EMPIRICAL BAYES TRANSDUCTIVE META-LEARNING WITH SYNTHETIC GRADIENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a meta-learning approach that learns from multiple tasks in a transductive setting, by leveraging unlabeled information in the query set to learn a more powerful meta-model. To develop our framework we revisit the empirical Bayes formulation for multi-task learning. The evidence lower bound of the marginal log-likelihood of empirical Bayes decomposes as a sum of local KL divergences between the variational posterior and the true posterior of each task. We derive a novel amortized variational inference that couples all the variational posteriors into a meta-model, which consists of a synthetic gradient network and an initialization network. The combination of local KL divergences and synthetic gradient network allows for backpropagating information from unlabeled data, thereby enabling transduction. Our results on the Mini-ImageNet and CIFAR-FS benchmarks for episodic few-shot classification outperform previous state-of-the-art methods.

1 INTRODUCTION

While supervised learning of deep neural networks can achieve or even surpass human-level performance (He et al., 2015; Devlin et al., 2018), they can hardly extrapolate the learned knowledge beyond the domain where the supervision is provided. The problem of solving rapidly a new task after learning several other similar tasks is called meta-learning (Schmidhuber, 1987; Bengio et al., 1991; Thrun & Pratt, 1998); typically, the data is presented in a two-level hierarchy such that each data point at the higher level is itself a dataset associated with a task, and the goal is to learn a meta-model that generalizes across tasks. In this paper, we focus on few-shot learning (Vinyals et al., 2016), an instance of meta-learning problems, where a task \( t \), in meta-testing, consists of an unlabeled set \( x_t := \{ x_{t,i} \}_{i=1}^{n_t} \) and a labeled set (aka support set) \( d_t := \{ (x_{t,i}, y_{t,i}) \}_{i=1}^{n_t} \), and the goal is to predict the corresponding labels, namely \( y_t = \{ y_{t,i} \}_{i=1}^{n_t} \), for the unlabeled set. In meta-training, \( y_t \) is provided as ground truth. The set \( d_t := (x_t, y_t) \) is sometimes referred to as query set.

A particular important distinction to make is whether each task is solved in a transductive or inductive manner. The inductive setting is what was originally proposed by Vinyals et al. (2016): we use \( d_t \) to train a model and test it on \( x_t \) (one example at a time). Transduction, however, has the advantage of being able to see all points in \( x_t \) before making predictions. We argue that the transductive setting is more relevant to the problem since, as in semi-supervised learning, an inductive learner can always be built from a transductive one (Chapelle et al., 2006). In fact, Nichol et al. (2018) notice that most of the existing meta-learning methods follow the transductive setting unintentionally since they use \( x_t \) implicitly via the batch normalization (Ioffe & Szegedy, 2015).

Due to the hierarchical structure of the data, it is natural to formulate meta-learning as an instance of hierarchical Bayes (HB) (Good, 1980; Berger, 1985), or alternatively, empirical Bayes (EB) (Robbins, 1985; Kucukelbir & Blei, 2014). The difference is that the latter restricts the learning of meta-parameters to point estimates. In this paper, we focus on the EB model, since it largely simplifies the training and testing without losing the strength of the HB formulation.

The idea of using HB or EB for meta-learning is not new: Amit & Meir (2018) derive an objective similar to that of HB using PAC-Bayesian analysis; Grant et al. (2017) can be understood as a EB method; Ravi & Beatson (2018) consider a HB extension to MAML and compute posteriors via amortized variational inference. However, unlike our proposal, these methods do not take advantage of the unlabeled set. Roughly speaking, they construct the variational
under review as a conference paper at ICLR 2020

Figure 1: (a) The generative and inference processes of the empirical Bayes model are depicted in solid and dashed arrows respectively, where the meta-parameters are denoted by dashed circles due to the point estimates. A comparison between MAML (6) and our method (SIB) is shown in (b) and (c). MAML is an inductive method since, for a task $t$, it first constructs a variational posterior $q_{\theta^K_t}$ as a function of the labeled set $d^t_l$, and then test on the unlabeled set $x_t$; while SIB constructs a better variational posterior as a function of both $d^t_l$ and $x_t$: it starts from an initialization $\theta^t_0(d^t_l)$, and then yields $\theta^K_t$ by running $K$ synthetic gradient steps on $x_t$.

posterior as a function of the labeled set $d^t_l$ without taking advantage of the unlabeled set $x_t$. The situation is similar in gradient based meta-learning methods (Finn et al., 2017; Ravi & Larochelle, 2016; Li et al., 2017b; Nichol et al., 2018; Flenerthag et al., 2019), and many other meta-learning methods (Vinyals et al., 2016; Snell et al., 2017; Gidaris & Komodakis, 2018), where the mechanisms used to generate the task-specific parameters rely on groundtruth labels, thus, there is no place for the unlabeled set to contribute. We argue that this is a suboptimal choice, which may lead to overfitting when the labeled set is small and hinder the possibility of zero-shot learning (when the labeled set is not provided). An exception is Liu et al. (2018). They reuse the label propagation algorithm (Zhu et al., 2003) for transductive inference within each task and show that transduction is useful for boosting the performance.

In this paper, we propose to use synthetic gradient (Jaderberg et al., 2017) to enable transductivity, such that the variational posterior is implemented as a function of the labeled set $d^t_l$ and the unlabeled set $x_t$. The synthetic gradient is produced by a neural network and learned to be a surrogate of the true gradient. The optimization process is similar to the inner gradient descent in MAML, but it iterates on the unlabeled set $x_t$ rather than on labeled set $d^t_l$, since it does not rely on $y_t$ to compute the true gradient. The labeled set for an unseen task is now optional, which is only used to generate the initialization in our case. In summary, our main contributions are the following:

1. In section 2 and section 3, we develop a novel empirical Bayes formulation with transduction for meta-learning. To perform amortized variational inference, we propose a parameterization for the variational posterior based on synthetic gradient descent, which incorporates the contextual information from all the inputs of the query set.

2. In section 4, we show in theory that a transductive variational posterior yields better generalization performance. Besides, we show that the proposed empirical Bayes formulation is equivalent to the information bottleneck principle considered by Achille & Soatto (2017). We thus call our method synthetic information bottleneck (SIB).

3. In section 5, we verify our proposal empirically. Our experimental results demonstrate that our method significantly outperforms the state-of-the-art meta-learning methods on standard few-shot classification benchmarks.

2 Meta-learning with Transductive Inference

The goal of meta-learning is to train a meta-model on a collection of tasks, such that it works well on another disjoint collection of tasks. Suppose that we are given a collection of $N$ tasks for training. The associated data is denoted by $\mathcal{D} := \{d_i := (x_i, y_i)\}_{i=1}^N$. In the case of few-shot learning, we are given in addition a support set $d^t_l$ for each task. In this section, we revisit the classical empirical Bayes model for meta-learning. Then, we propose to use a transductive scheme in the variational inference by constructing the variational posterior as a function of $x_t$. 

2
2.1 Empirical Bayes Model

Due to the hierarchical structure among data, it is natural to consider a hierarchical Bayes model with the marginal likelihood

$$p(D) = \int_{\psi} p(D|\psi)p(\psi) = \int_{\psi} \left[ \prod_{t=1}^{N} \int_{w_t} p(d_t|w_t)p(w_t|\psi) \right] p(\psi).$$  

(1)

The generative process is illustrated in Figure 1 (left, in solid arrows): first, a meta-parameter $\psi$ (aka hyper-parameter) is sampled from the hyper-prior $p(\psi)$; then, for each task, a task-specific parameter $w_t$ is sampled from the prior $p(w_t|\psi)$; finally, the dataset is drawn from the likelihood $p(d_t|w_t)$. Without loss of generality, we assume the log-likelihood model factorizes as

$$\log p_f(d_t|w_t) = \sum_{i=1}^{n} \log p_f(y_{t,i}|x_{t,i}, w_t) + \log p(x_{t,i}|w_t),$$

$$= -\frac{1}{n} \sum_{i=1}^{n} \ell_t(\hat{y}_{t,i}, f(x_{t,i}), w_t), y_{t,i}) + \text{constant},$$

(2)

which is the setting advocated by Minka (2005), in which the generative model $p(x_{t,i}|w_t)$ can be safely ignored since it is irrelevant to the prediction of $y_t$. To simplify the presentation, we still keep the notation $p(d_t|w_t)$ for the likelihood of the task $t$ and use $\ell_t$ to specify the discriminative model, which is also referred to as the task-specific loss, e.g., the cross entropy loss. The first argument in $\ell_t$ is the prediction, denoted by $\hat{y}_{t,i} = \hat{y}_{t,i}(f(x_{t,i}), w_t)$, which depends on the feature representation $f(x_{t,i})$ and the task-specific weight $w_t$.

Note that rather than following a fully Bayesian approach, we leave some random variables to be estimated by a frequentist approach, e.g., $f$ is a meta-parameter of the likelihood model shared by all tasks, for which we use a point estimate. As such, the posterior inference about these variables will be largely simplified. For the same reason, we derive the empirical Bayes (Robbins, 1985; Kucukelbir & Blei, 2014), which interprets $\psi$ in a frequentist way:

$$p_{\psi,f}(D) = \prod_{t=1}^{N} \int_{w_t} p_f(d_t|w_t)p_{\psi}(w_t).$$  

(3)

We highlight the meta-parameters as subscripts of the corresponding distributions to distinguish from random variables. Indeed, we are not the first to formulate meta-learning as empirical Bayes. The overall model formulation is essentially the same as the ones considered by Amit & Meir (2018); Grant et al. (2018); Ravi & Beatson (2018).

2.2 Amortized Inference with Transduction

As in standard probabilistic modeling, we derive an evidence lower bound (ELBO) on the log version of (3) by introducing a variational distribution $q_{\theta_t}(w_t)$ for each task with parameter $\theta_t$:

$$\log p_{\psi,f}(D) \geq \sum_{t=1}^{N} \mathbb{E}_{q_{\theta_t}} \left[ \log p_f(d_t|w_t) \right] - D_{\text{KL}}(q_{\theta_t}(w_t) \mid \mid p_{\psi}(w_t)).$$

(4)

The variational inference amounts to maximizing the ELBO with respect to $\theta_1, \ldots, \theta_N$, which together with the maximum likelihood estimation of the meta-parameters, we have the following optimization problem:

$$\min_{\psi, f} \min_{\theta_1, \ldots, \theta_N} \frac{1}{N} \sum_{t=1}^{N} \mathbb{E}_{q_{\theta_t}} \left[ -\log p_f(d_t|w_t) \right] + D_{\text{KL}}(q_{\theta_t}(w_t) \mid \mid p_{\psi}(w_t)).$$  

(5)

However, the optimization in (5), as $N$ increases, becomes more and more expensive in terms of the memory footprint and the computational cost. We therefore wish to bypass this heavy optimization and to take advantage of the fact that individual KL terms indeed share the same structure. To this end, instead of introducing $N$ different variational distributions, we consider a parameterized family of distributions in the form of $q_{\phi(t)}$, which is defined implicitly by a deep neural network $\phi$ taking
as input either $d_t^i$ or $x_t$ or both, that is, $q_{\phi}(d_t^i)$ or $q_{\phi}(d_t^i, x_t)$. This technique is known as amortized variational inference in the literature (Kingma & Welling, 2013; Rezende et al., 2014).

Since $d_t^i$ and $x_t$ are disjoint, the inference scheme is inductive with a variational posterior $q_{\phi}(d_t^i)$. As an example, MAML (Finn et al., 2017) is an inductive method forcing $q_{\phi}(d_t^i)$ to be a Dirac delta distribution, where $\phi(d_t^i) = \theta_t^k$, the $K$-th iterate of the stochastic gradient descent

$$\theta_t^{k+1} = \theta_t^k + \eta \nabla_{\theta_t} \mathbb{E}_{\theta_k \sim q_{\phi}(\theta)} \left[ \log p(d_t^i | w_t) \right] + \theta_t^0 = \phi. \quad (6)$$

Note that we overload $\phi$ to be both the learnable initialization as well as the amortization.

In this work, we consider a transductive inference scheme by using the entire $x_t$ to define the variational posterior as $q_{\phi(x_t)}$. Replacing each $q_{\phi}$ by $q_{\phi(x_t)}$, (5) can be written as

$$\min_{\psi, f} \min_{\phi} \frac{1}{N} \sum_{t=1}^{N} \left[ \mathbb{E}_{w_t \sim q_{\phi(x_t)}} \left[ \log p_f(y_t | w_t) \right] - D_{KL}(q_{\phi(x_t)}(w_t) \| p_{\psi}(w_t)) \right]. \quad (7)$$

It is also possible to define the variational posterior as $q_{\phi(x_t, w_t)} = q_{\phi(d_t^i)}$. However, to be consistent with testing, we do not take $y_t$ as input since for which we do not have access to during testing.

In fact, nothing prevents us to come up with an even better variational posterior $q_{\phi(x_t, d_t^i)}$. Shown in dashed arrows in Figure [1](a), which is again transductive by definition. In a nutshell, the meta-model includes $f, \psi$ from empirical Bayes and the amortization $\phi$ for inference.

### 3 Variational inference with synthetic gradients

It is however non-trivial to design a network architecture to implement the amortization $\phi(x_t)$ directly since $x_t$ is itself a dataset. The strategy adopted by neural processes (Garnelo et al., 2018) is to aggregate the information from all individual examples via a permutation invariant function. However, as pointed out by Kim et al. (2019), such a strategy tends to underfit $x_t$ because the aggregation does not necessarily attain the most relevant information for identifying the task-specific parameter. Attentive neural process (Kim et al., 2019) may alleviate this issue with a time complexity of $O(n^2)$ while the original neural processes only need $O(n)$ time. We instead design a neural network $\phi(x_t)$ to parameterize the optimization process of $\theta_t$. More specifically, consider a stochastic gradient descent on $\theta_t$ for optimizing (5) with step size $\eta$:

$$\theta_t^{k+1} = \theta_t^k - \eta \nabla_{\theta_t} D_{KL}(q_{\theta_t}(w) \| p_{\psi, f}(w | d_t^i)). \quad (8)$$

We would like to parameterize this optimization dynamics up to the $K$-th step via $\phi(x_t)$, such that $q_{\theta_t^K}$ is a good approximation of the optimum $q_{\theta_t^*}$. It consists of parameterizing

(a) the initialization $\theta_t^0$ and (b) the gradient $\nabla_{\theta_t} D_{KL}(q_{\theta_t}(w_t) \| p_{\psi, f}(w_t | d_t^i)).$

By doing so, $\theta_t^K$ becomes a function of $\phi, \psi$ and $x_t^i$ we therefore realize $q_{\phi(x_t)}$ as $q_{\theta_t^K}$.

For (a), we opt to either let $\theta_t^0 = \lambda$ to be a global data-independent initialization as in MAML (Finn et al., 2017) or let $\theta_t^0 = \lambda(d_t^i)$ with a few supervisions from the support set, where $\lambda$ can be implemented by a permutation invariant network described in Gidaris & Komodakis (2018). In the second case, the features of the support set will be first averaged in terms of their labels and then scaled by a learned vector of the same size.

For (b), the fundamental reason that we parameterize the gradient is because we do not have access to $y_t$ during the meta-testing phase. Note that we are able to follow (8) in meta-training to obtain $q_{\theta_t^K}(w_t) \propto p_f(d_t^i | w_t)p_{\psi}(w_t)$. To make a consistent parameterization in both meta-training and meta-testing, we thus discard $y_t$ when constructing the variational posterior. Regarding the true gradient, a key observation is that, under a reparameterization $w_t = w_t(\theta_t, \epsilon)$ with $\epsilon \sim p(\epsilon)$,

$$\nabla_{\theta_t} D_{KL}(q_{\theta_t}(w_t) \| p_{\psi, f}) = \mathbb{E}_{\epsilon} \left[ \frac{1}{n} \sum_{i=1}^{n} \frac{\partial \theta_t}{\partial y_t} \frac{\partial y_t}{\partial w_t} \frac{\partial w_t}{\partial \theta_t} \right] + \nabla_{\theta_t} D_{KL}(q_{\theta_t}(w_t) \| p_{\psi}) \cdot$$

$^1\theta_t^K$ is also dependent of $f$. We deliberately remove this dependency to simplify the update of $f$.  

Under review as a conference paper at ICLR 2020
where all the terms can be computed without \( y_t \) except for \( \frac{\partial \ell_t}{\partial y_{t,i}} \), thus, we introduce a deep neural network \( \xi(\hat{y}_{t,i}) \) to synthesize it. The idea of synthetic gradients was originally proposed by Jaderberg et al. (2017) to parallelize the back-propagation. Here, the purpose of \( \xi(\hat{y}_{t,i}) \) is to update \( \theta_t \) regardless of the groundtruth labels, which is slightly different from its original purpose. Besides, we do not introduce an additional loss to force \( \xi(\hat{y}_{t,i}) \) to approximate \( \frac{\partial \ell_t}{\partial y_{t,i}} \) since \( \xi(\hat{y}_{t,i}) \) will be learned to yield a reasonable \( \theta^K_t \) even without mimicking the true gradient.

To sum up, we have derived a particular implementation of \( \phi(x_t) \) by parameterizing the ideal mean-field update, namely (9), with respect to the query set \( d_t \), where the meta-model \( \phi \) includes an initialization network \( \lambda \) and a synthetic gradient network \( \xi \), such that the output of the amortization \( \phi(x_t) \) is \( \theta^K_t \) — the \( K \)-th iterate of the following update:

\[
\theta^{K+1}_t = \theta^K_t - \eta \left[ \mathbb{E}_c \left\{ \frac{1}{n} \sum_{i=1}^n \xi(\hat{y}_{t,i}) \frac{\partial \hat{y}_{t,i}}{\partial w_t} \frac{\partial w_t(\theta^K_t, c)}{\partial \theta_t} \right\} + \nabla_{\theta_t} D_{KL}(q_{\theta^K_t} \parallel p_\psi) \right].
\]  

(9)

The overall algorithm is depicted in Algorithm 1. We also make a side-by-side comparison with MAML shown in Figure 1. Rather than viewing (9) as an optimization process, it may be more precise to think of it as a part of the computation graph created in the forward-propagation. The computation graph of the amortized inference is shown in Figure 2.

As an extension, if we were deciding to estimate the feature network \( f \) in a Bayesian manner, we would have to compute higher-order gradients as in the case of MAML. This is impractical from a computational point of view and needs technical simplifications (Nichol et al., 2018). By introducing a series of synthetic gradient networks in a way similar to Jaderberg et al. (2017), the computation will be decoupled into computations within each layer, and thus becomes more feasible. We see this as a potential advantage of our method and leave this to our future work.

4 GENERALIZATION OF EMPIRICAL BAYES AND ITS CONNECTION TO INFORMATION BOTTLENECK

In this section, we study the generalization ability of the empirical Bayes (EB) model as well as its connection to the information bottleneck principle proposed by Achille & Soatto (2017).
Algorithm 1 Variational inference with synthetic gradients for empirical Bayes

1: **Input:** the dataset \( D \); the step size \( \eta \); the number of inner iterations \( K \); pretrained \( f \).
2: Initialize the meta-models \( \psi \), and \( \phi = (\lambda, \xi) \).
3: **while** not converged **do**
4: Sample a task \( t \) and the associated dataset \( d_t \) (plus optionally the support set \( d_t^{(i)} \)).
5: Compute the initialization \( \theta_t^0 = \lambda_t \) or \( \theta_t^0 = \lambda(d_t) \).
6: **for** \( k = 1, \ldots, K \) **do**
7: Compute \( \theta_t^k \) via (4).
8: **end for**
9: Compute \( w_t = w_t(\theta_t^k, \epsilon) \) with \( \epsilon \sim p(\epsilon) \).
10: Update \( \psi \leftarrow \psi - \eta \nabla_{\psi} D_{KL}(q_{\theta_t^k}(\psi)||p_\psi) \).
11: Update \( \phi \leftarrow \phi - \eta \nabla_{\phi} D_{KL}(q_{\phi}(x_t)\|p_f \cdot p_\psi) \).
12: Optionally, update \( f \leftarrow f + \eta \nabla_f \log p_f (d_t|w_t) \).
13: **end while**

From [3], we can see that EB implies a) \( \{d_t\}_{t=1}^N \) are independent of each other and b) a decomposition of the joint distribution (see Appendix A for the derivation)

\[
p_{\psi, f}(w_1, \ldots, w_N, D) = \prod_{i=1}^N p_{\psi, f}(w_t|d_t)p_{\psi, f}(d_t).
\]

Thus, rather than viewing the aggregated posterior

\[
R_{\text{aggregated}}(q) = \mathbb{E}_{d\sim D}\sum_{i=1}^N \ell_i(y_t, f(x_t, w_t))
\]

which, by variational inference, will be approximated by its variational counterpart \( q(w|d) = q(d)q(w|d) \), where \( q(d) \) can be seen as the underlying data distribution and \( q(w|d) \) is the variational posterior.

Consider an amortization \( \phi(d) \) and \( d = (x, y) \), then we can parameterize the variational posterior as \( q(w|d) = q_{\phi(d)}(w) \), or even simpler: \( q(w|d) = q(w|x) = q_{\phi(x)}(w) \). Note that both parameterizations result in transductive inference since, if we would like to test on \( x \), we will incorporate \( x \) to build a variational posterior. An inductive variational posterior is a special case such that \( q(w|d, d') = q(w|d') \).

Before going into details, we first introduce a few notations: the entropy is defined as \( H_p(x) := \mathbb{E}_{p(x)}[-\log p(x)] \); the mutual information is given by \( I_{p}(x; y) := D_{KL}(p(x, y)||p(x)p(y)) \); the cross entropy is defined as \( H_{p,q}(x) := \mathbb{E}_{p(x)}[-\log q(x)] \).

Assuming that \( q(w|d) \) can be optimized exactly to match the empirical Bayes model, we can then use it as a proxy to analyze the generalization performance of EB. Certainly, the quality of the optimization matters, we will also analyze this effect.

To this end, we first identify the empirical risk for a single task as \( L(w, d) := \frac{1}{n} \sum_{i=1}^n \ell_i(y_t, f(x_t, w), y_t) \). Then, the average empirical risk is the expectation over all possible samples and weights, which depends on the variational posterior and the data distribution:

\[
\hat{R}(q(w|d), q(d)) := \mathbb{E}_{d\sim D} \mathbb{E}_{w\sim q(w|d)} L(w, d)
\]

(11)

The true risk should be independent of any particular variational posterior. Inspired by [Xu & Raginsky 2017], we define the true risk as follows, where we sample the task-specific weight from the aggregated posterior \( q(w) \):

\[
R(q(w), q(d)) := \mathbb{E}_{w\sim q(w)} \mathbb{E}_{t\sim q(t)} \mathbb{E}_{(x_t, i, y_t, i)} \ell_i(y_t, f(x_t, i, w), y_t, i)
= \mathbb{E}_{w\sim q(w)} \mathbb{E}_{t\sim q(t)} \mathbb{E}_{d\sim q(d|t)} \frac{1}{n} \sum_{i=1}^n \ell_i(y_t, f(x_t, i, w), y_t)
= \mathbb{E}_{w\sim q(w)} \mathbb{E}_{d\sim q(d)} L(w, d).
\]

(12)

Finally, the generalization error is defined as

\[
\text{gen}(q(w|d), q(d)) := R(q(w), q(d)) - \hat{R}(q(w|d), q(d)).
\]

(13)

\footnote{When the context is clear, we ignore the parameters and write \( p_{\psi, f}(w, d) = p(w, d) \).}
Intuitively, the generalization error measures how much the empirical risk concentrated on the true risk is. That is, we would like to bound the tail of the distribution over all possible values of the empirical risk. By taking an expectation of the objective in (7) with respect to the choice of D, we have

\[
\mathbb{E}_{D \sim q(D)} \left[ \sum_{i=1}^{N} D_{KL}(q(w_i|d_i)||p(d_i|w)p(w)) \right] = \mathbb{E}_{d \sim q(d)} D_{KL}(q(w|d)||p(d|w)p(w)).
\]

We show in Theorem 1 that the quantity on the right hand side directly affects the generalization performance.

**Theorem 1.** Given distributions \(q(w|d), q(d), p(w) \) and \(p(d|w)\), if \(\ell_{1}(\hat{y}(w), y)\) is \(\sigma\)-subgaussian for all \(w\), the following inequalities hold:

\[
\mathbb{E}_{d \sim q(d)} D_{KL}(q(w|d)||p(d|w)p(w)) \geq I_q(w; d) + H_q(p(d|w)) \tag{14}
\]

\[
\geq \frac{n}{2\sigma^2} \text{gen}(q(w|d), q(d))^2 + \hat{R}(q(w|d), q(d));
\]

\[
\mathbb{E}_{d \sim q(d)} D_{KL}(q(w|d)||p(w,d)) \leq D_{KL}(q(w,d)||p(w,d)) . \tag{15}
\]

The equality in (14) holds if \(q(w) = p(w)\). The equality in (15) holds if \(q(d) = p(d)\).

The proof of Theorem 1 can be found in Appendix B.

**Implications of (14)** The inequality (14) basically says that (7) can be seen a regularized empirical risk minimization in which the regularization term is an upper bound of the mutual information between the weight and the sample. In general, there is a tradeoff between the generalization error and the empirical risk controlled by the coefficient \(\frac{1}{2\sigma^2}\), where \(n = |d|\) is the cardinality of \(d\). If \(n\) is small, then we are in the overfitting regime. This is the case of the inductive inference with variational posterior \(q(w|d^t)\), where the support set \(d^t\) is fairly small by the definition of few-shot learning. Consequently, we expect these methods to have large generalization errors in light of the above analysis. On the other hand, if we were following the transductive setting, the sample size \(n\) would be larger and thus would achieve a small generalization error. However, keeping increasing \(n\) will potentially over-regularize the model and thus yield negative effect. An empirical study on varying \(n\) can be found in Table 4 in the Appendix. Besides, the inequality (14) also suggests that the quality of the variational inference, measured by \(\mathbb{E}_{d \sim q(d)} D_{KL}(q(w|d)||p(w|d))\), largely affect the generalization error. If we can make this term diminishing to zero, we should be able to achieve zero generalization error and zero empirical risk.

Besides, (14) reveals a connection between empirical Bayes and information bottleneck (Tishby et al. 2000; Achille & Soatto 2017). The right hand side of (14) is exactly the IB objective considered by Achille & Soatto (2017) with a coefficient equal to 1. This connection is critical to the analysis since the generalization error is introduced by invoking its relationship to \(I(w; d)\). We find this connection interesting, thus, we call our method synthetic information bottleneck (SIB).

**Implications of (15)** An obvious message from the inequality (15) is that, if we choose appropriate likelihood, prior and variational posterior such that \(p(w, d)\) is aligned with \(q(w, d)\), the empirical Bayes model can actually be pretty good, that is, both the generalization error and the empirical risk are close to zero. If we look at the message more carefully, it also implies that a good likelihood is the key to represent the data well; the inference model is less important as long as the aggregated posterior coincides with the prior. A similar empirical finding was confirmed by Gidaris et al. (2019), who suggests to learn a better feature model, which in turn leads to a good likelihood model according to the definition in (2).

## 5 Experiments

In this section, we first validate our method on few-shot learning, and then on zero-shot learning. Note that many meta-learning methods, such as MAML, cannot do zero-shot learning since they rely on the support set.

### 5.1 Few-shot classification

Table 1: Average classification accuracies (with 95% confidence intervals) on the test-set of MiniImageNet and CIFAR-FS. For evaluation, we sample 2000 and 5000 episodes respectively for MiniImageNet and CIFAR-FS and test three different architectures as the feature extractor: Conv-4-64, Conv-4-128 and WRN-28-10. We train SIB with learning rate $\eta=1e-3$, $\lambda=1e-3$, $K=3$. Finally, for the predictor $\hat{y}_{ij}(\cdot, w)$, we adopt the cosine-similarity based classifier advocated by Chen et al. (2019) and Gidaris & Komodakis (2018).

5.1.1 Setup

Datasets We choose standard benchmarks of few-shot classification for this experiment. Each benchmark is composed of disjoint training, validation and testing classes. MiniImageNet is proposed by Vinyals et al. (2016), which contains 100 classes, split into 64 training classes, 16 validation classes and 20 testing classes, where each class consists of 600 image-label pairs and each image is of size $84 \times 84$. CIFAR-FS is proposed by Bertinetto et al. (2018), which is created by dividing the original CIFAR-100 into 64 training classes, 16 validation classes and 20 testing classes; each image is of size $32 \times 32$.

Network architectures Following Gidaris & Komodakis (2018); Qiao et al. (2018); Gidaris et al. (2019), we implement $f$ by a 4-layer convolutional network (Conv-4-64 or Conv-4-128) or a WideResNet (WRN-28-10) (Zagoruyko & Komodakis, 2016). We pretrain the feature network $f(\cdot)$ on the 64 training classes for a standard $64$-way classification. We reuse the feature averaging network proposed by Gidaris & Komodakis (2018) as our initialization network $\mathbf{X}(\cdot)$, which basically averages the feature vectors of all data points from the same class and then scales each feature dimension differently by a learned coefficient. For the synthetic gradient network $\xi(\cdot)$, we implement a three-layer MLP with hidden-layer size $8k$. Finally, for the predictor $\hat{y}_{ij}(\cdot, w)$, we adopt the cosine-similarity based classifier advocated by Chen et al. (2019) and Gidaris & Komodakis (2018).

Evaluation metrics In few-shot classification, a task (aka episode) $t$ consists of a query set $d_t^q$ and a support set $d_t^s$. When we say the task $f$ is $k$-way-$n^\prime$-shot we mean that $d_t^q$ is formed by first sampling $k$ classes from a pool of classes; then, for each sampled class, $n^\prime$ examples are drawn and a new

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>MiniImageNet, 5-way classification</th>
<th>CIFAR-FS, 5-way classification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>Matching Net (Vinyals et al. 2016)</td>
<td>Conv-4-64</td>
<td>44.2%</td>
<td>57%</td>
</tr>
<tr>
<td>MAML (Finn et al. 2017)</td>
<td>Conv-4-64</td>
<td>48.7±1.8%</td>
<td>63.1±0.9%</td>
</tr>
<tr>
<td>Prototypical Network (Snell et al. 2017)</td>
<td>Conv-4-64</td>
<td>49.4±0.8%</td>
<td>65.2±0.7%</td>
</tr>
<tr>
<td>Relation Net (Sung et al. 2018)</td>
<td>Conv-4-64</td>
<td>50.4±0.8%</td>
<td>65.3±0.7%</td>
</tr>
<tr>
<td>GNN (Satorras &amp; Bruna 2017)</td>
<td>Conv-4-64</td>
<td>50.3%</td>
<td>66.4%</td>
</tr>
<tr>
<td>R2-D2 (Bertinetto et al. 2018)</td>
<td>Conv-4-64</td>
<td>49.5±0.2%</td>
<td>65.4±0.2%</td>
</tr>
<tr>
<td>TPN (Liu et al. 2018)</td>
<td>Conv-4-64</td>
<td>55.5%</td>
<td>69.9%</td>
</tr>
<tr>
<td>Gidaris et al. (2019) Conv-4-64</td>
<td>Conv-4-64</td>
<td>54.8±0.4%</td>
<td>71.9±0.3%</td>
</tr>
<tr>
<td>SIB $\eta=0$ (Pre-trained feature)</td>
<td>Conv-4-64</td>
<td>50.0±0.4%</td>
<td>67.0±0.4%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=3$</td>
<td>Conv-4-64</td>
<td>58.0±0.6%</td>
<td>70.7±0.4%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=0$</td>
<td>Conv-4-128</td>
<td>53.62±0.79%</td>
<td>71.48±0.64%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=1$</td>
<td>Conv-4-128</td>
<td>58.74±0.89%</td>
<td>74.12±0.63%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=3$</td>
<td>Conv-4-128</td>
<td>62.59±1.02%</td>
<td>75.43±0.67%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=5$</td>
<td>Conv-4-128</td>
<td>63.26±1.07%</td>
<td>75.73±0.71%</td>
</tr>
<tr>
<td>TADAM (Oreshkin et al. 2018)</td>
<td>ResNet-12</td>
<td>58.5±0.3%</td>
<td>76.7±0.3%</td>
</tr>
<tr>
<td>SNAIL (Sanctoro et al. 2017)</td>
<td>ResNet-12</td>
<td>55.7±1.0%</td>
<td>68.9±0.9%</td>
</tr>
<tr>
<td>MetaOptNet-RR (Lee et al. 2019)</td>
<td>ResNet-12</td>
<td>61.4±0.6%</td>
<td>77.9±0.5%</td>
</tr>
<tr>
<td>MetaOptNet-SVM</td>
<td>ResNet-12</td>
<td>62.6±0.6%</td>
<td>78.6±0.5%</td>
</tr>
<tr>
<td>CTM (Li et al. 2019)</td>
<td>ResNet-18</td>
<td>64.1±0.8%</td>
<td>80.5±0.1%</td>
</tr>
<tr>
<td>Qiao et al. (2018)</td>
<td>WRN-28-10</td>
<td>59.0±0.4%</td>
<td>73.7±0.2%</td>
</tr>
<tr>
<td>LEO (Rusu et al. 2019)</td>
<td>WRN-28-10</td>
<td>61.8±0.1%</td>
<td>77.6±0.1%</td>
</tr>
<tr>
<td>Gidaris et al. (2019) WRN-28-10</td>
<td>WRN-28-10</td>
<td>62.9±0.5%</td>
<td>79.9±0.3%</td>
</tr>
<tr>
<td>SIB $\eta=0$ (Pre-trained feature)</td>
<td>WRN-28-10</td>
<td>60.6±0.4%</td>
<td>77.5±0.3%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=1$</td>
<td>WRN-28-10</td>
<td>67.3±0.5%</td>
<td>78.8±0.4%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=3$</td>
<td>WRN-28-10</td>
<td>69.6±0.6%</td>
<td>78.9±0.4%</td>
</tr>
<tr>
<td>SIB $\eta=1e-3$, $K=5$</td>
<td>WRN-28-10</td>
<td>70.0±0.6%</td>
<td>79.2±0.4%</td>
</tr>
</tbody>
</table>
We construct a dataset $d$ which suggests that SIB is a general method for few-shot learning. Then, we generate the weight for each dataset by calculating the mean of the inputs and shifting $D$. Denote by $I(x; d)$ the similarity between a query example and the support set while making use of intra- and inter-class extensions to our future work. We first look at a toy multi-task problem, where SIB is also applicable to zero-shot problems (i.e., no support set available). We leave these ideas as promised. For 1-shot learning, SIB (when $K = 3$ or $K = 5$) significantly outperforms previous methods on both MiniImageNet and CIFAR-FS. For 5-shot learning, the results are also comparable. It should be noted that the performance boost is consistently observed with different feature networks, which suggests that SIB is a general method for few-shot learning.

However, we also observe a potential limitation of SIB: when the support set is relatively large, e.g., 5-shot, with a good feature network like WRN-28-10, the initialization $\theta_k^0$ may already be close to some local minimum, making the updates later less important. For 5-shot learning, SIB is slightly worse than CTM (Li et al., 2019) and/or Gidaris et al. (2019). CMT (Li et al., 2019) can be seen as an alternative way to incorporate transduction – it measures the similarity between a query example and the support set while making use of intra- and inter-class relationships. Gidaris et al. (2019) uses in addition the self-supervision as an auxiliary loss to learn a richer and more transferable feature model. Both ideas are complementary to SIB. We leave these extensions to our future work.

5.2 Zero-shot regression: spinning lines

Since our variational posterior relies only on $x_t$, SIB is also applicable to zero-shot problems (i.e., no support set available). We first look at a toy multi-task problem, where $I(x_t; d)$ is tractable. Denote by $D_{\text{train}} := \{d_t\}_{t=1}^N$ the train set, which consists of datasets of size $n$: $d = \{(x_t, y_t)\}_{t=1}^n$. We construct a dataset $d$ by firstly sampling iid Gaussian random variables as inputs: $x_t \sim N(\mu, \sigma^2)$. Then, we generate the weight for each dataset by calculating the mean of the inputs and shifting.
Table 2: Multi-source domain adaptation results on PACS with ResNet-18 features. Three domains are used as the source domains keeping the fourth one as target.

<table>
<thead>
<tr>
<th>Method</th>
<th>Art</th>
<th>Cartoon</th>
<th>Sketch</th>
<th>Photo</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>JiGen (Carlucci et al., 2019)</td>
<td>84.9%</td>
<td>81.1%</td>
<td>79.1%</td>
<td>98.0%</td>
<td>85.7%</td>
</tr>
<tr>
<td>Rot (Xu et al., 2019)</td>
<td>88.7%</td>
<td>86.4%</td>
<td>74.9%</td>
<td>98.0%</td>
<td>87.0%</td>
</tr>
<tr>
<td>SIB-Rot $A = 0$</td>
<td>85.7%</td>
<td>86.6%</td>
<td>80.3%</td>
<td>98.3%</td>
<td>87.7%</td>
</tr>
<tr>
<td>SIB-Rot $K = 3$</td>
<td>88.9%</td>
<td>89.0%</td>
<td>82.2%</td>
<td>98.3%</td>
<td>89.6%</td>
</tr>
</tbody>
</table>

A more interesting zero-shot multi-task problem is unsupervised domain adaptation. We consider the only difference is the initialization network. Denote by $\theta$ the synthetic gradient network is able to identify the descent direction after meta-learning.

$$\ell_t \text{ is a mean squared error, implies that } p(y|x,w) = N(wx,1);$$

$$p_\psi(w) \text{ is a Gaussian distribution with parameters } \psi \in \mathbb{R}^2;$$

$$\text{the synthetic gradient network } \xi \text{ is a three-layer MLP with hidden size } 8.$$ 

In the experiment, we sample 240 tasks respectively for both $D_{train}$ and $D_{test}$. We learn SIB and BNN on $D_{train}$ for 150 epochs using the ADAM optimizer (Kingma & Ba, 2014), with learning rate $10^{-3}$ and batch size 8. Other hyperparameters are specified as follows: $n = 32, K = 3, \mu = 0, \sigma = 1, \mu_w = 1, \sigma_w = 0.1$. The results are shown in Figure 3. On the left, both $D_{KL}(q_{\theta_P}(w_t)||p(w_t|d_t))$ and $D_{KL}(p_\psi(w)||p(w))$ are close to zero indicating the success of the learning. More interestingly, in the middle, we see that $\theta^0_1, \theta^1_1, \ldots, \theta^2_1$ evolves gradually towards the ground truth, which suggests that the synthetic gradient network is able to identify the descent direction after meta-learning.

5.3 Zero-shot classification: unsupervised multi-source domain adaptation

A more interesting zero-shot multi-task problem is unsupervised domain adaptation. We consider the case where there exists multiple source domains and an unlabeled target domain. In this case, we treat each minibatch as a task. This makes sense because the difference in statistics between two minibatches are much larger than in the traditional supervised learning. The experimental setup is similar to few-shot classification described in Section 5.1 except that we do not have a support set and the class labels between two tasks are the same. Hence, it is possible to explore the relationship between class labels and self-supervised labels to implement the initialization network $\lambda$ without resorting to support set. We reuse the same model implementation for SIB as described in Section 5.1.

The only difference is the initialization network. Denote by $z_t := \{z_{ti}\}_{i=1}^n$ the set of self-supervised labels of task $t$, the initialization network $\lambda$ is implemented as follows:

$$\theta^t_1 = \lambda - \eta \nabla_\theta L_t \left( \tilde{z}_t(f(x_t), w_t(\theta, \epsilon)), f(x_t), z_t \right),$$

where $\theta^t_1$ is a global initialization similar to the one used by MAML; $L_t$ is the self-supervised loss; $\tilde{z}_t$ is the set of predictions of the self-supervised labels. One may argue that $\theta^t_1$ is a simpler solution. However, it is insufficient since the gap between two domains can be very large. The initial solution yielded by this is more dynamic in the sense that $\theta^t_1$ is adapted taking into account the information from $x_t$. 

$^3\lambda$ is overloaded to be both the network and its parameters.
In terms of experiments, we test SIB on the PACS dataset (Li et al., 2017a), which has 7 object categories and 4 domains (Photo, Art Paintings, Cartoon and Sketches), and compare with state-of-the-art algorithms for unsupervised domain adaptation. We follow the standard experimental setting (Carlucci et al., 2019), where the feature network is implemented by ResNet-18. We assign a self-supervised label $z_{t,i}$ to image $i$ by rotating the image by a predicted degree. This idea was originally proposed by Gidaris et al. (2018) for representation learning and adopted by Xu et al. (2019) for domain adaptation. The training is done by running ADAM for 60 epochs with learning rate $10^{-4}$. We take each domain in turns as the target domain. The results are shown in Table 2.

It can be seen that SIB-Rot ($K = 3$) improves upon the baseline SIB-Rot ($K = 0$) for zero-shot classification, which also outperforms state-of-the-art methods when the baseline is comparable.

6 Conclusion

We have presented an empirical Bayesian framework for meta-learning. To enable an efficient variational inference, we followed the amortized inference paradigm, and proposed to use a transductive scheme for constructing the variational posterior. To implement the transductive inference, we make use of two neural networks: a synthetic gradient network and an initialization network, which together enables a synthetic gradient descent on the unlabeled data to generate the parameters of the amortized variational posterior dynamically. We have studied the theoretical properties of the proposed framework and shown that it yields performance boost on MiniImageNet and CIFAR-FS for few-shot classification.

References


Then, we only need to show Lemma 1. A random variable $\psi, f$ Similarly, we have

Under review as a conference paper at ICLR 2020

**Proof.** Now, we are ready to prove Theorem 1.

**APPENDIX**

**A Decomposition implied by empirical Bayes**

In this section, we prove the statement in [10]. First of all,

$$p_{\psi, f}(w_1, \ldots, w_N, D) = p_{\psi, f}(w_1, \ldots, w_N \mid D) p_{\psi, f}(D)$$

$$= p_{\psi, f}(w_1, \ldots, w_N \mid D) \prod_{i=1}^{N} p_{\psi, f}(d_i)$$

Then, we only need to show

$$p_{\psi, f}(w_1, \ldots, w_N \mid D) = \frac{\prod_{i=1}^{N} p_f(d_i \mid w_i) p_{\psi}(w_i)}{\int_{w_1, \ldots, w_N} \prod_{i=1}^{N} p_f(d_i \mid w_i) p_{\psi}(w_i)} = \frac{\prod_{i=1}^{N} p_f(d_i \mid w_i) p_{\psi}(w_i)}{\prod_{i=1}^{N} \int_{w_i} p_f(d_i \mid w_i) p_{\psi}(w_i)} = \prod_{i=1}^{N} p_{\psi, f}(w_i \mid d_i).$$

**B Proof of Theorem 1**

Before going into details, we first introduce a few notations, definitions, and a technical lemma.

- The entropy is defined as $H_p(x) := \mathbb{E}_{p(x)}[-\log p(x)];$
- The mutual information is given by $I_p(x; y) := D_{KL}(p(x, y) \mid \mid p(x)p(y)).$
- Very often, we decompose the mutual information as $I_p(x; y) = H_p(x) - H_p(x \mid y)$ with $H_p(x \mid y) := \mathbb{E}_{p(x, y)}[-\log p(x \mid y)]$ the conditional entropy.
- The cross entropy is defined as $H_{p,q}(x) := \mathbb{E}_{p(x)}[-\log q(x)].$ Similarly, we have the cross conditional entropy $H_{p,q}(x \mid y) := \mathbb{E}_{p(x, y)}[-\log q(x \mid y)]$ and the cross mutual information $I_{p,q}(x; y) = H_{p,q}(x) - H_{p,q}(x \mid y)$.

**Definition 1.** A random variable $x$ is subgaussian if there exists a positive number $\sigma$ such that $\mathbb{E}[\exp(\lambda (x - \mathbb{E} x))] \leq \exp(\sigma^2 \lambda^2 / 2)$ for all $\lambda \in \mathbb{R}$.

**Lemma 1 (Xu & Raginsky [2017]).** If $g(x, y)$ is $\sigma$-subgaussian, then

$$|\mathbb{E}_{p(x, y)} g(x, y) - \mathbb{E}_{p(x)p(y)} g(x, y)| \leq \sqrt{2\sigma^2 I_p(x; y)}.$$  

Now, we are ready to prove Theorem 1.

**Proof.** The expected KL divergence can be rewritten as follows:

$$\mathbb{E}_{q(d)} D_{KL}(q(w \mid d) \| p(w \mid d)) = \mathbb{E}_{q(d)} \mathbb{E}_{q(w \mid d)} \left[ \log \frac{q(w \mid d)p(d)q(w)}{p(d \mid w)p(w)q(w)} \right]$$

$$= \mathbb{E}_{q(d)} \mathbb{E}_{q(w \mid d)} \left[ \log \frac{q(w \mid d)}{q(w)} \right] + \mathbb{E}_{q(d)} \mathbb{E}_{q(w \mid d)} \left[ -\log p(d \mid w) \right]$$

$$+ \mathbb{E}_{q(w)} \left[ \log \frac{q(w)}{p(w)} \right] + \mathbb{E}_{q(d)} \mathbb{E}_{q(w \mid d)} \left[ \log p(d) \right]$$

$$= I_{q}(w; d) + H_{q,p}(d \mid w) + D_{KL}(q(w) \| p(w)) - H_{q,p}(d)$$

$$= I_{q}(w; d) - I_{q,p}(d; w) + D_{KL}(q(w) \| p(w))$$

$$\geq I_{q}(w; d) - I_{q,p}(d; w) = I_{q}(w; d) - I_{q,p}(w; d).$$

(20)

Note that $I_{q,p}(d; w) = I_{q,p}(w; d)$ and $D_{KL}(q(w) \| p(w)) = 0$ iff $q(w) = \mathbb{E}_{q(d)} q(w \mid d) = p(w).$

Similarly, we have

$$\mathbb{E}_{q(d)} D_{KL}(q(w \mid d) \| p(d \mid w)p(w)) \geq I_{q}(w; d) + H_{q,p}(d \mid w).$$

(22)
We conduct an experiment on CIFAR-FS using Conv-4-64 feature network. The results are shown in Table 3. It can be seen that there is no improvement over SIB (\( \text{gen}(q(w|d), q(d)) \)). To further verify the effectiveness of the synthetic gradient descent, we implement an inductive version of SIB inspired by MAML, where the initialization \( \theta_0 \) is generated exactly the same way as SIB using \( \lambda(d_i^0) \), but it then follows the iterations in (6) as in MAML rather than follows the iterations in [2] as in standard SIB.

We conduct an experiment on CIFAR-FS using Conv-4-64 feature network. The results are shown in Table 3. It can be seen that there is no improvement over SIB (\( K = 0 \)) suggesting that the inductive approach is insufficient.

\[ \text{D VARYING n} \]

We notice that changing the size of \( d_i \) (i.e., \( n \)) during training does make a difference on testing.
Table 3: Average 5-way classification accuracies (with 95% confidence intervals) with Conv-4-64 on the test set of CIFAR-FS. For each test, we sample 5000 episodes containing 5 categories (5-way) and 15 queries in each category. We report the results with using different learning rate \( \eta \) as well as different number of updates \( K \). Note that \( K = 0 \) is the performance only using the pre-trained feature.

<table>
<thead>
<tr>
<th>( K )</th>
<th>( \eta )</th>
<th>( K ) = 0</th>
<th>( K = 1 )</th>
<th>( K = 3 )</th>
<th>( K = 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
<td>1-shot</td>
<td>5-shot</td>
<td>1-shot</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td></td>
<td>59.7 ± 0.5%</td>
<td>75.5 ± 0.4%</td>
<td>59.2 ± 0.5%</td>
</tr>
<tr>
<td>1</td>
<td>1e-1</td>
<td></td>
<td>59.9 ± 0.5%</td>
<td>75.3 ± 0.4%</td>
<td>60.0 ± 0.6%</td>
</tr>
<tr>
<td>3</td>
<td>1e-2</td>
<td></td>
<td>59.6 ± 0.5%</td>
<td>74.9 ± 0.4%</td>
<td>60.1 ± 0.6%</td>
</tr>
<tr>
<td>5</td>
<td>1e-3</td>
<td></td>
<td>59.9 ± 0.5%</td>
<td>74.9 ± 0.4%</td>
<td>60.0 ± 0.6%</td>
</tr>
<tr>
<td>1</td>
<td>1e-4</td>
<td></td>
<td>59.6 ± 0.5%</td>
<td>74.9 ± 0.4%</td>
<td>60.1 ± 0.6%</td>
</tr>
<tr>
<td>3</td>
<td>1e-5</td>
<td></td>
<td>59.6 ± 0.5%</td>
<td>74.9 ± 0.4%</td>
<td>60.1 ± 0.6%</td>
</tr>
<tr>
<td>5</td>
<td>1e-6</td>
<td></td>
<td>59.6 ± 0.5%</td>
<td>74.9 ± 0.4%</td>
<td>60.1 ± 0.6%</td>
</tr>
</tbody>
</table>

Table 4: Average classification accuracies on the validation set and the test set of Mini-ImageNet with backbone Conv-4-128. We modify the number of query images, i.e., \( n \), for each episode to study the effect on generalization.

<table>
<thead>
<tr>
<th>( n )</th>
<th>5-way, 5-shot</th>
<th>5-way, 1-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Validation</td>
<td>Test</td>
</tr>
<tr>
<td>3</td>
<td>77.97 ± 0.34%</td>
<td>75.91 ± 0.66%</td>
</tr>
<tr>
<td>5</td>
<td>78.14 ± 0.35%</td>
<td>76.01 ± 0.66%</td>
</tr>
<tr>
<td>10</td>
<td>78.30 ± 0.35%</td>
<td>76.22 ± 0.66%</td>
</tr>
<tr>
<td>15</td>
<td>77.53 ± 0.35%</td>
<td>75.43 ± 0.67%</td>
</tr>
<tr>
<td>30</td>
<td>76.21 ± 0.35%</td>
<td>74.04 ± 0.67%</td>
</tr>
<tr>
<td>45</td>
<td>75.65 ± 0.36%</td>
<td>73.27 ± 0.66%</td>
</tr>
</tbody>
</table>