LEARNING TO BALANCE: BAYESIAN META-LEARNING FOR IMBALANCED AND OUT-OF-DISTRIBUTION TASKS

Hae Beom Lee[1*], **Hayeon Lee**[1*], **Donghyun Na**[2*],
Saehoon Kim[3], **Minseop Park**[3], **Eunho Yang**[1,3], **Sung Ju Hwang**[1,3]
KAIST[1], TmaxData[2], AITRICS[3], South Korea
{haebeom.lee, hayeon926, eunhoy, sjhwang82}@kaist.ac.kr
donghyun_na@tmax.co.kr, {shkim, mike_seop}@aitrics.com

ABSTRACT

While tasks could come with varying the number of instances and classes in realistic settings, the existing meta-learning approaches for few-shot classification assume that the number of instances per task and class is fixed. Due to such restriction, they learn to *equally* utilize the meta-knowledge across all the tasks, even when the number of instances per task and class largely varies. Moreover, they do not consider distributional difference in unseen tasks, on which the meta-knowledge may have less usefulness depending on the task relatedness. To overcome these limitations, we propose a novel meta-learning model that *adaptively balances* the effect of the meta-learning and task-specific learning within each task. Through the learning of the balancing variables, we can decide whether to obtain a solution by relying on the meta-knowledge or task-specific learning. We formulate this objective into a Bayesian inference framework and tackle it using variational inference. We validate our Bayesian Task-Adaptive Meta-Learning (Bayesian TAML) on multiple realistic task- and class-imbalanced datasets, on which it significantly outperforms existing meta-learning approaches. Further ablation study confirms the effectiveness of each balancing component and the Bayesian learning framework.

1 INTRODUCTION

Despite the success of deep learning in many real-world tasks such as visual recognition and machine translation, such good performances are achievable at the availability of large training data, and many fail to generalize well in small data regimes. To overcome this limitation of conventional deep learning, recently, researchers have explored meta-learning (Schmidhuber, 1987; Thrun & Pratt, 1998) approaches, whose goal is to learn a model that generalizes well over distribution of tasks, rather than instances from a single task, in order to utilize the obtained meta-knowledge across tasks to compensate for the lack of training data for each task.

However, so far, most existing meta-learning approaches (Santoro et al., 2016; Vinyals et al., 2016; Snell et al., 2017; Ravi & Larochelle, 2017; Finn et al., 2017; Li et al., 2017) have only targeted an artificial scenario where all tasks participating in the multi-class classification problem have equal number of training instances per class. Yet, this is a highly restrictive setting, as in real-world scenarios, tasks that arrive at the model may have different training instances (task imbalance), and within each task, the number of training instances per class may largely vary (class imbalance). Moreover, the new task may come from a distribution that is different from the task distribution the model has been trained on (out-of-distribution task) (See (a) of Figure 1).

Under such a realistic setting, the meta-knowledge may have a varying degree of utility to each task. Tasks with small number of training data, or close to the tasks trained in meta-training step may want to rely mostly on meta-knowledge obtained over other tasks, whereas tasks that are out-of-distribution or come with more number of training data may obtain better solutions when trained in a task-specific

[*]Equal contribution

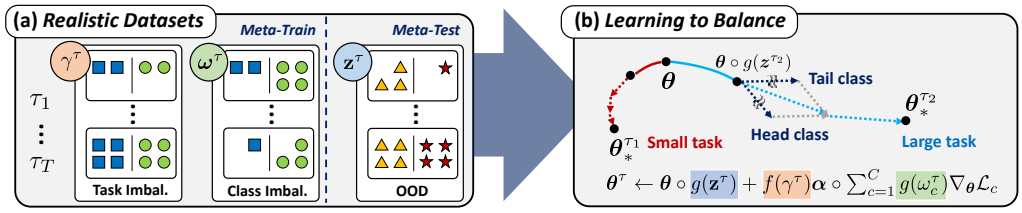


Figure 1: **Concept.** (a) To handle task imbalance, class imbalance and out-of-distribution (OOD) tasks, we introduce task-specific balancing variables $\boldsymbol{\gamma}^\tau$, $\boldsymbol{\omega}^\tau$, and $\mathbf{z}^\tau$, respectively. (b) With those variables, we learn to balance between the meta-knowledge $\boldsymbol{\theta}$ and task-specific update to handle imbalances and distributional shift.

manner. Furthermore, for multi-class classification, we may want to treat the learning for each class differently to handle class imbalance. Thus, to optimally leverage meta-learning under various imbalances, it would be beneficial for the model to task- and class-adaptively decide how much to use from the meta-learner, and how much to learn specifically for each task and class.

To this end, we propose a novel Bayesian meta-learning framework, which we refer to as Bayesian Task-Adaptive Meta-Learning (Bayesian TAML), that learns variables to adaptively balance the effect of meta- and task-specific learning. Specifically, we first obtain set-representations for each task, which are learned to convey useful statistics about the task or class distribution, such as mean, variance, and cardinality (the number of elements in the set), and then learn the distribution of three balancing variables as the function of the set: 1) *task-dependent learning rate multiplier*, which decides how far away to deviate from the meta-knowledge, when performing task-specific learning. Tasks with higher shots could benefit from taking gradient steps afar, while tasks with few shots may need to stay close to the initial parameter. 2) *class-dependent learning rate*, which decides how much information to use from each class, to automatically handle class imbalance where the number of instances per class can largely vary. 3) *task-dependent modulator for initial model parameter*, which modifies the shared initialization for each task, such that each task can decide how much and what to use from the shared initial model parameter and what to ignore based on its set representation. This is especially useful when handling out-of-distribution task, which may need to ignore some of the meta-knowledge.

We validate our model on CIFAR-FS and miniImageNet dataset, as well as a new dataset that consists of heterogeneous datasets, under a scenario where every class in each episode can have *any* number of shots, that leads to task and class imbalance, and where the dataset at meta-test time is different from that of meta-training time. The experimental results show that our Bayesian TAML significantly improves the performance over the existing approaches under these realistic scenarios. Further analysis of each component reveals that the improvement is due to the effectiveness of the balancing terms for handling task and class imbalance, and out-of-distribution tasks.

To summarize, our contribution in this work is threefold:

- We consider a novel problem of meta-learning under a realistic task distribution, where the number of instances across classes and tasks could largely vary, or the unseen task at the meta-test time is largely different from the seen tasks.
- For effective meta-learning with such imbalances, we propose a Bayesian task-adaptive meta-learning (Bayesian TAML) framework that can adaptively adjust the effect of the meta-learner and the task-specific learner, differently for each task and class.
- We validate our model on realistic imbalanced few-shot classification tasks with a varying number of shots per task and class and show that it significantly outperforms existing meta-learning models.

2 Related Work

Meta-learning Meta-learning (Schmidhuber, 1987; Thrun & Pratt, 1998) is an approach to learn a model to generalize over a distribution of task. The approaches in general can be categorized into either memory-based, metric-based, and optimization-based methods. A memory-based approach (Santoro et al., 2016) learns to store correct instance and label into the same memory slot and retrieve it later, in a task-generic manner. Metric-based approaches learn a shared metric

space (Vinyals et al., 2016; Snell et al., 2017). Snell et al. (2017) defines the distance between the instance and the class prototype, such that the instances are closer to their correct prototypes than to others. As for optimization-based meta-learning, MAML Finn et al. (2017) learns a shared initialization parameter that is optimal for any tasks within few gradient steps from the initial parameter. Meta-SGD (Li et al., 2017) improves upon MAML by learning the learning rate differently for each parameter. For effective learning of a meta-learner, meta-learning approaches adopt the episodic training strategy (Vinyals et al., 2016) which trains and evaluates a model over a large number of tasks, which are called meta-training and meta-test phase, respectively. However, existing approaches only consider an artificial scenario which samples the classification of classes with exactly the same number of training instances, both within each episode and across episodes. On the other hand, we consider a more challenging scenario where the number of shots per class and task could vary at each episode, and that the task given at meta-test time could be an out-of-distribution task.

Task-adaptive meta-learning The goal of learning a single meta-learner that works well for all tasks may be overly ambitious and leads to suboptimal performances for each task. Thus recent approaches adopt task-adaptively modified meta-learning models. Oreshkin et al. (2018) proposed to learn the temperature scaling parameter to work with the optimal similarity metric. Qiao et al. (2018) also suggested a model that generates task-specific parameters for the network layers, but it only trains with many-shot classes, and implicitly expects generalization to few-shot cases. Rusu et al. (2018) proposed a network type task-specific parameter producer, and Lee & Choi (2018) proposed to differentiate the network weights into task-shared and task-specific weights. Our model also aims to obtain task-specific parameter for each task, but is rather focused on learning how to balance between the meta-learning and task-/class-specific learning.

Probabilistic meta-learning Recently, a probabilistic version of MAML has been proposed (Finn et al., 2018), where they interpret a task-specific gradient update as a posterior inference process under variational inference framework. Kim et al. (2018) proposed Bayesian MAML with a similar motivation but with a stein variational inference framework and chaser loss. Gordon et al. (2018) proposed a probabilistic meta-learning framework where the paramter for a novel task is rapidly estimated under decision theoretic framework, given a set representation of a task. The motivation behind these works is to represent the inherent uncertainty in few-shot classification tasks. Our model also uses Bayesian modeling, but it focuses on leveraging the uncertainties of the meta-learner and the gradient-direction in order to balance between meta- and task- or class-specific learning.

3 Learning to Balance

We first introduce notations and briefly recap the model-agnostic meta-learning (MAML) by Finn et al. (2017). Suppose a task distribution $p(\tau)$ that randomly generates task τ consisting of a training set $\mathcal{D}^\tau = \{\mathbf{X}^\tau, \mathbf{Y}^\tau\}$ and a test set $\tilde{\mathcal{D}}^\tau = \{\tilde{\mathbf{X}}^\tau, \tilde{\mathbf{Y}}^\tau\}$. Then, the goal of MAML is to meta-learn the initial model parameter $\boldsymbol{\theta}$ as a meta-knowledge to generalize over the task distribution $p(\tau)$, such that we can easily obtain the task-specific predictor $\boldsymbol{\theta}^\tau$ in a single (or a few) gradient step from the initial $\boldsymbol{\theta}$. Toward this goal, MAML optimizes the following gradient-based meta-learning objective:

$$\min_{\boldsymbol{\theta}} \sum_{\tau \sim p(\tau)} \mathcal{L}(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}^\tau); \tilde{\mathcal{D}}^\tau) \tag{1}$$

where α denotes stepsize and $\mathcal{L}$ denotes empirical loss such as negative log-likelihood of observations. Note that by meta-learning the initial point $\boldsymbol{\theta}$, the task-specific predictor $\boldsymbol{\theta}^\tau = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}^\tau)$ can minimize the test loss $\mathcal{L}(\cdot; \tilde{\mathcal{D}}^\tau)$ even with $\mathcal{D}^\tau$ consisting of few samples. We can easily extend the Eq. (1), such that we obtain $\boldsymbol{\theta}^\tau$ with more than one inner-gradient steps from the initial $\boldsymbol{\theta}$.

However, the existing MAML framework has the following limitations that prevent the model from efficiently solving real-world problems involving class/task imbalance and out-of-distribution tasks.

1. **Class imbalance.** MAML does not provide any framework to handle class imbalance within each task. Therefore, classes with large number of training instances (head classes) may dominate the task-specific learning during the inner-gradient steps, yielding low performance on classes with fewer shots (tail classes).
2. **Task imbalance.** The model has a fixed number of inner-gradient steps and stepsize α across all tasks, which prevents the model from adaptively deciding how much to resort to

the meta-knowledge or how much to learn from the given dataset, depending on the number of the training examples per task.

3. **Out-of-distribution tasks.** The model assumes that the initial model parameter $\boldsymbol{\theta}$ will be equally useful for the unseen tasks, but for unseen tasks that are significantly different from the previously seen tasks, the initial parameter may be less useful.

3.1 TASK-ADAPTIVE META-LEARNING (TAML)

As shown in Figure 1 for the concepts, we introduce three balancing variables $\boldsymbol{\omega}^\tau, \boldsymbol{\gamma}^\tau, \mathbf{z}^\tau$ to tackle each problem mentioned above. How to generate these variables will be described in Section 4. Also, see the experimental section for how to generate the realistic tasks with class and task imbalance.

Tackling class imbalance. To handle class imbalance, we vary the learning rate of class-specific gradient for each inner-optimization step. Specifically, for class $c = 1, \dots, C$, we assume a set of class-specific scalars $\boldsymbol{\omega}^\tau = (\omega_1^\tau, \dots, \omega_C^\tau) \in [0,1]^C$ multiplied to each of the class-specific gradients $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_1^\tau), \dots, \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_C^\tau)$, where $\mathcal{D}_c^\tau$ is the set of instances and labels for class c. We expect ω_c^τ to be large for tail-classes to consider them more in the inner-optimization steps. We generate $\boldsymbol{\omega}^\tau$ with Softmax function, and denote the input to the function as $\tilde{\boldsymbol{\omega}}^\tau$.

Tackling task imbalance. To control whether to stay close to the initial parameter or deviate far from it, we introduce task-dependent learning-rate multipliers $\boldsymbol{\gamma}^\tau = (\gamma_1^\tau, \dots, \gamma_L^\tau) \in [0, \infty)^L$, such that for each layer $l = 1, \dots, L$, the learning rate becomes $\gamma_1^\tau \alpha, \gamma_2^\tau \alpha, \dots, \gamma_L^\tau \alpha$. We expect $\boldsymbol{\gamma}^\tau$ to be large for large tasks, such that they rely more on task-specific updates, while small tasks use small $\boldsymbol{\gamma}^\tau$ to benefit from the meta-knowledge. We generate $\boldsymbol{\gamma}^\tau$ with exponential function, and denote the input to the function as $\tilde{\boldsymbol{\gamma}}^\tau$.

Tackling out-of-distribution tasks. Lastly, we introduce $\mathbf{z}^\tau$ that modulates the initial parameter $\boldsymbol{\theta}$ for each task. We expect $\mathbf{z}^\tau$ to learn to relocate the initial $\boldsymbol{\theta}$ to a new appropriate starting point, such that out-of-distribution tasks can deviate from the shared initialization $\boldsymbol{\theta}$ if the current initialization is suboptimal for the given task. Define $\mathbf{z}^\tau = \mathbf{1} + \tilde{\mathbf{z}}^\tau$ for each channel of the network weights and $\mathbf{z}^\tau = \tilde{\mathbf{z}}^\tau$ for the biases. Then, we let $\boldsymbol{\theta}_0 \leftarrow \boldsymbol{\theta} \circ \mathbf{z}^\tau$ for the weights and $\boldsymbol{\theta}_0 \leftarrow \boldsymbol{\theta} + \mathbf{z}^\tau$ for the biases, where $\boldsymbol{\theta}_0$ denotes the new initialization modulated by $\mathbf{z}^\tau$. We define such operation as $\boldsymbol{\theta} * \mathbf{z}^\tau$.

A unified framework. Finally, we assemble all these components together into a single unified framework. With a slight abuse of notations, the update rule can be written as follows:

$$\boldsymbol{\theta}_0 = \boldsymbol{\theta} * \mathbf{z}^\tau, \tag{2}$$

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \boldsymbol{\gamma}^\tau \circ \boldsymbol{\alpha} \circ \sum_{c=1}^{C} \omega_c^\tau \nabla_{\boldsymbol{\theta}_{k-1}} \mathcal{L}(\boldsymbol{\theta}_{k-1}; \mathcal{D}_c^\tau) \quad \text{for } k = 1, \dots, K \tag{3}$$

where $\boldsymbol{\alpha}$ is a multi-dimensional global learning rate vector that is learned (Li et al., 2017), and the multiplication operator $\circ$ is appropriately defined. The last step $\boldsymbol{\theta}_K$ corresponds to the task-specific predictor $\boldsymbol{\theta}^\tau$.

3.2 BAYESIAN TASK-ADAPTIVE META-LEARNING

We next introduce the variational inference framework for the *input* of the three balancing variables, that we previously denote as $\tilde{\boldsymbol{\omega}}^\tau$, $\tilde{\boldsymbol{\gamma}}^\tau$, $\tilde{\mathbf{z}}^\tau$. Bayesian modeling allows to incorporate randomness in the posterior of $\boldsymbol{\phi}$, in the form of the ensemble of diverse task-specific predictors. By doing wo, we can effectively exploit the information in the given dataset $\mathcal{D}^\tau$. It also allows to find more robust latent structure of those balancing variables. We let the three variables share the same inference network pipeline to minimize the computational cost.

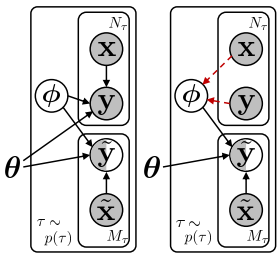


Figure 2: Graphical model. (a) Generative process. (b) Inference.

Firstly, define $\mathbf{X}^\tau = \{\mathbf{x}_n^\tau\}_{n=1}^{N_\tau}$ and $\mathbf{Y}^\tau = \{\mathbf{y}_n^\tau\}_{n=1}^{N_\tau}$ for training, and $\tilde{\mathbf{X}}^\tau = \{\tilde{\mathbf{x}}_m^\tau\}_{m=1}^{M_\tau}$ and $\tilde{\mathbf{Y}}^\tau = \{\tilde{\mathbf{y}}_m^\tau\}_{m=1}^{M_\tau}$ for test. Let $\boldsymbol{\phi}^\tau$ denote the collection of $\tilde{\boldsymbol{\omega}}^\tau$, $\tilde{\boldsymbol{\gamma}}^\tau$ and $\tilde{\mathbf{z}}^\tau$ for uncluttered notation. Then, inspired by Gordon et al. (2018) and Finn et al. (2018), we define the generative process for meta-learning framework as follows for each task τ (See Figure 2):

$$p(\mathbf{Y}^\tau, \tilde{\mathbf{Y}}^\tau, \boldsymbol{\phi}^\tau | \mathbf{X}^\tau, \tilde{\mathbf{X}}^\tau; \boldsymbol{\theta}) = p(\boldsymbol{\phi}^\tau) \prod_{n=1}^{N_\tau} p(\mathbf{y}_n^\tau | \mathbf{x}_n^\tau, \boldsymbol{\phi}^\tau; \boldsymbol{\theta}) \prod_{m=1}^{M_\tau} p(\tilde{\mathbf{y}}_m^\tau | \tilde{\mathbf{x}}_m^\tau, \boldsymbol{\phi}^\tau; \boldsymbol{\theta}) \tag{4}$$

for the complete data likelihood. Note that the deterministic $\boldsymbol{\theta}$ is shared across all the tasks.

4 VARIATIONAL INFERENCE

The goal of learning for each task τ is to maximize the log-likelihood of the joint dataset $\tilde{\mathcal{D}}^\tau$ and $\mathcal{D}^\tau$: $\log p(\tilde{\mathbf{Y}}^\tau, \mathbf{Y}^\tau | \tilde{\mathbf{X}}^\tau, \mathbf{X}^\tau; \boldsymbol{\theta})$. However, solving it involves the true posterior $p(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau, \tilde{\mathcal{D}}^\tau)$, which is intractable. Thus, we resort to amortized variational inference with a tractable form of approximate posterior $q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau, \tilde{\mathcal{D}}^\tau; \boldsymbol{\psi})$ parameterized by $\boldsymbol{\psi}$. Further, similarly to Ravi & Beatson (2018), we drop the dependency on the test dataset $\tilde{\mathcal{D}}^\tau$ for the approximate posterior, in order to make the two different pipelines consistent; one for meta-training where we observe the whole test dataset, and the other for meta-testing where the test labels are unknown. The form of our approximate posterior is now $q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})$. It greatly simplifies the inference framework, while ensuring to have a valid lower bound of the log evidence. Also, considering that performing the inner-gradient steps with the training dataset $\mathcal{D}^\tau$ automatically maximizes the training log-likelihood in MAML framework, we slightly modify the objective so that the expected log-likelihood term only involves the test examples with the appropriate scaling factor. The resultant form of the approximated lower bound that suits for our meta-learning purpose is as follows:

$$L_{\boldsymbol{\theta},\boldsymbol{\psi}}^\tau = \frac{N_\tau + M_\tau}{M_\tau} \sum_{m=1}^{M_\tau} \mathbb{E}_{q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})} \Big[\log p(\tilde{\mathbf{y}}_m^\tau | \tilde{\mathbf{x}}_m^\tau, \boldsymbol{\phi}^\tau; \boldsymbol{\theta}) \Big] - \mathrm{KL}[q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) \| p(\boldsymbol{\phi}^\tau)]. \tag{5}$$

We assume $q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})$ fully factorizes for each variable and also for each dimension as well:

$$q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) = \prod_c q(\tilde{\omega}_c^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) \prod_l q(\tilde{\gamma}_l^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) \prod_i q(\tilde{z}_i^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) \tag{6}$$

where we assume that each single dimension of $q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})$ follows univariate gaussian having trainable mean and variance. We also let each dimension of prior $p(\boldsymbol{\phi}^\tau)$ factorize into $\mathcal{N}(0, 1)$, such that the KL-divergence can have the especially simple closed form (Kingma & Welling, 2014).

The final form of the meta-training minimization objective with Monte-Carlo (MC) approximation for the expection in (5) is as follows:

$$\min_{\boldsymbol{\theta},\boldsymbol{\psi}} \frac{1}{M_\tau} \sum_{m=1}^{M_\tau} \frac{1}{S} \sum_{s=1}^{S} -\log p(\tilde{\mathbf{y}}_m^\tau | \tilde{\mathbf{x}}_m^\tau, \boldsymbol{\phi}_s^\tau; \boldsymbol{\theta}) + \frac{1}{N_\tau + M_\tau} \mathrm{KL}[q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}) \| p(\boldsymbol{\phi}^\tau)]. \tag{7}$$

where $\boldsymbol{\phi}_s^\tau \sim q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})$. We implicitly assume the reparameterization trick for $\boldsymbol{\phi}^\tau$ to obtain stable and unbiased gradient estimate w.r.t. $\boldsymbol{\psi}$ (Kingma & Welling, 2014). We set the MC sample size to $S = 1$ for meta-training for computational efficiency. When meta-testing, we perform MC approximation:

$$p(\tilde{\mathbf{y}}_*^\tau | \tilde{\mathbf{x}}_*^\tau; \boldsymbol{\theta}) = \mathbb{E}_q[p(\tilde{\mathbf{y}}_*^\tau | \tilde{\mathbf{x}}_*^\tau, \boldsymbol{\phi}^\tau; \boldsymbol{\theta})] \approx \frac{1}{S} \sum_{s=1}^{S} p(\tilde{\mathbf{y}}_*^\tau | \tilde{\mathbf{x}}_*^\tau, \boldsymbol{\phi}_s^\tau; \boldsymbol{\theta}), \quad \boldsymbol{\phi}_s^\tau \sim q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi}). \tag{8}$$

or we may naively approximate by taking the expectation inside for computational efficiency:

$$\mathbb{E}_q[p(\tilde{\mathbf{y}}_*^\tau | \tilde{\mathbf{x}}_*^\tau, \boldsymbol{\phi}^\tau; \boldsymbol{\theta})] \approx p(\tilde{\mathbf{y}}_*^\tau | \tilde{\mathbf{x}}_*^\tau, \mathbb{E}_q[\boldsymbol{\phi}^\tau]; \boldsymbol{\theta}). \tag{9}$$

In the experimental section, we show that Eq. (8) largely outperforms Eq. (9).

4.1 DATASET ENCODING

The main challenge in modeling our variational distribution $q(\boldsymbol{\phi}^\tau | \mathcal{D}^\tau; \boldsymbol{\psi})$ is to decide how to refine the training dataset $\mathcal{D}^\tau$ into informative representation, which is not trivial. This inference network should capture all the necessary statistical information to recognize any imbalances in the dataset $\mathcal{D}^\tau$. Mean-pooling (Edwards & Storkey, 2016) or sum-pooling (Zaheer et al., 2017) is frequently used

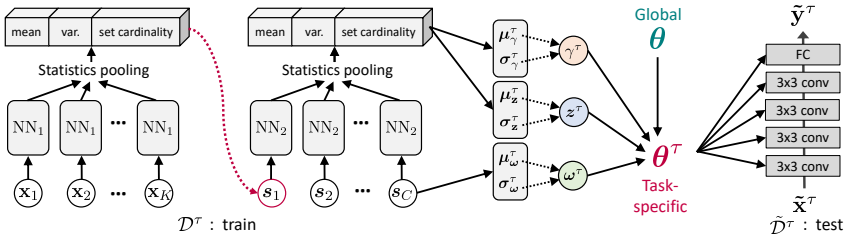


Figure 3: **Inference Network.** The proposed dataset encoder captures the instance-wise and class-wise statistics hierarchically, from which we infer three different balancing variables.

as a practical set-encoder, where each instance in the set is transformed by the shared nonlinearity, and then averaged or summed together to generate a single vector summarizing the set, followed by an additional nonlinearity. However, for the classification dataset $\mathcal{D}^\tau$ which is the *set of (class) sets*, those non-hierarchical pooling methods will perform poorly as they ignore the label information. Therefore, we use a two-layer hierarchical set encoder which first encodes each class as a set of samples and then encodes the set as the set of classes (see the Appendix B for the justification).

However, there is an additional limitation of mean pooling when using it to describe a task: it does not recognize the number of elements in the set [1]. This could be a critical limitation in recognizing the imbalances in the given task. Therefore we explicitly input the number of elements into the set encoder. Yet, the set cardinality alone is insufficient in capturing the distribution of the dataset. Suppose that we have a set containing replications of a single instance. Then, although the set has only limited information, the set encoding cannot recognize it and will overestimate the information. To prevent this, we encode the variance of the set as well.

Based on this intuition, we define the set encoder $\mathrm{StatisticsPooling}(\cdot)$ that generates the concatenation of the set statistics such as mean, variance, and cardinality. We use this encoder to first encode each class, and then the whole set of classes as follows:

$$\mathbf{v}^\tau = \mathrm{StatisticsPooling}\left(\{\mathrm{NN}_2\left(\mathbf{s}_c\right)\}_{c=1}^{C}\right), \quad \mathbf{s}_c = \mathrm{StatisticsPooling}\left(\{\mathrm{NN}_1(\mathbf{x})\}_{\mathbf{x}\in\mathbf{X}_c^\tau}\right)$$

for classes $c = 1, \dots, C$. $\mathbf{X}_c^\tau$ is the collection of class c examples in task τ. NN_1 and NN_2 are some neural networks parameterized by $\boldsymbol{\psi}$. The vector $\mathbf{v}^\tau$ finally summarizes the entire dataset $\mathcal{D}^\tau$ for few-shot classification. Note that the class-specific balancing variables $\tilde{\omega}_1^\tau, \dots, \tilde{\omega}_C^\tau$ are generated from $\mathbf{s}_1, \dots, \mathbf{s}_C$, and other task-specific balancing variables $\tilde{\boldsymbol{\gamma}}^\tau$ and $\tilde{\mathbf{z}}^\tau$ are generated from $\mathbf{v}^\tau$ with an additional affine transformation (See Figure 3).

5 Experiments

We next validate our method on multiple benchmark datasets with more realistic task distribution.

Datasets We validate our method on the following benchmark datasets. **CIFAR-FS:** This dataset (Bertinetto et al., 2019) is a variant of CIFAR-100 dataset that consists of 100 general object categories. For each class, there are 600 examples of 32×32 pixels. We split the dataset into $64/16/20$ classes for training/validation/test. **SVHN:** This dataset (Netzer et al., 2011) is frequently used as an out-of-distribution dataset of CIFAR dataset. It consists of $26,032$ color images of 32×32 pixels, from 10 digits classes. **miniImageNet:** This dataset (Vinyals et al., 2016) is a subset of the ImageNet dataset (Russakovsky et al., 2015). It consists of total 100 classes, each of which has 600 images resized into 84×84. We split the dataset into $64/16/20$ for training/validation/test. **CUB:** This dataset contains $11,788$ bird images from 200 classes. We resize the images into 84×84. We split the dataset into $100/50/50$ classes for training/validation/test. Since this dataset is fine-grained, we regard it as an out-of-distribution dataset of the coarse-grained miniImageNet dataset.

Realistic task distribution For the realistic task distribution $p(\tau)$, we first randomly sample $C = 5$ classes. Then with half probability we sample uniform-random number of instances in range $[1, 50]$ for each class to simulate *class imbalance*, and for the other half probability we also uniformly sample the class size from $[1, 50]$, but let all the classes share the same size to simulate *task imbalance*. The number of test examples (queries) is set to 15 for each class.

[1] We also found out that sum-pooling is very unstable in encoding set of variable size.

Table 1: **Any-shot classification results.** For each model, we run 3 independent trials and jointly test them over total $9,000 = 3 \times 3,000$ episodes. We report mean accuracies and 95% confidence intervals.

Meta-training	CIFAR-FS		miniImageNet	
Meta-test	CIFAR-FS	SVHN	miniImageNet	CUB
MAML (Finn et al., 2017)	56.89 ± 0.22	27.09 ± 0.12	60.50 ± 0.25	57.42 ± 0.25
ABML (Ravi & Beatson, 2018)	54.41 ± 0.23	24.47 ± 0.10	46.22 ± 0.20	46.42 ± 0.22
MT-NET (Lee & Choi, 2018)	56.86 ± 0.23	27.46 ± 0.12	63.40 ± 0.23	59.42 ± 0.23
Meta-SGD (Li et al., 2017)	72.71 ± 0.21	46.45 ± 0.24	69.95 ± 0.20	65.94 ± 0.22
Prototypical Networks (Snell et al., 2017)	73.24 ± 0.20	42.91 ± 0.18	69.11 ± 0.19	60.80 ± 0.19
Bayesian TAML	**75.15 ± 0.20**	**51.87 ± 0.23**	**71.46 ± 0.19**	**71.71 ± 0.21**

Table 2: **Multi-dataset classification results.**

Meta-training	Aircraft, Quickdraw, and VGG-Flower				
Meta-test	Aircraft	QuickDraw	VGG-Flower	Traffic Signs	Fashion-MNIST
MAML	31.29 ± 0.11	41.40 ± 0.16	38.75 ± 0.12	25.32 ± 0.11	38.47 ± 0.12
MT-NET	29.49 ± 0.13	41.24 ± 0.15	39.47 ± 0.12	23.62 ± 0.13	39.16 ± 0.16
Meta-SGD	49.71 ± 0.17	70.26 ± 0.16	59.41 ± 0.27	52.07 ± 0.35	62.71 ± 0.25
Prototypical Networks	50.63 ± 0.16	**72.31 ± 0.17**	65.52 ± 0.15	49.93 ± 0.18	64.26 ± 0.13
Bayesian TAML	**54.43 ± 0.16**	72.03 ± 0.16	**67.72 ± 0.16**	**64.81 ± 0.21**	**68.94 ± 0.13**

Experimental setup For all our experiments, we use 4-block convolutional neural networks with 32 channels, following Finn et al. (2017). For all the baselines and our model, we perform early-stopping based on the meta-validation set performance. We set the number of inner-gradient steps to 5 for training and 10 for testing, for all the gradient-based models. See the Appendix A for more details about the experimental setup.[2]

5.1 Main results

Any-shot classification Table 1 shows the results of the realistic any-shot classification. Firstly, for the gradient-based baselines, we see that MAML, ABML, and MT-NET fail in this realistic any-shot setting. However, Meta-SGD largely outperforms them, demonstrating the importance of meta-learning the element-wise learning rates to solve realistic any-shot classification problems. Prototypical Networks perform relatively well on in-distribution (ID) tasks but underperform Meta-SGD on out-of-distribution (OOD) tasks. This is because metric-based models like Prototypical Networks do not the take gradient steps for OOD tasks, such that the information in the dataset cannot be effectively incorporated into each task-specific predictor. Lastly, we see that our Bayesian TAML significantly outperforms all the baselines, on both ID and OOD datasets. Bayesian TAML especially performs well on OOD datasets. This is because when the given task is much different from the meta-knowledge, the model needs to be able to deviate far from the meta-knowledge, but it is not trivial for the other baselines.

Multi-dataset classification We further test our model under a more challenging setting where tasks could come from a highly heterogeneous dataset (Triantafillou et al., 2019). We meta-train with Aircrafts, Quickdraw and VGG-Flower dataset and meta-test with the five datasets including Traffic Signs and Fashion-MNIST. We see from Table 2 that Bayesian TAML also largely outperforms in this setting on most of the datasets. While prototypical Networks performs relatively well for this multi-dataset classification due to its flexibility in learning the metric space, our Bayesian TAML effectively combines the advantage of both metric-based and gradient-based approaches, by introducing the balancing variables that can flexibly modulate the gradient steps. See the Appendix for the detailed experimental setup.

5.2 Effectiveness of the Balancing Variables

We now validate the effectiveness of the three balancing variables, by adding in each of them to Meta-SGD one at a time to clearly show their effectiveness.

[2]Code is available at `https://github.com/haebeom-lee/l2b`

$\mathbf{z}^\tau$ for handling distributional shift. $\mathbf{z}^\tau$ modulates the initial model parameter $\boldsymbol{\theta}$, and decides what and how much to use from the meta-knowledge $\boldsymbol{\theta}$ based on the relatedness between the initial $\boldsymbol{\theta}$ and the task at hand. We validate the effectiveness of this balancing variable by examining the performance of the baseline methods and our models using datasets (SVHN, CUB) that are highly heterogeneous from the meta-training datasets (CIFAR-FS, miniImageNet). The results in Table 3 show that our model with only $\mathbf{z}^\tau$ can effectively handle these out-of-distribution (OOD) tasks.

We further observe that $\mathbf{z}^\tau$ relocates the initial $\boldsymbol{\theta}$ far from the initial parameter for these OOD tasks given at meta-test time (Figure 4), with larger displacements for highly heterogeneous tasks (Figure 4, right). This allows the model to either stick to, or deviate from the meta-knowledge based on the similarity between the tasks given at the meta-training and meta-test time.

Moreover, the last two rows of Table 3 show that the MC approximation in Eq. (8) largely outperforms the naive approximation in Eq. (9). This is because MC sampling allows to have larger variance for the distribution of $\boldsymbol{\theta} * \mathbf{z}^\tau$ for OOD tasks, to express uncertainty on them. Basically, as the tasks given at meta-test time becomes more different from the tasks seen during meta-training, the more random the initialization will become.

Meta-training	CIFAR-FS	miniImageNet
Meta-test	SVHN	CUB
MAML	27.09$\pm$0.12	57.42$\pm$0.25
Prototypical Networks	42.91$\pm$0.18	60.80$\pm$0.19
Meta-SGD	46.45$\pm$0.24	65.94$\pm$0.22
B. $\mathbf{z}$-TAML (naive approx.)	47.80$\pm$0.20	67.90$\pm$0.21
B. $\mathbf{z}$-TAML (MC approx.)	**52.29**$\pm$**0.24**	**69.11**$\pm$**0.22**

Table 3: Ablation study on distributional shift.

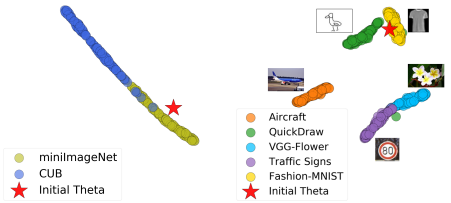
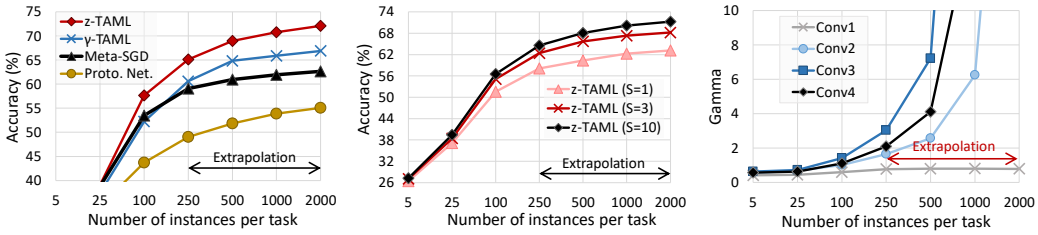


Figure 4: T-SNE visualization of $\boldsymbol{\theta}$ and $\boldsymbol{\theta} * \mathbf{z}^\tau$.

$\boldsymbol{\gamma}^\tau$ and $\mathbf{z}^\tau$ for handling task imbalance. We then examine how the two balancing variables, $\boldsymbol{\gamma}^\tau$ and $\mathbf{z}^\tau$ handle inter-task imbalance where tasks used for meta-learning have largely different number of instances. Figure 5(a) shows the performance of the baseline models and our models with each of the balancing variables, when the number of instances per task varies from 5 to 2000. We observe that our Bayesian $\mathbf{z}$-TAML and Bayesian $\boldsymbol{\gamma}$-TAML largely outperform Meta-SGD and Prototypical Networks, especially by large degree when the number of instances per task is large. Figure 5(b) shows that the effectiveness of $\mathbf{z}^\tau$ largely depends on the number of MC samples used in Eq. (8), demonstrating the importance of incorporating uncertainties in random initializations for handling task imbalance. We further observe from Figure 5(c) that the task-dependent learning rate multiplier $\boldsymbol{\gamma}^\tau$ exponentially grows as the number of instances per task increases. This agrees with our intuition that larger tasks should take larger inner-gradient steps, to learn more from the given task rather than resorting to meta-knowledge.



(a) Classification performance (b) MC sample size vs. Acc. (c) $\boldsymbol{\gamma}^\tau$

Figure 5: Ablation study on task imbalance (Meta-training: CIFAR-FS, Meta-test: SVHN).

$\boldsymbol{\omega}^\tau$ for handling class imbalance. $\boldsymbol{\omega}^\tau$ rescales the class-specific gradients to handle class imbalance where the number of instances per class (i.e. shot) largely varies. Table 4 shows the results under the varying degree of class imbalance across the task distribution. We observe that Bayesian $\boldsymbol{\omega}$-TAML outperforms Meta-SGD, especially by larger degree under higher class imbalance. Notably, Bayesian $\boldsymbol{\omega}$-TAML outperforms a heuristic balancing method which divides each class-specific gradient by the cardinality of each class set (Meta-SGD with $1/N$). The accuracy improvements over Meta-SGD in Figure 6 demonstrate that this heuristic balancing method overly emphasizes the tail classes with few training instances, thereby deteriorating the performance on classes with sufficiently large number of instances. On the other hand, our Bayesian $\boldsymbol{\omega}$-TAML learns the appropriate balancing variables which allow to obtain large performance gains on all classes.

Meta-training / Meta-test CIFAR-FS / CIFAR-FS	Number of instances per class 10	5 - 25	1 - 50
MAML	60.68±0.20	56.11±0.20	52.85±0.21
Meta-SGD	73.25±0.19	72.68±0.19	71.61±0.19
Meta-SGD with $1/N$	71.33±0.19	72.43±0.19	72.23±0.19
Prototypical Networks	72.88±0.19	**73.27±0.19**	72.57±0.19
Bayesian ω-TAML	**73.44±0.18**	73.20±0.18	**72.86±0.19**

Table 4: Ablation study on class imbalance.

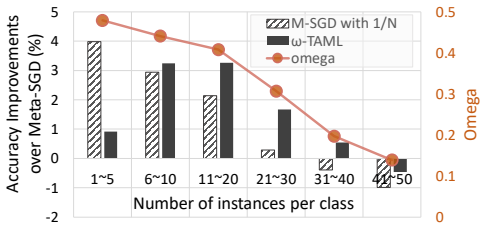


Figure 6: ω^τ and accuracy gains over M-SGD.

5.3 More ablation studies

Effectiveness of Bayesian modeling We further demonstrate the effectiveness of Bayesian modeling by comparing it with the deterministic version of our model in Eq. (9) (Deterministic TAML), where three balancing variables are no longer stochastic. We apply ℓ_2 regularization on the variables with coefficients that are equivalent to the KL-divergence in Eq. (5). The results in Table 5 show that the Bayesian TAML significantly outperforms its deterministic counterpart, especially with larger gap on the OOD task (SVHN). Figure 7 further shows that the balancing variable γ^τ for handling task imbalance, more sensitively reacts on Bayesian TAML than on Deterministic TAML, which suggests that Bayesian modeling amplifies the effect of the balancing variables.

Meta-training: CIFAR-FS	CIFAR-FS	SVHN
MAML	56.89±0.22	27.09±0.12
Meta-SGD	72.71±0.21	46.45±0.24
Prototypical Networks	73.24±0.20	42.91±0.18
Deterministic TAML	73.82±0.21	46.78±0.24
Bayesian TAML (Naive approx.)	73.52±0.20	47.15±0.20
Bayesian TAML (MC approx.)	**75.15±0.20**	**51.87±0.23**

Table 5: Classification performance of Bayesian and Deterministic TAML on seen and unseen dataset.

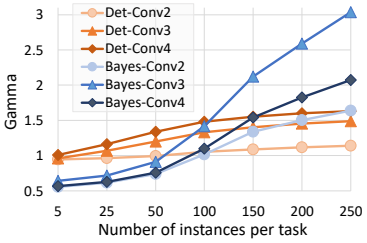


Figure 7: γ^τ vs. Bayesianness.

Dataset encoding Finally, we perform an ablation study of the set encoding methdos in Table 6 to validate the effectiveness of the proposed task encoding scheme. We see that our hierarchical dataset encoding, equipped with set cardinality and diagonal sample covariance, is far more effective than simple mean-pooling method (Zaheer et al., 2017; Edwards & Storkey, 2016).

Meta-training / Meta-test CIFAR-FS / CIFAR-FS	Hierarchical encoding ×	√
Mean	73.84±0.21	73.69±0.21
Mean + N	73.17±0.21	74.88±0.20
Mean + Var. + N	73.93±0.21	**75.15±0.20**

Table 6: Ablation study on dataset encoding schemes. N: Set cardinality.

6 Conclusion

We propose Bayesian TAML that learns to balance the effect of meta-learning and task-adaptive learning, to consider meta-learning under a more realistic task distribution where each task and class can have varying number of instances. Specifically, we encode the dataset for each task into hierarchical representations, and use it to modulate the original parameter, learning rate, and the class-specific gradients. We use a Bayesian framework to infer the posterior of these balancing variables, and propose an effective variational inference framework to solve for them. Our model outperforms existing meta-learning methods when validated on imbalanced few-shot classification tasks. Further analysis of each balancing variable shows that each variable effectively handles task imbalance, class imbalance, and out-of-distribution tasks respectively. We believe that our work makes a meaningful step toward application of meta-learning to real-world problems.

Acknowledgements This work was supported by Google AI Focused Research Award, Samsung Research Funding Center of Samsung Electronics (No. SRFC-IT1502-51), the Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korean Government MSIT (NRF-2018R1A5A1059921), and the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, and Artificial Intelligence Graduate School Program (KAIST))

REFERENCES

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *Proceedings of 6th International Conference on Learning Representations*. ICLR, 2019.

Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pp. 9516–9527, 2018.

Jonathan Gordon, John Bronskill, Matthias Bauer, Sebastian Nowozin, and Richard E Turner. Meta-learning probabilistic inference for prediction. *arXiv preprint arXiv:1805.09921*, 2018.

Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *The 2013 international joint conference on neural networks (IJCNN)*, pp. 1–8. IEEE, 2013.

Takashi Kawashima Jongmin Kim Jonas Jongejan, Henry Rowley and Nick Fox-Gieg. The quick, draw! – a.i. experiment. 2016. URL `http://quickdraw.withgoogle.com`.

Taesup Kim, Jaesik Yoon, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. *arXiv preprint arXiv:1806.03836*, 2018.

Diederik P. Kingma and Max Welling. Auto encoding variational bayes. In *2nd International Conference on Learning Representations*. 2014.

Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *International Conference on Machine Learning*, pp. 2933–2942, 2018.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

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

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pp. 722–729. IEEE, 2008.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 719–729. 2018.

Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L Yuille. Few-shot image recognition by predicting parameters from activations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7229–7238, 2018.

Sachin Ravi and Alex Beatson. Amortized bayesian meta-learning. 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International conference on machine learning*, 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

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

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 4077–4087, 2017.

Sebastian Thrun and Lorien Pratt (eds.). *Learning to Learn*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. ISBN 0-7923-8047-9.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *arXiv preprint arXiv:1903.03096*, 2019.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

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

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 3391–3401. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/6931-deep-sets.pdf`.

A EXPERIMENTAL SETUP

A.1 BASELINES

We describe the baseline models and our task-adaptive learning to balance model.

1) MAML. The Model-Agnostic Meta-Learning (MAML) model by (Finn et al., 2017), which aims to learn the global initial model parameter, from which we can take a few gradient steps to get task-specific predictors.

2) ABML. This model interprets MAML under hierarchical Bayesian framework, but they propose to share and amortize the inference rules across both global initial parameters as well as the task-specific parameters.

3) MT-NET. A gradient-based meta-learning model proposed by Lee & Choi (2018). The model obtains a task-specific parameter only w.r.t. a subset of the whole dimension (M-Net), followed by a linear transformation to learn a metric space (T-Net)

4) Meta-SGD. A base MAML with the learnable learning-rate vector (without any restriction on sign) element-wisely multiplied to each step inner-gradient (Li et al., 2017).

5) Prototypical Networks. A metric-based few-shot classification model proposed by (Snell et al., 2017). The model learns the metric space based on Euclidean distance between class prototypes and query embeddings.

6) Bayesian TAML. Our model to balance model that can adaptively balance between meta- and task-specific learners for each task and class.

A.2 ANY-SHOT CLASSIFICATION.

We describe more detailed settings for any-shot classification.

CIFAR-FS and SVHN: We train all models for $50K$ steps with meta-batch size set to 4. The learning rate is set to 0.001 for all the baselines and our models. We set the inner learning rate to 0.01 for MAML, MT-NET, and ABML, which we empirically verified to perform better than other values.

miniImageNet and CUB: For all models, the number of meta-training iterations is set to $80K$ and we set the meta-batch size to 1. The learning rate is set to 0.0001 for all the baselines and our models. Inner learning rate is set to 0.01 for MAML, MT-NET and ABML.

A.3 MULTI-DATASET EXPERIMENT

For multi-dataset experiment, we construct a subset of the whole collection of the Meta-Dataset (Triantafillou et al., 2019). We resize the images into to 32×32 pixels. We generate each episode with the maximum number of training instances for each class to 50. For each task, We uniformly select one dataset among Aircrafts, Quickdraw and VGG-Flower and randomly sample 10 classes from the dataset for every task.

Aircraft: We split this dataset (Maji et al., 2013) into 70/15/15 for training/validation/test with 100 examples for each class.

Quick Draw: We split this dataset (Jonas Jongejan & Fox-Gieg, 2016) into 241/52/52 for training/validation/test with randomly sampled 200 examples for each class.

VGG Flower: This dataset (Nilsback & Zisserman, 2008) contains 40 between 258 images for each class and we split this dataset into 71/16/15 for training/validation/test. In meta-test phase, we test our model on additional out-of-distribution datasets, Traffic Signs and Fashion-MNIST including three in-distribution datasets.

Traffic Signs: This dataset (Houben et al., 2013) has only test dataset with 43 classes. Each class has 900 examples.

Fashion-MNIST: We use test dataset of Fashion-MNIST (Xiao et al., 2017) in meta-test phase. This dataset has 10 classes with 1000 examples.

We keep the number of classes for each task, meta-batch size, meta-training iterations, and learning rate as 10, 3, $60K$, and 0.001, respectively for all models. We set the inner learning rate to 0.01.

A.4 INFERENCE NETWORK ARCHITECTURE

We describe the network architecture of the inference network that takes a classification dataset as an input and generates three balancing variables as output.

Shared encoder $\mathrm{NN}_1 : \mathbf{X}_c^\tau \rightarrow \mathbf{s}_c$

3 × 3 Conv2d with 10-dim with ReLU
2 × 2 Max pool
3 × 3 Conv2d with 10-dim with ReLU
2 × 2 Max pool fc layer with 64-dim
Statistics Pooling across element representations of class c
fc layer with the 4-dim across the statistics with ReLU

Shared encoder $\mathrm{NN}_2 : \mathbf{s}_1, \ldots, \mathbf{s}_C \rightarrow \mathbf{v}^\tau$

fc layer with 128-dim with ReLU
fc layer with 32-dim
Statistics Pooling across class representations
fc layer with the 4-dim across the statistics with ReLU

Generate ω_c^τ: $\mathbf{s}_c^\tau \rightarrow \mu_{\omega_c}^\tau, \sigma_{\omega_c}^\tau$

fc layer with 64-dim and ReLU
fc layer to generate $\mu_{\omega_c}^\tau$ and *fc* layer to generate $\sigma_{\omega_c}^\tau$ as scalar values (1-dim)

Generate γ^τ: $\mathbf{v}^\tau \rightarrow \boldsymbol{\mu}_\gamma^\tau, \boldsymbol{\sigma}_\gamma^\tau$

fc layer with 64-dim and ReLU
fc layer to generate $\boldsymbol{\mu}_\gamma^\tau$ and *fc* layer to generate $\boldsymbol{\sigma}_\gamma^\tau$ per layer (4-dim)

Generate $\mathbf{z}^\tau$: $\mathbf{v}^\tau \rightarrow \boldsymbol{\mu}_\mathbf{z}^\tau, \boldsymbol{\sigma}_\mathbf{z}^\tau$

fc layer with 64-dim and ReLU
fc layer to generate $\boldsymbol{\mu}_\mathbf{z}^\tau$ and *fc* layer to generate $\boldsymbol{\sigma}_\mathbf{z}^\tau$ per output channel (256-dim)

B JUSTIFICATION FOR SET-OF-SETS STRUCTURE.

Based on the previous justification of DeepSets (Zaheer et al., 2017), we can easily justify the Set-of-Sets structure proposed in the main paper as well, in terms of the two-level permutation invariance properties required for any classification dataset. The main theorem of DeepSets is:

Theorem 1. *A function f operating on a set $\mathbf{X} \in \mathcal{X}$ is a valid set function (i.e. permutation invariant), iff it can be decomposed as $f(\mathbf{X}) = \rho_2(\sum_{\mathbf{x} \in \mathbf{X}} \rho_1(\mathbf{x}))$, where ρ_1 and ρ_2 are appropriate nonfcities.*

See (Zaheer et al., 2017) for the proof. Here we apply the same argument twice as follows.

1. A function f operating on a set of representations $\{\mathbf{s}_1, \ldots, \mathbf{s}_C\}$ (we assume each $\mathbf{s}_c$ is an output from a shared function g) is a valid set function (i.e. permutation invariant w.r.t. the order of $\{\mathbf{s}_1, \ldots, \mathbf{s}_C\}$), *iff* it can be decomposed as $f(\{\mathbf{s}_1, \ldots, \mathbf{s}_C\}) = \rho_2(\sum_{c=1}^{C} \rho_1(\mathbf{s}_c))$ with appropriate nonfcities ρ_1 and ρ_2.

2. A function g operating on a set of examples $\{\mathbf{x}_{c,1}, \ldots, \mathbf{x}_{c,N}\}$ is a valid set function (i.e. permutation invariant w.r.t. the order of $\{\mathbf{x}_{c,1}, \ldots, \mathbf{x}_{c,N}\}$) *iff* it can be decomposed as $g(\{\mathbf{x}_{c,1}, \ldots, \mathbf{x}_{c,N}\}) = \rho_4(\sum_{i=1}^{N} \rho_3(\mathbf{x}_{c,i}))$ with appropriate nonfcities ρ_3 and ρ_4.

Inserting $\mathbf{s}_c = g(\{\mathbf{x}_{c,1}, \ldots, \mathbf{x}_{c,N}\})$ into the expression of f, we arrive at the following valid composite function operating on a set of sets:

$$f\left(\{g\left(\{\mathbf{x}_{c,1}, \ldots, \mathbf{x}_{c,N}\}\right)\}_{c=1}^{C}\right) = \rho_2\left(\sum_{c=1}^{C} \rho_1\left(\rho_4\left(\sum_{i=1}^{N} \rho_3\left(\mathbf{x}_{c,i}\right)\right)\right)\right) \tag{10}$$

Let F denote the composite of f and (multiple) g and let NN_2 denote the composite of ρ_1 and ρ_4. Further define $\mathrm{NN}_1 := \rho_3$ and $\mathrm{NN}_3 := \rho_2$. Then, we have

$$F\Big(\big\{\{\mathbf{x}_{1,1}, \ldots, \mathbf{x}_{1,N}\}, \ldots, \{\mathbf{x}_{C,1}, \ldots, \mathbf{x}_{C,N}\}\big\}\Big) = \mathrm{NN}_3\left(\sum_{c=1}^{C} \mathrm{NN}_2\left(\sum_{i=1}^{N} \mathrm{NN}_1\left(\mathbf{x}_{c,i}\right)\right)\right) \tag{11}$$

where C is the number of classes and N is the number of examples per class. See Section A.4 for the actual encoder structure.