XB-MAML: Learning Expandable Basis Parameters for Effective Meta-Learning with Wide Task Coverage

Jae-Jun Lee **Sung Whan Yoon**†

{johnjaejunlee95, shyoon8}@unist.ac.kr
Graduate School of Artificial Intelligence, Ulsan National Institute of Science and Technology (UNIST)
† Corresponding Author

Abstract

Meta-learning, which pursues an effective initialization model, has emerged as a promising approach to handling unseen tasks. However, a limitation remains to be evident when a meta-learner tries to encompass a wide range of task distribution, e.g., learning across distinctive datasets or domains. Recently, a group of works has attempted to employ multiple model initializations to cover widely-ranging tasks, but they are limited in adaptively expanding initializations. We introduce XB-MAML, which learns expandable basis parameters, where they are linearly combined to form an effective initialization to a given task. XB-MAML observes the discrepancy between the vector space spanned by the basis and fine-tuned parameters to decide whether to expand the basis. Our method surpasses the existing works in the multi-domain meta-learning benchmarks and opens up new chances of meta-learning for obtaining the diverse inductive bias that can be combined to stretch toward the effective initialization for diverse unseen tasks.

1 INTRODUCTION

Humans have the capability to learn unknown or unseen tasks without explicit prior learning. When encountering unseen data or learning tasks, humans tap into their meta-knowledge to reason and adapt to the unfamiliar context, drawing upon connections with their past experiences. This innate adaptability in humans is closely related to the concept of 'learning to learn'. In contrast, deep learning algorithms (LeCun et al. (2015)) aim to replicate the humans' learning capability but typically rely on sample-wise likelihood maximization which leans toward memorizing encountered data rather than pursuing meta-knowledge. As a consequence, modern learning frameworks are prone to fall into critical difficulties, i.e., limited generalization across varying environments.

To tackle the long-lasting challenge, the research field of meta-learning has emerged, drawing inspiration from humans' rapid adaptability to unseen environments by leveraging meta-knowledge across previous experiences. Specifically, meta-learning aims to effectively solve new tasks by utilizing the commonality acquired from previous learning tasks. In the domain of classification, meta-learning can be broadly categorized into two main groups: Metric-based methods, encompassing methods such as Matching Network, ProtoNet, TapNet, and TADAM (Vinyals et al. (2016); Snell et al. (2017); Yoon et al. (2019); Oreshkin et al. (2018)), and optimization-based methods such as Model-Agnostic Meta-Learning (MAML), Reptile, and Meta-SGD (Finn et al. (2017); Nichol et al. (2018); Li et al. (2017)). Albeit the success of prior works in improving the adaptability to novel tasks, they often fail to handle a wide range of tasks across varying domains, or contexts (Triantafillou et al. (2020)).

By recognizing the hardship of capturing the widely-ranging task distribution through a single meta-trained model parameter, the concept of employing multiple meta-trained initializations has gained attention in recent times. As noticeable trials, TSA-MAML by Zhou et al. (2021) tries to build clusters of similar task parameters to employ per-cluster initialization, and MUSML by Jiang et al. (2022) meta-trains multiple subspaces that cover a wider range of task parameters. The prior works with multi-initializations show obvious limitations in two perspectives: i) The

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

number of initializations is predefined before training, and it is not expandable even if more initializations are required. ii) Utilization of multiple initializations is restricted to selecting one of them so that the combinatorial way of multi-initialization to enlarge the coverage of task distributions is infeasible.

In this paper, we introduce a multi-initialization approach called XB-MAML with two distinctive advantages: i) Expandability of initializations and ii) Combinatorial usage of multiple initializations to provide a better initialization for a given task. Specifically, XB-MAML incrementally incorporates additional initialization to adaptively cover a wider range of tasks, i.e., the set of initializations is expandable according to the given task distribution. Also, each initialization works as a 'basis' in parameter space, where the meta-trained initializations are linearly combined to form an effective initialization point for the given task. When the current set of bases falls short of covering task distribution, XB-MAML adaptively employs an additional model parameter, which is likened to increasing the rank of the basis to enable more effective task-specific adaptation across a wide range of complex tasks. XB-MAML gradually progresses towards the rank of basis that excels in task adaptation and attains performance convergence. Also, XB-MAML covers the parameter space spanned by the linear combination of the meta-trained bases so that it provides a widened coverage of parameters that cannot be obtained by an individual initialization. XB-MAML offers a novel strategy to obtain the diverse inductive bias in meta-learning that can be combined to stretch toward the effective initialization for diverse unseen tasks.

In extensive experiments, our XB-MAML shows significant improvements over previous works on the challenging multi-domain few-shot classification in benchmarks datasets: *Meta-Datasets*-ABF/BTAF/CIO.

2 RELATED WORK

2.1 Recent Advances of Meta-Learning

Metric-based Method After the early works of metric-based meta-learning, e.g., Matching Nets by Vinyals et al. (2016) and ProtoNets by Snell et al. (2017), the explicit task-adaptation strategies are adopted in metric learning: TADAM with conditioned representation (Oreshkin et al. (2018)), TPN with graph-based metric computations (Liu et al. (2019b)), TapNet with task-adaptive projection (Yoon et al. (2019)). Recently, cross-attention has been shown to be effective in few-shot learning (Hou et al. (2019)), and the transformer architecture with in-depth attention draws substantial gains (Lai et al. (2023)).

Optimization-based Method The optimization-based approach aims to train an initialization parameter that is prepared for quick adaptation to the task parameters via bi-level optimization. Beyond the pioneering work called MAML (Finn et al. (2017)), the computation-efficient variants such as Reptile with the first-order approximation of Hessian (Nichol et al. (2018)) and iMAML with implicit gradients (Rajeswaran et al. (2019)) have been proposed. As another branch, the probabilistic modeling of the initialization has been explored, e.g., Bayesian MAML by Yoon et al. (2018) and PLATIPUS by Finn et al. (2018). Our XB-MAML is also built on the bi-level optimization by MAML and it incorporates extra initialization by sampling from the Gaussian-based modeling of initializations which is motivated by PLATIPUS.

2.2 Multi-domain Meta-learning

The prior meta-learning works are often limited in training the multi-domain few-shot tasks (Triantafillou et al. (2020)). To tackle the issue, MMAML (Vuorio et al. (2019)) employs modulation networks for task grouping, adapting with modulated meta-parameters via gradient updates. HSML (Yao et al. (2019)) clusters feature-represented tasks hierarchically, adjusting their parameters correspondingly. Another approach, ARML (Yao et al. (2020)), utilizes prototype-based relational structures and a meta-knowledge graph to disseminate information within its components. These methods primarily focus on task classification or clustering, adapting or selecting tasks within justified groupings to effectively cover a wide array of task distributions. XB-MAML does not rely on task clustering but meta-trains a set of bases that linearly combine to cover the given task distributions.

2.3 Multi-initialization Approches

A group of recent methods has investigated the utilization of multiple initializations to widen the task coverage. Here, we analyze two distinctive approaches.

Clustering of Tasks A method called TSA-MAML (Zhou et al. (2021)) utilizes a pretrained MAML to perform k-means clustering to group similar tasks and assign one initialization for each cluster, which is the per-cluster centroid. Additional episodic learning is then done to further meta-train the centroid initializations. This process maps each episode to the closest initialization, which can reduce the gap between the initialization and its task-specific model. However, it raises a crucial concern that TSA-MAML strongly relies on the pretrained MAML which is not tailored for the multi-initialization setting. Also, an extensive extra computational burden is required to run the pre-

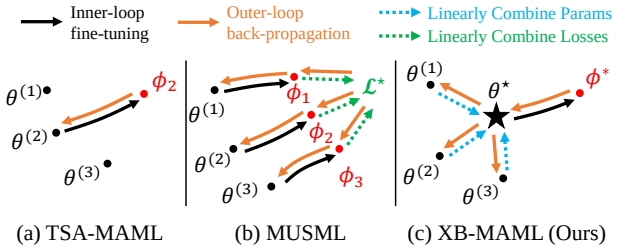


Figure 1: Illustration of bi-level optimization with three initializations $\{\theta^{(m)}\}_{m=1}^{3}$. (a) TSA-MAML selects one initialization with the smallest loss. (b) MUSML separately fine-tunes and meta-updates each initialization. (c) XB-MAML forms the initialization $\theta^{\star}$ via linear combination and jointly meta-updates them.

Table 1: Comparisons of multi-initialization meta-learners

	Methods		
	TSA-MAML	**MUSML**	**XB-MAML**
Expandability	✗	✗	✓
Use of initial. in training	Select & update the best one	Parallelly use & update	Linear combination
Use of initial. in testing	Use the best one	Use the best one	Linear combination
Extra modules	Pretrained MAML	Subspaces (FC Layers)	✗

training of MAML. Lastly, for each task, the best initialization with the smallest loss is selected and meta-updated so that TSA-MAML is limited to naively partitioning the task distribution into clusters.

Learning Subspaces Another noteworthy approach called MUSML (Jiang et al. (2022)) introduces subspace learning and leverages multi-initializations. This method quickly learns the significance of subspaces, which consist of simple additional fully connected layers placed right after the feature extractor, effectively serving as a classifier. MUSML parallelly fine-tunes initializations to the given task and meta-updates them with weighed loss, i.e., a large weight is assigned for the initialization with a smaller meta-loss. The process allows multi-initializations to be located properly for covering the task distribution. As TSA-MAML does, however, MUSML does not address a combinatorial usage of multi-initializations.

It is crucial to acknowledge that the aforementioned methods rely on a predefined number of initializations. It prohibits the prior methods to expand the size of initializations even if additional initializations are required. In contrast, our XB-MAML adaptively increases initializations if they are needed. We emphasize that XB-MAML provides a collaborative way of multiple initializations by linearly combining them to build an effective initialization for the given task. This enables XB-MAML to meta-train the basis of parameters that are required to further stretch out the initialization to the near side of task-specific parameters, which has never been anticipated by the related works. In Figure 1 and Table 1, the key differences of XB-MAML and the related works are presented.

3 METHOD: XB-MAML

3.1 Preliminaries

Problem Formulation We follow the episode construction for N-way K-shot few-shot classification setting with the support/query protocol (Vinyals et al. (2016)). Specifically, we sample a batch of tasks $\{\mathcal{T}_i\}_{i=1}^{\mathcal{B}} = \{(\mathcal{S}_i, \mathcal{Q}_i) : i = 1, \cdots, \mathcal{B}\}$, where each task $\mathcal{T}_i$ is sampled from the task distribution $p(\mathcal{T})$. Here, $\mathcal{B}$ is the number of tasks in a batch, and $\mathcal{S}_i$ and $\mathcal{Q}_i$ represent the support and query sets of $\mathcal{T}_i$. To elaborate further, $\mathcal{S}_i$ and $\mathcal{Q}_i$ consist sets of input-label pairs: $\mathcal{S}_i = \{(x_{i,j}^{\mathcal{S}}, y_{i,j}^{\mathcal{S}})\}_{j=1}^{K}$ and $\mathcal{Q}_i = \{(x_{i,j}^{\mathcal{Q}}, y_{i,j}^{\mathcal{Q}})\}_{j=1}^{Q}$, where $\mathcal{S}_i$ contains K samples, referred to as the number of shots, and Q represents the number of query samples.

We introduce additional notations for clarity: $f(\cdot;\theta)$ denotes the output from the model parameterized by $\theta \in \mathbb{R}^d$. Also, $\mathcal{L}(\mathcal{D};\theta) = \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} l(f(x;\theta), y)$ is the averaged loss value for the samples in dataset $\mathcal{D}$ with loss function $l(\cdot,\cdot)$.

Review of MAML MAML by Finn et al. (2017) involves bi-level optimization of inner and outer loop processes. Through repetitive learning $\{\mathcal{T}_i\}_{i=1}^{\mathcal{B}}$, MAML optimizes an θ tailored to the bi-level optimization. In the inner loop, initialization θ is rapidly updated for the task using support set $\mathcal{S}_i$. We labeled the process as **Inner-Loop**(θ, k), where k is the number of updates in the inner loop (referring Equation 1). In the equation, we assume one-step fine-tuning with $k = 1$ to obtain task parameter ϕ_i from θ. In the outer loop, the initialization is meta-updated based on the loss incurred by processing query set $\mathcal{Q}_i$ with the fine-tuned parameter (referring Equation 2).

$$\phi_i \leftarrow \theta - \alpha\nabla_\theta \mathcal{L}(\mathcal{S}_i;\theta) \tag{1}$$

$$\theta^{*} \leftarrow \theta - \frac{\beta}{\mathcal{B}}\nabla_\theta \sum_{i=1}^{\mathcal{B}} \mathcal{L}(\mathcal{Q}_i;\phi_i), \tag{2}$$

where α and β are the learning rates for inner and outer optimization, respectively. By repeating the inner and outer loop optimization via task batches, the initialization θ converges to the meta-trained initialization θ^{*} which is ready for quickly adapting to a given novel task.

3.2 Overview of the Proposed Method

Our XB-MAML handles multiple initializations so that we newly denote the set of initializations, i.e.,

$\Theta = \{\theta^{(m)}\}_{m=1}^{M}$, where M is the number of initializations. Here, we describe our method by letting M as a variable, but it does not lose generality in presenting the methodology. After that, we present the algorithmic way to increase M to incorporate additional initialization. When focusing on inner and outer optimizations of XB-MAML for task $\mathcal{T}_i$, it starts from computing the loss value by processing the given support samples with each initialization parameter, i.e., $\{\mathcal{L}_i^{(m)}\}_{m=1}^{M}$. XB-MAML then prepares a new initialization $\theta_i^\star$ which is the linear combination of multi-initializations with coefficients $\{\sigma_i^{(m)}\}_{m=1}^{M}$ from softmax computation of the minus losses $\{-\mathcal{L}_i^{(m)}\}_{m=1}^{M}$:

$$\theta_i^\star = \sum_{m=1}^{M} \sigma_i^{(m)} \theta^{(m)}. \tag{3}$$

$\theta^\star$ is then further fine-tuned within the support set, as done by MAML (referring Equation 1), and evaluated on the query sets. Along with the loss from queries, we additionally apply a regularization loss $\mathcal{L}_{reg}$ by calculating the dot products between multi-initializations in order to enforce the orthogonality between initializations:

$$\mathcal{L}_{total,i}^{(m)} = \mathcal{L}(\mathcal{Q}_i; \phi_i^\star) + \mathcal{L}_{reg}^{(m)} \tag{4}$$

Finally, $\theta^\star$ is meta-updated in accordance, where the meta-update chains are further linked to the multi-initializations so that M initializations are eventually meta-updated:

$$\theta^{(m)} \leftarrow \theta^{(m)} - \frac{\beta}{\mathcal{B}} \sum_{i=1}^{\mathcal{B}} \nabla_{\theta^{(m)}} \mathcal{L}_{total,i}^{(m)} \tag{5}$$

In Equation 4 and 5, formulas are given per initialization. Through the iterative process as done by MAML, XB-MAML meta-trains multiple initializations. The pseudocode of the process is described in Algorithm 1. In line 14, Algorithm 2 determines whether we expand the basis, so let us describe the exact rule for the basis expansion as follows.

3.3 Expandable Basis Parameters

In this section, we describe how XB-MAML expands the multi-initializations. For given Θ, each individual initialization can be regarded as linearly independent due to the regularization loss so that it forms a basis in its parameter space. According to the formal definition of a basis, Θ can construct subspace $V \in \mathbb{R}^{M \times d}$ within the parameter space with rank M. XB-MAML aims to utilize the multi-initializations as bases to cover space V, which is the key process to obtain a wide range of tasks. Consequently, when task parameters are hard to be represented on space V, we trigger to increase the number of initializations. It can be understood as the expansion of the rank of Θ from $M \rightarrow M+1$.

Algorithm 1 Training procedures for XB-MAML

Hyperparameter: k: number of inner loop steps, $\mathcal{B}$: batch size, γ: temperature scaling factor, η: regularization hyperparameter
Require: $p(\mathcal{T})$: task distribution,
Require: $\{\mathcal{T}_i\}_{i=1}^{\mathcal{B}}$, where $\mathcal{T}_i \sim p(\mathcal{T})$
Parameter: $\Theta = \{\theta^{(m)}\}_{m=1}^{M}$: Set of initializations

1: Initialize Θ
2: **while** not done **do**
3: **for** $i = 1 : \mathcal{B}$ **do**
4: $\{\mathcal{L}^{(m)}\}_{m=1}^{M} = \{\mathcal{L}(\mathcal{S}_i; \theta^{(m)})\}_{m=1}^{M}$
5: $\{\sigma_i^{(m)}\}_{m=1}^{M} = \frac{\exp(-\mathcal{L}^{(m)}/\gamma)}{\sum_{(m')} \exp(-\mathcal{L}^{(m')}/\gamma)}$
6: $\theta_i^\star = \sum_{m=1}^{M} \sigma_i^{(m)} \theta^{(m)}$
7: $\phi_i^\star = \textbf{Inner-Loop}(\theta^\star, k)$
8: **end for**
9: **for** $m = 1 : M$ **do**
10: $\mathcal{L}_{reg}^{(m)} = \frac{\eta}{M-1} \sum_{m!=j} \theta^{(m)} \cdot (\theta^{(j)})^\intercal$
11: $\{\mathcal{L}_{total,i}^{(m)}\}_{i=1}^{\mathcal{B}} = \{\mathcal{L}(\mathcal{Q}_i; \phi_i^\star) + \mathcal{L}_{reg}^{(m)}\}_{i=1}^{\mathcal{B}}$
12: $\theta^{(m)} \leftarrow \theta^{(m)} - \frac{\beta}{\mathcal{B}} \sum_{i=1}^{\mathcal{B}} \nabla_{\theta^{(m)}} \mathcal{L}_{total,i}^{(m)}$
13: **end for**
14: Apply Algorithm 2
15: **end while**

Algorithm 2 Expanding basis for XB-MAML

Require: $\phi_i^\star$: finetuned parameters
Require I: index of epoch, $\mathcal{B}$: batch size
Hyperparameter: c: threshold

1: Initialize $\mathcal{E}[I] = 0$, count $= 0$
2: Construct Subspace $V = \text{span}\left\{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}\right\}$
3: **for** $i = 1 : \mathcal{B}$ **do**
4: $\phi_{i,proj}^\star$: Projection $\phi_i^\star$ onto subspace V
5: $\epsilon = \frac{||\phi_i^\star - \phi_{i,proj}^\star||_2^2}{||\phi_i^\star||_2^2}$
6: $\mathcal{E}[I] += \epsilon / \mathcal{B}$
7: **end for**
8: **if** $\mathcal{E}[I] > \mathcal{E}[I-1]$ **then**
9: count = count + 1
10: **else**
11: count = 0
12: **end if**
13: **if** count $> c$ **then**
14: Add initial model ▷ Eq.6
15: **end if**

Condition for Expanding Basis We introduced an intuitive metric to determine whether increasing the model parameters is necessary. As shown in Algorithm 2, we first make a subspace V within the set of basis vectors $\{\theta^{(m)}\}_{m=1}^{M}$. After executing the process

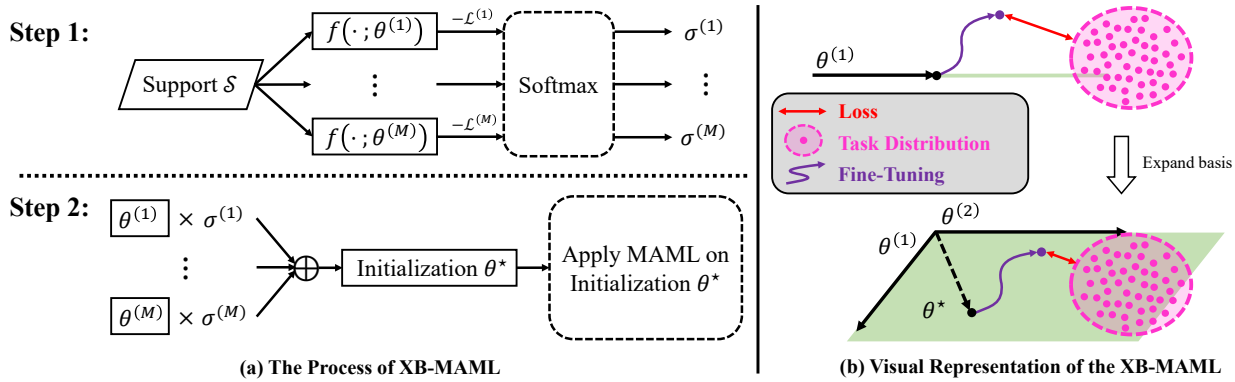


Figure 2: The conceptual illustrations of XB-MAML: **(a)** outlines the XB-MAML learning process. **(b)** illustrates the visual representation of the scenario that XB-MAML expands extra basis.

outlined in lines 3-13 of Algorithm 1, it projects the fine-tuned parameters $\phi^\star$ into its subspace S, which is represented as $\phi^\star_{proj}$. Following this, we calculate the ratio between $||\phi^\star - \phi^\star_{proj}||_2^2$ and $||\phi^\star||_2^2$, denoted as $\epsilon = \frac{||\phi^\star - \phi^\star_{proj}||_2^2}{||\phi^\star||_2^2}$, which serves as the primary metric for deciding whether to increase the initial model parameters or not. When task parameters $\phi^\star$ largely deviate from space V spanned by the meta-trained initializations, ϵ increases. It says that XB-MAML should increase the rank of basis to cover the diverged task parameters. After the sufficient training without adding extra initialization, there comes a point where ϵ starts to rise, signifying that the gap between $\phi^\star$ and $\phi^\star_{proj}$ becomes more significant than the power of $\phi^\star$. This phenomenon indicates that the projection error gains prominence. Based on the intuition, when ϵ continues to increase during the number of episodes, it becomes necessary to add more initializations. Specifically, for the current batch at I-th epoch, ϵ values are averaged across the tasks, i.e., $\mathcal{E}[I] += \epsilon/\mathcal{B}$. When $\mathcal{E}[I]$ is larger than the previous one, i.e., $\mathcal{E}[I] > \mathcal{E}[I-1]$, we increase a counter by $+1$. When the counter reaches a certain threshold c, it adds extra initialization. Algorithm 2 fully describes the condition for basis expansion.

Way to Expand Basis Instead of naively opting for a random parameter as an additional initialization, we employ a particular strategy involving the sampling of parameters from a probabilistic perspective. We build a Gaussian distribution with the average of the current initializations for mean, and white noise λI, where λ controls the variance and $I \in \mathbb{R}^{d\times d}$ is an identity matrix. We sample additional basis $\theta^{(M+1)}$ from the Gaussian distribution:

$$\theta^{(M+1)} \sim \mathcal{N}(\mu, \lambda I), \tag{6}$$

where μ is the average parameter of Θ. This probabilistic sampling allows XB-MAML to explore stochastic variants of the new initialization. Although the newly adopted basis is not forced to be orthogonal to the current bases, we confirm that XB-MAML quickly tunes the new basis to be orthogonal to search additional dimension. The overall outline of XB-MAML[1] is illustrated at Figure 2.

4 EXPERIMENTS

In this section, we present a comprehensive overview of experimental settings and results. We start by providing detailed descriptions of the experimental settings. Moreover, our observations demonstrate the improvement of our XB-MAML in few-shot classification for multiple and cross-domain classifications, surpassing the performance of previous studies. Additional results, such as single-domain datasets and their cross-domain classification, or experiments on larger backbone, are provided in Appendix C.

4.1 Experimental Settings

Datasets Description We primarily utilized benchmark datasets for our experiments of classification, which have been commonly used in previous works. Specifically, we employed three multi-domain datasets: *Meta-Datasets*-ABF/BTAF/CIO (Yao et al. (2019), Zhou et al. (2021), Jiang et al. (2022)). In addition, we also included three experiments on single-domain datasets: CIFAR-FS, *mini*-ImageNet, and *tiered*-ImageNet (Bertinetto et al. (2019), Ravi and Larochelle (2017), Ren et al. (2018)). Detail descriptions of each dataset is provided in Appendix A.

Hyperparameters Settings For the learning rates, we use $\alpha = 0.05$ and $\beta = 0.0007$ for the inner and outer loop. We conducted training up to 80,000 epochs across multiple datasets, with a batch size of 2. During

[1]XB-MAML github code is available at `https://github.com/johnjaejunlee95/XB-MAML`

Table 2: 5-way 5-shot accuracies on *Meta-Datasets*-ABF with 95% confidence intervals

Methods	AIRCRAFT	BIRD	FUNGI	Average
MAML (Finn et al. (2017))	69.70 ± 0.33	68.36 ± 0.72	53.65 ± 0.93	63.91
ProtoNet (Snell et al. (2017))	70.07 ± 0.14	71.49 ± 0.25	54.21 ± 0.31	65.26
HSML (Yao et al. (2019))	68.29 ± 0.56	70.11 ± 0.85	56.28 ± 1.01	64.89
ARML (Yao et al. (2020))	69.94 ± 0.78	71.55 ± 0.33	53.61 ± 0.89	65.04
TSA-MAML (5 init) (Zhou et al. (2021))	74.67 ± 0.77	71.06 ± 0.14	55.68 ± 0.27	66.14
MUSML (3 init) (Jiang et al. (2022))	**75.46 ± 0.89**	70.01 ± 0.56	50.40 ± 0.75	65.29
XB-MAML (4 init)	74.39 ± 0.38	**75.17 ± 0.67**	**56.85 ± 0.14**	**68.80**
MUSML(3 init) + Transduction	**79.23 ± 0.98**	76.21 ± 0.77	58.24 ± 0.80	71.22
XB-MAML(4 init) + Transduction	77.62 ± 0.89	**77.78 ± 0.39**	**58.34 ± 0.14**	**71.24**

Table 3: 5-way 5-shot accuracies on *Meta-Datasets*-BTAF with 95% confidence intervals

Methods	BIRD	TEXTURE	AIRCRAFT	FUNGI	Average
MAML (Finn et al. (2017))	67.69 ± 0.89	45.91 ± 0.54	66.93 ± 0.45	50.43 ± 0.80	57.74
ProtoNet (Snell et al. (2017))	71.97 ± 0.74	47.65 ± 0.49	69.96 ± 0.87	54.49 ± 0.41	60.02
HSML (Yao et al. (2019))	72.01 ± 0.65	49.00 ± 0.96	70.34 ± 0.68	55.21 ± 0.80	61.64
ARML (Yao et al. (2020))	71.30 ± 0.44	50.48 ± 0.22	70.44 ± 0.80	56.76 ± 0.56	62.25
TSA-MAML (5 init) (Zhou et al. (2021))	68.05 ± 0.94	49.61 ± 0.33	73.99 ± 0.46	53.36 ± 0.20	62.25
MUSML (4 init) (Jiang et al. (2022))	70.84 ± 0.32	49.63 ± 0.98	**75.73 ± 0.65**	49.74 ± 0.75	61.91
XB-MAML (5 init)	**75.49 ± 0.12**	**50.95 ± 0.93**	73.33 ± 0.16	**57.15 ± 0.71**	**64.23**
MUSML (4 init) + Transduction	75.32 ± 0.33	52.69 ± 0.15	**77.01 ± 0.92**	56.67 ± 0.81	65.45
XB-MAML (5 init) + Transduction	**75.45 ± 0.90**	**53.90 ± 0.85**	76.11 ± 0.43	**58.34 ± 0.93**	**66.20**

Table 4: 5-way 5-shot accuracies on *Meta-Datasets*-CIO with 95% confidence intervals

Methods	CIFAR-FS	*mini*-ImageNet	Omniglot	Average
MAML (Finn et al. (2017))	68.72 ± 0.43	59.84 ± 0.97	96.51 ± 0.68	75.02
ProtoNet (Snell et al. (2017))	69.53 ± 0.72	61.40 ± 0.64	97.67 ± 0.20	76.20
HSML (Yao et al. (2019))	70.81 ± 0.97	62.45 ± 0.44	96.34 ± 0.11	76.53
ARML (Yao et al. (2020))	70.40 ± 0.57	62.89 ± 0.48	96.80 ± 0.14	76.70
TSA-MAML (5 init) (Zhou et al. (2021))	69.35 ± 0.26	61.20 ± 0.20	98.65 ± 0.02	76.40
MUSML (3 init) (Jiang et al. (2022))	67.97 ± 0.65	59.00 ± 1.64	92.99 ± 0.41	73.41
XB-MAML (6 init)	**74.90 ± 0.35**	**65.63 ± 0.12**	**98.89 ± 0.09**	**79.81**
MUSML (3 init) + Transduction	75.03 ± 0.38	65.54 ± 0.54	96.84 ± 0.11	79.14
XB-MAML (6 init) + Transduction	**76.87 ± 0.73**	**68.66 ± 0.53**	**98.41 ± 0.04**	**81.31**

meta-validation/test, we evaluated model performance on 600 tasks for each dataset in the multiple datasets. Lastly, to ensure a fair comparison, we reproduced all compared approaches via PyTorch, including MAML, ProtoNet, HSML, ARML, TSA-MAML, and MUSML. Moreover, because some methods used augmentations while others did not, we opted not to apply augmentations except normalization to all methods, including ours. Details of the hyperparameter settings are provided in Appendix B.2.

4.2 Results

Multiple Domain Datasets Classification Tables 2, 3, and 4 demonstrate the outstanding results of XB-MAML in the 5-way 5-shot classification. This performance gain remains consistent across three multi-domain datasets: *Meta-Datasets*-ABF/BTAF/CIO. The most significant enhancement is observed in *Meta-Datasets*-CIO, which surpasses previous methods by approximately +3%. We also achieve performance gains of around +2% on remaining datasets.

Transduction Setting Since the original MUSML (Jiang et al. (2022)) applies a transduction setting that utilizes both support and query sets for task-adaptation, we ran experiments in a similar transduction setting for the comparison, as referenced by TPN (Liu et al. (2019b)). TPN modifies loss term with additional regularization by:

$$\mathcal{L}_{trans} = \mathcal{L}_{total} + \sum_{i=1}^{\mathfrak{N}} \sum_{j=1}^{\mathfrak{N}} W_{(i,j)} ||f(x_i;\phi) - f(x_j;\phi)||_2^2$$

where, $W_{(i,j)} = \exp\left(-\frac{1}{2\eta^2}\left|\left|f_e(x_i;\phi) - f_e(x_j;\phi)\right|\right|_2^2\right)$

Here, $\mathfrak{N} = N \times N_s + N_q$, where N, N_s, and N_q is the number of ways and the number of samples in the support and query sets. Additionally, $f_e(\cdot;\phi)$ is the output from backbone of the model parameterized by ϕ, yielding feature representation vectors, while η is a scaling factor fixed at a value of 0.25. As the result, our method still outperforms MUSML at experiments in transduction settings.

Cross-domain Classification We also conducted experiments with cross-domain datasets, where a model is trained on one specific domain and then evaluated on other unseen domains. The results presented in Table 5 reveal a substantial performance improvement, with a gain of approximately up to +4%. This outcome validates that XB-MAML effectively generates effective initial model parameters in unseen domains. Here, we abbreviated *Meta-datasets*-ABF/BTAF/CIO as ABF/BTAF/CIO.

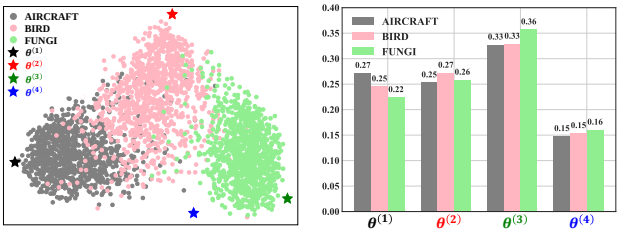
Table 5: 5-way 5-shot cross-domain classification

Methods	MAML	TSA-MAML	MUSML	**XB-MAML**
ABF→BTAF	58.27	58.67	57.96	**61.15**
ABF→CIO	60.82	62.90	62.35	**66.73**
BTAF→ABF	63.84	65.79	64.08	**68.53**
BTAF→CIO	62.69	64.24	62.99	**69.82**
CIO→ABF	46.38	48.58	48.55	**51.60**
CIO→BTAF	44.31	46.24	45.89	**49.03**

5 ANALYSIS

5.1 t-SNE Visualization

t-SNE plots could be one of the effective choices to illustrate the distributions of model parameters. We illustrate the t-SNE visualization of the fine-tuned parameters and all initialization parameters across multiple datasets, particularly focusing on the *Meta-Datasets*-ABF experiments. As shown in Figure 3, it is evident that the initialized model parameters are well-distributed (depicted by the colored marker $\star$), while the fine-tuned parameters have been clustered according to their respective datasets (depicted by the colored dots). Notably, Figure 3b demonstrates the alignment of coefficients with the t-SNE visualization concept. For instance, the coefficient associated with the AIRCRAFT dataset represents the largest value in $\theta^{(1)}$, and this corresponds to the proximity of $\theta^{(1)}$ to the AIRCRAFT cluster. Furthermore, it is noteworthy that $\theta^{(4)}$ has a comparatively less impact than the others, but still crucial in adapting to various domains, which is positioned in a way that facilitates easy adaptation to any domain.



(a) t-SNE Visualization (b) Normalized Coefficients σ

Figure 3: Visualization results in *Meta-Datasets*-ABF. (a) illustrates a t-SNE plot comparing the finetuned parameters and initial model parameters. (b) represents the average normalized coefficients σ indicating the impact of each $\theta^{(m)}$ when linearly combined with the datasets.

5.2 Tendency of ϵ

Figure 4 displays the impact of our parameter addition metric. When the moving average shows a sustained rise over multiple epochs, we decide to add more initial model parameters. We selected a threshold c as 500, to monitor the behavior of ϵ for a certain steps. The sudden rise in ϵ, almost up to 1, is due to the instability in constructing the subspaces after adding an initialization. This instability occurs when increasing the number of initializations. However, it rapidly decreases and converges within a few steps, indicating that it stabilizes in constructing the basis. After a few steps of adopting new initializations, ϵ settles down near to 0, which means that the introduced initializations are sufficient to cover the task distribution.

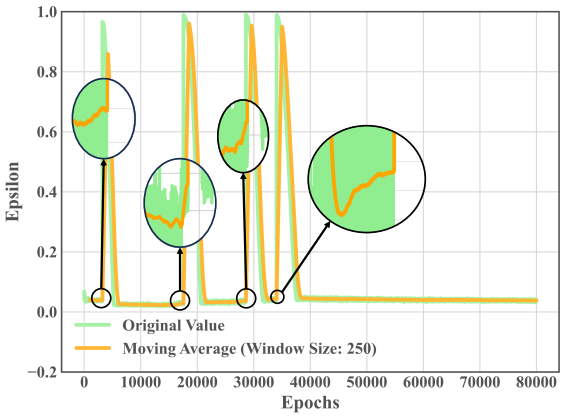


Figure 4: ϵ on *Meta-Datasets*-BTAF

5.3 Singular Value Decomposition

Figure 5 displays the normalized singular values at each datasets from Singular Value Decomposition (SVD) of the multi-initializations. All singular values fall within the range of approximately 0.1 to 0.3.

These results ensure that no single model parameter dominates, and most model parameters exhibit similarity, in all datasets. Also, this observation implies that the majority of model parameters has spanned effectively in parameter space.

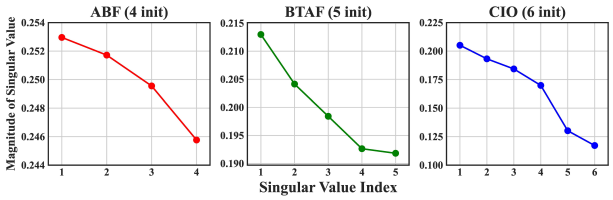


Figure 5: Normalized singular values of initializations

5.4 Computational Complexity

As our method adaptively increases the number of initializations, concerns about computational costs may arise. To address these concerns, we conducted a comparative analysis of our approach and two other methods in terms of computational complexity. We assume that the computational complexity of the forward and backward processes is $O(n)$, where n represents the number of model weight parameters. Additionally, we assume that the backpropagation during the outer loop process is $O(n^2)$, which includes the Hessian matrix computation. M denotes the number of initializations. As a result, our approach demonstrates efficiency, imposing no substantial computational burden in comparison to others, as demonstrated in Table 6.

Table 6: Analysis of computational complexity

Methods	Inner Loop	Outer Loop	Total
TSA-MAML	$O(3n)$	$O(n^2)$	$\boldsymbol{O(n^2)}$
MUSML	$O(M(n+2n'))^{\dagger}$	$O(Mn^2)^{\dagger\dagger}$	$O(Mn^2)$
XB-MAML	$O((M+2)n)$	$O(n^2)$	$\boldsymbol{O(n^2)}$

†: n' is the number of subspace weight parameters in MUSML (Jiang et al. (2022)).
††: Applied relaxation operation, which can be updated model parameters simultaneously (Liu et al. (2019a)).

5.5 Ablation Studies

Fixed Number of Model Parameters In the ablation study, we first introduced a variant of XB-MAML that trains with a fixed number of initialized models, given the same hyperparameter settings. For instance, as XB-MAML utilizes 6 initializations in the case of *Meta-Datasets*-CIO, we compare it with a variant of XB-MAML that maintains this number of initializations at 6 from the beginning of the training. The results presented in Figure 6 and Table 7 demonstrate that starting with a single initialized model proves to be more efficient than starting with a fixed number of model. First, Figure 6 illustrates this efficiency, which is particularly noticeable in the faster convergence of XB-MAML compared to XB-MAML+fixed. Also, Table 7 highlights the final outcomes, indicating that XB-MAML outperforms XB-MAML+fixed with the latter representing method that starts with a fixed number of initialized models.

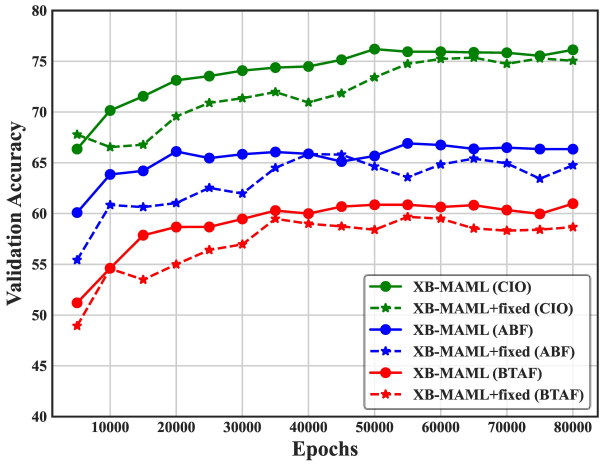


Figure 6: Validation accuracies of XB-MAML and XB-MAML+fixed

Table 7: Results of XB-MAML and XB-MAML+fixed

Datasets	**XB-MAML**	XB-MAML+fixed
ABF (4 init)	**68.80 ± 0.47**	67.43 ± 0.38
BTAF (5 init)	**64.23 ± 0.55**	62.19 ± 0.76
CIO (6 init)	**79.81 ± 0.23**	77.35 ± 0.14

Other Choices to Compute $\boldsymbol{\sigma}$ When answering the reason why we choose 'Softmax of minus-loss' ($\exp(-\mathcal{L})$) method in computing σ, we opt to use a simple function that outputs a larger positive $\sigma^{(m)}$ when the loss $\mathcal{L}^{(m)}$ is small, ensuring normalized σ, likewise the attention module. This directly makes us to choose 'Softmax of minus-loss'. For the ablation study, we additionally compared our method to other choices, including 'Equal coefficient', where $\sigma^{(m)} = 1/M$, 'Inverse loss coefficient', where $\sigma^{(m)} = 1/\mathcal{L}^{(m)}$, and 'Softmax of minus-loss $\times M$', where $M\exp(-\mathcal{L})$, ensuring that the sum of all values exceeds 1. As the results in Table 8, our choice is shown to be the best.

Table 8: Results of the various ways to compute σ

	$1/M$	$1/\mathcal{L}$	$M\exp(-\mathcal{L})$	$\boldsymbol{\exp(-\mathcal{L})}$
ABF	65.32	68.30	64.67	**68.80**
BTAF	61.57	63.98	61.04	**64.23**
CIO	76.18	77.34	76.62	**79.81**

Sensitivity of c As an ablation on c, which determines the expansion of bases as described in Algorithm 2, we deviated it from 250 to 1,000. As the results, shown in Table 9, we found that $c = 500$ is the best. This observation suggests that too small value of c leads to more frequent addition of a new basis, which hinders sufficient training. Also, too large c could suppress the expansion of the basis, which hampers the ability to cover a wide range of task distribution.

Table 9: Accuracies for various choices of c

c	250	**500**	750	1000
ABF	67.68 (6 init)	**68.80 (4 init)**	67.00 (2 init)	64.02 (1 init)
BTAF	63.29 (8 init)	**64.23 (5 init)**	62.99 (2 init)	58.12 (1 init)
CIO	75.39 (10 init)	**79.81 (6 init)**	77.91 (4 init)	77.46 (3 init)

Sensitivity of λ As our method relies on Gaussian sampling when adding additional initializations, the hyperparameter λ plays a crucial role in controlling the variance of the resulting Gaussian distribution. To explore its impact, we conducted the ablation studies with varying values of λ, specifically $\lambda = \{0.005,\ 0.01,\ 0.05,\ 0.1\}$. As shown in Figure 7, we provided experimental results on multi-domain datasets with several choices of λ. These results clearly show that selecting an appropriate value for λ is important for achieving better performance. When λ becomes too large or too small, it adversely impacts the performance. A large λ introduces high uncertainty in the sampling process, hindering effective learning. Conversely, a small λ results in a sampling process that closely samples around the current initializations, which could disrupt the construction of an optimal subspace to cover the task distribution. We use $\lambda = 0.01$, which shows the best performance.

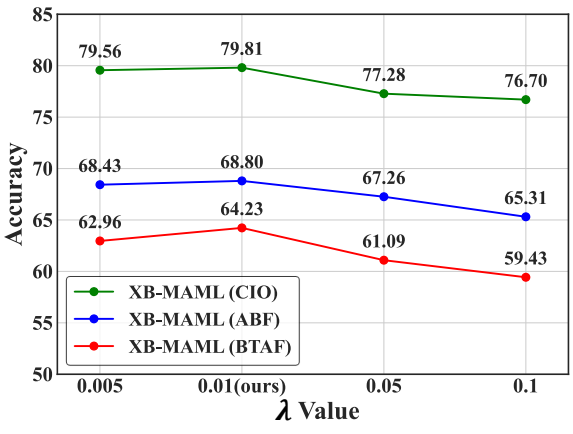


Figure 7: Sensitivity of λ

The Effect of Number of Initializations Here, we analyze the impact of the allowed maximum number of initializations of XB-MAML. When only 2 initializations are allowed, we stop expanding the initializations beyond 2. As shown in Table 10, the results indicate that while performance begins to converge at 4 initializations, further increasing the number of initializations still enhances performance. This suggests that having an adequate number of initialized models is essential. Also, the results confirm that XB-MAML gradually expands its initializations to reach the optimal rank.

Table 10: Results with the various maximum number of initializations for *Meta-Datasets*-CIO

Datasets	1 init	2 init	3init	4 init	5 init	**6 init**
CIO	75.02	77.76	78.27	79.44	79.38	**79.82**

The Effect of $\mathcal{L}_{\mathbf{reg}}$ As our approach introduces an additional dot product regularization loss to encourage orthogonality among the initial model parameters, it prompts questions about how this regularization loss influences the span of the set Θ, which acts as a basis, and overall performance. Table 11 shows that without $\mathcal{L}_{reg}$, XB-MAML fails to fully span the initializations and to enforce orthogonality (as indicated by cosine similarity), leading to the performance degradation.

Table 11: Results with $\mathcal{L}_{reg}$ for *Meta-Datasets*-ABF

Loss	Accuracy	Cosine Similarity†
With $\mathcal{L}_{\mathbf{reg}}$	**68.80 ± 0.47**	**3.95×10^{-5}**
Without $\mathcal{L}_{reg}$	64.32 ± 0.76	7.50×10^{-1}

†: Averaged cosine similarity between initializations

6 CONCLUSION

We introduce XB-MAML, a novel meta-learning approach that adaptively increases the number of initialized models and refines the initialization points through linear combinations, contributing to more efficient meta-learning. The extensive analysis illustrates that XB-MAML competently covers complex and diverse task distributions, particularly in the context of multi-domain and cross-domain classification. Furthermore, we enhanced the performance by treating initialized models as bases, and enforcing orthogonality among them through regularization loss, resulting in the improved performance compared to the absence of such regularization. Finally, our method achieved state-of-the-art results on multi-domain datasets and their cross-domain classifications. We hope this work could provide new perspectives in the research field of meta-learning when solving diverse unseen tasks.

Acknowledgements

This work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01336, Artificial Intelligence Graduate School Program (UNIST)), (No. 2021-0-02201, Federated Learning for Privacy Preserving Video Caching Networks) (No. 2023-RS-2022-00156361, Innovative Human Resource Development for Local Intellectualization support program), and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1C1C1012797).

References

beejisbrigit, Y. C. (2018). 2018 fgcvx fungi classification challenge. Kaggle.

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

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. (2014). Describing textures in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3606–3613.

Deleu, T., Würfl, T., Samiei, M., Cohen, J. P., and Bengio, Y. (2019). Torchmeta: A Meta-Learning library for PyTorch. Available at: https://github.com/tristandeleu/pytorch-meta.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Finn, C., Xu, K., and Levine, S. (2018). Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31.

Hou, R., Chang, H., MA, B., Shan, S., and Chen, X. (2019). Cross attention network for few-shot classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Jiang, W., Kwok, J., and Zhang, Y. (2022). Subspace learning for effective meta-learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pages 10177–10194. PMLR.

Krizhevsky, A., Nair, V., and Hinton, G. (2010). Cifar-10 (canadian institute for advanced research). *http://www.cs.toronto.edu/kriz/cifar.html*, 5(4):1.

Lai, J., Yang, S., Wu, W., Wu, T., Jiang, G., Wang, X., Liu, J., Gao, B.-B., Zhang, W., Xie, Y., and Wang, C. (2023). Spatialformer: Semantic and target aware attentions for few-shot learning. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 37(7):8430–8437.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Li, Z., Zhou, F., Chen, F., and Li, H. (2017). Meta-sgd: Learning to learn quickly for few shot learning. In *CoRR*.

Liu, H., Simonyan, K., and Yang, Y. (2019a). DARTS: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*.

Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S. J., and Yang, Y. (2019b). Learning to propagate labels: Transductive propagation network for few-shot learning. In *International Conference on Learning Representations (ICLR)*.

Maji, S., Kannala, J., Rahtu, E., Blaschko, M., and Vedaldi, A. (2013). Fine-grained visual classification of aircraft. Technical report.

Nichol, A., Achiam, J., and Schulman, J. (2018). On first-order meta-learning algorithms. In *ArXiv Preprint arXiv:1803.02999*.

Oh, J., Yoo, H., Kim, C., and Yun, S.-Y. (2021). Boil: Towards representation change for few-shot learning. In *International Conference on Learning Representations (ICLR)*.

Oreshkin, B., Rodríguez López, P., and Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31.

Raghu, A., Raghu, M., Bengio, S., and Vinyals, O. (2020). Rapid learning or feature reuse? towards understanding the effectiveness of maml. In *International Conference on Learning Representations (ICLR)*.

Rajeswaran, A., Finn, C., Kakade, S. M., and Levine, S. (2019). Meta-learning with implicit gradients. In

Advances in Neural Information Processing Systems (NeurIPS), volume 32.

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

Ren, M., Ravi, S., Triantafillou, E., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations (ICLR)*.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 30.

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.-A., and Larochelle, H. (2020). Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations (ICLR)*.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., and Others (2016). Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 29.

Vuorio, R., Sun, S.-H., Hu, H., and Lim, J. J. (2019). Multimodal model-agnostic meta-learning via task-aware modulation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). *The Caltech-UCSD Birds-200-2011 Dataset*. Number CNS-TR-2011-001.

Yao, H., Wei, Y., Huang, J., and Li, Z. (2019). Hierarchically structured meta-learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 7045–7054. PMLR.

Yao, H., Wu, X., Tao, Z., Li, Y., Ding, B., Li, R., and Li, Z. (2020). Automated relational meta-learning. In *International Conference on Learning Representations (ICLR)*.

Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. (2018). Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31.

Yoon, S. W., Seo, J., and Moon, J. (2019). TapNet: Neural network augmented with task-adaptive projection for few-shot learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 7115–7123. PMLR.

Zhou, P., Zou, Y., Yuan, X.-T., Feng, J., Xiong, C., and Hoi, S. (2021). Task similarity aware meta learning: Theory-inspired improvement on maml. In *Uncertainty in Artificial Intelligence (UAI)*, pages 23–33.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [**Yes**/No/Not Applicable]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [**Yes**/No/Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [**Yes**/No/Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes/No/**Not Applicable**]
   (b) Complete proofs of all theoretical results. [Yes/No/**Not Applicable**]
   (c) Clear explanations of any assumptions. [Yes/No/**Not Applicable**]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [**Yes**/No/Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [**Yes**/No/Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [**Yes**/No/Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [**Yes**/No/Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [**Yes**/No/Not Applicable]
   (b) The license information of the assets, if applicable. [**Yes**/No/Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [**Yes**/No/Not Applicable]
   (d) Information about consent from data providers/curators. [**Yes**/No/Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [**Yes**/No/Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Yes/No/**Not Applicable**]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/**Not Applicable**]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/**Not Applicable**]

XB-MAML: Supplementary Materials

A DATASET DESCRIPTION

A.1 Single-domain Datasets

***mini*-ImageNet**: *mini*-ImageNet, as introduced by Vinyals et al. (2016), contains 100 classes with 600 images each, where the images have a size of 84×84×3. It is derived from a subset of the ImageNet dataset (Deng et al. (2009)). The data split follows the protocols proposed by Ravi and Larochelle (2017), involving 64 classes for the training split, 16 classes for the validation split, and 20 classes for the test split.

***tiered*-ImageNet**: *tiered*-ImageNet, first referenced by Ren et al. (2018), is composed of 608 classes distributed across 34 categories, and the images have a size of 84×84×3, which is derived from ImageNet datasets. Among these categories, 20 are allocated for training, 6 for validation, and 8 for testing. Each category further contains a varying number of classes. In total, the training split contains 351 classes with 448,695 images, the validation split contains 97 classes with 124,261 images, and the test split contains 160 classes with 206,209 images.

CIFAR-FS: CIFAR-FS is originally comprised by Bertinetto et al. (2019). This dataset is constructed by randomly sampling from CIFAR-100 datasets (Krizhevsky et al. (2010)), and it follows a similar data splitting strategy as the *mini*-ImageNet case. The images in CIFAR-FS have a size of 32×32×3 and are divided into 64/16/20 classes for train/validation/test splits, where each class containing 600 images.

A.2 Multiple-domain Datasets

We present three multi-domain datasets for classification, denoted as *Meta-Datasets*-ABF/BTAF/CIO. The dataset labels are defined as follows: 'A' for Aircraft, 'B' for Bird, 'C' for CIFAR-FS, 'F' for Fungi, 'I' for *mini*-ImageNet, 'O' for Omniglot, and 'T' for Texture. All the images in these datasets have been resized to 84×84×3. Additionally, as following the work by Yao et al. (2019), we have partitioned each of these datasets into 64/12/20 for Train/Validation/Test splits, with the exception of the Omniglot and Texture cases. We provide descriptions for all datasets except CIFAR-FS and *mini*-ImageNet, which are covered separately in Section A.1.

Aircraft: The Aircraft dataset, officially published by Maji et al. (2013), consists of images of 102 different types of aircraft, with 100 images for each type.

Bird: The Bird dataset, proposed by Wah et al. (2011), comprises 200 different bird species and a total of 11,788 images. In alignment with the protocol presented in the work by Yao et al. (2019). We have randomly chosen 100 bird species, each with 60 images.

Fungi: The Fungi dataset contains 1,500 distinct fungi species that has more than 100,000 images, which has been introduced by beejisbrigit (2018). We filtered out species with fewer than 150 images, following the method described in Yao et al. (2019). Subsequently, we have randomly selected 100 species, each having 150 images.

Omniglot: The Omniglot dataset proposed by Lake et al. (2015) comprises 20 instances within 1,623 characters from 50 different alphabets. Specifically, each instance is handwritten by different persons. We have splitted 1623 characters into 792/172/659 for Train/Validation/Test.

Texture: The Texture dataset comprises 5,640 texture images from 47 classes, with each class containing 120 images, which has been provided by Cimpoi et al. (2014). In addition, we split these classes into 30/7/10 for Train/Validation/Test.

B MODEL AND HYPERPARAMETER DESCRIPTION

B.1 Model Architecture

We adopt standard Conv-4 backbone (Vinyals et al. (2016); Finn et al. (2017)) with four convolutional blocks as a feature extractor. Each block comprises 3×3 convolutional layers with 64 filters, 1 stride, 1 padding, along with a batch normalization layer, ReLU activation functions, and 2×2 max pooling (Conv 3×3 - BatchNorm - ReLU - MaxPool 2×2). In last, we use a linear classifier that involves adding a fully-connected layer at the end of the Conv-4 backbone. All implementations follow the framework provided by Torchmeta (Deleu et al. (2019)). All our experiments were run on a single NVIDIA RTX A5000 or A6000 processor.

B.2 Hyperparameters Settings

We adopt the following hyperparameter settings described in Table 12.

Table 12: Hyperparameter Settings

Settings	Single-Domain Datasets	Multi-Domain Datasets
Epochs	60,000	80,000
Batch Size (1-shot, 5-shots)	4, 2	4, 2
Inner-Loop Update Steps (Train, Test)	3, 7	3, 7
Inner-Loop Learning Rate (α)	0.03	0.05
Outer-Loop Learning Rate (β)*	0.001	0.001
Epoch Threshold (c)	500	500
White Noise (λ)	0.01	0.01
Temperature Scaling (γ)	5	8
Dot Products Regularizer (η)	0.0005	0.001

*: We decrease β by multiplying 0.8 at every 20,000 epochs.

C ADDITIONAL RESULTS

C.1 Single Datasets Classification

In this section, we present the evaluation results on the single-domain datasets classification, which were excluded from the main paper due to space limitations. Table 13 showcases the 5-way 5-shot performance of XB-MAML in single datasets classification, demonstrating notable improvements compared to other approaches, achieving an improvement of approximately +1%. These results highlights the effectiveness of XB-MAML in handling single-domain datasets and its outstanding performance in comparison to existing methods, akin to its performance in multi-domain datasets.

Table 13: 5-way 5-shot Accuracies on single-domain datasets with 95% confidence intervals

Methods	CIFAR-FS	*mini*-ImageNet	*tiered*-ImageNet
MAML (Finn et al. (2017))	72.41 ± 0.28	63.54 ± 0.19	65.58 ± 0.15
ProtoNet (Snell et al. (2017))	72.48 ± 0.29	66.17 ± 0.15	68.32 ± 0.14
HSML (Yao et al. (2019))	74.34 ± 0.37	65.11 ± 0.32	67.18 ± 0.45
ARML (Yao et al. (2020))	74.76 ± 0.56	65.56 ± 0.71	67.77 ± 0.63
TSA-MAML (5 init) (Zhou et al. (2021))	72.74 ± 0.19	64.29 ± 0.11	66.40 ± 0.24
MUSML (2 init) (Jiang et al. (2022))	73.09 ± 0.79	64.46 ± 0.32	68.05 ± 0.31
XB-MAML*	**75.82 ± 0.26**	**66.69 ± 0.56**	**68.91 ± 0.38**

*: XB-MAML finally adopts 2 for CIFAR-FS, and 4 initializations for both *mini* & *tiered*-ImageNet.

C.2 Additional Results for Cross-domain Classification

We provide additional results for cross-domain classification, as shown in Table 14. Our method exhibits outstanding performance compared to other approaches, with consistent improvements of up to approximately +1%. It confirms that XB-MAML is consistently effective for the various cases of cross-domain settings.

Table 14: 5-way 5-shot accuracies on cross-domain classification

Train Datasets	CIFAR-FS		*mini*-ImageNet		*tiered*-ImageNet	
Test Datsets	*mini*	*tiered*	CIFAR	*tiered*	CIFAR	*mini*
MAML (Finn et al. (2017))	55.97	51.30	59.14	56.52	56.93	57.89
TSA-MAML (Zhou et al. (2021))	56.43	51.87	60.21	59.34	60.34	61.84
MUSML (Jiang et al. (2022))	56.79	52.30	60.23	59.69	60.92	61.23
XB-MAML	**57.67**	**53.40**	**61.58**	**61.74**	**61.91**	**62.01**

C.3 Additional Results in Bigger Backbone

Given that previous experiments were conducted exclusively using the Conv-4 backbone, there could be some concerns that how XB-MAML would work in a larger backbone. Consequently, we also experimented with ResNet-12, which is also widely utilized as a standard backbone in the meta-learning field. As shown in Table 15, XB-MAML also shows the best performance in the ResNet-12 backbone compared to other methods. Interestingly, the number of bases decreased to 2 or 3. We conjecture that it is due to the sufficient number of parameters already given in the larger backbone that reduces the demands of additional initializations. These results are based on the average of 3 experimental runs.

Table 15: Multi-domain classification results with the ResNet-12 backbone

	MAML	TSA-MAML	**XB-MAML**
ABF	67.76	68.28	**70.83 (2 init)**
BTAF	63.57	65.02	**67.60 (2 init)**
CIO	82.46	82.74	**84.03 (3 init)**

D ADDITIONAL ABLATION STUDIES

In the recent investigation of the meta-learning research field, the significance of the feature extractor and classifier has become more pronounced, prompting an investigation into which one has a greater impact on performance. ANIL by Raghu et al. (2020) suggests that rapid adaptation of the classifier can only yield performance results nearly to MAML (Finn et al. (2017)), which encourages the reuse of a single feature representation across tasks. In contrast, BOIL (Oh et al. (2021)) argues that fine-tuning the feature extractor during the inner loop with fixed classifiers proves exceptionally effective where it leverages the diversity of feature representations.

Hence, we investigate the efficacy of the feature extractor, i.e., body, and classifier, i.e., head, in the context of our approach. We divide the analysis into two components: increasing the classifier only and increasing the feature extractor only. We denote these variants of XB-MAML as XB-MAML-head and XB-MAML-body, where multi-heads are adopted with a single-body initialization and multi-bodies are adopted with a single-head initialization, respectively. Here, we itemize the related key questions as follows:

Q1 Is it essential to introduce a number of classifiers to manage a wide range of task distributions?

Q2 Is it required to introduce multiple feature extractors to effectively extract the diverse and complex representations across domains?

Q3 Is there any synergy of multi-body and multi-head in XB-MAML?

Table 16: 5-way 5-shot accuracies on XB-MAML and its variants with 95% confidence intervals

Domain	Datasets	**XB-MAML**	XB-MAML-head	XB-MAML-body
Single	CIFAR-FS	**75.82 ± 0.26 (2 init)**	73.17 ± 0.19 (2 init)	74.02 ± 0.46 (3 init)
	mini-ImageNet	**66.69 ± 0.56 (4 init)**	63.12 ± 0.35 (2 init)	64.12 ± 0.21 (3 init)
	tiered-ImageNet	**68.91 ± 0.38 (4 init)**	67.05 ± 0.26 (2 init)	67.54 ± 0.36 (3 init)
Multi	*Meta-Datasets*-ABF	**68.80 ± 0.49 (4 init)**	66.94 ± 0.18 (2 init)	67.84 ± 0.73 (8 init)
	Meta-Datasets-BTAF	**64.23 ± 0.27 (5 init)**	61.24 ± 0.24 (2 init)	63.95 ± 0.49 (8 init)
	Meta-Datasets-CIO	**79.81 ± 0.11 (6 init)**	77.50 ± 0.32 (4 init)	78.34 ± 0.13 (10 init)

Q1 links to the prior understanding of the importance of the classifier part as pointed out by ANIL. On the other hand, **Q2** is related to the observation by BOIL which argues to diversify the feature extractor part via fine-tuning. Finally, **Q3** is for clarifying the synergetic effect of multi-body and multi-head via XB-MAML.

Regarding **Q1**, our findings based on Table 16 suggest that while incorporating multiple heads, i.e., classifiers, can indeed yield meaningful gains over MAML (referring to the accuracies in the main paper), too many head initializations are not required. It is because the classifier primarily comprises fully-connected layers, which are simple linear models that can be efficiently fine-tuned during the inner loop process with a minimal number of initializations. However, it seems that we need a few number of head initializations ranging from 2 to 4.

Conversely, when answering to **Q2**, the feature extractor requires a greater number of initializations compared to XB-MAML-head, as indicated in Table 16. Given that the feature extractor is responsible for representing feature distributions, it may require multiple initializations to effectively encompass a broad spectrum of features from distinctive domains. This aspect highlights that the feature extractor primarily contributes to the construction of an optimal subspace within the parameter space. Moreover, this is also closely tied to the idea that introducing diversity in feature representation is more critical than diversifying selection ability.

Going beyond the previous inquiries, let us think of the answer of **Q3**. We observe that XB-MAML which meta-trained multi-initializations for both body and head, i.e., an essential aspect of our method, yields the synergetic gains compared to XB-MAML-body and XB-MAML-head. Moreover, our method reconciles the multi-body and multi-head cases to find fewer initializations than XB-MAML-body but more than XB-MAML-head. This observation underscores the synergy between increasing the feature extractor and classifier, facilitating a more diverse task adaptation within feature representation and classifier ability.