Two Sides of Meta-Learning Evaluation: In vs. Out of Distribution

Amrith Setlur[1*] **Oscar Li**[2*] **Virginia Smith**[2]
asetlur@cs.cmu.edu oscarli@cmu.edu smithv@cmu.edu
[1]Language Technologies Institute [2]Machine Learning Department
School of Computer Science, Carnegie Mellon University

Abstract

We categorize meta-learning evaluation into two settings: *in-distribution* [ID], in which the train and test tasks are sampled *iid* from the same underlying task distribution, and *out-of-distribution* [OOD], in which they are not. While most meta-learning theory and some FSL applications follow the ID setting, we identify that most existing few-shot classification benchmarks instead reflect OOD evaluation, as they use disjoint sets of train (base) and test (novel) classes for task generation. This discrepancy is problematic because—as we show on numerous benchmarks—meta-learning methods that perform better on existing OOD datasets may perform significantly worse in the ID setting. In addition, in the OOD setting, even though current FSL benchmarks seem befitting, our study highlights concerns in 1) reliably performing model selection for a given meta-learning method, and 2) consistently comparing the performance of different methods. To address these concerns, we provide suggestions on how to construct FSL benchmarks to allow for ID evaluation as well as more reliable OOD evaluation. Our work[†] aims to inform the meta-learning community about the importance and distinction of ID vs. OOD evaluation, as well as the subtleties of OOD evaluation with current benchmarks.

1 Introduction

Meta-learning considers learning algorithms that can perform well over a distribution of tasks [19, 37]. To do so, a meta-learning method first learns from a set of tasks sampled from a training task distribution (*meta-training*), and then evaluates the quality of the learned algorithm using tasks from a test task distribution (*meta-testing*). The test task distribution can be the same as the training task distribution (a scenario we term *in-distribution* generalization evaluation or ID evaluation) or a different task distribution (*out-of-distribution* generalization evaluation or OOD evaluation).

In this work, we argue that there is a need to carefully consider current meta-learning practices in light of this ID vs. OOD categorization. In particular, meta-learning is commonly evaluated on few-shot learning (FSL) benchmarks, which aim to evaluate meta-learning methods' ability to learn sample-efficient algorithms. Current benchmarks primarily focus on image classification and provide training tasks constructed from a set of train (base) classes that are completely disjoint and sometimes extremely different from the test (novel) classes used for test tasks. As we discuss in Section 3, this design choice imposes a natural shift in the train and test task distribution that makes current benchmarks reflective of OOD generalization. However, there are a number of reasons to also consider the distinct setting of ID evaluation. First, whether in terms of methodology or theory, many works motivate and analyze meta-learning under the assumption that train and test tasks are sampled *iid* from the same distribution (see Section 2). Second, we identify a growing number of applications, such as federated learning, where there is in fact a need for sample-efficient algorithms

[*]Authors contributed equally to this paper.
[†]Code available at https://github.com/ars22/meta-learning-eval-id-vs-ood.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

that can perform ID generalization. Crucially, we show across numerous benchmarks that methods that perform well OOD may perform significantly worse in ID settings. Our results highlight that it is critical to clearly define which setting a researcher is targeting when developing new meta-learning methods, and we provide tools for modifying existing benchmarks to reflect both scenarios.

Beyond this, we also re-examine current OOD FSL benchmarks and analyze how the shift in the train and test task distributions may impact the reliability of OOD evaluations. We point out two concerns which we believe are not widely considered in the meta-learning community. First, unlike areas such as domain generalization where model selection challenges are more widely discussed [18, 23], we conduct to the best of our knowledge the first rigorous study demonstrating the difficulty of model selection due to the shift in the validation and test task distributions in FSL benchmarks. Second, because the OOD scenario in meta-learning does not assume a specific test task distribution, there is room for different test distributions to be used for evaluation. We show that comparing which meta-learning method performs better can be unreliable not only over different OOD FSL benchmarks, but also within a single benchmark depending on the number of novel classes.

Our main contributions are: **i)** We clearly outline both ID and OOD FSL evaluation scenarios and explain why most popular FSL benchmarks target OOD evaluation (Section 3). **ii)** We provide realistic examples of the ID scenario and show that the performance of popular meta-learning methods can drastically differ in ID vs. OOD scenarios (Section 4). **iii)** For existing OOD FSL benchmarks, we highlight concerns with a) current model selection strategies for meta-learning methods, and b) the reliability of meta-learning method comparisons (Section 5). **iv)** To remedy these concerns, we suggest suitable modifications to the current FSL benchmarks to allow for ID evaluation, and explain how to construct FSL benchmarks to provide more reliable OOD evaluation. Our hope in highlighting these evaluation concerns is for future researchers to consider them when evaluating newly proposed meta-learning methods or designing new FSL benchmarks.

2 Related Work

Current FSL benchmarks. A plethora of few-shot image classification benchmarks (*e.g., mini*-ImageNet (*mini* in short) [42], CIFAR-FS [4]) have been developed for FSL evaluation. These benchmarks typically provide three disjoint sets of classes (base, validation, novel) taken from standard classification datasets, *e.g.,* ImageNet or CIFAR-100 [36, 32, 42]. Training, val, and test tasks are then constructed from these classes respectively, which, as we discuss in Section 3, *can induce a shift in their corresponding task distributions*. Distribution mismatch can be particularly large with non-random splits created at the super class level, *e.g.,* FC-100 [30], or dataset level, *e.g.,* Meta-Dataset [40]. Recently, Arnold and Sha [2] propose an automated approach to construct different class splits from the same dataset to allow for varying degrees of task distribution shifts; Triantafillou et al. [41] separate the distribution shifts on Meta-Dataset into weak vs. strong generalization depending on whether the novel classes are taken from the used training datasets or not. Both works find that the meta-learning methods that perform better in one distribution shift scenario might perform worse in another, providing further evidence to our OOD performance comparison inconsistency argument in Section 5.2. Beyond these canonical ways of task construction through a set of classes, Ren et al. [35] propose new benchmarks in a new flexible few-shot learning (FFSL) setting, where the aim is to classify examples into a context instead of an object class. During testing, they perform OOD evaluation on unseen contexts. Inspired by their approach, we also provide an FSL benchmark where tasks are specified by contexts (Section 4), though we differ by exploring ID evaluation.

Mismatch between meta-learning theory/methodology and evaluation. Despite theoretical works which analyze meta-learning OOD generalization [11, 13], there are many theoretical meta-learning works [e.g., 1, 3, 8, 22] that first assume the train and test tasks are *iid* sampled from the same distribution despite validating their analyses on OOD FSL benchmarks. Additionally, several popular meta-learning methods [25, 16, 31] that are motivated in the ID scenario are largely evaluated on OOD benchmarks (see Appendix A). Prior work of Lee et al. [24] explores ID vs. OOD; however they treat the FSL setup as ID when the disjoint base and novel classes are from the same dataset and OOD only when they are not (*e.g.,* base from ImageNet, novel from CUB). We emphasize that even the disjointedness of base and novel from the same dataset can create a task distribution shift and hence unlike [24] we do not consider current FSL benchmarks (like *mini*) to be ID (Section 3).

FSL ID evaluation. We are unaware of any FSL classification benchmarks that are explicitly advertised for ID evaluation. As we discuss in Section 4, a growing number of works [7, 12, 20, 22, 27]

use meta-learning for personalized federated learning, but do not clearly discuss the in-distribution nature of these benchmarks nor how they differ from standard FSL benchmarks. In recent work, Chen et al. [10] extend current FSL benchmarks to evaluate their proposed method both ID and OOD, but only to further improve OOD performance on the original FSL benchmark's novel classes. Our work uses a similar setup for constructing ID evaluations with current OOD FSL benchmarks, but focuses on showing that certain meta-learning methods/design choices can improve OOD performance at the cost of ID performance. Prior work [17, 34] on incremental few-shot/low-shot learning also explores performance on both base and novel classes simultaneously. However, they differ in their methodology, as they use supervised learning (not meta-learning) to classify over the base classes.

OOD evaluation in other fields. Train and test distribution shifts are also found in domain generalization [5, 29] where the goal is to find a model that works well for a different test environment. Due to this shift, Gulrajani and Lopez-Paz [18] specifically discuss difficulties in performing model selection in domain generalization benchmarks. They argue that it is the responsibility of the method designer (not benchmark designer) to determine model selection strategies for their method, and propose several model selection methods, mainly targeting hyperparameter selection. Unlike domain generalization, FSL benchmarks often have a pre-determined disjoint set of validation classes to construct validation tasks, so the need for developing model selection methods may be less obvious. In Section 5, motivated by [18], we explore strategies for model selection for meta-learning. However, in contrast to [18], we focus on algorithm snapshot selection for meta-learning, which is required by hyperparameter selection as a subroutine (exact definitions see Section 5).

3 FSL benchmarks: Background & Focus on OOD evaluation

Background and notation. In this work, we employ a general definition of a meta-learning FSL task: a task $\mathcal{T}$ is a distribution over the space of support and query dataset pairs (S, Q), where S, Q are two sets of examples from an example space $\mathcal{X} \times \mathcal{Y}$. The support set S is used by an algorithm to produce an adapted model which is then evaluated by the corresponding query set Q. Each time we interact with a task $\mathcal{T}$, an (S, Q) pair is sampled *iid* from $\mathcal{T}$, the mechanism for which depends on the specific application; we provide multiple examples below. As discussed in Section 1, meta-learning aims to learn an algorithm over a distribution of tasks $\mathbb{P}(\mathcal{T})$. During meta-training, we assume access to N pairs of $\{(S_i, Q_i)\}_{i\in[N]}$ sampled from a *training task distribution*[‡] $\mathbb{P}_{\text{tr}}(\mathcal{T})$ in the following way: first, N tasks are sampled *iid* from the training task $\mathcal{T}_i \sim \mathbb{P}_{\text{tr}}(\mathcal{T})$; then, for each task $\mathcal{T}_i$, a support query pair $(S_i, Q_i) \sim \mathcal{T}_i$ is sampled *iid*. During meta-testing, we assume there is a *test task distribution* $\mathbb{P}_{\text{te}}(\mathcal{T})$ where fresh (S, Q) samples are similarly sampled based on $P_{\text{te}}(\mathcal{T})$. We define a learning scenario to be in-distribution (ID) if $\mathbb{P}_{\text{tr}} = \mathbb{P}_{\text{te}}$ and out-of-distribution (OOD) if $\mathbb{P}_{\text{tr}} \neq \mathbb{P}_{\text{te}}$. Whenever $\mathbb{P}_{\text{tr}} = \mathbb{P}_{\text{te}}$, the induced train and test marginal distributions of (S, Q) are also identical.

Construction of (S, Q) pairs in FSL. Most popular FSL benchmarks share a similar structure: they provide three disjoint sets of classes: base classes $\mathcal{C}_B$, validation classes $\mathcal{C}_V$, and novel classes $\mathcal{C}_N$, where any class c in these sets specifies a distribution $\mathbb{P}_c$ over the example space $\mathcal{X}$. An n-way k-shot q-query task $\mathcal{T}_{\boldsymbol{c}}$ in these benchmarks is specified by a length n non-repeated class tuple $\boldsymbol{c} = (c_1, \ldots, c_n)$ where $\boldsymbol{c} \in [\mathcal{C}^n] \coloneqq \{(d_1, \ldots, d_n) \in \mathcal{C}^n : d_i \neq d_j, \forall i \neq j\}$. $\mathcal{T}_{\boldsymbol{c}}$ generates random (S, Q) pairs in the following way: For every class c_i, k support examples $S_i \sim (\mathbb{P}_{c_i})^k$ and q query examples $Q_i \sim (\mathbb{P}_{c_i})^q$ are sampled. The support and query set is formed by the union of such labelled examples from each class: $S = \cup_{i=1}^n \{(x, i), \forall x \in S_i\}$, $Q = \cup_{i=1}^n \{(x, i), \forall x \in Q_i\}$. By specifying the base and novel classes, the FSL benchmark has provided a collection of tasks for training $\{\mathcal{T}_{\boldsymbol{c}} : \boldsymbol{c} \in [\mathcal{C}_B^n]\}$ and test $\{\mathcal{T}_{\boldsymbol{c}} : \boldsymbol{c} \in [\mathcal{C}_N^n]\}$. These sets can be extremely large. For example, in *mini*, which has 64 base classes, the total number of 5-way training tasks is $\frac{64!}{(64-5)!} \approx 9.1 \times 10^8$. However, it is not explicitly specified what underlying task distribution $\mathbb{P}(\mathcal{T})$ generates these sets of tasks. We believe this may have led prior work to discuss FSL benchmarks in the context of ID evaluation [e.g., 1, 3, 8], contrary to what we outline below.

Current FSL benchmarks target OOD evaluation. We now informally discuss our reasoning for arguing that current FSL benchmarks reflect OOD evaluation (we provide a more formal proof by contradiction in Appendix B). In particular, if the training and test tasks in FSL benchmarks are indeed *iid* sampled from the same underlying task distribution, then this underlying distribution must be induced by a distribution over class tuples of an even larger class set $\mathcal{C}_L$ ($\mathcal{C}_L \supseteq (\mathcal{C}_B \cup \mathcal{C}_V \cup \mathcal{C}_N)$).

[‡]We note that "training task distribution" here refers to the true underlying distribution of training tasks, not the empirical distribution supported over the finite set of sampled training tasks.

We consider the following dichotomy:

i) when $|\mathcal{C}_L| = \mathcal{O}(nN)$: In this case, the total number of classes nN covered by the sampled tasks (counting repetition) is greater than the number of underlying classes. Then with high probability, both the training and test tasks would each cover a significant portion of all the classes in $\mathcal{C}_L$, making it extremely unlikely to have an empty intersection as in the current FSL benchmarks.

ii) when $|\mathcal{C}_L| = \Omega(nN)$: In this alternative case, the total number of classes (even counting repetitions) used by sampled tasks is still smaller than the number of underlying classes $|\mathcal{C}_L|$. Thus the sampled tasks cannot cover all the underlying classes. Under this regime, the number of classes covered by the training tasks alone would scale linearly with N, as repeating an already-seen class in a new task sample is relatively rare. Since FSL benchmarks typically use a large number of training tasks during meta-training ($N > 10^3$), it is improbable that all the training tasks would together only cover a very low number of classes (64 in the case of *mini*).

Randomized class partitions do not imply randomized task partitions. Another issue that may cause some to view the current FSL benchmarks as performing ID evaluation is that in some of these benchmarks, the base, val, novel classes are random partitions of *iid* drawn classes from a class level distribution (specifically *mini*ImageNet, CIFAR-FS; but not FC-100, *tiered*ImageNet as the classes are not partitioned randomly). The logic here is that in standard machine learning practice, randomly partitioning *iid* sampled data points into train and test guarantees that the train and test samples are drawn *iid* from the same underlying distribution. However, it is important to notice that *the first class citizen in common FSL benchmarks is not a class, but a task* (represented by a class tuple). So, only a randomized partition of *iid* sampled class tuples would guarantee in-distribution sampling.

How can we view $\mathbb{P}_{\text{tr}}, \mathbb{P}_{\text{te}}$ in common FSL benchmarks? Based on the discussion above, we need to view train and test tasks in current FSL benchmarks as coming from different distributions, *i.e.,* $\mathbb{P}_{\text{tr}} \neq \mathbb{P}_{\text{te}}$. In order to ensure that both sets of tasks are still sampled *iid* from their respective distributions, it is convenient to view the train/test tasks as being *iid* sampled from a uniform distribution over all possible train/test class tuples induced by $\mathcal{C}_B/\mathcal{C}_N$ *i.e.,* $\mathbb{P}_{\text{tr}} = \mathbb{P}_{\mathcal{C}_B} \coloneqq \text{Unif}(\{\mathcal{T}_{\boldsymbol{c}} : \boldsymbol{c} \in [\mathcal{C}_B^n]\})$ and test $\mathbb{P}_{\text{te}} = \mathbb{P}_{\mathcal{C}_N} \coloneqq \text{Unif}(\{\mathcal{T}_{\boldsymbol{c}} : \boldsymbol{c} \in [\mathcal{C}_N^n]\})$ — a view which we will adopt in the rest of the paper.

4 Evaluating In-Distribution Performance

Although (as discussed in Section 3) current FSL benchmarks target OOD evaluation, we now explore example applications where ID generalization is instead required, and provide easily constructible benchmarks mirroring these scenarios. As we will show, this distinction is important because meta-learning methods may perform markedly different in ID vs OOD scenarios.

Example 1 **(Federated Learning):** Multiple works [7, 12, 20, 22, 27] have considered applying meta-learning methods in federated learning, in which the goal is to learn across a distributed network of devices [28, 26]. Meta-learning can produce personalized models for unseen devices/users, improving over a single globally-learned model's performance. In this setting, a popular benchmark is the FEMNIST [6] handwriting recognition dataset. For FEMNIST, we assume there exists a distribution of writers $\mathbb{P}(\texttt{id})$ in a federated network where each writer (with a unique `id`) is associated with a few-shot classification problem to recognize over the different character classes the writer has written for. We associate each writer `id` with a task $\mathcal{T}_{\texttt{id}}$ which randomly generates a support set with one random example per class and a query set with varying number of random examples per class.

ID evaluation on FEMNIST. We are given a total of ~ 3500 writers sampled *iid* from $\mathbb{P}(\texttt{id})$ and we randomly partition them into a $2509/538/538$ split for training, validation, and test tasks, following similar practices used in prior FL work [20, 7]. Note that this random split is performed at the task/`id` level. As such, we can treat the training and test tasks as being sampled *iid* from the same task distribution, unlike current FSL benchmarks.

Example 2 **(Online Recommendation):** Ren et al. [35] propose the use of *Zappos* [43] dataset as a meta-learning benchmark where each task is a binary classification of shoe images into an attribute context. This mimics an online shopping recommendation problem, where each user has different shoe preferences based on specific shoe attributes (hence a single global predictive model would not perform well), and the recommendation system must quickly learn a user's likes/dislikes through a few interactions. In this simplified setup, we fix a universal set of shoe attributes $\mathcal{A}$, and each user's preference is represented by a specific pair of unique attributes $\boldsymbol{a} = (a_1, a_2)$, $a_1 \neq a_2$. A task $\mathcal{T}_{\boldsymbol{a}}$ representing a user with attribute preference $\boldsymbol{a}$ generates 2-way k-shot q-query (S, Q) pair by *iid* sampling $k + q$ examples both from the set of images that carry both attributes in $\boldsymbol{a}$ (positive

Table 1: Ranking in () of meta-algorithms' test performance on **i)** ID benchmarks FEMNIST, Zappos-ID (with either 1000 or 50 training tasks); and **ii)** OOD FSL benchmark *mini*ImageNet.

Dataset /	FEMNIST	Zappos-ID			*mini*ImageNet
Method	*w1s	2w10s	2w5s (1000 train tasks)	2w5s (50 train tasks)	5w5s
PN	$^{(1)}94.72 \pm 0.41\%$	$^{(1)}88.40 \pm 0.13\%$	$^{(1)}86.58 \pm 0.15\%$	$^{(1)}77.67 \pm 0.17\%$	$^{(3)}76.22 \pm 0.14\%$
Ridge	$^{(1)}94.71 \pm 0.42\%$	$^{(2)}88.01 \pm 0.14\%$	$^{(2)}85.56 \pm 0.16\%$	$^{(2)}74.75 \pm 0.16\%$	$^{(2)}77.20 \pm 0.15\%$
SVM	$^{(3)}94.22 \pm 0.45\%$	$^{(3)}87.75 \pm 0.14\%$	$^{(3)}85.12 \pm 0.16\%$	$^{(3)}74.06 \pm 0.17\%$	$^{(1)}77.72 \pm 0.15\%$
FO-MAML	N/A	$^{(4)}81.90 \pm 0.14\%$	$^{(4)}80.14 \pm 0.15\%$	$^{(4)}69.85 \pm 0.18\%$	$^{(4)}75.96 \pm 0.17\%$

examples) and from the set that does not (negative examples). Our task distribution is a uniform distribution over tasks of all attribute pairs $\mathbb{P}_{\mathcal{A}}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}} : \boldsymbol{a} \in [\mathcal{A}^2]\})$.

ID evaluation on Zappos. Unlike in [35], where Zappos is used to measure OOD performance by having disjoint train and test attributes $\mathcal{A}_{\text{tr}} \cap \mathcal{A}_{\text{te}} = \phi$, in this work we use Zappos for ID evaluations by *iid* sampling both meta-train and meta-test tasks from the same distribution $\mathbb{P}_{\mathcal{A}}(\mathcal{T})$. Through this modification of Ren et al.'s setup, we sample 1000 / 50 attribute pairs from an attribute set $|\mathcal{A}| = 36$ to construct 1000 / 50 training tasks (each with a randomly sampled (S, Q)) and evaluate ID performance on another 25000 test tasks sampled in the same way. Our evaluation setup captures a realistic setting where the goal is to learn an algorithm that can generalize to the entire online shopper population despite being trained only on a randomly chosen subset of shoppers.

Remark. Even with ID evaluation it is possible to encounter unseen classes/attributes in meta-test tasks, specifically when the number of meta-train tasks is smaller than the number of underlying classes/attributes (Section 3). However, a suitable number of *iid* sampled meta-train tasks is needed to ensure good performance, which would naturally encompass a larger set of meta-train classes than those considered by OOD FSL benchmarks. For example, there are still 16 attribute pairs from the test tasks that are unseen in the 1000 training tasks on Zappos-ID, but the dataset is still in-distribution since the sampling distributions of train and test attribute pairs (and thus of tasks) are identical.

ID benchmark results. We evaluate the ID performance of four popular meta-learning methods: Prototypical Networks (PN) [39], MetaOptNet-SVM (SVM) [25], MetaOptNet-Ridge Regression (RR) [25, 4] and FOMAML [15] on our identified ID FSL benchmarks (Table 1). Since FEMNIST's tasks have varying number of ways, FOMAML cannot be directly used due to the logit layer shape mismatch. We note that the performance order of the four methods are consistent on all three ID benchmarks yet surprisingly **almost completely opposite** to the performance order observed on the OOD benchmark *mini* (except for FOMAML). In terms of the actual performance differences, we notice that the ID performance advantage of PN over SVM becomes particularly large ($> 3\%$) when we reduce of the number of training tasks for Zappos-ID to 50; in contrast, on the OOD benchmark *mini*, SVM instead outperforms PN by 1.5% (a significant number as many newly proposed meta-learning methods often only report improvements over previous methods by 1.5% or less). These performance differences make it clear that the performance ranking flips between ID and OOD indeed exist, and as a result, these common OOD FSL benchmarks (like *mini*) cannot be used to compare ID performance without modifications, giving further evidence to the danger of such practices (see Section 2). To further understand this phenomenon, we propose a way to also enable ID performance evaluation over these common OOD FSL benchmarks and see if there still exists a difference in ID, OOD performance orders.

Modifying OOD FSL benchmarks for ID evaluation. From our previous discussion, we have shown that we can think of FSL training tasks as being sampled *iid* from the task distribution $\mathbb{P}_{\mathcal{C}_B} \coloneqq \text{Unif}(\{\mathcal{T}_{\boldsymbol{c}} : \boldsymbol{c} \in [\mathcal{C}_B^n]\})$. To conduct an in-distribution evaluation, we need to sample fresh test tasks *iid* from $\mathbb{P}_{\mathcal{C}_B}$. For a freshly sampled $\mathcal{T}_{\boldsymbol{c}} \sim \mathbb{P}_{\mathcal{C}_B}$, we also need to *iid* sample a fresh support query pair $(S, Q) \sim \mathcal{T}_{\boldsymbol{c}}$. To ensure independent sampling from the already seen meta-training (S, Q) pairs, we need to introduce new examples from $\mathbb{P}_c$ for each class $c \in \mathcal{C}_B$. Thus we construct slightly modified versions of four common FSL benchmarks **i)** *mini*ImageNet-Mod (*mini*-M) [42], in which we find (≈ 700) unused examples (from ImageNet) for each base class and use them to evaluate the performance over $\mathbb{P}_{\mathcal{C}_B}$. Here the original 600 examples of each base class are still only used for meta-training. **ii)** CIFAR-FS-Mod (*cifar*-M) [30], FC-100-Mod (FC-M) [4], and *tiered*ImageNet-Mod (*tiered*-M) [33]: As we don't have additional samples for base classes, we randomly partition each base class's current examples into an approximate $80/20$ split where the training tasks are constructed using the former and the latter is reserved for ID evaluation.

ID vs. OOD conflicts still exist. In addition to evaluating the four aforementioned meta-learning methods, we also consider two supervised pretraining methods: the supervised learning baseline (SB)

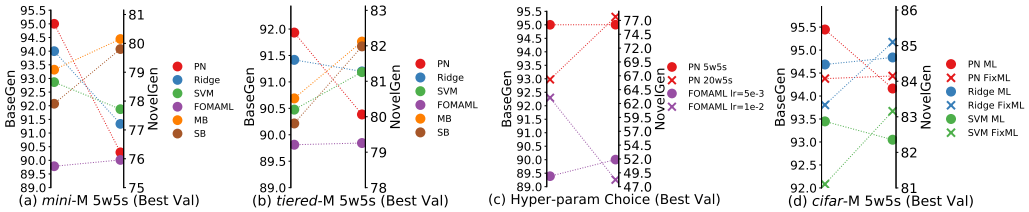


Figure 1: We show the BaseGen and NovelGen performance tradeoff (for best validation snapshots): over the choice of a set of four meta-learning and two supervised pre-training methods on *mini*-M (a) and *tiered*-M (b); over the number of ways to train PN on *mini*-M and different learning rates to train FOMAML on FC-M (c); over the use of FIX-ML (S, Q) generation strategy or not (ML) with SVM, RR and PN on *cifar*-M in (d).

[9] and Meta-Baseline (MB) [10]. Both of these methods have been shown to outperform many meta-learning methods on the current OOD FSL benchmarks. For succintness, we call the generalization performance over the training task distribution $\mathbb{P}_{\mathcal{C}_B}$ *BaseGen* and performance over $\mathbb{P}_{\mathcal{C}_N}$ *NovelGen* (we will refer to performance on validation tasks from $\mathbb{P}_{\mathcal{C}_V}$ *ValGen* in a later section). We plot the BaseGen and NovelGen performances of the aforementioned methods on *mini*ImageNet-Mod and *tiered*ImageNet-Mod in Figure 1(a)(b) (other datasets see Appendix E Figure 4(a)(b)), with a dotted line connecting BaseGen and NovelGen value of same learned algorithm snapshot of a meta-learning method. We see that the **ID/OOD performance order flips still exist within current FSL benchmarks themselves** (the dotted lines of different methods cross each other frequently), showing that **the issue of improving OOD at the expense of ID is a common realistic phenomenon for multiple benchmarks**. In addition, despite outperforming all meta-learning methods on NovelGen, the non-meta-learning methods SB and MB cannot beat the best meta-learning methods on BaseGen, which demonstrates their restrictive advantage only in the OOD setting. More broadly speaking, these OOD FSL datasets are constructed with a belief that there exists a commonality between the training and test task distributions so that an algorithm capturing this commonality using the training distribution alone would also generalize to the test tasks. Thus the phenomenon of improving OOD while sacrificing ID means the algorithm has in some sense failed to learn the essence/commonality from the given training tasks. Additionally, we see that the *BaseGen ranking of the four meta-learning methods over these common FSL bechmarks are exactly the same as the ranking over our two newly proposed ID benchmarks in Table 1*. We suggest that researchers who want to also test on their proposed methods' ID generalization performance can perform BaseGen evaluation method in our proposed way as it is a simple addition to their existing NovelGen evaluation setup.

Certain OOD training choices might harm ID generalization. In addition to checking the discrepancy of ID vs OOD generalization comparison of different meta-learning methods, we now ask, for a given meta-learning method, whether the meta-training choices found to be most effective on NovelGen would still be optimal for BaseGen.

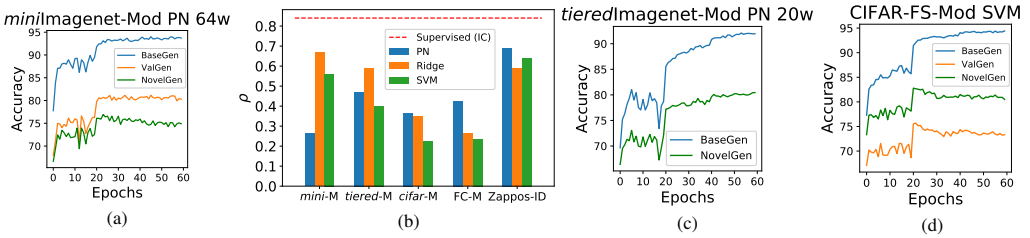
i) **Meta-training hyperparameters:** In Figure 1(c) we see that a higher learning rate when training FOMAML finds algorithms that have higher BaseGen, whereas lower learning rates are better for NovelGen. Additionally, we see that the proposed technique of training with more ways for PN in [39] can lead to better NovelGen performance but worse BaseGen performance than the 5-way trained PN whose training and test task configurations match.

ii) **Meta-training (S, Q) generation alternatives:** It was found in [38] that always using the same support examples for every base class when constructing S (FIX-ML) can improve NovelGen performance for several meta-learning methods over multiple OOD FSL benchmarks. However, restricting the training (S, Q)'s diversity sampled from $\mathbb{P}_{\mathcal{C}_B}$ seems counter-intuitive and we wouldn't expect to improve the in-distribution generalization BaseGen. In Figure 1(d), we indeed see that FIX-ML only improves NovelGen performance at the expense of BaseGen by training over a much less diverse set of tasks.

These two observations above caution us that **some training techniques to boost test accuracy on FSL benchmarks might only work for the OOD scenario but not the ID scenario.**

5 Challenges With Out-of-Distribution Evaluation

After identifying some example benchmarks for ID evaluation, we come back to the current OOD FSL benchmarks to further examine some subtle challenges in OOD evaluation. Here, instead of focusing on the distinction between ID vs. OOD, we now look at some reliability and inconsistency problems



(a) (b) (c) (d)

Figure 2: We plot the (Base, Val, Novel)Gen progression of 64(max)-way trained PN on *mini*-M in (a) and of SVM trained on *cifar*-M in (d). In (b) we compute the Kendall rank correlation coefficient (ρ) between the validation and test rankings of model snapshots for IC (trained on *cifar*) and algorithm snapshots for PN, SVM, RR on OOD datasets *mini*-M (last 40 epochs), *cifar*-M (last 40), FC-M (last 10), *tiered*-M (last 20) and ID dataset Zappos-ID (last 30); in (c) we show the BaseGen tracking NovelGen for 20-way trained PN on *tiered*-M.

within the OOD evaluation itself. In particular, we highlight two concerns: **1)** Despite providing a validation set of classes for task construction during meta-validation, it is not clear whether this is a reliable way to perform model selection. **2)** As there is no requirement on how similar the training and test task distributions need to be in the OOD setting, there can be inconsistencies in method comparisons when the evaluation setup is slightly modified.

5.1 Model Selection

To compare a set of meta-learning methods on the OOD test task distribution, one needs to *select* a representative algorithm learned by each method to be evaluated on the test task distribution. To do so, one should first decide what set of hyperparameter configurations to choose from for a given meta-learning method (we define the entire set of all hyperparameters for a training run as a *hyperparameter config*). For each such considered config, after its training is completed, we need to choose one of the algorithm snapshots saved during training to represent it (which we call *snapshot selection*). Then the set of hyperparameter configs are compared based on their respectively selected snapshots, and a single config is then chosen among them (which we term *hyperparameter selection*). This config's selected snapshot will represent the given meta-learning method to be finally evaluated on the test tasks. We refer to the combined problem of hyperparameter and snapshot selection as model selection. (See Appendix F.1 for a simplified example of snapshot and hyperparameter selection; see F.2 for the distinction between the commonly used technique *early-stopping* and snapshot selection.)

Snapshot vs Hyperparameter selection. If the snapshot selection strategy cannot reliably identify a snapshot with good test performance, it is possible that some hyperparameter configs will be unfairly represented by a mischosen snapshot, leading to erroneous hyperparameter selection. Thus we believe snapshot selection is a more fundamental problem and we focus our analyses on it (more on hyperparameter selection in Appendix F.5). In OOD FSL benchmarks, a reserved set of validation classes (disjoint from the set of novel classes) is provided, which as we have argued, provides tasks which are not sampled *iid* from the test task distribution. As a result, we ask: *is performing validation on the given validation tasks reliable, and, if not, are there other options?* In contrast, there is typically no need for such concerns in standard ID supervised learning, where the validation set is sampled *iid* from the test distribution.

5.1.1 *Option 1*: Snapshot selection tradition using ValGen.

By providing a set of validation classes, it has become the default practice for meta-learning methods to use ValGen performance for snapshot selection. However, because $\mathbb{P}_{\mathcal{C}_V}$ and $\mathbb{P}_{\mathcal{C}_N}$ are different tasks distributions, it is not clear whether a higher ValGen performance is *strongly correlated with* a higher NovelGen performance. In Figure 2(a), we plot the progression of ValGen and NovelGen of a 64-way trained PN on *mini*ImageNet-Mod 5w5s tasks. We notice that ValGen is consistently higher than NovelGen, indicating that meta-val performance is not an accurate estimator of NovelGen. More importantly, we see trendwise that while ValGen is generally non-decreasing, NovelGen starts to decrease after epoch 30. Thus the snapshot selected according to the best ValGen is not the snapshot with the best possible meta-test performance. In fact, this loss of NovelGen performance due to choosing the best ValGen model instead of the actual best NovelGen model can be particularly large, with values being 1.1% for SVM, 1.2% for RR, 0.7% for PN, and 0.9% for FOMAML on the FC-100 dataset. These performances losses for each method are especially concerning considering the differences among the best possible NovelGen performance of these different methods are often smaller than 1.5%.

Ranking similarity analysis. In light of the above observation, we ask a more general quantitative question: *How **similar** is the ranking of the training snapshots using meta-val performance (ValGen) **to** the ranking using the meta-test performance (NovelGen)?* To answer this, we compute the Kendall rank correlation coefficient[§] ρ [21] between the ValGen and NovelGen rankings of algorithm snapshots trained on four OOD FSL benchmarks (Figure 2(b)) and our ID benchmark Zappos-ID whose validation and test tasks come from the same distribution $\mathbb{P}_{\mathcal{A}}$. More concretely, for each meta-learning method and dataset combination, we save the algorithm snapshots (one from each epoch) throughout meta-training and rank these algorithm snapshots according to their ValGen and NovelGen value respectively. Then ρ is computed between these two rankings for this specific (meta-learning method,dataset) combination. For snapshot selection through ValGen to work reliably, we need ρ to be close to 1. For context, we also compute ρ for a standard supervised image classification problem (IC), where train, val and test examples are sampled from the same example-level distribution.

Unreliabity of ValGen snapshot selection. From Figure 2(b), we see that when using validation samples generated *iid* from the test distribution (Zappos-ID and supervised learning IC), the value of ρ is consistently higher than the OOD benchmarks, indicating the validation performance can more reliably track the trend of test performance in the ID setting than on the OOD FSL benchmarks. In particular, for the *cifar*-M and FC-M datasets, the ValGen ranking of algorithm snapshots seems to be only weakly correlated with the true meta-test ranking for all the meta-learning methods. In fact, the meta-val and meta-test rankings of the most useful snapshots can sometimes even be negatively correlated ($\rho \approx -0.12 < 0$ over all snapshots after epoch 30 in the training scenario shown in Figure 2(a)). These results show that on the OOD FSL benchmarks, snapshot selection using the pre-assigned validation tasks can sometimes be **unreliable/unable to identify a snapshot candidate with top-tier meta-test performance among all the snapshots.**

5.1.2 *Option 2*: Snapshot selection alternative using BaseGen.

Beyond using OOD validation tasks for snapshot selection, inspired by the domain generalization community [23, 18], we can alternatively also consider using the ID performance over the training task distribution for snapshot selection. Perhaps due to the lack of an ID evaluation setup in common FSL benchmarks, this possibility has not been widely considered. Enabled by our modifications of current FSL benchmarks, we can now evaluate the ID generalization performance (BaseGen) throughout training in addition to ValGen.

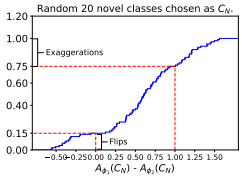
We plot how BaseGen and NovelGen progress for meta-learning methods trained on two different datasets in 2(c)(d). Here we see that on *tiered*-M, the BaseGen and NovelGen of PN both increase fairly consistently; thus picking the algorithm snapshot with the highest BaseGen performance (roughly the end-of-training snapshot) would also give close-to-best NovelGen. However, on *cifar*-M, after the learning rate drop at epoch 20, SVM's BaseGen keeps improving while NovelGen starts deteriorating. In this case, selecting snapshots according to the best BaseGen would pick a much worse snapshot than picking according to the best ValGen. (For concrete numbers of how much snapshot selection through BaseGen vs. ValGen could impact the chosen snapshot's NovelGen performance in each of these two cases, see Appendix F.3.) This ambiguity of whether ID or OOD Validation snapshot selection is better has also been documented in domain generalization, where Gulrajani and Lopez-Paz [18] find ID model selection can perform better in some settings while Koh et al. [23] find OOD validation model selection is better in others. Despite this ambiguity, we believe the commonly neglected **in-distribution (BaseGen) snapshot selection approach should be considered by users of OOD FSL benchmarks as a viable alternative** to the default ValGen selection approach in proper settings.

5.2 Inconsistencies in Meta-learning Method Performance Comparisons

After discussing concerns regarding OOD model selection for *a given meta-learning method*, we now analyze the reliability and consistency of conclusions drawn from comparing *different meta-learning methods*' OOD performance on these benchmarks. In particular, we focus on two cases:

***Inconsistency example 1*: Limited number of novel classes in a single benchmark.** Since in OOD FSL we specifically care about the learned algorithms' ability to quickly learn many unseen concepts, we should not be satisfied with an algorithm performing well only on tasks constructed from a small number of pre-selected novel classes. However, for many widely-used FSL benchmarks

[§] $\rho \in [-1, 1]$, $\rho \approx 0$ means no correlation, while $\rho = 1/\rho = -1$ means exactly same/opposite rankings.



(a)

		ϵ	$\lvert\mathcal{C}_B\rvert$	$\lvert\mathcal{C}_N\rvert$	$\lvert\mathcal{C}_L\rvert$	CF	IE,0.5%
(IC)	*mini*-M	0.5%	100 imgs/class			3%	1%
(i)	*mini*-M	0.5%	64	20	120	15%	25%
(ii)	*mini*-M	0.5%	64	20	552	20%	24%
(iii)	*mini*-M	1.0%	64	20	552	10%	12%
(iv)	*tiered*-M	0.5%	64	20	552	20%	23%
(v)	*tiered*-M	0.5%	64	160	552	0%	7%
(vi)	*tiered*-M	0.5%	351	20	552	9%	15%
(vii)	*tiered*-M	0.5%	351	160	552	0%	1%

(b)

Dataset	ZAPPOS-OOD		*mini*-OOD
Methods	2w10s	2w5s	5w5s
PN	$^{(1)}80.51 \pm 0.13\%$	$^{(1)}74.67 \pm 0.14\%$	$^{(3)}76.22 \pm 0.14\%$
Ridge	$^{(1)}80.51 \pm 0.13\%$	$^{(2)}73.50 \pm 0.14\%$	$^{(2)}77.20 \pm 0.15\%$
SVM	$^{(3)}79.70 \pm 0.13\%$	$^{(3)}72.84 \pm 0.14\%$	$^{(1)}77.72 \pm 0.15\%$
FO-MAML	$^{(4)}72.89 \pm 0.14\%$	$^{(4)}68.12 \pm 0.15\%$	$^{(4)}75.96 \pm 0.17\%$

(c)

Figure 3: In (a), we show the CDF plot of $A_{\phi_1}(\mathcal{C}_N) - A_{\phi_2}(\mathcal{C}_N)$ over 100 randomly chosen sets of $\mathcal{C}_N$ with 20 (out of total $|\mathcal{C}_L|$=120) novel classes each. In (b), for different values of true performance difference ϵ and values of underlying class size $|\mathcal{C}_L|$, training class size ($|\mathcal{C}_B|$), and evaluation class size ($|\mathcal{C}_N|$); we show the percentage of conclusion flips (CF) and improvement exaggerations (IE) with $\delta = 0.5\%$ computed over 100 evaluations. In (c), we demonstrate the inconsistencies in performance rankings for PN, SVM, RR, and FOMAML on two OOD benchmarks: Zappos-OOD and *mini*-OOD.

(*mini*ImageNet, CIFAR-FS, FC-100), only 20 novel classes are used for meta-testing. Ideally, even if we don't measure it, we would hope that our learned algorithm would also generalize to tasks made by other sets of classes different from the fixed small set of novel classes.

Formal setup. We suppose the existence of a much larger collection of classes $\mathcal{C}_L$, where the novel classes $\mathcal{C}_N$ used for meta-testing is a small random subset with each element class sampled uniformly and non-repeatedly from $\mathcal{C}_L$ and fixed thereafter for NovelGen evaluation. Ideally, our goal is to evaluate an algorithm snapshot ϕ on the task distribution $\mathbb{P}_{\mathcal{C}_L}$ (denote this performance by $A_\phi(\mathcal{C}_L)$), yet during evaluation we only have access to the novel classes in $\mathcal{C}_N$ and thus we can only compute the performance $A_\phi(\mathcal{C}_N)$. It is easy to see that when we randomize over different choices of $\mathcal{C}_N$, the expected performance over the sampled novel classes would match the true performance: $\mathbb{E}_{\mathcal{C}_N}[A_\phi(\mathcal{C}_N)] = A_\phi(\mathcal{C}_L)$. However, when using a single randomly sampled novel set, the estimator $A_\phi(\mathcal{C}_N)$ can have high variance (see Appendix F.6). Instead of relying on $A_\phi(\mathcal{C}_N)$ to directly estimate $A_\phi(\mathcal{C}_L)$, we ask a more relaxed question: for a pair of algorithms A_{ϕ_1} and A_{ϕ_2} (given by two meta-learning methods), if the true performance $A_{\phi_1}(\mathcal{C}_L) - A_{\phi_2}(\mathcal{C}_L) = \epsilon > 0$, how frequently will we observe an opposite conclusion *i.e.,* $\mathbb{P}(A_{\phi_1}(\mathcal{C}_N) < A_{\phi_2}(\mathcal{C}_N))$ over a randomly sampled $\mathcal{C}_N$ (we call this event *conclusion flip*)? Additionally, it is also possible that the observed performance difference on $\mathcal{C}_N$ is greater than the true difference ϵ by some amount $\delta > 0$. In this case, the NovelGen observation would make the algorithm snapshot ϕ_1 look better than ϕ_2 more than it actually is on $\mathcal{C}_L$. Thus we also ask what is the probability of $\mathbb{P}(A_{\phi_1}(\mathcal{C}_N) - A_{\phi_2}(\mathcal{C}_N) > \epsilon + \delta)$ and we call such events *improvement exaggerations*. To answer both these questions empirically, we first suggest some larger class sets $\mathcal{C}_L$ for *mini* and *tiered*. For both we select unused classes from ImageNet disjoint from the base and validation classes. For *tiered*, we use all the remaining $1000 - 351$ (base) -97 (novel) $= 552$ classes as $\mathcal{C}_L$ while for *mini*, we randomly choose 120 or 552 (to match $|\mathcal{C}_L|$ in *tiered*) unused classes. We fix these $\mathcal{C}_L$ choices in the following analysis.

Checking the frequency of conclusion flips and exaggerations. Figure 3(a) shows the empirical CDF of the performance differences $A_{\phi_1}(\mathcal{C}_N) - A_{\phi_2}(\mathcal{C}_N)$ computed over 100 randomly sampled size-20 novel class sets $\mathcal{C}_N$ for a fixed pair of PN and RR algorithm snapshots whose true performance difference over the larger 120 classes $\mathcal{C}_L$ is $\epsilon = 0.5\%$. In 15% of the cases the performance order is flipped from the true order, while in 25% of them improvements are exaggerated by more than 0.5% (total difference greater than 1%). Moreover, for some of the performance order flips, the observed performance difference can be quite negative $< -0.5\%$ thus significantly opposite to the true performance order. (Here for each run we evaluate both methods on 20,000 tasks sampled from $\mathbb{P}_{\mathcal{C}_N}$ in order to significantly reduce the randomness in estimating the true $A_{\phi_1}(\mathcal{C}_N), A_{\phi_2}(\mathcal{C}_N)$.)

Comparison to supervised learning. We check for the conclusion flip and improvement exaggeration frequency when only a random subset of the full test set (100 randomly drawn test images from each base class in *mini*-M) is used to compare two supervised learning image classification models with the same full test set performance difference of $\epsilon = 0.5$ (row (IC) in Table 3(b)). Here we see that **compared to supervised learning, the chances of getting an incorrect performance comparison** (row (i) in Table 3(b)) **is much higher for the meta-learning OOD FSL benchmarks** when evaluating only on 20 randomly chosen novel classes (as done in several FSL benchmarks).

Larger $|\mathcal{C}_L|$ makes it even less reliable but larger ϵ helps. If we were to care about an even larger set of underlying classes ($|\mathcal{C}_L| = 552$) despite still using only 20 random novel classes for evaluation

comparison, the conclusions are even less reliable (Table 3(b) (i) vs (ii)). On the other hand, we do see that the performance comparison becomes comparatively more consistent if the true performance difference ϵ is higher (1% in (iii) compared to 0.5% in (ii)), despite that there still exists a statistically significant chance (10%) of getting an opposite conclusion.

OOD evaluations in current FSL benchmarks. In practice, because **1)** we never specify exactly what and how big the underlying set of classes that we care about is, and **2)** some of the recent meta-learning methods (SVM vs PN on *cifar* in Table 2 of [25], R2-D2 vs GNN on *mini* in Table 1 of [4], FIX-ML [38]) sometimes only improve over the prior works by $< 1\%$, we believe researchers should **be aware of the possibility of getting a performance conclusion that is inconsistent** over a single randomly chosen and fixed set of 20 novel classes used by some of these benchmarks.

Actionable suggestions. Since the size of the unknown underlying larger class set $\mathcal{C}_L$ and the true performance difference ϵ might not be something one can directly control when designing the OOD benchmark, we now discuss two actionable choices that can reduce the chances of conclusion flips:

i) **Use more novel classes in the evaluation:** By comparing (iv) vs (v) and (vi) vs (vii) in Table 3, we see that the frequency of conclusion flips and improvement exaggerations are **much** lower when 160 novel classes are used as opposed to 20 when $|\mathcal{C}_L|$ is the same.
ii) **Train on more base classes:** The *tiered* dataset has more base classes (351 compared to 64 for *mini*) to train on. When comparing PN and RR snapshots trained on a modified version of *tiered* with fewer (randomly sampled 64 out of 351 to match *mini*) base classes, we see that the CF frequency is twice as high compared to when 351 base classes are used (Table 3(b)(iv) vs (vi)).

Based on these two trends, for more reliable comparisons of meta-learning methods' OOD performance we suggest using datasets like *tiered*ImageNet and MetaDataset (both with much larger set of base and novel classes) in addition to the smaller benchmarks like *mini*ImageNet, CIFAR-FS, and FC-100, which some recent works [e.g., 30, 4] still solely rely upon.

***Inconsistency example 2*: Inconsistency across multiple OOD FSL benchmarks.** Unlike the ID scenario where the training and test task distribution are the same, the similarity between training and test distributions in the OOD FSL benchmarks can vary significantly. Ideally, we want a meta-learning method to be consistently better on multiple OOD benchmarks with different type/degree of distribution shifts. Since Ren et al. [35] originally use the Zappos dataset for OOD evaluation, we also perform a similar evaluation on new attribute pairs based on their setup. At test time, we use an attribute set $\mathcal{A}'$ disjoint from the one used in the Zappos-ID setup $\mathcal{A}$, and sample attribute pairs from $\mathcal{A}'$ only. This induces a test task distribution $\mathbb{P}_{\mathcal{A}'}$ different from the training task distribution $\mathbb{P}_{\mathcal{A}}$. We evaluate different meta-learning methods on these Zappos-OOD tasks to see if the performance order is consistent with other OOD FSL benchmarks (Table 3(c)). Here we see that despite SVM outperforming RR and PN on *mini* NovelGen, the performance order of these three methods are completely flipped on Zappos-OOD. Similar observations can be made from TADAM underperforming PN in Table 2 of [35] despite TADAM being shown to outperform PN on the other more commonly-used FSL benchmarks. *This inconsistency over different types of OOD FSL benchmarks is in stark contrast to the consistency of performance rankings over the* 6 *different ID benchmarks* (FEMNIST, Zappos-ID, and the BaseGen results of the 4 current FSL benchmarks (Section 4)). Based on these findings, we caution meta-learning researchers to **be aware of such conclusion inconsistencies over different OOD FSL scenarios** and **reason carefully about the generality of their empirical findings** when using only specific types of OOD datasets.

6 Conclusion

In this paper, we categorize meta few-shot learning evaluation into two settings: in-distribution (ID) and out-of-distribution (OOD). After explaining why common FSL benchmarks reflect OOD evaluation, we identify realistic needs for ID FSL evaluation and provide new benchmarks as well as suggestions on how to modify existing OOD FSL benchmarks to allow for ID evaluation. Through experiments performed on these ID benchmarks, we demonstrate a surprising phenomenon that many meta-learning methods/training techniques improve OOD performance while sacrificing ID performance. Beyond this, through quantitative analyses, we show that even in the OOD scenario, current FSL benchmarks may present subtle challenges with both model selection for a given meta-learning method and reliable performance comparisons of different methods. For these concerns, we provide initial suggestions and alternatives with the hope of alleviating these issues. Overall, we aim to raise awareness about the dichotomy of FSL evaluation and to motivate the meta-learning community to collectively reason about ways to improve both ID and OOD methodology and evaluation.

Acknowledgements. This work was supported in part by the National Science Foundation Grant IIS1838017, a Google Faculty Award, a Facebook Faculty Award, and the CONIX Research Center. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the NSF or any other funding agency.

References

[1] M. Al-Shedivat, L. Li, E. Xing, and A. Talwalkar. On data efficiency of meta-learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1369–1377. PMLR, 2021.

[2] S. M. Arnold and F. Sha. Embedding adaptation is still needed for few-shot learning. *arXiv preprint arXiv:2104.07255*, 2021.

[3] Y. Bai, M. Chen, P. Zhou, T. Zhao, J. D. Lee, S. Kakade, H. Wang, and C. Xiong. How important is the train-validation split in meta-learning? *arXiv preprint arXiv:2010.05843*, 2020.

[4] L. Bertinetto, J. F. Henriques, P. H. Torr, and A. Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019.

[5] G. Blanchard, G. Lee, and C. Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24:2178–2186, 2011.

[6] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[7] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

[8] J. Chen, X.-M. Wu, Y. Li, Q. LI, L.-M. Zhan, and F.-l. Chung. A closer look at the training strategy for modern meta-learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[9] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang. A closer look at few-shot classification. *International Conference on Learning Representations*, 2019.

[10] Y. Chen, X. Wang, Z. Liu, H. Xu, and T. Darrell. A new meta-baseline for few-shot learning. *arXiv preprint arXiv:2003.04390*, 2020.

[11] S. S. Du, W. Hu, S. M. Kakade, J. D. Lee, and Q. Lei. Few-shot learning via learning the representation, provably. *arXiv preprint arXiv:2002.09434*, 2020.

[12] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33, 2020.

[13] A. Fallah, A. Mokhtari, and A. Ozdaglar. Generalization of model-agnostic meta-learning algorithms: Recurring and unseen tasks. *arXiv preprint arXiv:2102.03832*, 2021.

[14] C. Finn and S. Levine. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=HyjC5yWCW`.

[15] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

[16] C. Finn, K. Xu, and S. Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527, 2018.

[17] S. Gidaris and N. Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018.

[18] I. Gulrajani and D. Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[19] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[20] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[21] M. G. Kendall. *Rank correlation methods*. Griffin, London, 3d ed. edition, 1962.

[22] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar. Adaptive gradient-based meta-learning methods. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/f4aa0dd960521e045ae2f20621fb4ee9-Paper.pdf.

[23] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020.

[24] H. B. Lee, H. Lee, D. Na, S. Kim, M. Park, E. Yang, and S. J. Hwang. Learning to balance: Bayesian meta-learning for imbalanced and out-of-distribution tasks. *International Conference on Learning Representations*, 2020.

[25] K. Lee, S. Maji, A. Ravichandran, and S. Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019.

[26] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[27] S. Lin, G. Yang, and J. Zhang. A collaborative learning framework via federated meta-learning. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 289–299, Los Alamitos, CA, USA, dec 2020. IEEE Computer Society. doi: 10.1109/ICDCS47774.2020.00032. URL https://doi.ieeecomputersociety.org/10.1109/ICDCS47774.2020.00032.

[28] H. B. McMahan et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1), 2021.

[29] K. Muandet, D. Balduzzi, and B. Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013.

[30] B. N. Oreshkin, P. Rodriguez, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, 2018.

[31] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pages 113–124, 2019.

[32] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

[33] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

[34] M. Ren, R. Liao, E. Fetaya, and R. Zemel. Incremental few-shot learning with attention attractor networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/e833e042f509c996b1b25324d56659fb-Paper.pdf.

[35] M. Ren, E. Triantafillou, K.-C. Wang, J. Lucas, J. Snell, X. Pitkow, A. S. Tolias, and R. Zemel. Flexible few-shot learning with contextual similarity. *arXiv preprint arXiv:2012.05895*, 2020.

[36] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[37] J. Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[38] A. Setlur, O. Li, and V. Smith. Is support set diversity necessary for meta-learning? *arXiv preprint arXiv:2011.14048*, 2020.

[39] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017.

[40] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol, et al. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020.

[41] E. Triantafillou, H. Larochelle, R. Zemel, and V. Dumoulin. Learning a universal template for few-shot dataset generalization. In *International Conference on Machine Learning*, 2021.

[42] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[43] A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 192–199, 2014.

Appendix

Appendix Outline

A. Assumption (ID) and Evaluation (OOD) Mismatch Examples

B. Formal Analysis on Why Current FSL Benchmarks Are OOD

C. Overview of Notations for (S, Q) Sampling in ID and OOD Benchmarks

D. Dataset Preprocessing and Hyperparameter Details

E. Additional Results on Evaluating ID Performance

F. Additional Discussion and Results on OOD Evaluation

A Assumption (ID) and Evaluation (OOD) Mismatch Examples

To illustrate the mismatch between meta-learning theory/methodology and evaluation (Section 2), below are some examples of works that motivate commonly-used meta-learning methods in the in-distribution setting, but largely evaluate empirical performance on OOD FSL benchmarks. For convenience, we provide exact lines from the original works that refer to the ID scenario. Our aim is not to draw attention to these works specifically, but to highlight the ubiquity of the divide between theory and practice in current meta-learning literature.

- Lee et al. [25] (Section 3.1): "It is often **assumed that the training and test set are sampled from the same distribution** and the domain is mapped to a feature space using an embedding model f_ϕ parameterized by ϕ ".
- Rajeswaran et al. [31] (Section 2.1): "... a collection of **meta-training** tasks $\{\mathcal{T}_i\}_{i=1}^M$ **drawn from** $P(\mathcal{T})$ ... At **meta-test** (deployment) time, when presented with a dataset $\mathcal{D}_j^{\text{tr}}$ corresponding to a **new task** $\mathcal{T}_j \sim P(\mathcal{T})$." Notice that the training and test tasks are all sampled from the same task distribution $P(\mathcal{T})$.
- Finn et al. [16] (Section 3): "To do so, meta-learning algorithms require a set of **meta-training and meta-testing tasks drawn from some distribution** $p(\mathcal{T})$. The key assumption of learning-to-learn is that the tasks in this distribution share common ... ".

B Formal Analysis on Why Current FSL Benchmarks Are OOD

In this section we provide formal arguments for the informal statements in Section 3, which explain why it is improbable for the train and test tasks in the current FSL benchmarks to be *iid* sampled from the same underlying distribution.

Formal Setup. If we believe that both train and test tasks in current FSL benchmarks are sampled from the same underlying task distribution, then this shared task distribution (where each task is specified by a class tuple) must cover a larger set of underlying classes $\mathcal{C}_L$ which would contain both the base classes and novel classes as subsets $\mathcal{C}_L \supseteq (\mathcal{C}_B \cup \mathcal{C}_N)$. For convenience, we represent the classes in this set with $\mathcal{C}_L \coloneqq \{1, \ldots, L\}$ where the task distribution (from which the train and test tasks are *iid* sampled) is induced by a probabilistic distribution over $n-$way non-repeating tuples $\boldsymbol{c} \coloneqq (c_1, \ldots, c_n) \in [\mathcal{C}_L^n]$, denoted by $\mathbb{P}_L(\boldsymbol{c})$. To sample a task from this larger task distribution, we sample $\boldsymbol{c} \sim \mathbb{P}_L$ and take the corresponding task $\mathcal{T}_{\boldsymbol{c}}$. Notice that this task distribution can be more general than $\mathbb{P}_{\mathcal{C}_N}$ or $\mathbb{P}_{\mathcal{C}_B}$, as $\mathbb{P}_L(\boldsymbol{c})$ does not have to be a uniform distribution over all possible class tuples.

Definition 1 (Probability of observing a class in a single draw)**.** *The indicator event of observing a class i anywhere in a randomly drawn class tuple $\boldsymbol{c} \sim \mathbb{P}_L$ can be represented by $\sum_{j=1}^n \mathbb{1}(c_j = i)$, since it is impossible to observe the same class more than once in the same tuple. We denote the probability of this event by $p_i \coloneqq \mathbb{P}\left(\left(\sum_{j=1}^n \mathbb{1}(c_j = i)\right) = 1\right) = \sum_{j=1}^n \mathbb{P}(\mathbb{1}(c_j = i) = 1) = \sum_{j=1}^n \mathbb{P}(c_j = i)$.*

Lemma 1. *The sum of the probability of observing a class i in a single class tuple draw over all the classes $i \in \mathcal{C}_L$ is equal to n, i.e., $\sum_{i=1}^L p_i = n$.*

Proof of Lemma 1. We know that $\sum_{i=1}^{L} p_i = \sum_{i=1}^{L} \sum_{j=1}^{n} \mathbb{P}(c_j = i)$ by simply plugging in the definition of p_i. Since $\sum_{i=1}^{L} \mathbb{P}(c_j = i) = 1$, by exchanging the summations we get, $\sum_{i=1}^{L} p_i = \sum_{j=1}^{n} \sum_{i=1}^{L} \mathbb{P}(c_j = i) = \sum_{j=1}^{n}[1] = n$. □

Assumption 1 (Every class must have nonzero probability to be sampled)**.** *To avoid degeneracy, we assume that each class has a minimum non-zero probability of being sampled in a class tuple:* $\forall i \in \{1, \ldots, L\}$, $1 \geq p_i \geq \frac{\gamma n}{L}$, *where* $\gamma \in (0, 1]$. *Notice* γ *is strictly greater than 0 to avoid the degenerate case where a class would almost surely never be sampled in any class tuple. If there exists such a class, then we can prune the set* $\mathcal{C}_L$ *accordingly and use the pruned set (which now has every class with nonzero probability) as our new* $\mathcal{C}_L$.

Remark. Note that the task distribution induced by the probability values $\{p_i\}_{i=1}^{L}$ in Assumption 1 is a relaxed form of the uniform distribution Unif$(\{\mathcal{T}_{\boldsymbol{c}} \coloneqq \boldsymbol{c} \in [\mathcal{C}_L^n]\})$ over all non-repeating class tuples spawned by $\mathcal{C}_L$. This case can be recovered by setting $p_i = \frac{n}{L}$, $\forall i \in [L]$.

Suppose there are N total *iid* random draws $\{\boldsymbol{c}^{(k)}\}_{k=1}^{N}$ of class tuples from $\mathbb{P}_L$ (every $\boldsymbol{c}^{(k)} \in [\mathcal{C}_L^n]$), then the event of observing a class i in any of these N class tuple draws is exactly the complement of the event that the class does not appear in any of these tuples.

Definition 2 (Observing a class at least once in N draws)**.** *We denote the indicator random variable of observing a class* i *in any of the* N *draws by*

$$X_{i,N} \coloneqq 1 - \mathbb{1}\left(\left(\sum_{k=1}^{N} \mathbb{1}(i \in \boldsymbol{c}^{(k)})\right) = 0\right) \in \{0, 1\}. \tag{1}$$

Then we have $\mathbb{E}[X_{i,N}] = \mathbb{P}(X_{i,N} = 1) = 1 - (1 - p_i)^N$. *We denote the random variable representing the total number of unique classes observed in* N *draws as* Z, *which can be expressed by*

$$Z = \sum_{i=1}^{L} X_{i,N}. \tag{2}$$

Remark. We note that the total number of unique classes seen (Z) in N *iid* draws **1)** must have at least n classes (even after a single class tuple is sampled, there would already be n different classes seen) and cannot be greater than the total number of classes possible, *i.e.,* $Z \in [n, L]$, and **2)** cannot be greater than the total number of (possibly overlapping) classes drawn, *i.e.,* $Z \leq nN$.

Lemma 2. *For notational convenience, let* $q_i \coloneqq 1 - p_i$. *Then, by Equation* (1) *and Assumption 1 we have:*

(a) $\mathbb{E}[Z] = L - \sum_{i=1}^{L}(1 - p_i)^N = L - \sum_{i=1}^{L} q_i^N$, (3)

(b) For $\{q_i\}_{i=1}^{L}$, $0 \leq q_i \leq 1 - \frac{\gamma n}{L}$ *and* $\sum_{i=1}^{L} q_i = L - n$. (4)

Now that we have set up the problem formulation, we provide Theorems 1, 2 describing properties of $\mathbb{E}[Z]$ and $\mathbb{V}[Z]$ which we will use to analyze the dichotomy described in the main paper (Section 3).

B.1 Lower Bound on $\mathbb{E}[Z]$

To achieve a lower bound of $\mathbb{E}[Z]$, we need to analyze the worst case class tuple distribution that makes the value $L - \sum_{i=1}^{L} q_i^N$ as small as possible. This amounts to maximizing the value of $\sum_{i=1}^{L} q_i^N$ under the constraints for $\{q_i\}_{i=1}^{L}$ described in Lemma 2. We present an upper bound for this constrained maximization objective below.

Theorem 1 (Lower bound on $\mathbb{E}[Z]$)**.** *The optimal value of the following constrained optimization problem in* (5) *is upper bounded by* $L\left(1-\frac{\gamma n}{L}\right)^N$.

$$\begin{aligned} &\max_{\{q_i\}_{i=1}^L} \quad \sum_{i=1}^{L} q_i^N \\ &\textit{subject to} \quad 0 \leq \; q_i \leq \; 1-\frac{\gamma n}{L}, \; \forall i \in [L] \\ &\qquad\qquad \sum_{i=1}^{L} q_i = L-n \end{aligned} \tag{5}$$

As a result, for Z *defined in* (2)*, directly applying* (3) *we get* $\mathbb{E}[Z] \geq L\left(1-(1-\frac{\gamma n}{L})^N\right)$.

To prove Theorem 1, we first provide a lemma describing the structure of the solution to the optimization problem in (5).

Lemma 3 (Structure of the optimal solution to (5))**.** *The optimal solution to optimization Objective* (5) *has the following form: out of the* L *variables* $\{q_i\}_{i=1}^L$*,* K *of them have value* $1-\frac{\gamma n}{L}$*,* $(L-K-1)$ *of them have value* 0*, and the last remaining variable has the value* $(L-n)-K(1-\frac{\gamma n}{L})$*, which must still be in the range of* $\left[0, 1-\frac{\gamma n}{L}\right]$*. This directly implies that the integer* K *must be satisfy* $\frac{L^2-nL}{L-\gamma n}-1 \leq K \leq \frac{L^2-nL}{L-\gamma n}$.

Proof of Lemma 3. Let us denote the optimal solution to Objective (5) by $(q_1^*, \ldots, q_n^*)$. Suppose that there exists a pair $q_k^*,\, q_j^*,\; k \neq j$, such that neither of them equals 0 or $1-\frac{\gamma n}{L}$. Then by changing the values of $q_k^*,\, q_j^*$ to be either $(1-\frac{\gamma n}{L}, q_k^*+q_j^*-1+\frac{\gamma n}{L})$ or $(0, q_k^*+q_j^*)$, the new q tuple would still be feasible while the value of Objective 5 would strictly improve because the function $q_k^N+q_j^N$ is strongly convex over $\mathbb{R}_{++}^2$. (Recall that a convex function over a closed interval can only take maximum value at either one of its two endpoints.) As a result, there can be no more than a single q_i^* in the optimal solution $(q_1^*, \ldots, q_L^*)$ that has a value of neither 0 nor $1-\frac{\gamma n}{L}$. Now, we denote the total number of q_i's in the optimal solution that has the value of $1-\frac{\gamma n}{L}$ by $K \in \mathbb{Z}$, then there must be at least $L-K-1$ values of 0, with the remaining term $(L-n)-K(1-\frac{\gamma n}{L}) \in \left[0, 1-\frac{\gamma n}{L}\right]$. Manipulating this inequality of K gives us the feasible range of K, $\frac{L^2-nL}{L-\gamma n}-1 \leq K \leq \frac{L^2-nL}{L-\gamma n}$. □

Proof of Theorem 1. We know from Lemma 3 that the optimal solution to the constrained maximization problem in Theorem 1 is given by the optimal solution to the reduced objective below.

$$\max_{K \in \{0,\ldots,L\}} \quad K \cdot \left(1-\frac{\gamma n}{L}\right)^N + \left[L-n-K\cdot\left(1-\frac{\gamma n}{L}\right)\right]^N \tag{6}$$

$$\text{subject to} \quad \frac{L^2-nL}{L-\gamma n}-1 \leq \; K \leq \; \frac{L^2-nL}{L-\gamma n} \tag{7}$$

Let the optimal value of K (minimum value if multiple are optimal) in the above optimization problem be K^*. Then the optimal value of the above is upper bounded by:

$$\begin{aligned} &K^* \cdot \left(1-\frac{\gamma n}{L}\right)^N + \left[L-n-K^*\cdot\left(1-\frac{\gamma n}{L}\right)\right]^N \\ &\leq (K^*+1)\cdot\left(1-\frac{\gamma n}{L}\right)^N \\ &\leq L\cdot\left(1-\frac{\gamma n}{L}\right)^N \end{aligned}$$

As a result, directly applying (3), we have:

$$\mathbb{E}[Z] \geq L\cdot\left[1-\left(1-\frac{\gamma n}{L}\right)^N\right]. \tag{8}$$

□

B.2 Upper bound on $\mathbb{V}[Z]$

We apply the Efron-Stein inequality to obtain the upper bound of the variance of Z, which we state here for convenience.

Lemma 4 (Efron-Stein's inequality)**.** *Let $S : \mathcal{Y}^N \to \mathbb{R}$ be a measurable function that is permutation invariant. Let the random variable U be given by $U = S(Y_1, \ldots, Y_N)$, where $(Y_1, \ldots, Y_N)$ is a random vector of N independent random variables in $\mathcal{Y}^N$. Then, we have:*

$$\mathbb{V}[U] \leq \frac{1}{2} \sum_{i=1}^{N} \mathbb{E}(U - U_i')^2, \tag{9}$$

where $U_i' = S(Y_1, ..., Y_i', ..., Y_N)$, and $\forall i \in [N]$, Y_i and Y_i' are drawn iid from the same distribution.

Theorem 2 (Upper bound on $\mathbb{V}[Z]$)**.** *For Z defined in* (2)*, the variance $\mathbb{V}[Z] \leq \frac{1}{2} n^2 N$.*

Proof of Theorem 2. By directly applying Lemma 4 on the permutation invariant measurable function $S : [\mathcal{C}_L^n]^N \to \mathbb{R}$, where $Z = S(\boldsymbol{c}^{(1)}, \ldots, \boldsymbol{c}^{(N)}) \coloneqq \sum_{i=1}^{L} X_{i,N}$, we have the variance $\mathbb{V}[Z] \leq \frac{1}{2} \sum_{i=1}^{N} \mathbb{E}(Z - Z_i')^2 \leq \frac{1}{2} \sum_{i=1}^{N} n^2 = \frac{1}{2} n^2 N$. Note that the last inequality holds because when we swap out one observed class tuple $\boldsymbol{c}^{(k)}$ (among the N total) with a different one $\boldsymbol{c}^{(k)'}$ (to get Z_i'), the total number of unique classes we observe can change by at most n, *i.e.,* $|Z - Z_i'| \leq n$. □

Using Theorems 1, 2, we now show that it is extremely unlikely for the commonly used FSL benchmarks, which have a (relatively) small number of train classes and disjoint train/test classes, to have their training and test tasks sampled *iid* from the same underlying task distribution. We break this analysis into the dichotomy presented in the main paper (Section 3).

B.3 When $|\mathcal{C}_L|$ is Small ($L = \mathcal{O}(nN)$)

Definition 3. *Let Z_{tr} be the total number of unique classes observed in N_{tr} iid drawn train class tuples (tasks) from $\mathbb{P}_L$ and similarly let Z_{te} be the total number of unique classes observed in N_{te} iid drawn test class tuples (tasks) from $\mathbb{P}_L$. Furthermore, denote the set of indices of unique train classes by $\mathcal{I}_{\text{tr}} \coloneqq \{j : \; X_{j,N_{\text{tr}}} = 1, j \in [N_{\text{tr}}]\}$ and the set of unique test classes be $\mathcal{I}_{\text{te}} \coloneqq \{j : X_{j,N_{\text{te}}} = 1, j \in [N_{\text{te}}]\}$. Under this notation, the probability of observing disjoint sets of train and test classes among the N_{tr} and N_{te} randomly drawn train and test class tuples can be denoted by $\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi)$.*

Theorem 3 (Upper bounding the probability of having disjoint train, test classes)**.** $\mathbb{P}(Z_{\text{tr}} + Z_{\text{te}} \leq L) \leq 4\left(1 - \frac{\gamma n}{L}\right)^{min(N_{\text{tr}}, N_{\text{te}})}$. *As a result,* $\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi) \leq \mathbb{P}(Z_{\text{tr}} + Z_{\text{te}} \leq L) \leq 4\left(1 - \frac{\gamma n}{L}\right)^{min(N_{\text{tr}}, N_{\text{te}})}$.

Proof of Theorem 3. Since the random variable $L - Z_{\text{tr}} \geq 0$, using Markov's inequality we have:

$$\begin{aligned}
\mathbb{P}\left(Z_{\text{tr}} \leq \frac{L}{2}\right) &= \mathbb{P}\left(L - Z_{\text{tr}} \geq L - \frac{L}{2}\right) && (10)\\
&\leq \frac{L - \mathbb{E}[Z_{\text{tr}}]}{(L - \frac{L}{2})} \\
&\leq \frac{2(L - \mathbb{E}[Z_{\text{tr}}])}{L} \\
&\leq 2\left(1 - \frac{\gamma n}{L}\right)^{N_{\text{tr}}}. \qquad \text{(using Theorem 1)} && (11)
\end{aligned}$$

Similarly, since $L - Z_{\text{te}} \geq 0$, we have

$$\mathbb{P}\left(Z_{\text{te}} \leq \frac{L}{2}\right) \leq 2\left(1 - \frac{\gamma n}{L}\right)^{N_{\text{te}}}. \tag{12}$$

Since $\mathbb{P}\left(Z_{\text{tr}} + Z_{\text{te}} \leq L\right) \leq \mathbb{P}\left(\left(Z_{\text{tr}} \leq \frac{L}{2}\right) \cup \left(Z_{\text{te}} \leq \frac{L}{2}\right)\right)$, applying the union bound yields:

$$\mathbb{P}\left(Z_{\text{te}} + Z_{\text{te}} \leq L\right) \leq 4\left(1 - \frac{\gamma n}{L}\right)^{\min(N_{\text{tr}}, N_{\text{te}})}. \tag{13}$$

When $Z_{\text{tr}} + Z_{\text{te}} > L$, by pigeonhole principle, the two sets of class indices $\mathcal{I}_{\text{tr}}$ and $\mathcal{I}_{\text{te}}$ must have non-empty intersection, *i.e.,* $\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} \neq \phi$. Taking the contra-positive of this claim, we see that $\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi$ implies the event $Z_{\text{tr}} + Z_{\text{te}} \leq L$. As a result, we have $\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi) \leq \mathbb{P}(Z_{\text{tr}} + Z_{\text{te}} \leq L) \leq 4\left(1 - \frac{\gamma n}{L}\right)^{\min(N_{\text{tr}}, N_{\text{te}})}$. □

Corollary 1. *If enough samples are observed i.e., if* $\min(N_{\text{tr}}, N_{\text{te}}) \geq \frac{\ln(4/\rho)L}{\gamma n}$*, then the probability of having no training and test classes intersection is upper bounded by* $\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi) \leq \rho$.

Proof of Corollary 1. By logarithm inequality $\ln(1+x) \geq \frac{x}{1+x}$, we have $\ln(L/(L-\gamma n)) \geq \gamma n/L$. Taking the reciprocal of the two sides, we have $\frac{L}{\gamma n} \geq \frac{1}{\ln(L/(L-\gamma n)}$. As a result, $\min(N_{\text{tr}}, N_{\text{te}}) \geq \frac{\ln(4/\rho)L}{\gamma n} \geq \frac{\ln(4/\rho)}{\ln(L/(L-\gamma n))} = \frac{\ln(\rho/4)}{\ln(1-\frac{\gamma n}{L})} = \log_{(1-\frac{\gamma n}{L})}(\frac{\rho}{4})$, where the last step uses the change of basis equality of logarithm. Thus, by $\min(N_{\text{tr}}, N_{\text{te}}) \geq \log_{(1-\frac{\gamma n}{L})}(\frac{\rho}{4})$, we have

$$\begin{aligned}
&\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi) \\
&\leq \mathbb{P}(Z_{\text{tr}} + Z_{\text{te}} \leq L) \\
&\leq 4\left(1 - \frac{\gamma n}{L}\right)^{\min(N_{\text{tr}}, N_{\text{te}})} \\
&\leq 4\left(1 - \frac{\gamma n}{L}\right)^{\log_{(1-\frac{\gamma n}{L})}(\frac{\rho}{4})} \\
&= 4 \cdot \frac{\rho}{4} \\
&= \rho.
\end{aligned}$$

□

Remark. From Corollary 1, we see that when the number of tasks sampled is larger than a multiple of the number of underlying classes (for example when $\rho = 0.01$, $\gamma = 0.5$, $n = 5$, $\min(N_{\text{tr}}, N_{\text{te}}) \geq \frac{\ln(4/\rho)L}{\gamma n} \approx 2.39L$), and equivalently, $L = \mathcal{O}(nN)$, the probability of having no training and test task classes intersecting is upper bounded by $\mathbb{P}(\mathcal{I}_{\text{tr}} \cap \mathcal{I}_{\text{te}} = \phi) \leq \rho$ (in our example, the probability is upper bounded by $\rho = 0.01$, which is a statistically rare event). In summary, in this case, we show that when $L \leq cnN$ for some small constant c, the probability of having no intersection between the training and test task classes is extremely small because it is very likely that the training tasks and test tasks would each cover a majority ($\geq 50\%$) of the entire set of classes.

B.4 When $|\mathcal{C}_L|$ is Large ($L = \Omega(nN)$)

In this alternate case, we analyze the scenario where the underlying set of classes is larger than the total number of tasks we sampled, for which we make the following assumption:

Assumption 2. *$L \geq nN$, i.e., even if we observe all the classes in the randomly drawn N tuples to be distinct, we still would not exhaust the much larger underlying set $\mathcal{C}_L$. In this setting, $L = \Omega(nN)$.*

Corollary 2. *By Assumption 2 and the Bernoulli inequality* $(1+x)^r \leq 1 + \frac{rx}{1-(r-1)x}$, $x \in (-1, \frac{1}{r-1}), r > 1$*, substituting* $x = \frac{-\gamma n}{L}$, $r = N$*, we can further lower bound the RHS of Equation* (8) *in Theorem 1:*

$$\mathbb{E}[Z] \geq \frac{\gamma n N}{(1+\gamma)} \tag{14}$$

Theorem 4 (Unlikely to observe only a small number of unique classes)**.** *For $\eta \in (0,1)$, the probability of observing totally fewer than $\frac{\eta\gamma nN}{1+\gamma}$ classes in N iid class tuple samples from $\mathbb{P}_L$ is at most $\frac{(1+\gamma)^2}{2(1-\eta)^2\gamma^2 N}$.*

Proof.

$$
\begin{aligned}
&\mathbb{P}\left(Z \leq \frac{\eta\gamma nN}{(1+\gamma)}\right) && (15)\\
&\leq \mathbb{P}(Z \leq \eta\,\mathbb{E}[Z]) \\
&= \mathbb{P}(Z \leq \mathbb{E}[Z] - (1-\eta)\,\mathbb{E}[Z]) \\
&\leq \mathbb{P}(|Z - \mathbb{E}[Z]| \geq (1-\eta)\,\mathbb{E}[Z]) \\
&\leq \frac{\mathbb{V}[Z]}{((1-\eta)\,\mathbb{E}[Z])^2} && \text{(by Chebyshev's inequality)} \\
&\leq \frac{\frac{1}{2}n^2N}{(1-\eta)^2\,\mathbb{E}[Z]^2} && \text{(using Theorem 2)} \\
&\leq \frac{\frac{1}{2}n^2N}{(1-\eta)^2\left(\frac{\gamma nN}{(1+\gamma)}\right)^2} && \text{(using Corollary 2)} \\
&= \frac{(1+\gamma)^2}{2(1-\eta)^2\gamma^2N} && (16)
\end{aligned}
$$

□

In summary, in this case, when $L \geq nN$, the probability of observing only a small fraction of nN classes in N tuple draws, scales with $1/N$. Because in practice a very large number of training tasks are used ($N \geq 10^5$), the probability of only observing fewer than hundreds of classes ($Z \leq 10^3$) in N class tuple samples would be extremely small. This means that we shouldn't treat the large number of tasks used during meta-training as being sampled *iid* from an underlying task distribution under the assumption that the number of task samples hasn't exceeded the total number of underlying classes.

B.5 Concluding Remarks

In the first part of the dichotomy, we show using Theorem 3 that when the number of underlying classes is smaller than nN, it is highly unlikely for the train and test classes to be completely disjoint.

In the second part of the dichotomy, we show using Theorem 4 that when the number of underlying classes is larger than nN, it is unlikely to observe only a few (very small fraction of nN) unique train classes — in fact the number of unique train classes observed would roughly speaking scale linearly with the number of task samples N.

Conclusion on why current FSL benchmarks target OOD. Note that in current FSL benchmarks **i)** there is no overlap of classes observed in the train and test tasks; and **ii)** the number of train (base) classes observed (*e.g.,* 64 for *mini*) is much smaller than the total number of train tasks (*e.g.,* $\approx 10^6$ for *mini*). Thus, the two sides of the dichotomy (above) when taken together leads us to reject the hypothesis/assumption of *iid* sampled train and test tasks in the current FSL benchmarks.

C Overview of Notations for (S, Q) Sampling in ID and OOD Benchmarks

Table 2: An overview of the notations used to describe each of the two steps: i) sampling the task from the training/test task distribution, and ii) sampling (S,Q) pair from the task; for the OOD benchmarks (*mini*, *cifar*, FC, *tiered*, Zappos-OOD) and ID benchmarks (FEMNIST, Zappos-ID).

Benchmark / Steps		**Step 1:** $\mathcal{T} \sim \mathbb{P}(\mathcal{T})$	**Step 2:** $(S, Q) \sim \mathcal{T}$
mini, *cifar*, FC, *tiered* (OOD)	Train	$\mathcal{T}_{\boldsymbol{c}_B} \sim \mathbb{P}_{\mathcal{C}_B} \coloneqq \text{Unif}(\{\mathcal{T}_{\boldsymbol{c}_B} : \boldsymbol{c}_B \in [\mathcal{C}_B^n]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{c}_B}$
	Test	$\mathcal{T}_{\boldsymbol{c}_N} \sim \mathbb{P}_{\mathcal{C}_N} \coloneqq \text{Unif}(\{\mathcal{T}_{\boldsymbol{c}_N} : \boldsymbol{c}_N \in [\mathcal{C}_N^n]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{c}_N}$
Zappos-OOD (OOD)	Train	$\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}} : \boldsymbol{a} \in [\mathcal{A}^2]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{a}}$
	Test	$\mathcal{T}_{\boldsymbol{a}'} \sim \mathbb{P}_{\mathcal{A}'}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}'} : \boldsymbol{a}' \in [\mathcal{A}'^2]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{a}'}$
FEMNIST (ID)	Train	$\mathcal{T}_{\texttt{id}} \sim \mathbb{P}(\texttt{id})$	$S, Q \sim \mathcal{T}_{\texttt{id}}$
	Test	$\mathcal{T}_{\texttt{id}} \sim \mathbb{P}(\texttt{id})$	$S, Q \sim \mathcal{T}_{\texttt{id}}$
Zappos-ID (ID)	Train	$\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}} : \boldsymbol{a} \in [\mathcal{A}^2]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{a}}$
	Test	$\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}} : \boldsymbol{a} \in [\mathcal{A}^2]\})$	$S, Q \sim \mathcal{T}_{\boldsymbol{a}}$

D Dataset Preprocessing and Hyperparameter Details

Here we first provide some details on the logic used to construct the ID benchmark Zappos-ID and its OOD counterpart Zappos-OOD. We then list the set of hyperparameter configurations used to train the meta-learning methods PN, RR, SVM, FOMAML and the supervised learning baselines MB, SB and IC on each of the benchmarks in the paper.

D.1 Zappos Preprocessing

Recall that the Zappos dataset is motivated through an online shopping recommendation problem, where each task is a binary classification of shoe images into an attribute context. Every online user is represented by such a task, where the user's preference for shoes is specified by the corresponding shoe attribute context. We consider a simplified setting where we fix a set of universal shoe attributes $\mathcal{A}$ and each user's preference is specified exactly by a pair of attributes $\boldsymbol{a} = (a_1, a_2) \in \mathcal{A}^2,\ a_1 \neq a_2$. The Zappos-ID and Zappos-OOD FSL benchmarks we use are derived from the UT Zappos50k corpus which consists of 50,025 shoe images each annotated with a list of attributes the shoe possess.

Attributes Selection. We limit the subset of attributes we consider to the 78 considered by Ren et al. [35] (Table 7). Recall that the task distribution we consider is the uniform distribution $\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}(\mathcal{T}) = \text{Unif}(\{\mathcal{T}_{\boldsymbol{a}} : \boldsymbol{a} \in [\mathcal{A}^2]\})$ (Table 2) over all non-repeating attribute pairs in $\mathcal{A}$. In order to ensure that each attribute pair in $\mathcal{A}$ has at least 20 shoes carrying both the attributes (feasible pairs), we only consider the uniform distribution over such feasible attribute pairs. Thus, we reduce the original set of 78 attributes to 66, since 12 of the attributes where found to be infeasible with every other attribute in the original set.

Determining $\mathcal{A}, \mathcal{A}'$. For the Zappos-ID benchmark we use the set of attributes specified by $\mathcal{A}$ (of size 36) to *iid* sample 1000 (or 50) train and 25000 test tasks. On the other hand, as mentioned in Section 5 we use a disjoint set of attributes $\mathcal{A}'$ (of size 30) to sample 25000 test tasks for Zappos-OOD (see Table 3 for the exact sets). To determine this partition, we first consider a graph of 66 nodes, where each node represents an attribute and an undirected edge between a pair of attribute node is weighted by the number of shoe images (in the corpus) that have both attributes. Using spectral clustering, we find an approximate min-cut bipartition of this graph. In other words, we partition the entire set of attributes into two subsets in a way that reduces the number of images which carry pairs of attributes that are not in the same subset. This graph partition gives us the split of a 36-attribute set $(\mathcal{A})$ and a 30-attribute set $(\mathcal{A}')$.

Table 3: We show the disjoint set of attributes $\mathcal{A}, \mathcal{A}'$ for the Zappos-ID/OOD datasets. For the Zappos-ID dataset we use the set of attributes in $\mathcal{A}$ to *iid* sample train and test tasks $\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}$. For the Zappos-OOD dataset the train tasks are still sampled using $\mathcal{A}$ *i.e.,* $\mathcal{T}_{\boldsymbol{a}} \sim \mathbb{P}_{\mathcal{A}}$ but the test tasks are sampled using $\mathcal{A}'$ *i.e.,* $\mathcal{T}_{\boldsymbol{a}'} \sim \mathbb{P}_{\mathcal{A}'}$.

$\mathcal{A}$	Category.Boots	Category.Sandals	Closure.Ankle.Strap	Closure.Ankle.Wrap
	Closure.Buckle	Closure.Bungee	Closure.Button.Loop	Closure.Elastic.Gore
	Closure.Pull.on	Closure.Sling.Back	Closure.Snap	Closure.T.Strap
	Closure.Toggle	Closure.Zipper	Gender.Girls	Gender.Women
	HeelHeight.High.heel	HeelHeight.Short.heel	Material.Rubber	Material.Suede
	SubCategory.Ankle	SubCategory.Clogs.and.Mules	SubCategory.Flats	SubCategory.Heel
	SubCategory.Heels	SubCategory.Knee.High	SubCategory.Mid.Calf	SubCategory.Over.the.Knee
	ToeStyle.Almond	ToeStyle.Center Seam	ToeStyle.Closed Toe	ToeStyle.Open Toe
	ToeStyle.Peep Toe	ToeStyle.Pointed Toe	ToeStyle.Round Toe	ToeStyle.Snip Toe
$\mathcal{A}'$	Category.Shoes	Category.Slippers	Closure.Hook.and.Loop	Closure.Lace.up
	Closure.Monk.Strap	Closure.Slip.On	Gender.Boys	Gender.Men
	Material.Corduroy	Material.Silk	Material.Wool	SubCategory.Boat.Shoes
	SubCategory.Crib.Shoes	SubCategory.Firstwalker	SubCategory.Loafers	SubCategory.Oxfords
	SubCategory.Prewalker	SubCategory.Slipper.Flats	SubCategory.Sneakers.and.Athletic.Shoes	ToeStyle.Algonquin
	ToeStyle.Apron Toe	ToeStyle.Bicycle Toe	ToeStyle.Bump Toe	ToeStyle.Capped Toe
	ToeStyle.Medallion	ToeStyle.Moc Toe	ToeStyle.Snub Toe	ToeStyle.Square Toe
	ToeStyle.Wide Toe Box	ToeStyle.Wingtip		

D.2 Hyperparameter settings

For all the experiments performed in our paper, we have run grid search to tune both the meta-learning method-specific hyperparameters and the optimization hyperparameters for each meta-learning method. For the existing OOD FSL benchmarks, we found that the best hyperparameters are often the same as what was reported in the original paper. Additionally, the absolute performance and performance orders we report on OOD FSL benchmarks match with other works after the hyperparameter tuning – indicating that we have been fair in representing each meta-learning method with its best hyperparameter setting. For our newly identified in-distribution benchmarks, we took care in tuning these hyperparameters for each method to ensure fairness of comparison.

In Table 4 we list the optimization and other algorithm-specific hyperparameters for each meta-learning method, dataset pair. For **optimization hyperparameters**, we describe

(a) the total number of epochs (each with 1000 iterations of gradient updates except for FEMNIST, Zappos-ID, and Zappos-OOD where each epoch depends on number of training tasks);
(b) step (staircase) learning rate schedule (lr: $e_1(r_1) - e_2(r_2) - \ldots - e_n(r_n)$ where r_i is the value of the learning rate and e_i is the epoch number at which r_i is first set);
(c) the number of tasks in a minibatch to compute one gradient update (task batch size or task BS);
(d) for all experiments we use SGD optimizer (Nesterov Momentum 0.9).

Other meta-learning method specific hyperparameters: The scale-factor refers to a constant factor that is multiplied to the logits for each class, before passing them through softmax. In some cases these are fixed through training, and in others they are learnable. For other method-specific hyperparameters that we borrow directly from the original paper, we provide the references.

i) During meta-training, we perform the same **data augmentation** used in Chen et al. [9] for *mini*-M and used in Lee et al. [25] for *cifar*-M, FC-M, *tiered*-M. For FEMNIST, Zappos-ID and Zappos-OOD we do not perform any data augmentations.
ii) We use a **weight decay** of $5e-4$ for all datasets except for FEMNIST for which a weight decay of $1e-2$ is used to prevent overfitting.
iii) We use the Resnet-12 **backbone** for all our experiments except for FEMNIST which is made up of the relatively easier tasks of digit classification. For FEMNIST, we use a four layer Conv-64 model backbone.

Computational Resources. For all experiments we use (at most) four NVIDIA GEFORCE GTX 1080Ti GPU cards. A single run of PN, RR, SVM, and FOMAML on the *mini*-M dataset takes ≈ 12 hrs, 48 hrs, 48 hrs, and 72 hrs respectively. For experiments on Zappos and FEMNIST, except for FOMAML which takes about 24 hrs, all the other experiments take no more than 5 hrs to complete training.

Table 4: Hyperparameter details for different algorithms and datasets in Sections 4, 5.

Alg / Dataset	Optimization hyperparameters	Other hyperparameters
PN/ (*mini*-M, *cifar*-M, *tiered*-M)	60 Epochs lr: 0(0.1)-20(6e-3)-40(1.2e-3)-50(2.4e-4) task BS: 4(5-way), 1(>5-way)	scale-factor 10 euclidean metric [39]
PN/ FC-M	15 Epochs lr: 0(0.1)-5(6e-3) task BS: 4(5-way), 1(>5-way)	scale-factor 10 euclidean metric [39]
(RR, SVM)/ (*mini*-M, *cifar*-M, *tiered*-M)	60 Epochs lr: 0(0.1)-20(6e-3)-40(1.2e-3)-50(2.4e-4) task BS: 8 (always 5 way)	learnable scale other as in Lee et al. [25]
(RR, SVM)/ FC-M	30 Epochs lr: 0(0.1)-20(6e-3) task BS: 8 (always 5 way)	scale-factor 7 other as in Lee et al. [25]
FOMAML/ (*cifar*-M, FC-M)	60 Epochs lr: 0(0.01)-20(6e-3)-40(1.2e-3) task BS: 4 (always 5 way)	scale-factor 1 inner loop step size: 0.01 inner loop steps: 5 (train), 20 (test) other as in Finn and Levine [14]
FOMAML/ (*mini*-M, *tiered*-M)	70 Epochs lr: 0(1e-2)-35(1e-3)-65(1e-4) task BS: 4 (always 5 way)	scale-factor 1 inner loop step size: 0.01 inner loop steps: 5 (train), 20 (test) other as in Finn and Levine [14]
(PN, RR, SVM, FOMAML)/ (Zappos-ID, Zappos-OOD)	60 Epochs lr: 0(0.1)-30(6e-3) task BS: 4	PN: scale-factor 10, euclidean metric [39] RR: learnable scale SVM: learnable scale FOMAML: inner loop steps: 5 (train), 20 (test), other as in Finn and Levine [14]
(PN, RR, SVM, FOMAML)/ FEMNIST	100 Epochs lr: 0(1e-3)-50(1e-4) task BS: 5	PN: scale-factor 10, euclidean metric [39] RR: learnable scale SVM: learnable scale FOMAML: inner loop steps: 5 (train), 20 (test), other as in Finn and Levine [14]
(SB, MB)/ (*mini*-M, *cifar*-M, *tiered*-M, FC-M)	100 Epochs lr: 0(0.1)-90(1e-2) batch size: 128	SB, MB: We project features in to unit norm during meta-train and use euclidean metric with scale-factor 10 in meta-test [39] MB only: we further finetune (lr=1e-3) on the meta-learning objective for additional 30 epochs.
IC/ (*cifar*-M, *mini*-M)	100 Epochs lr: 0(0.1)-90(1e-2) batch size: 128	Figure 2(b) (*cifar*-M): For supervised learning image classification (IC) baseline we collect all the images belonging to the base classes in *cifar*-M and randomly split them into 80%(train)-10%(val)-10%(test). The val and test rankings used for computing the Kendall coefficient (ρ) are obtained using the val and test splits. Figure 3(b) (*mini*-M): The supervised learning IC baseline is trained on 600 train images from each base class in *mini*-M. After training is completed, we identify two IC models that differ by $\epsilon = 0.5\%$ in terms of their generalization performance over a test set made up of all the unused examples (~ 700) from each base class. To test the frequency of conclusion flips, for each of the 100 comparison runs, a random test subset is sampled from this test set with 100 examples sampled from each class. The chosen IC model pair is then evaluated over this test subset and their performance difference is recorded for this comparison run.

E Additional Results on Evaluating ID Performance

In this section, we first present additional results on the choice of a different meta-learning method or the usage of a different (S, Q) sampling strategy (FIX-ML) for the same meta-learning method can lead to improvements in OOD performance at the cost of ID performance. Then, we show how reducing the number of training tasks is unlikely to change the performance order of meta-learning methods in the ID benchmark Zappos-ID — even though by reducing the number of training tasks, the number of unseen attribute pairs at test time increases. Finally, we present additional results how the degree of train/test task distribution mismatch can impact the performance order of meta-learning methods.

E.1 Additional results on ID/OOD Performance Tradeoffs

In the main text (Section 4, Figure 1), we have compared the BaseGen and NovelGen performance of the best validation snapshots from different meta-learning methods (PN, SVM, RR, FOMAML) and supervised trained baselines (SB, MB) on the *mini*-M and *tiered*-M datasets. Here, we show how the performance order of the same set of methods *can also flip* in the ID (BaseGen) and OOD (NovelGen) settings **on two additional datasets**: *cifar*-M and FC-M (Figure 4(a),(b)). Additionally, in the main text, we have shown how switching the choice of the (S, Q) sampling strategy to one with fixed support sets (FIX-ML Setlur et al. [38]) can also lead to improvements in the OOD performance at the cost of ID performance on the *cifar*-M dataset. In Figure 4(c),(d) we provide further evidence of this performance tradeoff **on two more datasets** *mini*-M and FC-M.

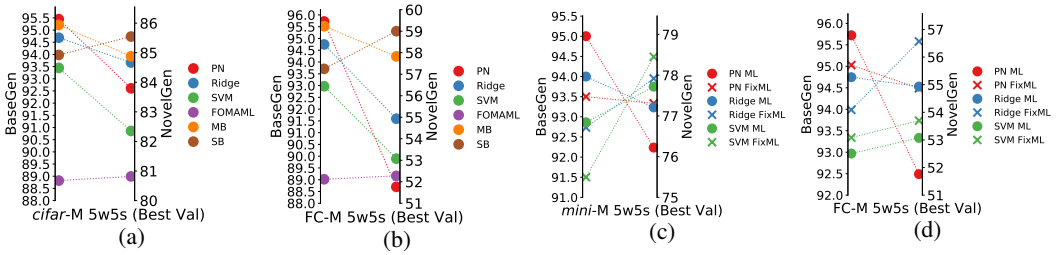


Figure 4: BaseGen and NovelGen performance tradeoff (for best validation snapshots): over the choice of a set of four meta-learning and two supervised pre-training methods on *cifar*-M (a) and FC-M (b); over the use of FIX-ML (S, Q) generation strategy or not (ML) with SVM, RR and PN on *mini*-M in (c) and FC-M in (d).

E.2 In-distribution Performance with Reduced Number of Training Tasks

Now we analyze the impact of the number of training tasks on the performance order of meta-learning methods in ID scenario. For the 2w5s results on the Zappos-ID dataset in Table 1 we used 1000 (or 50) training tasks and 25000 test tasks. Here, we also consider using 250 training tasks (while still using the same number of test tasks). As we have discussed in Section 4, it is possible to encounter unseen classes (attribute pairs in the case of Zappos) at test time even while evaluating a meta-learning method in the ID scenario. When we reduce the number of training tasks, we observe more unseen attribute pairs at test time. For 250 train tasks we observe 153 and for 50 train tasks we observe 269 unseen attribute pairs at test time.

Table 5: We analyze the 2w5s performance order of PN, SVM, RR and FOMAML on the Zappos-ID dataset with reduced number (50, 250) of train tasks and compare the performance order (ranking in parentheses) observed with the larger set of 1000 train tasks.

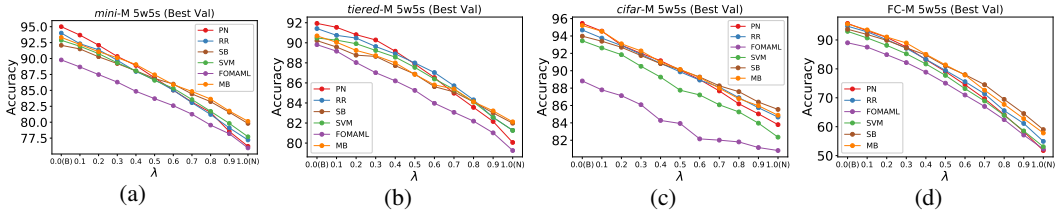
Methods / # Train tasks	50	250	1000
PN	$^{(1)}77.67 \pm 0.17\%$	$^{(1)}81.67 \pm 0.16\%$	$^{(1)}86.58 \pm 0.15\%$
Ridge	$^{(2)}74.75 \pm 0.16\%$	$^{(2)}80.84 \pm 0.15\%$	$^{(2)}85.56 \pm 0.16\%$
SVM	$^{(3)}74.06 \pm 0.17\%$	$^{(3)}80.15 \pm 0.17\%$	$^{(3)}85.12 \pm 0.16\%$
FO-MAML	$^{(4)}69.85 \pm 0.18\%$	$^{(4)}73.20 \pm 0.16\%$	$^{(4)}80.14 \pm 0.15\%$

In Table 5 we can see that even with reduced training tasks and more unseen attribute pairs at test time the performance order of PN, RR, SVM and FOMAML is retained. Note that the same order is observed on the other ID benchmark FEMNIST. This also matches the BaseGen performance order obtained after doing ID evaluations on modified FSL benchmarks. This result further confirms that the performance evaluations done on ID datasets are much more consistent than OOD datasets.

E.3 Degree of Mismatch between Train ($\mathbb{P}_{\mathcal{C}_B}$) and Test ($\mathbb{P}_{\mathcal{C}_N}$) Distributions

In Section 4 we notice that the performance order of meta-learning methods in the ID scenario (Zappos-ID) is quite different from that of the OOD FSL benchmarks (*e.g., mini*). We believe this is mainly because of the OOD nature of FSL benchmarks, which we also proof formally in Appendix B. Moreover, since the type/degree of the mismatch between training and test distributions can vary for different FSL benchmarks, the performance order of popular methods is not as consistent as we would like them to be on these OOD benchmarks (see Section 5.2).

Range of test task distributions. To further analyze the impact of the degree of distribution mismatch on the performance order, for each FSL benchmark we construct **a range of new task distributions that are increasingly dissimilar to the train task distribution** $\mathbb{P}_{\mathcal{C}_B}$ and **similar to the test task distribution** $\mathbb{P}_{\mathcal{C}_N}$. We denote this set of task distributions as $\{\mathbb{P}_{\text{BN}}(\lambda), \lambda \in [0, 1]\}$. When sampling an n-way task from $\mathbb{P}_{\text{BN}}(\lambda)$, instead of performing the first step of sampling outlined in Table 2, we sample the class tuple $\boldsymbol{c} = (c_1, \ldots, c_n) \in [(\mathcal{C}_B \cup \mathcal{C}_N)^n]$ where c_i's are sequentially sampled non-repeatedly from $\mathcal{C}_N$ uniformly with probability λ and from $\mathcal{C}_B$ with probability $1 - \lambda$. The second step of sampling (S, Q) from the task distribution $\mathcal{T}_{\boldsymbol{c}}$ is done in the usual way. It is clear that $\mathbb{P}_{\text{BN}}(0) = \mathbb{P}_{\mathcal{C}_B}$ and $\mathbb{P}_{\text{BN}}(1) = \mathbb{P}_{\mathcal{C}_N}$. In addition, when $\lambda = |\mathcal{C}_N|/(|\mathcal{C}_N| + |\mathcal{C}_B|)$, $\mathbb{P}_{\text{BN}}(\lambda) = \mathbb{P}_{\mathcal{C}_B \cup \mathcal{C}_N}$ which describes a continual learning setting where the evaluation task is made up of classes uniformly sampled from the union of base and novel classes. Notice that the task distribution $\mathbb{P}_{\text{BN}}(\lambda)$ is not the same as the mixture distribution $\lambda \cdot \mathbb{P}_{\mathcal{C}_N} + (1 - \lambda) \cdot \mathbb{P}_{\mathcal{C}_B}$ because a single task from $\mathbb{P}_{\text{BN}}(\lambda)$ can have both classes from $\mathcal{C}_N$ and $\mathcal{C}_B$. With this set of new task distributions defined, we evaluate our learned algorithm snapshot for each meta-learning method not only over the training ($\mathbb{P}_{\mathcal{C}_B}$) and the test ($\mathbb{P}_{\mathcal{C}_N}$) task distribution but also over distributions from this interpolation set. We plot the performances in Figure 5.



(a) (b) (c) (d)

Figure 5: Comparison of PN, SVM, RR, FOMAML, SB and MB's performance (best validation snapshots) on the set of distributions $\{\mathbb{P}_{\text{BN}}(\lambda)\}$ for (a) *mini*-M, (b) *tiered*-M, (c) *cifar*-M, and (d) FC-M datasets.

Performance order depends on degree of mismatch. In Figure 5 we first notice that the performance drops in a monotonically non-increasing way as λ increases for each dataset/method combination. More importantly, we note that the performance order of the methods depends on the degree of mismatch. If the test task distribution is very similar to the train task distribution, then the performance order is mostly retained as the ID performance order (*e.g.,* for $\lambda \leq 0.2$ in Figure 5(a)). Also in most cases the lines don't cross each other more than once, which indicates that if the degree of mismatch crosses a certain threshold then it is unlikely for the performance order to switch again for a given pair of methods. However, without seeing any test tasks during meta-training, it is difficult to know the degree of training and test task distribution mismatch, which makes it more difficult to predict when a given method will start performing better/worse. For example, the performance order changes at a lower value of λ for *mini* and *cifar*, as compared to *tiered*.

F Additional Discussion and Results on OOD Evaluation

F.1 Simplified Example of Snapshot Selection and Hyperparameter Selection

To further explain our existing definition of hyperparameter and snapshot selection in Section 5.1, we accompany our original definitions with a concrete simplified scenario: suppose we want to train

Prototypical Network (PN) with two hyperparameter configs: training with learning rate 1e-3 vs. with learning rate 1e-4 (with all other hyperparameters the same).

[Snapshot Selection] Training under each hyperparameter config would generate a sequence of algorithm snapshots (most often with one snapshot saved after each training epoch). The term snapshot selection refers to the procedure of choosing one snapshot from each hyperparameter config: one from the 1e-3 learning-rate optimized PN algorithm snapshot trajectory, and one from the 1e-4 learning-rate optimized PN algorithm snapshot trajectory. There can be multiple strategies for snapshot selection, for example, picking **1)** the last snapshot at the end of training; **2)** the snapshot that has the lowest training loss; **3)** the snapshot that has the best BaseGen performance; or **4)** the snapshot that has the best ValGen performance.

[Hyperparameter Selection] Once an algorithm snapshot is chosen for each considered hyperparameter config, we need to decide which hyperparameter config's (lr 1e-3 or lr 1e-4) selected snapshot to choose to be evaluated on the test task distribution with its performance recorded as the meta-learning method PN's performance. The procedure of deciding which hyperparameter config's snapshot to choose is called the hyperparameter selection problem. Similar to snapshot selection, there can be multiple strategies for hyperparameter selection: e.g., choosing the hyperparameter config whose selected snapshot has the best BaseGen or the best ValGen performance.

F.2 Distinction between Snapshot Selection and Early Stopping

Early stopping is not the same as what we mean by snapshot selection in our paper. In standard supervised learning, *early stopping* involves keeping track of the performance over an *iid* validation dataset and stopping training after the validation performance starts to consistently deteriorate. It mainly serves as an approach to avoid overfitting and to save unnecessary computations if one believes further training would never improve the test performance. In terms of deciding which snapshot to choose, early stopping is often equivalent to the strategy of selecting the snapshot that has the highest validation accuracy. In contrast, in our paper, the term *snapshot selection* refers to the **general problem** of deciding which snapshot to select for a given training run, **as opposed to a specific choice of selection strategy**. As we have explained in Section 5.1.1, the strategy of picking the snapshot with the best meta-validation performance (early stopping) might not be the best strategy in the OOD scenario. There exist other snapshot selection strategies (different from early stopping), such as picking the snapshot with the lowest training loss or the best BaseGen performance.

F.3 Differences in NovelGen Performance When Using Best BaseGen vs. ValGen for Snapshot Selection

In Section 5.1 we compare different model selection strategies in the out-of-distribution scenario where we show that in some cases BaseGen performance can track NovelGen (Figure 2(c)) while in other cases the ValGen performance may be better correlated with NovelGen (Figure 2(d)). In Table 6 we present NovelGen (test) performance results on two different datasets when the snapshot is chosen using the best BaseGen snapshot (row 1) vs the best ValGen snapshot (row 2), and compare them against the best possible NovelGen across all snapshots (row 3). Here we notice that there can be a **significant difference in the selected snapshot's NovelGen performance if we use one selection strategy instead of another** and there isn't a single snapshot selection strategy that yields the best results for multiple OOD scenarios.

Table 6: NovelGen performance for i) PN trained on *tiered*-M; and ii) SVM trained on *cifar*-M evaluated using the snapshot chosen with best BaseGen/ValGen/NovelGen performance.

Snapshot selection strategy	PN on *tiered*-M (Figure 2(c))	SVM on *cifar*-M (Figure 2(d))
Best BaseGen	80.45%	80.49%
Best ValGen	80.05%	82.65%
Best NovelGen	80.45%	82.79%

F.4 What to Consider While Designing Model Selection and Comparison Strategies for Both ID and OOD Performance?

In Section 5.1 we discuss in detail the implications of different model selection strategies on the final NovelGen performance of the selected snapshot specific to current OOD FSL benchmarks. In more generic settings, it is likely that one may wish to design model selection methods when one cares about both ID and OOD performance. In such a case we believe that the optimal model selection and comparison strategy would depend on: i) the final comparison metric; ii) the type of task access for model selection.

i) **Method comparison metric**: We first need to define a metric for the final evaluation of a meta-learning method's selected algorithm snapshot. One way is to individually evaluate ID and OOD performance and record it as a 2-tuple. In this case, a meta-learning method is said to outperform another only if its selected snapshot is better in each component of the 2-tuple. In this case, it is very possible that there does not exist a meta-learning method that clearly outperforms all others in this metric as we have seen from Figure 1(a)(b). Another way is to evaluate the performance on a mixture task distribution, with the training and OOD test distributions weighted by fixed probability weights (as we do in Appendix E.3). As we have seen in Appendix E.3 (Figure 5), different probability weighting of the two distributions can result in different conclusions of which meta-learning method works the best.

ii) **Task access during model selection**: We also need to specify what type of task samples are available during model selection. While it is reasonable to assume there are additional fresh *iid* samples from the training distribution (e.g. by holdout training set), it depends on the specific application to know whether there exist task samples from the OOD task distribution. Different scenarios may arise based on this: i) In federated learning applications, one might be allowed to evaluate meta-learned algorithm snapshots over a small sample of users (tasks) from the OOD user population before deployment; or ii) In case one does not have such OOD task samples, a proxy distribution (e.g. validation task distribution in FSL benchmarks) may still be available and *iid* task samples drawn from it could be used for model selection. However, its utility would depend on how similar it is to the actual OOD distribution and we have seen examples in Figure 2(a) that using samples from such proxy task distributions might also be suboptimal.

We hope that our work advocates for more discussion and development of model comparison metrics and selection strategies while taking into these considerations described above.

F.5 Hyperparameter Selection Strategies

In Section 5.1 we discuss the distinction between snapshot selection and hyperparameter selection and how the former is called as a subroutine while determining the best snapshot to represent a specific hyperparameter configuration. As snapshot selection is a ubiquitous problem we focus on analyzing it and exploring alternative strategies (in the context of meta-learning) in the main paper. Motivated by the work of Gulrajani and Lopez-Paz [18], we now discuss some ways of performing hyperparameter selection specifically for settings similar to the current FSL benchmarks where the few-shot classification tasks are determined by a class tuple.

Because the current FSL benchmarks have set aside a specific validation set of classes, it would appear that using tasks generated by these classes is the only option for hyperparameter selection. However, as mentioned in [18], there are also other alternatives.

Cross validated hyperparameter selection. Instead of using a single set of validation classes, one could rely on cross validation. Here, for a given hyperparameter configuration, one can train multiple times, where each training run is done on tasks generated by a different subset of the training (base) classes (or the union of base and val classes) and the proxy performance is calculated on tasks generated by the remaining classes not used in training. Finally, the performance is averaged over all runs for the given hyperparameter configuration and the hyperparameter with the highest performance would be chosen.

However, this approach would still require performing snapshot selection for each run, thus it remains unclear how to best perform snapshot selection in this case. Despite this, cross-validated hyperparameter evaluation could potentially be more reliable than using a single set of validation classes for hyperparameter selection (as done on the current FSL benchmarks) but it would also be

more computationally expensive and we leave further investigation of this hyperparameter selection approach as future work.

Allowing restricted oracle evaluation over the test task distribution. Instead of completely forbidding access to the test task distribution, one can allow a limited number of test task distribution performance evaluations for a given meta-learning method. In this case it becomes the responsibility of designer of the meta-learning method to decide how to best distribute this fixed number of evaluations wisely over different hyperparameter and snapshot choices.

F.6 Variance of performance $A(\mathcal{C}_N)$ over randomly sampled $\mathcal{C}_N$

In Section 5.2 Inconsistency example 1, we have shown how limited number of novel classes in the evaluation can lead to a high chance of conclusion flips. This high degree of unreliability stems from the variance of the performance estimator $A(\mathcal{C}_N)$ which only uses a subset of the larger underlying class set $\mathcal{C}_L$. In Figure 6 we plot the histogram of the random variable $A(\mathcal{C}_N)$ randomized over the choice of novel classes $\mathcal{C}_N$. We notice that the performance standard deviation is 2.49% on *mini* and 3.1% on *tiered*. The standard deviation on *tiered* is higher since the number of underlying classes $|\mathcal{C}_L|$ is larger in *tiered* ($= 552$). When the variance of the novel accuracy is as high as what we have observed here, it becomes very hard to clearly determine the better meta-learning method. To alleviate this, we provide some actionable suggestions like choosing benchmarks with more base classes during training and more novel classes during evaluation (see Section 5.2).

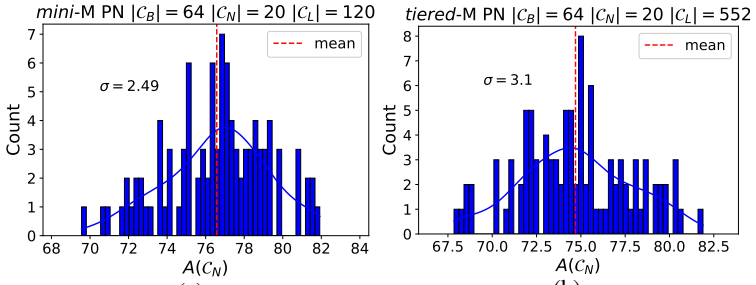


Figure 6: Histogram plots (over 100 runs) of the OOD accuracy $A(C_N)$ for (a) a PN trained on *mini* and evaluated on random 20 out of 120 novel classes; and (b) a PN trained on 64 *tiered* base-classes (see Section 5.2) and evaluated on random 20 out of 552 novel classes.