), CIFAR10 and CIFAR100 (Krizhevsky et al., 2009). We train LeNet for MNIST and FMNIST and WideResNet-16-4 (Zagoruyko and Komodakis, 2016) for SVHN, CIFAR10 and CIFAR100. The OOD training set for the methods which rely on OOD training, including ours, is $300,000$ random images as released by Hendrycks et al. (2018) as 80 million tiny images (Torralba et al., 2008) is no longer available.

Metrics. Following convention, we define an in-domain sample as positive and an OOD sample as negative. The true positive rate (TPR) is $\text{TPR} = \frac{\text{TP}}{\text{TP+FN}}$ and the false positive rate (FPR) is $\text{FPR} = \frac{\text{FP}}{\text{FP+TN}}$, where TP, FN, FP and TN are true positive, false negative, false positive and true negative respectively. We report our results on FPR-95 with further results on AUROC and calibration in Supplementary Section B. FPR-95 is the FPR when the TPR is 95%. The metric can be interpreted as the probability that a negative sample will be classified as positive when 95% of samples are correctly classified as positive. A lower score is better.

Baselines. We compare PreLoad against baseline methods in two settings: trained from scratch and fine-tuned. In the former, we compare against a DNN trained on in-domain data, referred to as Standard, OOD training baselines including a "None" class method (Kristiadi et al., 2022b) referred to as NC, Outlier Exposure (Hendrycks et al., 2018) referred to as OE and a generative modeling baseline (Mukhoti et al., 2023) referred to as DDU. All the methods are developed starting from identical neural network architectures and we select optimal hyper-parameters for

[1] https://github.com/serenahacker/PreLoad

Table 1: OOD data detection using the FPR-95 metric when the OOD data is far away from the training data. We present the average result of 5 runs with error bars. Lower numbers are better.

Dataset	**Trained from Scratch**					**Finetuned**			
	Standard	**DDU**	**NC**	**OE**	**PreLoad**	**OE-FT**	**DOE-FT**	**Energy-FT**	**PreLoad-FT**
MNIST									
FarAway	100.0±0.0	**0.0±0.0**	**0.0±0.0**	56.6±19.6	**0.0±0.0**	99.0±0.4	56.8±18.1	100.0±0.0	**0.0±0.0**
FarAway-RD	99.9±0.0	**0.0±0.0**	99.9±0.1	99.8±0.0	**0.0±0.0**	99.5±0.1	99.8±0.1	100.0±0.0	**0.0±0.0**
F-MNIST									
FarAway	100.0±0.0	**0.0±0.0**	53.5±22.5	100.0±0.0	**0.0±0.0**	100.0±0.0	99.6±0.4	38.4±8.9	**0.0±0.0**
FarAway-RD	100.0±0.0	**0.0±0.0**	100.0±0.0	100.0±0.0	**0.0±0.0**	100.0±0.0	100.0±0.0	81.6±8.8	**0.0±0.0**
SVHN									
FarAway	99.4±0.2	**0.0±0.0**	80.0±20.0	99.4±0.4	**0.0±0.0**	99.3±0.3	99.9±0.1	100.0±0.0	**0.0±0.0**
FarAway-RD	99.8±0.1	**0.0±0.0**	80.0±20.0	85.4±7.6	**0.0±0.0**	93.1±2.5	99.3±0.6	100.0±0.0	**0.0±0.0**
CIFAR-10									
FarAway	100.0±0.0	**0.0±0.0**	20.0±20.0	100.0±0.0	**0.0±0.0**	100.0±0.0	100±0.0	100.0±0.0	**0.0±0.0**
FarAway-RD	99.7±0.2	**0.0±0.0**	40.0±24.5	100.0±0.0	**0.0±0.0**	99.5±0.3	99.6±0.4	100.0±0.0	**0.0±0.0**
CIFAR-100									
FarAway	100.0±0.0	**0.0±0.0**	20.0±20.0	100.0±0.0	**0.0±0.0**	100.0±0.0	100.0±0.0	100.0±0.0	**0.0±0.0**
FarAway-RD	100.0±0.0	**0.0±0.0**	20.0±20.0	100.0±0.0	**0.0±0.0**	100.0±0.0	100.0±0.0	100.0±0.0	**0.0±0.0**

Table 2: OOD detection results on image classification data reporting the FPR-95 metric. The results are averaged over 6 OOD test sets and five runs for each instance. Lower numbers are better.

Dataset	**Trained from Scratch**					**Finetuned**			
	Standard	**DDU**	**NC**	**OE**	**PreLoad**	**OE-FT**	**DOE-FT**	**Energy-FT**	**PreLoad-FT**
MNIST	10.9±2.3	47.7±6.9	**3.3±1.2**	**5.5±1.9**	6.6±2.0	**4.7±1.7**	**4.3±1.6**	**8.4±2.4**	**8.4±2.5**
F-MNIST	70.6±4.0	35.1±7.7	**2.2±0.5**	31.7±5.5	**2.3±0.6**	31.7±5.9	20.7±3.9	**14.5±2.9**	**12.4±2.3**
SVHN	23.7±1.3	8.0±1.6	**2.1±0.9**	**1.7±0.7**	**1.1±0.5**	**1.6±0.6**	**1.3±0.5**	7.2±0.9	**0.8±0.3**
CIFAR10	51.1±3.6	38.0±4.8	**5.7±1.7**	11.7±2.1	**6.0±1.8**	20.0±2.8	**15.1±2.5**	**15.6±3.6**	**12.0±2.5**
CIFAR100	77.2±1.9	66.0±6.4	**27.5±5.2**	60.2±4.1	**25.9±4.8**	70.6±2.5	54.3±4.4	49.4±4.6	**39.5±5.2**

PreLoad based on maximizing in-distribution validation accuracy. Standard, OE, NC, and PreLoad are all trained for 100 epochs from scratch. DDU trains a Gaussian Mixture Model over the Standard method for OOD detection. Finetuned (FT) baselines include OE (OE-FT), DOE (DOE-FT) and Energy-FT (Liu et al., 2020). PreLoad-FT and the FT baselines are initialized from a Standard model and are fine-tuned over 10 epochs using the respective losses. Training details can be found in Supplementary Section A.1.

5.1 Far-Away Data

We present results on two types of far-away data: FarAway (Hein et al., 2019) and FarAway Random Direction (FarAway-RD). A FarAway sample s is defined as $s = tu_x$, where u_x has the shape of a training sample x and contains values sampled from a uniform distribution on the interval [0, 1) and t is some constant. On the other hand, a FarAway-RD sample s' can be defined as $s' = u_x + tv$, where u_x and t are as previously defined and v is a sample from the unit sphere such that $\|v\| = 1$. As the name suggests, Faraway-RD can scale the data in a random direction. For all the experiments we have fixed $t = 10,000$. Note that far-away data defined as such is unbounded, i.e., in $\mathbb{R}^n$, whereas realistic images are in $[0, 1]^n$. In the subsequent section we present results on realistic images.

Table 1 shows the FPR-95 metric when we evaluate on the two types of far-away data for models trained on each dataset. We observe that both versions of our method, PreLoad, and PreLoad-FT, achieve perfect FPR-95 of 0 on all the datasets on both types of far-away data. The Standard method is the worst with OE and DOE-FT also doing poorly. Energy-FT, which incorporates an energy function into the loss also does not do well in this setting. NC performs better in some scenarios such as on FarAway on MNIST but generally has high variability between different runs. DDU, which trains a Gaussian Mixture Model on top of the neural network embedding, unsurprisingly, achieves perfect results as well.

5.2 OOD Benchmarks

Next, we present our results on standard OOD benchmarks which evaluate a more realistic scenario for the evaluation of an image classifier. Models trained on MNIST and F-MNIST are evaluated on each other and E-MNIST (Cohen et al., 2017), K-MNIST (Clanuwat et al., 2018) and grey-scale CIFAR (CIFAR-Gr). Mod-

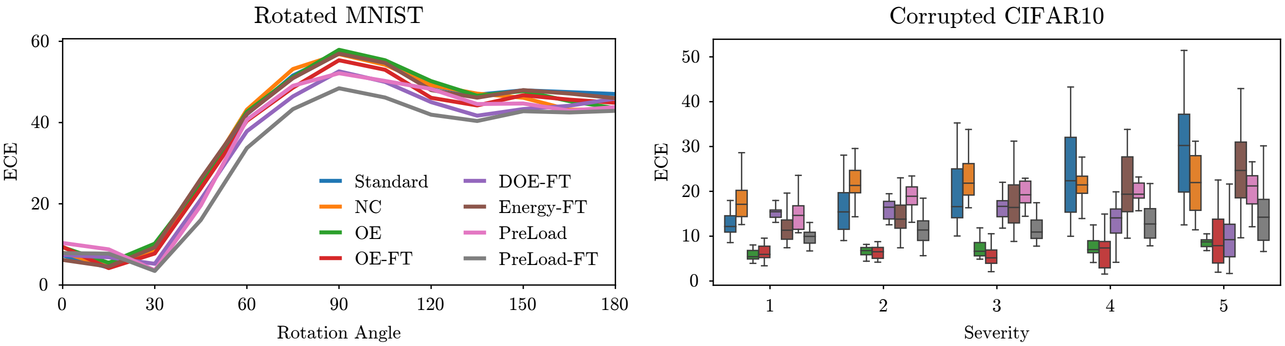

Figure 3: Calibration results, measured on the ECE metric, on Rotated MNIST and CIFAR10-C following Ovadia et al. (2019).

els trained on SVHN, CIFAR10 and CIFAR100 are evaluated on each other as well as LSUN classroom (LSUN-CR) (Yu et al., 2015) and Fashion MNIST 3D (FMNIST-3D). Additionally, all models are evaluated on uniform noise shaped like the relevant images and smooth noise (Hein et al., 2019), obtained by permuting, blurring and contrast-rescaling the original training data. Further information is provided in the Supplementary Section A.2.

Table 2 presents the FPR-95 averaged over all the OOD evaluation sets with error bars. Detailed results are in Section B. We have indicated in bold the best "Trained from scratch" and best "Finetuned" results in each row. Note that we take into account the error bars when highlighting the best results.

When training from scratch, PreLoad along with NC performs the best on F-MNIST, SHVN, CIFAR10, and CIFAR100. On MNIST, the NC method and OE perform better. Note that DDU which can prevent arbitrarily high confidence on far-away data is always significantly worse than our method on realistic OOD data. In the FT setting, we observe that Energy-FT and DOE-FT are better than OE-FT, however, our FT method performs the best.

We note that extra class methods, such as NC and ours, use the confidence of the additional class to detect OOD, unlike other methods such as Standard or OE which use $\max P(y|x)$ amongst all the classes. Since the additional class is trained on OOD data, such methods tend to perform better on OOD detection.

5.3 Dataset Shifts

Once we have established that our method performs well on far-away and realistic OOD data, we evaluate model calibration under data shift. Calibration is an important measure of uncertainty quantification. We evaluate calibration using the ECE (confidence ECE following Guo et al. 2017) metric with 15 bins. We use Rotated MNIST (Ovadia et al., 2019) and Corrupted CIFAR10 (CIFAR10-C) (Hendrycks and Dietterich, 2018) for evaluating on data shift.

We observe in Figure 3 that on Rotated MNIST, as we increase the rotation angle, PreLoad-FT performs the best followed by DOE-FT and PreLoad. PreLoad-FT scores the lowest ECE when the angle moves beyond 30. On CIFAR10-C we observe that as we increase corruption severity, ECE for Standard degrades the most followed by NC and Energy-FT. The OE and OE-FT methods perform the best followed by PreLoad-FT and DOE-FT. We observe that the FT methods do better than the methods trained from scratch.

Kristiadi et al. (2022b) suggest that NC, which is an extra class method, may do worse on dataset shift as it uses the confidence of the additional class which is trained on OOD data. Corrupted data may resemble OOD and therefore the calibration would be off. Our method on the other hand demonstrates that carefully designed extra-class methods can be better calibrated under dataset shift.

6 CONCLUSION

In this work, we have presented PreLoad, an OOD detection method that provably fixes arbitrarily high confidence in neural networks on far-away data. PreLoad works by training an extra class which produces larger logits as test samples move farther from the training data. Unlike all other baselines, PreLoad fulfills each of our three desiderata: a) maintain the simplicity of standard discriminative training b) provably fix arbitrarily high confidence on far-away data and c) perform well on realistic OOD samples. Future work could include training PreLoad with perturbed data such as adversarial examples, and adapting it to OOD detection in language models.

Acknowledgements

This research was partly funded by the Waterloo Apple PhD Fellowship, Natural Sciences and Engineering Council of Canada, and the David R. Cheriton Graduate Scholarship. Resources used in this work were provided by Huawei Canada, the Province of Ontario, the Government of Canada through CIFAR, companies sponsoring the Vector Institute https://vectorinstitute.ai/partners/ and the Natural Sciences and Engineering Council of Canada.

References

Arora, Raman, Basu, Amitabh, Mianjy, Poorya, and Mukherjee, Anirbit. Understanding deep Neural Networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

Blundell, Charles, Cornebise, Julien, Kavukcuoglu, Koray, and Wierstra, Daan. Weight uncertainty in Neural Network. In *International Conference on Machine Learning*. PMLR, 2015.

Clanuwat, Tarin, Bober-Irizar, Mikel, Kitamoto, Asanobu, Lamb, Alex, Yamamoto, Kazuaki, and Ha, David. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Cohen, Gregory, Afshar, Saeed, Tapson, Jonathan, and Van Schaik, Andre. Emnist: Extending MNIST to handwritten letters. In *2017 international joint conference on Neural Networks*. IEEE, 2017.

Gal, Yarin and Ghahramani, Zoubin. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*. PMLR, 2016.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, 2014.

Guo, Chuan, Pleiss, Geoff, Sun, Yu, and Weinberger, Kilian Q. On calibration of modern Neural Networks. In *International Conference on Machine Learning*. PMLR, 2017.

Gupta, Kartik, Rahimi, Amir, Ajanthan, Thalaiyasingam, Mensink, Thomas, Sminchisescu, Cristian, and Hartley, Richard. Calibration of Neural Networks using Splines. In *International Conference on Learning Representations*, 2020.

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

Hendrycks, Dan and Dietterich, Thomas. Benchmarking Neural Network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

Hendrycks, Dan, Mazeika, Mantas, and Dietterich, Thomas. Deep anomaly detection with Outlier Exposure. In *International Conference on Learning Representations*, 2018.

Kingma, Diederik P and Welling, Max. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kristiadi, Agustinus, Hein, Matthias, and Hennig, Philipp. Being Bayesian, even just a bit, fixes overconfidence in ReLU Networks. In *International Conference on Machine Learning*. PMLR, 2020.

Kristiadi, Agustinus, Hein, Matthias, and Hennig, Philipp. An infinite-feature extension for Bayesian ReLU nets that fixes their asymptotic overconfidence. *Advances in Neural Information Processing Systems*, 34, 2021.

Kristiadi, Agustinus, Eschenhagen, Runa, and Hennig, Philipp. Posterior refinement improves sample efficiency in Bayesian Neural Networks. 2022a.

Kristiadi, Agustinus, Hein, Matthias, and Hennig, Philipp. Being a bit frequentist improves bayesian Neural Networks. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022b.

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

Kull, Meelis, Perello Nieto, Miquel, Kängsepp, Markus, Silva Filho, Telmo, Song, Hao, and Flach, Peter. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with Dirichlet calibration. *Advances in neural information processing systems*, 32, 2019.

Kumar, Aviral, Sarawagi, Sunita, and Jain, Ujjwal. Trainable calibration measures for Neural Networks from kernel mean embeddings. In *International Conference on Machine Learning*. PMLR, 2018.

Lakshminarayanan, Balaji, Pritzel, Alexander, and Blundell, Charles. Simple and scalable predictive uncertainty estimation using Deep Ensembles. *Advances in neural information processing systems*, 30, 2017.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998.

Lee, Kimin, Lee, Kibok, Lee, Honglak, and Shin, Jinwoo. A simple unified framework for detecting out-of-distribution samples and Adversarial Attacks. *Advances in Neural Information Processing Systems*, 31, 2018.

Lin, Tsung-Yi, Goyal, Priya, Girshick, Ross, He, Kaiming, and Dollár, Piotr. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017.

Liu, Weitang, Wang, Xiaoyun, Owens, John, and Li, Yixuan. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33, 2020.

Louizos, Christos and Welling, Max. Multiplicative normalizing flows for variational Bayesian Neural Networks. In *International Conference on Machine Learning*.

Meinke, Alexander and Hein, Matthias. Towards Neural Networks that provably know when they don't know. In *International Conference on Learning Representations*, 2020.

Minderer, Matthias, Djolonga, Josip, Romijnders, Rob, Hubis, Frances, Zhai, Xiaohua, Houlsby, Neil, Tran, Dustin, and Lucic, Mario. Revisiting the calibration of modern Neural Networks. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., and Vaughan, J. Wortman, editors, *Advances in Neural Information Processing Systems*, volume 34. Curran Associates, Inc., 2021.

Mukhoti, Jishnu, Kirsch, Andreas, van Amersfoort, Joost, Torr, Philip HS, and Gal, Yarin. Deep deterministic uncertainty: A new simple baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Müller, Rafael, Kornblith, Simon, and Hinton, Geoffrey E. When does Label Smoothing help? *Advances in neural information processing systems*, 32, 2019.

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

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

Ovadia, Yaniv, Fertig, Emily, Ren, Jie, Nado, Zachary, Sculley, David, Nowozin, Sebastian, Dillon, Joshua, Lakshminarayanan, Balaji, and Snoek, Jasper. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

Thulasidasan, Sunil, Chennupati, Gopinath, Bilmes, Jeff A, Bhattacharya, Tanmoy, and Michalak, Sarah. On Mixup training: Improved calibration and predictive uncertainty for deep Neural Networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Torralba, Antonio, Fergus, Rob, and Freeman, William T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 30(11), 2008.

Wang, Qizhou, Ye, Junjie, Liu, Feng, Dai, Quanyu, Kalander, Marcus, Liu, Tongliang, Hao, Jianye, and Han, Bo. Out-of-distribution detection with implicit outlier transformation. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=hdghx6wbGuD.

Xiao, Han, Rasul, Kashif, and Vollgraf, Roland. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

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

Zagoruyko, Sergey and Komodakis, Nikos. Wide Residual Networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhang, Xiang and LeCun, Yann. Universum prescription: Regularization using unlabeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:

   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes]

   (b) The license information of the assets, if applicable. [Yes]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

   (d) Information about consent from data providers/curators. [Not Applicable]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Preventing Arbitrarily High Confidence on Far-Away Data in Point-Estimated Discriminative Neural Networks Supplementary Materials

Appendix A Experimental Details

A.1 Training Details

Our code is publicly available at: https://github.com/serenahacker/PreLoad.

We trained all models on a single 12GB-Tesla P-100 GPU. All results are averaged over 5 random seeds. Models were either trained from scratch or fine-tuned. When trained from scratch, all models were trained for 100 epochs and when fine-tuned, the Standard model was trained for 100 epochs with a further 10 epochs of fine-tuning. Note that OE-FT and Energy-FT do not introduce any new parameters to the Standard network, so all the parameters are sufficiently initialized during the 100 epochs of pre-training. On the contrary, PreLoad-FT introduces additional parameters, w_{k+1} and b_{k+1}. In order to initialize them properly and maintain a fair comparison between the algorithms, we pre-train the new weights using the objective $\min_{w_{k+1}, b_{k+1}} \mathcal{R}(f_{\psi,W}(x))$ for 10 epochs. All other parameters are frozen. After that, we fine-tune all the parameters θ using the objective $\min_{\theta} \mathcal{R}(f_{\psi,W}(x))$ where $\theta = \{\psi, W\}$ as in Algorithm 1—note that W here also includes w_{k+1} and b_{k+1}.

Note that in our implementation, we restrict the weights w_{k+1} to the positive orthant by setting $w_{k+1} = \exp(w'_{k+1})$ for some weights w'_{k+1}.

In all experiments, we used a batch size of 128 and a Cosine Annealing Scheduler for the learning rate. We tuned some of the hyper-parameters for PreLoad, PreLoad-FT and Energy-FT using WandB[2] sweeps. Each sweep consisted of 50 runs. Optimal hyper-parameter values were selected based on the highest evaluation accuracy. Table 3 lists our tuning strategy for each algorithm. Note that wd stands for weight decay, lr is learning rate, m_{in} and m_{ood} are the in-domian and OOD margin parameters for Energy-FT, and λ is a constant that scales the OOD loss. Note the for Energy-FT the OOD loss has both an in-domain and an out-of-domain component. Also note that for DOE-FT, λ controls the relative weight of the cross-entropy loss and the OE loss (Wang et al., 2023).

Tables 4 to 8 list the important hyper-parameters for all the methods for all the data sets. The values in bold were obtained after hyper-parameter tuning. For DOE-FT, we mostly used the same hyper-parameters as specified in Wang et al. (2023) (i.e. $\beta = 0.6$, $\lambda = 1$, warmup epochs = 5, perturbation steps = 1). We changed α to be uniformly sampled from $\{1e^{-2}, 1e^{-3}, 1e^{-4}\}$ (as opposed to $\{1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}\}$) as we found that this resulted in significantly higher validation accuracy for CIFAR-10 and CIFAR-100. Note that for the CIFAR datasets, we use WideResNet-16-4, while the authors of DOE, use WideResNet-40-2.

[2] https://github.com/wandb/wandb

Table 3: Hyper-parameter Tuning Strategies

Method	Tuning Method	Parameter Ranges
Energy-FT	Random Search	m_{in}: -30 to 0 m_{ood}: -30 to 0
PreLoad	Bayesian Optimization	lr: 1.0×10^{-3} to 5×10^{-1} wd: 1.0×10^{-5} to 1.0×10^{-3}
PreLoad-FT-Init	Random Search	λ: 1.0×10^{-2} to 1.0 lr: 1.0×10^{-4} to 1.0×10^{-1} wd: 1.0×10^{-6} to 1.0
PreLoad-FT	Random Search	λ: 1.0×10^{-2} to 1.0 lr: 1.0×10^{-4} to 1.0×10^{-1} wd: 1.0×10^{-6} to 1.0×10^{-3}

Table 4: MNIST Hyper-Parameters

Methods	Optimizer	Learning Rate	Weight Decay	λ	In Margin	Out Margin
Standard	Adam	1.0×10^{-3}	5.0×10^{-4}	-	-	-
NC	Adam	1.0×10^{-3}	5.0×10^{-4}	-	-	-
OE	Adam	1.0×10^{-3}	5.0×10^{-4}	5.0×10^{-1}	-	-
Pre-Load	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{0}	-	-
OE-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	5.0×10^{-1}	-	-
DOE-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{0}	-	-
Energy-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{-1}	**-3.6**	**-25.0**
PreLoad-FT-Init	Adam	$\mathbf{4.1 \times 10^{-3}}$	$\mathbf{3.1 \times 10^{-4}}$	$\mathbf{8.0 \times 10^{-1}}$	-	-
PreLoad-FT	Adam	$\mathbf{4.1 \times 10^{-3}}$	$\mathbf{3.1 \times 10^{-4}}$	$\mathbf{8.0 \times 10^{-1}}$	-	-

A.2 OOD Test Sets

In addition to MNIST, F-MNIST, SVHN, CIFAR-10 and CIFAR-100, we evaluate on the following OOD test sets.

- E-MNIST consists of handwritten letters and is in the same format as MNIST (Cohen et al., 2017).
- K-MNIST consists of handwritten Japanese (Hiragana script) and is in the same format as MNIST (Clanuwat et al., 2018).
- CIFAR-Gr consists of CIFAR-10 images converted to greyscale.
- LSUN-CR consists of real images of classrooms (Yu et al., 2015).
- FMNIST-3D consists of F-MNIST images converted from single channel to three channels with identical values.

Appendix B Additional Results

In Table 9, we present the complete FPR-95 results for all the methods and datasets that were used to compute the average results reported in Table 2. Note that results for Far-Away and Far-Away-RD are not included in the averages. Additionally, Table 10 presents results with the AUROC metric. AUROC is the area under the receiver operator curve (ROC). The ROC plots the TPR against FPR. AUROC can be interpreted as the probability that a model under test ranks a random positive sample higher than a random negative sample. We report AUROC as a percentage between 0 and 100 where higher the better. Tables 11 and 12 present the accuracy and calibration scores respectively.

Table 5: F-MNIST Hyper-Parameters

Methods	**Optimizer**	**Learning Rate**	**Weight Decay**	λ	**In Margin**	**Out Margin**
Standard	Adam	1.0×10^{-3}	5.0×10^{-4}	-	-	-
NC	Adam	1.0×10^{-3}	5.0×10^{-4}	-	-	-
OE	Adam	1.0×10^{-3}	5.0×10^{-4}	5.0×10^{-1}	-	-
Pre-Load	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{0}	-	-
OE-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	5.0×10^{-1}	-	-
DOE-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{0}	-	-
Energy-FT	Adam	1.0×10^{-3}	5.0×10^{-4}	1.0×10^{-1}	$\mathbf{6.4 \times 10^{-2}}$	**-4.0**
PreLoad-FT-Init	Adam	$\mathbf{6.0 \times 10^{-3}}$	$\mathbf{2.5 \times 10^{-3}}$	$\mathbf{2.6 \times 10^{-2}}$	-	-
PreLoad-FT	Adam	$\mathbf{6.0 \times 10^{-3}}$	$\mathbf{2.5 \times 10^{-3}}$	$\mathbf{2.6 \times 10^{-2}}$	-	-

Table 6: SVHN Hyper-Parameters

Methods	**Optimizer**	**Learning Rate**	**Weight Decay**	**Momentum**	λ	**In Margin**	**Out Margin**
Standard	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
NC	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
OE	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	5.0×10^{-1}	-	-
Pre-Load	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{0}	-	-
OE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	$5.0 \times 10-1$	-	-
DOE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{0}	-	-
Energy-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{-1}	**-5.7**	**-12.3**
PreLoad-FT-Init	SGD	$\mathbf{2.9 \times 10^{-2}}$	$\mathbf{2.8 \times 10^{-6}}$	9.0×10^{-1}	$\mathbf{2.2 \times 10^{-1}}$	-	-
PreLoad-FT	SGD	$\mathbf{2.9 \times 10^{-2}}$	$\mathbf{2.8 \times 10^{-6}}$	9.0×10^{-1}	$\mathbf{2.2 \times 10^{-1}}$	-	-

Table 7: CIFAR-10 Hyper-Parameters

Methods	**Optimizer**	**Learning Rate**	**Weight Decay**	**Momentum**	λ	**In Margin**	**Out Margin**
Standard	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
NC	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
OE	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	5.0×10^{-1}	-	-
Pre-Load	SGD	$\mathbf{7.3 \times 10^{-2}}$	$\mathbf{7.6 \times 10^{-4}}$	9.0×10^{-1}	1.0×10^{0}	-	-
OE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	5.0×10^{-1}	-	-
DOE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{0}	-	-
Energy-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{-1}	**-9.9**	**-5.7**
PreLoad-FT-Init	SGD	$\mathbf{6.1 \times 10^{-2}}$	$\mathbf{1.8 \times 10^{-6}}$	9.0×10^{-1}	$\mathbf{2.0 \times 10^{-2}}$	-	-
PreLoad-FT	SGD	$\mathbf{6.1 \times 10^{-2}}$	$\mathbf{1.8 \times 10^{-6}}$	9.0×10^{-1}	$\mathbf{2.0 \times 10^{-2}}$	-	-

Table 8: CIFAR-100 Hyper-Parameters

Methods	**Optimizer**	**Learning Rate**	**Weight Decay**	**Momentum**	λ	**In Margin**	**Out Margin**
Standard	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
NC	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	-	-	-
OE	SGD	1.0×10^{-1}	5.0×10^{-4}	9.0×10^{-1}	5.0×10^{-1}	-	-
Pre-Load	SGD	$\mathbf{4.5 \times 10^{-1}}$	$\mathbf{1.2 \times 10^{-4}}$	9.0×10^{-1}	1.0×10^{0}	-	-
OE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	5.0×10^{-1}	-	-
DOE-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{0}	-	-
Energy-FT	SGD	1.0×10^{-3}	5.0×10^{-4}	9.0×10^{-1}	1.0×10^{-1}	**-14.5**	**-10.3**
PreLoad-FT-Init	SGD	$\mathbf{6.3 \times 10^{-3}}$	$\mathbf{1.1 \times 10^{-4}}$	9.0×10^{-1}	$\mathbf{2.3 \times 10^{-2}}$	-	-
PreLoad-FT	SGD	$\mathbf{6.3 \times 10^{-3}}$	$\mathbf{1.1 \times 10^{-4}}$	9.0×10^{-1}	$\mathbf{2.3 \times 10^{-2}}$	-	-

Table 9: FPR-95, Complete Results

Datasets	Standard	DDU	NC	OE	PreLoad	OE-FT	DOE-FT	Energy-FT	PreLoad-FT
MNIST									
F-MNIST	8.1±0.9	21.8±1.1	0.0±0.0	0.2±0.0	0.0±0.0	0.3±0.1	0.0±0.0	3.7±0.7	0.1±0.0
E-MNIST	32.1±0.4	18.5±0.3	17.0±0.5	27.4±0.2	29.7±0.4	24.8±0.3	22.9±0.4	36.5±0.3	33.1±2.8
K-MNIST	10.9±0.3	4.6±0.1	3.1±0.4	5.4±0.2	9.8±0.6	3.4±0.2	2.9±0.2	10.1±0.6	17.2±3.5
CIFAR-Gr	0.0±0.0	62.6±8.2	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
Uniform	14.1±7.8	81.3±13.1	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
Smooth	0.0±0.0	97.5±2.4	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
FarAway	100.0±0.0	0.0±0.0	0.0±0.0	56.6±19.6	0.0±0.0	99.0±0.4	56.8±18.1	100.0±0.0	0.0±0.0
FarAway-RD	99.9±0.0	0.0±0.0	99.9±0.1	99.8±0.0	0.0±0.0	99.5±0.1	99.8±0.1	100.0±0.0	0.0±0.0
F-MNIST									
MNIST	74.8±1.4	0.6±0.2	6.8±0.5	65.4±1.1	6.7±2.1	68.4±1.5	51.3±0.9	36.9±3.5	28.8±3.4
E-MNIST	72.3±0.7	2.6±0.6	1.7±0.2	55.4±1.6	1.4±0.3	60.4±0.8	32.6±2.2	28.5±1.7	15.2±3.2
K-MNIST	73.6±0.6	0.4±0.1	4.7±0.6	58.1±0.9	5.7±1.2	60.6±1.0	38.6±1.0	21.1±1.3	22.6±2.6
CIFAR-Gr	85.4±1.0	84.3±3.8	0.0±0.0	0.0±0.0	0.0±0.0	0.1±0.0	0.0±0.0	0.1±0.0	0.0±0.0
Uniform	91.9±2.8	33.2±18.9	0.1±0.1	11.3±9.6	0.0±0.0	0.3±0.2	2.0±0.9	0.3±0.1	2.0±1.3
Smooth	25.6±1.4	89.6±0.6	0.0±0.0	0.2±0.0	0.0±0.0	0.4±0.1	0.0±0.0	0.2±0.1	5.8±5.8
FarAway	100.0±0.0	0.0±0.0	53.5±22.5	100.0±0.0	0.0±0.0	100.0±0.0	99.6±0.4	38.4±8.9	0.0±0.0
FarAway-RD	100.0±0.0	0.0±0.0	100.0±0.0	100.0±0.0	0.0±0.0	100.0±0.0	100.0±0.0	81.6±8.8	0.0±0.0
SVHN									
CIFAR-10	20.2±0.6	8.3±0.5	0.0±0.0	0.0±0.0	0.0±0.0	0.1±0.0	0.0±0.0	4.7±0.2	0.0±0.0
LSUN-CR	24.8±0.9	2.5±0.5	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	3.5±0.3	0.0±0.0
CIFAR-100	23.4±0.6	8.9±0.4	0.0±0.0	0.1±0.0	0.0±0.0	0.5±0.0	0.1±0.0	7.5±0.2	0.0±0.0
FMNIST-3D	26.5±0.4	25.8±2.2	0.0±0.0	0.0±0.0	0.0±0.0	0.6±0.3	0.0±0.0	14.9±1.1	0.0±0.0
Uniform	33.5±3.9	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.9±0.1	0.0±0.0
Smooth	14.0±0.5	2.5±0.4	12.6±1.2	10.0±0.3	6.8±1.0	8.6±0.3	7.4±0.4	11.8±1.3	4.6±0.2
FarAway	99.4±0.2	0.0±0.0	80.0±20.0	99.4±0.4	0.0±0.0	99.3±0.3	99.9±0.1	100.0±0.0	0.0±0.0
FarAway-RD	92.8±2.1	0.0±0.0	80.0±20.0	85.4±7.6	0.0±0.0	93.1±2.5	99.3±0.6	100.0±0.0	0.0±0.0
CIFAR-10									
SVHN	42.1±6.5	45.1±2.4	0.5±0.1	2.7±0.6	0.4±0.1	8.1±2.2	3.1±0.5	2.9±0.6	4.4±2.1
LSUN-CR	50.1±1.1	64.1±1.7	0.6±0.1	6.1±0.3	0.7±0.2	20.7±1.2	11.5±0.8	7.9±0.3	4.3±0.2
CIFAR-100	58.8±0.6	71.0±0.4	26.5±0.1	33.4±0.3	27.3±0.1	44.9±0.4	37.1±0.4	34.0±0.3	35.9±0.2
FMNIST-3D	38.9±1.1	35.7±5.4	3.5±0.4	11.5±0.6	3.9±0.4	15.8±1.0	12.3±0.9	8.2±0.6	7.9±0.8
Uniform	76.3±15.5	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	19.8±19.6	0.0±0.0
Smooth	40.4±5.1	12.4±1.8	3.4±0.7	16.3±3.0	3.6±0.6	30.2±1.5	26.4±4.1	20.6±2.6	19.4±6.0
FarAway	100.0±0.0	0.0±0.0	20.0±20.0	100.0±0.0	0.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	0.0±0.0
FarAway-RD	99.7±0.2	0.0±0.0	40.0±24.5	100.0±0.0	0.0±0.0	99.5±0.3	99.6±0.4	100.0±0.0	0.0±0.0
CIFAR-100									
SVHN	71.0±1.1	82.4±4.1	21.9±5.3	60.2±3.0	19.0±2.6	68.2±1.8	58.6±6.4	50.0±6.1	46.1±6.5
LSUN-CR	78.1±0.7	88.1±0.6	12.1±0.8	62.8±2.1	17.7±0.6	71.7±0.8	39.6±1.1	48.9±1.1	36.6±0.7
CIFAR-10	79.2±0.4	92.8±0.1	79.4±0.3	80.7±0.4	80.5±0.6	80.1±0.6	85.8±0.5	79.7±0.3	88.9±0.1
FMNIST-3D	65.6±2.2	90.2±0.9	7.2±0.5	56.0±3.0	13.9±1.5	61.3±1.6	32.0±1.6	34.4±3.6	46.4±2.8
Uniform	96.6±3.3	0.0±0.0	0.0±0.0	41.0±22.1	0.0±0.0	75.8±14.6	42.2±16.3	22.7±19.1	0.0±0.0
Smooth	72.7±1.3	42.5±1.2	44.1±4.9	60.4±3.4	24.2±4.1	66.6±1.2	67.7±5.3	61.0±2.7	19.0±3.7
FarAway	100.0±0.0	0.0±0.0	20.0±20.0	100.0±0.0	0.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	0.0±0.0
FarAway-RD	100.0±0.0	0.0±0.0	20.0±20.0	100.0±0.0	0.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	0.0±0.0

Table 10: AUROC, Complete Results

Datasets	**Standard**	**DDU**	**NC**	**OE**	**PreLoad**	**OE-FT**	**DOE-FT**	**Energy-FT**	**PreLoad-FT**
MNIST									
F-MNIST	98.3±0.1	96.9±0.1	100.0±0.0	99.8±0.0	100.0±0.0	99.7±0.0	99.9±0.0	99.2±0.2	100.0±0.0
E-MNIST	90.2±0.1	91.6±0.1	96.4±0.1	92.8±0.0	89.0±0.2	93.7±0.1	94.5±0.1	90.2±0.1	87.7±1.1
K-MNIST	97.5±0.1	98.9±0.0	99.3±0.1	98.6±0.1	97.5±0.2	99.0±0.1	99.1±0.0	97.7±0.1	95.2±1.1
CIFAR-Gr	99.8±0.0	94.3±0.4	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0
Uniform	96.7±0.5	93.2±0.8	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	99.6±0.1	100.0±0.0
Smooth	100.0±0.0	89.4±2.7	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0
FarAway	1.1±0.1	100.0±0.0	100.0±0.0	59.8±16.0	100.0±0.0	7.3±4.6	43.9±18.1	0.0±0.0	100.0±0.0
FarAway-RD	1.2±0.1	100.0±0.0	16.4±1.0	1.5±0.1	100.0±0.0	2.1±0.3	0.6±0.1	0.0±0.0	100.0±0.0
F-MNIST									
MNIST	80.1±0.6	99.7±0.1	98.4±0.1	84.4±0.7	98.4±0.5	82.9±0.4	88.4±0.4	90.5±1.0	92.4±1.2
E-MNIST	82.6±0.4	99.3±0.1	99.6±0.1	88.8±0.7	99.7±0.1	87.5±0.6	93.8±0.7	93.5±0.4	96.1±1.0
K-MNIST	83.1±0.4	99.7±0.0	99.0±0.1	89.9±0.2	98.9±0.2	89.1±0.1	93.8±0.2	95.7±0.3	95.1±0.6
CIFAR-Gr	83.8±0.6	85.0±1.2	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	99.8±0.0	100.0±0.0
Uniform	74.3±3.3	95.7±1.2	99.9±0.1	98.6±1.1	100.0±0.0	99.9±0.0	99.7±0.2	99.8±0.1	99.6±0.3
Smooth	95.9±0.2	59.5±1.2	100.0±0.0	100.0±0.0	100.0±0.0	99.9±0.0	100.0±0.0	99.1±0.0	98.6±1.3
FarAway	2.6±0.2	100.0±0.0	49.0±21.5	2.2±0.3	100.0±0.0	1.8±0.2	1.8±1.5	61.6±8.9	100.0±0.0
FarAway-RD	2.8±0.2	100.0±0.0	5.3±0.5	2.3±0.2	100.0±0.0	1.7±0.2	0.4±0.1	18.5±8.8	100.0±0.0
SVHN									
CIFAR-10	95.9±0.1	98.3±0.1	100.0±0.0	100.0±0.0	100.0±0.0	99.9±0.0	100.0±0.0	98.8±0.0	100.0±0.0
LSUN-CR	95.6±0.1	99.1±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	99.1±0.1	100.0±0.0
CIFAR-100	95.1±0.1	98.2±0.1	100.0±0.0	100.0±0.0	100.0±0.0	99.9±0.0	100.0±0.0	98.2±0.0	100.0±0.0
FMNIST-3D	94.5±0.4	96.0±0.3	100.0±0.0	100.0±0.0	100.0±0.0	99.7±0.1	100.0±0.0	96.9±0.3	100.0±0.0
Uniform	93.2±0.9	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	99.6±0.0	100.0±0.0
Smooth	96.9±0.2	99.2±0.1	97.1±0.3	98.0±0.1	98.4±0.2	98.3±0.1	98.5±0.1	97.4±0.2	98.9±0.0
FarAway	1.8±0.6	100.0±0.0	20.2±20.0	2.0±1.2	100.0±0.0	3.9±1.1	1.1±0.2	0.0±0.0	100.0±0.0
FarAway-RD	19.9±5.9	100.0±0.0	20.2±20.0	38.0±16.5	100.0±0.0	23.0±7.2	3.3±2.3	0.0±0.0	100.0±0.0
CIFAR-10									
SVHN	94.4±1.0	91.1±0.5	99.6±0.1	99.4±0.1	99.7±0.1	98.6±0.3	99.1±0.1	98.9±0.1	99.1±0.3
LSUN-CR	92.6±0.2	87.4±0.3	99.7±0.0	99.0±0.0	99.7±0.0	97.1±0.1	97.9±0.1	98.4±0.0	99.1±0.0
CIFAR-100	90.0±0.0	80.7±0.2	94.9±0.0	94.1±0.1	94.3±0.1	92.3±0.1	93.0±0.0	93.4±0.0	92.9±0.0
FMNIST-3D	94.7±0.2	94.5±0.8	99.3±0.1	98.3±0.1	99.1±0.1	97.7±0.1	97.9±0.2	98.4±0.1	98.5±0.2
Uniform	88.2±3.0	100.0±0.0	99.8±0.2	100.0±0.0	99.8±0.0	100.0±0.0	100.0±0.0	97.4±1.8	100.0±0.0
Smooth	93.6±1.1	96.1±0.7	99.0±0.1	97.5±0.4	98.9±0.1	96.3±0.4	96.5±0.4	96.9±0.4	97.1±0.7
FarAway	3.4±0.1	100.0±0.0	80.1±19.9	3.6±0.0	100.0±0.0	5.1±0.3	1.5±0.4	0.0±0.0	100.0±0.0
FarAway-RD	5.0±1.3	100.0±0.0	60.3±24.3	3.6±0.0	100.0±0.0	10.1±3.6	34.2±16.6	0.0±0.0	100.0±0.0
CIFAR-100									
SVHN	82.9±0.5	72.0±2.4	96.1±0.9	86.7±0.7	96.6±0.3	84.0±0.9	89.6±1.2	91.7±1.0	92.1±1.0
LSUN-CR	80.1±0.5	73.5±0.6	97.4±0.1	86.6±0.4	96.6±0.0	83.0±0.4	93.1±0.3	92.0±0.2	93.6±0.1
CIFAR-10	77.4±0.2	65.7±0.3	80.8±0.1	77.2±0.2	79.1±0.5	77.1±0.2	74.6±0.3	79.3±0.1	73.9±0.2
FMNIST-3D	85.2±0.9	67.6±1.1	98.3±0.1	88.4±0.7	97.2±0.2	86.5±0.6	94.7±0.2	94.7±0.5	92.8±0.4
Uniform	62.3±7.7	99.9±0.1	100.0±0.0	87.1±8.5	100.0±0.0	73.8±10.2	94.8±1.6	95.9±2.9	100.0±0.0
Smooth	75.4±1.2	84.9±2.2	89.9±1.2	81.1±1.3	95.1±1.0	78.3±0.7	79.9±1.7	82.7±1.5	96.1±0.8
FarAway	0.4±0.0	100.0±0.0	80.2±19.8	0.8±0.0	100.0±0.0	0.9±0.1	0.1±0.1	0.0±0.0	100.0±0.0
FarAway-RD	1.1±0.3	100.0±0.0	80.2±19.8	0.9±0.2	100.0±0.0	1.0±0.1	1.5±0.5	0.0±0.0	100.0±0.0

Table 11: Accuracy

	MNIST	**F-MNIST**	**SVHN**	**CIFAR-10**	**CIFAR-100**
Standard	99.5	92.7	97.4	94.9	77.3
DDU	99.5	92.7	97.4	94.9	77.3
NC	99.4	92.6	97.3	92.8	73.6
OE	99.5	92.7	97.4	95.5	77.1
PreLoad	99.5	92.3	97.3	93.5	71.9
OE-FT	99.5	92.8	97.3	94.8	77.1
DOE-FT	99.5	92.8	97.3	94.6	74.1
Energy-FT	99.5	92.8	97.4	94.9	76.8
PreLoad-FT	99.5	92.4	97.1	94.5	76.8

Table 12: Calibration measured using the ECE score

	MNIST	**F-MNIST**	**SVHN**	**CIFAR-10**	**CIFAR-100**
Standard	7.1	12.2	8.9	10.6	13.3
DDU	7.1	12.2	8.9	10.6	13.3
NC	7.1	7.4	9.0	15.2	8.6
OE	6.4	11.5	9.1	6.1	13.1
PreLoad	8.9	6.1	7.3	11.9	9.8
OE-FT	7.1	11.6	10.0	5.7	16.0
DOE-FT	7.3	7.5	8.2	15.7	16.9
Energy-FT	6.7	11.8	12.2	10.3	15.5
PreLoad-FT	8.6	3.9	10.3	8.1	11.1