HyperMix: Out-of-Distribution Detection and Classification in Few-Shot Settings

Nikhil Mehta¹,²,* Kevin J Liang²,* Jing Huang² Fu-Jen Chu² Li Yin² Tal Hassner²
Duke University¹ Meta² Equal Contribution
nikhilmehta.dce@gmail.com

Abstract

Out-of-distribution (OOD) detection is an important topic for real-world machine learning systems, but settings with limited in-distribution samples have been underexplored. Such few-shot OOD settings are challenging, as models have scarce opportunities to learn the data distribution before being tasked with identifying OOD samples. Indeed, we demonstrate that recent state-of-the-art OOD methods fail to outperform simple baselines in the few-shot setting. We thus propose a hypernetwork framework called HyperMix, using Mixup on the generated classifier parameters, as well as a natural out-of-episode outlier exposure technique that does not require an additional outlier dataset. We conduct experiments on CIFAR-FS and MiniImageNet, significantly outperforming other OOD methods in the few-shot regime.

1. Introduction

Out-of-Distribution (OOD) detection [20, 25, 61] has attracted much attention from the research community due to its practical implications, but almost all recent work has primarily considered OOD in standard supervised settings with a large number of samples. These many examples provide much information, e.g., clear boundaries, for distinguishing In-Distribution (IND) samples from future OOD inputs. However, access to abundant IND data is not always available in many settings, necessitating algorithms that can learn from few examples. Such few-shot settings [5, 59] commonly arise where data is expensive or difficult to come by [1, 36], quick adaptation to new classes is necessary [51], or models are personalized to users [15].

Although few-shot OOD (FS-OOD) detection (Figure 1) is a natural fit for many real-world scenarios [15, 27, 66], the intersection of these two problems is relatively underexplored, and solutions are not entirely straightforward as FSL and OOD each exacerbate some of the major challenges of the other: In few-shot learning, methods are generally starved for generalizability due to the few examples to learn from [11, 35], but in an open-world setting, one must also be wary of over-generalizing, which risks misclassifying OOD samples as one of the IND classes. Conversely, for OOD detection, having a limited number of samples can handicap learning the IND data distribution, making it significantly more challenging to detect those that are OOD, especially since neither the IND classes nor OOD classes are known ahead of time during offline meta-training. As we empirically show, popular few-shot and OOD methods tend to struggle in the FS-OOD setting. In particular, we find that many recent state-of-the-art OOD methods fail to outperform the simple baseline of maximum softmax predictive probability (MSP) [25] in the few-shot setting.

To address this gap, we develop a novel framework for few-shot OOD detection that brings together strategies from few-shot learning and OOD detection approaches. In particular, we adopt a hypernetwork-based [7, 21] approach that we call HyperMix. HyperMix utilizes Mixup [64] to augment the samples in a few-shot episode in two distinct ways. While Mixup has been used to perform augmentations in the input [64] or feature [52] space, we propose Mixup in the parameter space of hypernetwork-generated classifier
weights. We show that this greatly improves generalization and OOD detection in few-shot settings.

In addition, previous OOD methods found that exposure to OOD samples during training can be helpful [17, 26, 65]. Such methods, however, risk overfitting to the chosen OOD set and make the assumption of a relevant OOD set being available during training. This assumption is at odds with the unpredictable nature of OOD samples in the real world [33, 61]. Instead, we show that we can naturally incorporate outlier exposure through the meta-learning process common to few-shot methods [47, 48, 54], without an additional outlier dataset: classes not sampled during a particular meta-training task can serve as OOD samples.

We rigorously test our proposed methods on standard FSL benchmarks CIFAR-FS [6, 29] and MiniImageNet [12, 54], but in an OOD setting. Our results clearly show that the method we propose outperforms popular OOD methods, as well as the MSP baseline, in FS-OOD settings. We will release the code to promote the reproduction of our results. In summary, we make the following contributions:

1. We empirically show that the current state-of-the-art OOD detection methods have limited success in the few-shot learning scenario: the baseline MSP detector [25] and a related variant [38] are more reliable than other OOD methods in few-shot settings.

2. We propose a support-set augmentation for hypernetworks that mixes generated weight parameters and out-of-episode (OOE) Mixup [64] samples to supplement the query set. We name the combination of these two techniques HyperMix.

3. Through extensive experiments on CIFAR-FS and MiniImageNet, we show that our proposed HyperMix leads to improved FS-OOD detection.

2. Related Work

Out-of-Distribution (OOD) Detection. As artificial intelligence is increasingly being deployed for real-world use cases, understanding failure modes has become critical for the safety of such systems [2, 3, 37, 55]. One important axis is how such systems react when presented with unknown inputs [61]. Adversarial examples [20, 31, 50] have infamously demonstrated the ability to induce arbitrary responses in neural networks, but even untampered, normal examples can induce unexpected responses when not from the training distribution [24, 41]. While machine learning models do tend to be overconfident on OOD samples, the confidence on such samples is often lower than that of in-distribution samples and thus can be used as a simple way of identifying OOD samples [25]. Subsequent methods have sought to improve OOD detection with more sophisticated strategies, such as learning class-specific mean and covariance of embeddings and then using the Mahalanobis distance [34]. ODIN [38] uses small perturbations and temperature scaling to further separate IND and OOD score distributions. pNML [8] proposed to use a specific generalization error, the predictive normalized maximum likelihood regret, as the confidence score for OOD detection.

Outlier exposure has also been a popular technique: by showing the model OOD examples and explicitly optimizing for lower confidence, these models are often more robust to outliers. Some approaches source outliers from other datasets [13, 26, 65], while others generate them [33, 53]. However, the outlier distribution is often unknown in advance, and outliers exposed during training, whether from other datasets or synthesized, may not be representative.

Few-Shot Learning (FSL). While many deep learning advancements were enabled by large supervised datasets [12, 30], for many problems, labeled data can be scarce or expensive; being able to learn from few samples has thus been an active field of research recently. We direct interested readers to surveys [5, 59] for a more thorough treatment, highlighting several relevant works here.

Many few-shot methods are metric-based, which seek to learn an embedding such that classification can be done by comparing query and support samples in the learned feature space. A prominent example of this are Prototypical Networks [47], which meta-learns a feature extractor such that the mean of the support sample embeddings for a class can be used for Euclidean distance nearest neighbors. Mahalanobis distance [8], Earth Mover’s Distance [63], and cosine similarity [10] have also been utilized. Alternatively, methods such as RelationNet [48] learn a metric for comparing samples. POODLE [32] leverages OOD samples to help the model ignore irrelevant features.

Approaches with hypernetworks [21]—auxiliary models that generate weights for the primary task model—have also been popular for FSL. Learnnet [7] proposed generating full weights of a deep discriminative model from a single sample, scaling parameters by factorizing linear and convolutional layers. Hypernetworks have subsequently appeared in several FSL works, such as using task embeddings to influence model weights [42, 58], generating task-specific classifiers on top of a pre-trained encoder [45, 46], or synthesizing convolutional filters for object detection [43, 62].

Few-Shot OOD Detection. Few attempts have been made to solve the challenging problem of OOD detection under the few-shot constraint until recently. FS-OOD [57] introduced Out-of-Episode Classifier (OEC) that utilizes examples from the same dataset but not the current episode and showed its effectiveness in few-shot OOD detection. OOD-MAML [28] incorporated model-agnostic meta-learning (MAML) [16], a popular meta-learning approach for FSL, and proposed to synthesize OOD examples from unknown
classes via gradient updating of special meta-parameters to help the model learn a sharper decision boundary. FROB [14] performed self-supervised few-shot negative data augmentation on the distribution confidence boundary and combined it with outlier exposure.

3. Background

OOD Detection methods seek to identify unfamiliar OOD inputs during inference. More precisely, OOD methods [25] commonly frame the problem as training a model on “in-distribution” data \( \{x_{\text{IND}}, y_{\text{IND}}\} \sim D_{\text{IND}} \), but during test, inputs are drawn from an expanded distribution \( D_{\text{test}} = D_{\text{IND}} \cup D_{\text{OOD}} \), with \( D_{\text{OOD}} \) being the out-of-distribution data. Generally, the OOD distribution is unknown during training, though some methods utilizing outlier exposure [26] may assume access to an auxiliary dataset of OOD samples \( D_{\text{OOD}} \) during training, which may be distinct from the inference time outliers \( D_{\text{OOD}} \). Samples from this dataset \( \tilde{x} \sim D_{\text{OOD}} \) are used to augment the IND samples during training. A common label to assign such OOD samples is one of maximum entropy: \( p(\tilde{y}|\tilde{x}) = 1/N \) for an \( N \)-way classification problem.

Few-Shot Classification is typically framed as a \( K \)-shot \( N \)-way classification problem, where the goal is to leverage \( K \times N \) labeled examples from \( N \) unseen classes to learn a model \( \mathcal{M} \) that can classify \( Q \) unlabeled examples from the same classes. We denote the labeled \( K \times N \) examples as \( D_s = \{x_s, y_s\}_{k=1}^{KN} \) and the \( Q \) unlabeled examples as \( D_q = \{x_q\}_{q=1}^{Q} \). The labeled \( D_s \) and the unlabeled \( D_q \) examples are often referred to as support and query examples of the Few-Shot Classification (FSC) task.

To produce an effective model \( \mathcal{M}^* \) for novel classes \( C_n \), many FSC methods [47,54] meta-train \( \mathcal{M} \) on many \( K \)-shot \( N \)-way classification tasks sampled from a large labeled dataset of base classes \( C_b \), which are assumed to be distinct from the classes in \( C_n \), i.e. \( C_n \cap C_b = \emptyset \). Each task sampled from the training dataset of base classes is referred to as an FSC episode, which contains a support and query set. Specifically, for episode \( i \), \( \{D_s^{(i)}, D_q^{(i)}\} \sim D_{\text{train}} \), where \( D_{\text{train}} \) contains labeled examples from classes \( C_b \), and \( \{D_s^{(i)}, D_q^{(i)}\} \) contain labeled examples from classes \( C_n^{(i)} \subset C_b \) and \( \mathcal{C}_n^{(i)} = N \). The optimal classifier is produced by solving the following optimization problem:

\[
\mathcal{M}^* = \arg\min_{\mathcal{M}} \mathbb{E}_{(D_s^{(i)}, D_q^{(i)})} \left[ \mathcal{L}_{\text{EP}} \left( \mathcal{M}(D_s^{(i)}); D_q^{(i)} \right) \right].
\]

(1)

where \( \mathcal{L}_{\text{EP}} \left( \mathcal{M}(D_s^{(i)}); D_q^{(i)} \right) = \mathbb{E}_{D_s^{(i)}} \left[ \mathcal{L} \left( \tilde{y}^{(i)}; \tilde{x}^{(i)} \right) \right] \).

(2)

Here the prediction \( \tilde{y}^{(i)} = \mathcal{M}(D_s^{(i)})(x^{(i)}) \) and \( \mathcal{L}_{\text{EP}} \) corresponds to the loss for the \( i \)-th episode.

Mixup [64] is a data augmentation technique aiming to linearize model behaviour between samples. Given a pair of inputs \((x_1, y_1), (x_2, y_2) \sim D \) drawn from the data distribution, Mixup constructs an augmented sample \((\tilde{x}, \tilde{y})\) as a convex combinations of the two inputs and labels:

\[
\tilde{x} = \lambda x_1 + (1 - \lambda) x_2
\]

(3)

\[
\tilde{y} = \lambda y_1 + (1 - \lambda) y_2
\]

(4)

with \( \lambda \in [0, 1] \) being a weighting factor drawn randomly; Uniform and Beta distributions are common. These augmented samples are then used as additional training data.

Figure 2. A diagram of the proposed ParamMix, for a 3-way 3-shot example episode. Support samples (orange path): To generate the classifier, each of the \( KN \) inputs \( x_s \) (with label \( y_s \), denoted by color) in the support set are fed to the feature extractor \( F \); Mixup-augmented samples are also processed, resulting in \( 2KN \) embeddings. Each embedding is then passed to hypernetwork \( H \) to generate a corresponding classifier weight. The final \( N \) classifier weights \( W \) are calculated by weighted aggregation based on the samples’ Mixup labels (Equation 7). Query samples (blue path): To perform inference, we compose the feature extractor and classifier to produce the prediction \( y = \mathcal{C}(F(x_q)) \), using \( W \) previously generated from the support set.
4. Methods

4.1. FS-OOD Classification and Detection

We formulate FS-OOD detection and classification as the intersection of the OOD detection and few-shot classification problems. Given an N-way K-shot support set $D_s = \{x_s, y_s\}_{s=1}^{KN}$ (in-distribution data $D_{IND}$) and query samples $D_q = \{x_q\}_{q=1}^{Q}$, where $x_q$ is drawn from either the $N$ classes or the OOD distribution $D_{OOD}$, the goal is to classify each query $x_q$ as one of the $N$ classes or as OOD.

As FSC aims to generalize to new tasks during meta-test, the FS-OOD detection problem poses a notable challenge over standard OOD detection in that the IND data distribution $D_{IND}$ is unknown during meta-training, in addition to the usually unknown OOD distribution $D_{OOD}$. Instead, a model that can quickly adapt to a new IND distribution given a few samples must be learned.

4.2. Hypernetwork Framework

Because of their strong adaptation capabilities [18, 45], we adopt a hypernetwork [21] framework for FS-OOD: We instantiate model $M$ as a feature extractor $F$ and a linear classifier $C$ parameterized by $W$. Following previous works [10], we first pre-train $F$ on the dataset of base classes $D_b$. Rather than directly learning $W$ by gradient descent, we instead learn a hypernetwork $H$ that generates the parameters for $C$ from the support set. A hypernetwork-based model allows us to leverage the learning to learn paradigm, training a universal architecture model which can be quickly adapted to a new task by generating task-specific weights. Unlike fine-tuning and second-order procedures (inner and outer loop), hypernetworks are “training-free” when producing a classifier for a new task, resulting in faster adaptation and lower computational requirements.

Meta-training of $H$. The hypernetwork $H$ is trained on many classification tasks sampled from the dataset of base classes $D_b$ in the meta-training stage. This meta-training is done in an episodic fashion by drawing $M$ few-shot tasks from $D_{train}$. For each episode $i$, the hypernetwork takes the support embeddings extracted using $F$ and generates sample-specific weight codes $H(F(x_s^{(i)}))$, which are used to generate the classifier weights as follows:

$$w_n = \frac{1}{K} \sum_{s=1}^{KN} 1[y_s=n] H\left(F(x_s^{(i)})\right).$$

Let the classifier weights derived from the support set of episode $i$ be $W(D_s^{(i)}) = \{w_1, \ldots, w_n\}$. The class-logits for queries are predicted as $\hat{y}_q = C\left(F(x_q) \mid W(D_s^{(i)})\right)$. The loss function at the meta-training stage is as follows:

$$L^* = \text{argmin}_{W} \mathbb{E}_{(D_s^{(i)}, D_s^{(i)})} \left[ L_{EP} \left(W(D_s^{(i)}); D_s^{(i)}\right) \right],$$

where $L_{EP} \left(W(D_s^{(i)}); D_s^{(i)}\right) = \mathbb{E}_{D_s^{(i)}} \left[ L_{CCE} \left(\hat{y}_q, y_s\right)\right]$ and $L_{CCE} \left(\hat{y}_q, y_s\right) = -y_q \cdot \log(\hat{y}_q)$ is the Categorical Cross Entropy (CCE) loss. Note, the loss gradient in (6) can also be backpropagated into $F$ to fine-tune the feature extractor, providing some improvement.

OOD Detection during Inference. After meta-training the model, we use the maximum softmax probability [25] as the score for the OOD detection task during inference. In particular, we define the score $s(x_q)$ for a given query $x_q$ at meta-testing as $s(x_q) = \max_c \hat{y}_{q,c}$, where $\hat{y}_{q,c}$ is the $c^{th}$ component of the softmax vector $\hat{y}_q = C\left(F(x_q) \mid W\right)$. If $s(x_q) < \tau$, with $\tau$ being the OOD threshold, we call the query an OOD sample. However, note that the OOD detection task is evaluated using metrics that alleviate the need to set a fixed threshold $\tau$ (see Section 5.2).

4.3. HyperMix

We find our hypernetwork framework for FS-OOD outperforms other common few-shot approaches, but such a set-up alone does not incorporate any OOD detection capabilities. We thus propose HyperMix: a support and query set augmentation technique based on Mixup [64] for training the hypernetwork during the meta-training stage. In contrast to the existing Mixup data augmentation technique, which uses linear Mixup of the model inputs and target labels, HyperMix uses Mixup in two novel ways uniquely adapted to our hypernetwork framework and FS-OOD settings, which we call ParamMix and OOE-Mix.

**ParamMix.** ParamMix augments the support set using Mixup and uses weighted aggregation to compute the classifier weights. Let the mixed support samples and the corresponding label be denoted as $\tilde{x}_s$ and $\tilde{y}_s$ respectively. The mixing parameter $\lambda_{PM}$ is sampled from a fixed Beta distribution, and the extracted parameters of the classifiers are calculated using weighted aggregation of sample-specific weight codes, where the aggregation weights are determined based on the probabilities of mixed samples. ParamMix modifies the weight generation in (5) for soft labels:

$$w_n = \frac{1}{S} \sum_{s=1}^{S} \sum_{s=1}^{S} \tilde{y}_s^{(s)} H\left(F(\tilde{x}_s)\right).$$

where $\tilde{x}_s = \lambda_{PM} x_s^{(1)} + (1-\lambda_{PM}) x_s^{(2)}$ is a mixed sample formed from two support set images, $\tilde{y}_s^{(s)}$ corresponds to the probability of $\tilde{x}_s$ belonging to the label $n$, and $S$ is the total number of support samples after augmentation with Mixup. We summarize ParamMix in Figure 2.
OOE-Mix. Outlier exposure [17, 26, 65] has been shown to be highly effective for OOD detection, but it often assumes access to a dataset of outliers, which is often practical. On the other hand, we note that the meta-learning process of sampling a subset of the base classes $C_b^{(i)}$ to construct a few-shot episode means there are classes $C_b \setminus C_b^{(i)}$ available that are not part of the $N$-way classification for episode $i$. We refer to the samples from $C_b^{(i)}$ as in-episode (INE) and samples from $C_b \setminus C_b^{(i)}$ as out-of-episode (OOE). As the OOE samples are from the same dataset, not only are these samples representative and of the same domain as the INE data, but the mutually exclusive nature of the labels means that they are also not in-distribution. We can thus consider the ample samples from $C_b \setminus C_b^{(i)}$ as realistic “near” outliers [60] for this particular few-shot episode.

While we can expose OOE samples during meta-training as is, we find mixing INE and OOE samples to be highly effective. Therefore, we propose OOE-Mix to augment the query set with mixed INE and OOE samples (Figure 3). Specifically, we sample INE and OOE samples as:

$$x_1^{(i)} \sim D^{(i)}_{INE}; \quad x_2^{(i)} \sim D^{(i)}_{OOE}; \quad p(y|z^{(i)}) = 1/N$$  \hspace{1cm} (8)

As we consider the OOE samples to be OOD, we assign OOE samples with a uniform distribution label, to enforce maximal uncertainty. Similar to Equations 3 and 4, we then apply Mixup, but specifically between pairs of INE and OOE samples:

$$\tilde{x}^{(i)} = \lambda_{OM} x_1^{(i)} + (1 - \lambda_{OM}) x_2^{(i)}$$  \hspace{1cm} (9)

$$p(y|\tilde{z}^{(i)}) = \lambda_{OM} p(y|x_1^{(i)}) + (1 - \lambda_{OM}) p(y|x_2^{(i)})$$  \hspace{1cm} (10)

with $\lambda_{OM} \sim \text{Beta}(\alpha_{OM}, \beta_{OM})$. We augment the query dataset with the generated mixed samples while optimizing the loss function in Equation (6). We find this helpful as the addition of Mixup between INE and OOE samples encourages smooth behaviour in the classifier’s decision space. This helps with generalization, a critical component for few-shot settings.

5. Experiments

5.1. Datasets

We run experiments on popular few-shot learning datasets CIFAR-FS [6] and MiniImageNet [54], but we adapt them for few-shot OOD detection. In particular, for each meta-testing task, we include query images from the unsampled meta-test classes (those not in the $N$-way task) as OOD inputs (see Evaluation Protocol in Section 5.2).

CIFAR-FS consists of $32 \times 32$ color images from the CIFAR-100 dataset [29], split into 64, 16, and 20 classes for meta-training, meta-validation, and meta-testing, with 600 images per class.

MiniImageNet [54] is a common few-shot benchmark consisting of $84 \times 84$ color images, with 100 classes of 600 examples each. Similar to CIFAR-FS, we use 64 classes for meta-training, 16 for validation, and 20 for meta-testing.

5.2. Set-up

Our experimental settings closely resemble the traditional few-shot classification benchmarks, with the primary difference being that half the meta-test query samples are drawn from outside of the $N$-way classification problem, i.e., OOD samples. We use 5-shot and 10-shot settings during the meta-training phase. All experiments were done on a GeForce GTX 1080 GPU with 12GB memory.

Implementation of $\mathcal{F}$. We implement the feature extractor $\mathcal{F}$ as a ResNet-12 [23], which has 4 residual blocks with 3 convolutional layers of 64, 160, 320, and 640 3 x 3 kernels. The first three blocks are followed by $2 \times 2$ average pooling, resulting in features of size 640. Inspired by recent self-supervised learning methods for few-shot classification [19, 39, 44], we train $\mathcal{F}$ using a combination of supervision and self-supervised criteria. In particular, we follow the strategy of [44], where the image-label pair $(x, y)$ is augmented to produce $(x(r), y, r)$, where $x(r)$ is the image $x$ rotated by $r$ degrees and $r \in \{0, 90, 180, 270\}$. In the pre-training stage, a stack of two linear classifiers are learned on top of $\mathcal{F}$, where the first classifier predicts the class logits $\hat{y}$ for the $|C_b|$-way classification task, and the second classifier predicts the rotation logits $\hat{r}$. The model is trained using $\mathcal{L}_{CCE}(\text{softmax}(\hat{y}), y) + \mathcal{L}_{BCE}(\text{sigmoid}(\hat{r}), r)$, where CCE and BCE correspond to the Categorical and Binary Cross-Entropy loss. After convergence, the two linear classifiers are discarded [9], and $\mathcal{F}$ is used to extract image embeddings in the meta-training stage. During pre-training of $\mathcal{F}$, the batch size was set to 64. Each image in the batch was rotated three times at 90, 180, and 270 degrees, resulting in an effective batch size of 256. We train $\mathcal{F}$ for 200 epochs using Stochastic Gradient Descent (SGD) optimizer with Nesterov momentum [49] of 0.9 and weight decay of 5e-4. We start with an initial learning rate of 0.1 and decay by 0.2 at 60, 120, and 160 epochs.
Table 1. FS-OOD Classification and Detection. *OE is the outlier exposure method and thus requires auxiliary data, which we do not assume in our experiments; we instead use OOE sampling as the auxiliary outliers. Best value shown in bold; second best shown in italics.

<table>
<thead>
<tr>
<th>Method</th>
<th>5-shot 5-way</th>
<th>10-shot 5-way</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IND Acc ↑</td>
<td>AUROC ↑</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine-tune w/ MSP [25]</td>
<td>80.38 ± 0.41</td>
<td>72.00 ± 0.40</td>
</tr>
<tr>
<td>MSP [25]</td>
<td>80.09 ± 0.42</td>
<td>75.42 ± 0.37</td>
</tr>
<tr>
<td>DM [34]</td>
<td>80.09 ± 0.42</td>
<td>75.31 ± 0.37</td>
</tr>
<tr>
<td>OEC [56]</td>
<td>76.45 ± 0.93</td>
<td>74.32 ± 0.94</td>
</tr>
<tr>
<td>OEC* [26]</td>
<td>78.31 ± 0.41</td>
<td>73.13 ± 0.45</td>
</tr>
<tr>
<td>Hypernetwork</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP [25]</td>
<td>78.89 ± 0.45</td>
<td>78.38 ± 0.39</td>
</tr>
<tr>
<td>Entropy [40]</td>
<td>78.89 ± 0.45</td>
<td>69.67 ± 0.43</td>
</tr>
<tr>
<td>ODIN [38]</td>
<td>78.89 ± 0.45</td>
<td>78.73 ± 0.38</td>
</tr>
<tr>
<td>ODIN [38]</td>
<td>78.89 ± 0.45</td>
<td>63.67 ± 0.22</td>
</tr>
<tr>
<td>pNML [8]</td>
<td>78.89 ± 0.45</td>
<td>60.41 ± 0.49</td>
</tr>
<tr>
<td>OEC [56]</td>
<td>77.51 ± 0.44</td>
<td>79.11 ± 0.34</td>
</tr>
<tr>
<td>OEC* [26]</td>
<td>79.46 ± 0.42</td>
<td>81.68 ± 0.35</td>
</tr>
<tr>
<td>Hypernetwork</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParamMix (ours)</td>
<td>78.45 ± 0.49</td>
<td>79.24 ± 0.37</td>
</tr>
<tr>
<td>OOE-Mix (ours)</td>
<td>80.88 ± 0.42</td>
<td>81.11 ± 0.35</td>
</tr>
<tr>
<td>HyperMix (ours)</td>
<td>81.33 ± 0.41</td>
<td>82.43 ± 0.33</td>
</tr>
</tbody>
</table>

Implementation of \( H \). We parameterize the hypernetwork as a multi-layered perceptron, containing two fully-connected layers of size 256. We generate the classifier weights using the procedure described in Equations 5 and 7. Note that while the number of parameters generated in Equation 5 corresponds to the number of classes \( N \), in practice, we generate \( N + 1 \) parameters and split them into classifier weights and biases. We meta-train \( H \) for 50 epochs with SGD optimizer having a fixed learning rate of 0.001. Each epoch had 200 batches, where each batch contained up to 4 episodes/tasks. The input images are augmented using random crop, color jitter, and horizontal flip.

Evaluation Protocol. We sample 400 episodes randomly from the meta-test set containing novel classes \( C_N \) that the model has not previously encountered. In each episode, the model is given a labeled support set of \( KN \) samples that is referred to as the IND samples, and a set of 20 unlabeled queries. Of the 20 queries, 10 belong to the \( N \) IND classes and the remaining 10 are OOD. In our experiments, we consider the near-OOD detection task, which is considered a harder problem compared to far-OOD detection [60]. Unlike far-OOD detection, which uses OOD samples from a different dataset for evaluation, in near-OOD detection, we reserve classes from the same dataset as OOD during evaluation. In our experiments, we consider the classes outside a given episode as OOD during evaluation. Note that both IND and OOD samples belong to classes that are not known a priori to the model. We evaluate the model on the few-shot \( N \)-way classification task for IND queries, and the binary OOD detection task. We use classification accuracy to measure the few-shot classification performance, and AUROC and FPR@90 to measure the OOD detection performance.

5.3. Baselines

Few-shot OOD detection and classification is an under-explored problem, meaning there are relatively few previous works. Standard few-shot methods lack a mechanism to reject outlier samples [36], which leads to misclassifications for all OOD samples. Thus, we instead primarily compare against several recently proposed OOD methods, with restricted numbers (shots) of IND samples:

- Maximum Softmax predictive Probability (MSP) [25]:
MSP uses maximum probability from the model as the IND confidence score.

- **Entropy** [40]: It uses the predictive entropy of the output probabilities as the OOD score.
- **ODIN** [38]: ODIN also uses MSP; however, it employs temperature scaling and input perturbation based on the gradient of the loss function to increase the margin between maximum probability obtained for IND and OOD samples.
- **Deep Mahalanobis (DM)** [34]: DM calculates the class-conditional mean and covariance for each class to calculate the confidence score of query samples at each layer of the encoder. Scores are computed by a logistic regression model learned with validation data.
- **Predictive Normalized Maximum Likelihood (pNML)** [8]: pNML uses a generalization error based on predictive normalized maximum likelihood regret as the OOD score.
- **Outlier Exposure (OE)** [26]: OE approaches expose OOD samples to the model, trained with a uniform distribution label in the meta-training stage. Note that this assumes the availability of an auxiliary outlier dataset, which we do not use in our experiments. Instead, we show OE with our proposed strategy of leveraging OOE samples from $C_h \setminus c^{(i)}$.
- **Out-of-Episode Classifier (OEC)** [56]: OEC is a recently proposed method for FS-OOD detection, which uses binary cross-entropy objective to learn an IND/OOD binary classifier. Similar to OE, we provide OOE samples to OEC during meta-training.

We primarily perform our comparisons using the hypernetwork implementation described in Section 5.2, but we also perform experiments with some of the baselines using fine-tuning [10] and ProtoNet [47] as the few-shot learning algorithm, to demonstrate the efficacy of our hypernetwork. We further describe how we adapted the above methods to the FS-OOD setting in Appendix A. In addition to our proposed HyperMix, we also perform ablations, analyzing the two components of our approach individually:

- **ParamMix**: We apply Mixup to the parameters $W$ generated by hypernetwork $H$ as described in Section 4.3, but without OOE samples during meta-training.
- **OOE-Mix**: Similar to OE [26], we provide OOE samples to the model from the unsampled base classes, and additionally apply Mixup between pairs of INE and OOE samples as described in Section 4.3.

Note that our proposed HyperMix is the combination of ParamMix and OOE-Mix. The hyperparameters for all the methods are chosen based on grid search done on validation dataset. The chosen hyperparameters and candidates are discussed in Appendix B.

### Table 2. ParamMix, OOE-Mix, and Hyper-Mix on mixed samples on CIFAR-FS. Best values shown in bold; second best in italics.

<table>
<thead>
<tr>
<th>Method</th>
<th>IND Acc</th>
<th>AUROC</th>
<th>FPR@90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test OOD Dataset: INE-OOE Mixture</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParamMix</td>
<td>80.94 ± 0.93</td>
<td>85.47 ± 0.65</td>
<td>50.26 ± 1.61</td>
</tr>
<tr>
<td>OOE-Mix</td>
<td>84.24 ± 0.47</td>
<td>85.41 ± 0.66</td>
<td>50.59 ± 1.60</td>
</tr>
<tr>
<td>HyperMix</td>
<td>84.51 ± 0.78</td>
<td>85.41 ± 0.66</td>
<td>50.59 ± 1.60</td>
</tr>
</tbody>
</table>

### 5.4. Few-shot In-Distribution Classification and OOD Detection

We report FS-OOD performance on CIFAR-FS [6] and MiniImageNet [54] in Table 1. We quantify OOD detection performance with area under the receiver operating characteristic curve (AUROC) and false positive rate at 90% recall rate (FPR@90); we simultaneously also measure classification accuracy of in-distribution samples (IND Acc). Note that many of the baselines are post-hoc methods on the same meta-trained model and thus have the same IND accuracy.

#### Choice of Few-shot Framework.

Our proposed hypernetwork framework for FS-OOD outperforms more basic approaches like fine-tuning or ProtoNet. For example, for the MSP baseline, a hypernetwork approach improves AUROC and FPR@90 by almost 5% and 9%, respectively, compared to ProtoNet on 5-shot 5-way classification on FS-CIFAR-100; this grows to a 6.38% and 17.55% improvement in FS-CIFAR-100 AUROC and FPR@90 compared to MSP with fine-tuning. Similar improvements to OOD detection can be seen in the 10-shot 5-way case.

#### Performance of Popular OOD Detection Methods.

We observe that several recent OOD detection methods outperform the simple baseline MSP in the FS-OOD setting. In particular, we found that popular OOD methods such as DM and pNML had limited success in detecting OOD samples; DM for example has almost 15% worse AUROC and almost 23% worse FPR@90 compared to MSP, which stands in sharp contrast to its large improvements in the many-shot setting. We posit that this is due to the limited number of support samples per episode, leading to the empirical covariance/data matrix used in these approaches to be degenerate. We indeed find this to be the case when comparing the singular values of the covariance matrix used in DM for few-shot and many-shot settings (see Appendix C). This limits the generalization ability of these methods in differentiating IND from OOD samples.

**HyperMix**. In contrast, we observe that our proposed HyperMix performs the best. Using HyperMix at meta-training, the model not only does better on the OOD detection task, but also leads to improved IND accuracy compared to the baselines with hypernetworks. We see that the
combination of both ParamMix and OOE-Mix is important. While each individually is comparable or does better than the baselines, the combination of the two significantly outperforms the baseline methods.

Why is the combination of ParamMix and OOE-Mix especially effective? We hypothesize that each makes HyperMix robust to certain types of outliers: ParamMix smooths the inter-support class decision boundary, while OOE-Mix regularizes behavior on outside classes. We verify this by testing ParamMix, OOE-Mix, and HyperMix on outliers created by mixing inlier class samples (IN-INE) and inliers with OOE samples (INE-OOE) in Table 2; we indeed observe ParamMix and OOE-Mix have different tendencies, and that HyperMix tends to combine their strengths.

5.5. Noisy Labels

Even in few-shot settings, outliers can have noisy labels and could be present in the support sets, and such mislabeled samples can have a significant impact on performance [36]. We thus also evaluate all the models in the FS-OOD settings when noisy labels are present during meta-testing, reporting OOD detection results in Table 3 and few-shot accuracy in Figure 4. We restrict our experiments to hypernetwork-based few-shot learning, as it saw the strongest OOD detection performance in Table 1. We experiment with varying levels of noise present in the support set ranging from 10% to 40% of the total support samples. For each evaluation scenario, we randomly select IND samples \((x, y)\) from the support set and replace them with OOD samples \((\bar{x}, y)\). To make this scenario especially challenging, we pair each IND class with an OOE class, to model real-world classes being similar [22]. As expected, we observe that all methods, including ours, see worse performance with higher noise levels. On the other hand, we observe that HyperMix maintains a performance improvement gap even up to 40% support set noise.

6. Conclusion

Standard few-shot methods are unequipped to deal with OOD samples during test time and will tend to misclassify them as one of the IND classes by default. At the same time, while recently proposed methods have led to improvements in OOD detection in settings with abundant in-distribution data, we find that many of them struggle in few-shot settings, failing to outperform the simple MSP [25] baseline. To address this gap, we show that a hypernetwork-based framework for FS-OOD outperforms ProtoNet or fine-tuning based approaches. We further show that exposing the model to out-of-episode (OOE) outliers mixed with the query set during meta-training helps prepare the model for test OOD samples, and mixup of hypernetwork weights generated from the support set further helps the model generalize. This combination of augmentation techniques, which we call HyperMix, strongly outperforms baseline methods in FS-OOD experiments on FS-CIFAR-100 and MiniImageNet in a variety of settings.

Table 3. Noisy FS-OOD Detection (10-shot 5-way). HyperMix outperforms the baseline methods even in noisy support set settings.

<table>
<thead>
<tr>
<th>Method</th>
<th>AUROC ±</th>
<th>FPR@90 ±</th>
<th>AUROC ±</th>
<th>FPR@90 ±</th>
<th>AUROC ±</th>
<th>FPR@90 ±</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP [25]</td>
<td>79.93 ± 0.36</td>
<td>57.15 ± 0.79</td>
<td>77.35 ± 0.38</td>
<td>62.28 ± 0.78</td>
<td>72.17 ± 0.42</td>
<td>70.11 ± 0.73</td>
</tr>
<tr>
<td>Entropy [40]</td>
<td>71.61 ± 0.83</td>
<td>69.16 ± 1.40</td>
<td>70.66 ± 0.41</td>
<td>71.92 ± 0.70</td>
<td>67.93 ± 0.43</td>
<td>76.56 ± 0.66</td>
</tr>
<tr>
<td>HyperMix</td>
<td>80.33 ± 0.36</td>
<td>55.69 ± 0.78</td>
<td>77.79 ± 0.38</td>
<td>60.88 ± 0.77</td>
<td>72.56 ± 0.42</td>
<td>68.98 ± 0.73</td>
</tr>
<tr>
<td>ODIN [38]</td>
<td>63.52 ± 0.18</td>
<td>80.92 ± 0.24</td>
<td>60.33 ± 0.18</td>
<td>83.63 ± 0.22</td>
<td>57.40 ± 0.18</td>
<td>85.60 ± 0.20</td>
</tr>
<tr>
<td>gSNL [8]</td>
<td>62.77 ± 0.91</td>
<td>74.40 ± 1.09</td>
<td>60.64 ± 0.93</td>
<td>75.56 ± 1.12</td>
<td>59.81 ± 0.92</td>
<td>78.61 ± 1.04</td>
</tr>
<tr>
<td>OE [56]</td>
<td>81.60 ± 0.35</td>
<td>56.71 ± 0.80</td>
<td>78.46 ± 0.37</td>
<td>63.59 ± 0.77</td>
<td>72.70 ± 0.41</td>
<td>71.95 ± 0.70</td>
</tr>
<tr>
<td>OEC [56]</td>
<td>81.05 ± 0.74</td>
<td>57.68 ± 0.83</td>
<td>76.19 ± 0.51</td>
<td>64.17 ± 0.94</td>
<td>71.11 ± 0.78</td>
<td>74.46 ± 1.05</td>
</tr>
<tr>
<td>ParamMix</td>
<td>81.02 ± 0.36</td>
<td>55.28 ± 0.78</td>
<td>78.74 ± 0.37</td>
<td>60.16 ± 0.78</td>
<td>73.80 ± 0.40</td>
<td>67.94 ± 0.73</td>
</tr>
<tr>
<td>HyperMix</td>
<td>81.94 ± 0.34</td>
<td>53.67 ± 0.79</td>
<td>79.69 ± 0.36</td>
<td>58.80 ± 0.80</td>
<td>74.87 ± 0.40</td>
<td>67.00 ± 0.76</td>
</tr>
<tr>
<td>MinImageNet [54]</td>
<td>81.94 ± 0.34</td>
<td>53.67 ± 0.79</td>
<td>79.69 ± 0.36</td>
<td>58.80 ± 0.80</td>
<td>74.87 ± 0.40</td>
<td>67.00 ± 0.76</td>
</tr>
</tbody>
</table>

Figure 4. FS accuracy on FS-CIFAR-100 (left) and MiniImageNet (right) with varying levels of label noise in the support set.
References


[41] Khanh Xuan Nguyen and Brendan O’Connor. Posterior calibration and exploratory analysis for natural language processing models. In EMNLP, 2015. 2


