Uncertainty in Model-Agnostic Meta-Learning using Variational Inference

Cuong Nguyen
University of Adelaide

Thanh-Toan Do
University of Liverpool

Gustavo Carneiro
University of Adelaide

firstname.lastname@{adelaide.edu.au, liverpool.ac.uk}

Abstract

We introduce a new, rigorously-formulated Bayesian meta-learning algorithm that learns a probability distribution of model parameter prior for few-shot learning. The proposed algorithm employs a gradient-based variational inference to infer the posterior of model parameters for a new task. Our algorithm can be applied to any model architecture and can be implemented in various machine learning paradigms, including regression and classification. We show that the models trained with our proposed meta-learning algorithm are well calibrated and accurate, with state-of-the-art calibration and classification results on three few-shot classification benchmarks (Omniglot, mini-ImageNet and tiered-ImageNet), and competitive results in a multi-modal task-distribution regression.

1. Introduction

Machine learning, in particular deep learning, has thrived during the last decade, producing results that were previously considered to be infeasible in several areas. For instance, outstanding results have been achieved in speech and image understanding [1–4], and medical image analysis [5]. However, the development of these machine learning methods typically requires a large number of training samples to achieve notable performance. Such requirement contrasts with the human ability of quickly adapting to new learning tasks using few "training" samples. This difference may be due to the fact that humans tend to exploit prior knowledge to facilitate the learning of new tasks, while machine learning algorithms often do not use any prior knowledge (e.g., training from scratch with random initialisation [6]) or rely on weak prior knowledge to learn new tasks (e.g., training from pre-trained models [7]). This challenge has motivated the design of machine learning methods that can make more effective use of prior knowledge to adapt to new learning tasks using few training samples [8].

Such methods assume the existence of a latent distribu-tion over classification or regression tasks that share a common structure. This common structure means that solving many tasks can be helpful to solve a new task, sampled from the same task distribution, even if it contains a limited number of training samples. For instance, in *multi-task learning* [9], an agent simultaneously learns the shared representation of many related tasks and a main task that are assumed to come from the same domain. The extra information provided by this multi-task training tends to regularise the main task training, particularly when it contains few training samples. In *domain adaptation* [10, 11], a learner transfers the shared knowledge of many training tasks drawn from one or several source domains to perform well on tasks (with small training sets) drawn from a target domain. *Bayesian learning* [12] has also been explored, where prior knowledge is represented by a probability density function on the parameters of the visual classes' probability models. In *learning to learn* or *meta-learning* [13, 14], a meta-learner extracts relevant knowledge from many tasks learned in the past to facilitate the learning of new future tasks.

From the methods above, meta-learning currently produces state-of-the-art results in many benchmark few-shot learning datasets [15–22]. Such success can be attributed to the way meta-learning leverages prior knowledge from several training tasks drawn from a latent distribution of tasks, where the objective is to perform well on unseen tasks drawn from the same distribution. However, a critical issue arises with the limited amount of training samples per task combined with the fact that most of these approaches [15, 16, 18, 19, 23] do not try to estimate model uncertainty – this may result in overfitting. This issue has been recently addressed with Laplace approximation to estimate model uncertainty, involving the computationally hard estimation of a high-dimensional covariance matrix [24], and with variational Bayesian learning [20, 25] containing sub-optimal point estimate of model parameters and inefficient optimisation.

In this work, we propose a new variational Bayesian learning by extending model-agnostic meta-learning (MAML) [19] based on a rigorous formulation that is

efficient and does not require any point estimate of model parameters. In particular, compared to MAML [19], our approach explores probability distributions over possible values of meta-parameters, rather than having a fixed value. Learning and prediction using our proposed method are, therefore, more robust due to the perturbation of learnt meta-parameters that coherently explains data variability. Our evaluation shows that the models trained with our proposed meta-learning algorithm is at the same time well calibrated and accurate, with competitive results in terms of Expected Calibration Error (ECE) and Maximimum Calibration Error (MCE), while outperforming state-of-the-art methods in some few-shot classification benchmarks (Omniglot, mini-ImageNet and tiered-ImageNet).

2. Related Work

Meta-learning has been studied for a few decades [13, 14, 26], and recently gained renewed attention with the use of deep learning methods. As meta-learning aims at the unique ability of learning how to learn, it has enabled the development of training methods with limited number of training samples, such as few-shot learning. Some notable meta-learning approaches include memory-augmented neural networks [15], deep metric learning [18, 23], learning how to update model parameters [16, 27] and learning good prior [19] using gradient update. These approaches have generated some of the most successful meta-learning results, but they lack the ability to estimate model uncertainty. Consequently, their performances may suffer in uncertain environments and real world applications.

Bayesian meta-learning techniques have, therefore, been developed to incorporate uncertainty into model estimation. Among those, MAML-based meta-learning has attracted much of research interest due to the straightforward use of gradient-based optimisation. LLAMA [24] uses Laplace method to extend the point estimates made in MAML to Gaussian distributions to improve the robustness of the trained model, but the need to estimate and invert the Hessian matrix makes this approach computationally challenging, particularly for large-scale models used in deep learning. Variational inference (VI) addresses such scalability issue – remarkable examples of VI-based methods are PLATIPUS [25], BMAML [20] and the methods similar to our proposal, Amortised Bayesian Meta-Learning (ABML) [28] and VERSA [29] [1]. However, PLATIPUS optimises the lower bound of data prediction, leading to the need to approximate a joint distribution between the task-specific and meta parameters. This approximation complicates the implementation and requires a point estimate of the task-specific parameters to reduce the complexity of the

estimation of this joint distribution. Employing point estimate may, however, reduce its ability to estimate model uncertainty. BMAML uses a closed-form solution based on Stein Variational Gradient Descent (SVGD) that simplifies the task adaptation step, but it relies on the use of a kernel matrix, which increases its computational complexity. ABML uses the both train and validation subsets to update meta-parameters, potentially resulting in overfitting. VERSA takes a slightly different approach by employing an external neural network to learn the variational distribution for certain parameters, while keeping other parameters shared across all tasks. Another inference-based method is Neural Process [30] that employs the train-ability of neural networks to model a Gaussian-Process-like distribution over functions to achieve uncertainty quantification in few-shot learning. However, due to the prominent weakness of Gaussian Process that suffers from cubic complexity to data size, this might limit the scalability of Neural Process and makes it infeasible for large-scale datasets.

Our approach, in contrast, is a straightforward extension of MAML, which uses VI to model the distributions of task-specific parameters and meta-parameters, where we do not require the use of point estimate of any term, nor do we need to compute Hessian or kernel matrices or depend on an external network. Our proposed algorithm can be considered a rigorous and computationally efficient Bayesian meta-learning algorithm. A noteworthy non-meta-learning method that employs Bayesian methods is the neural statistician [31] that uses an extra variable to model data distribution within each task, and combines that information to solve few-shot learning problems. Our proposed algorithm, instead, does not introduce additional parameters, while still being able to extract relevant information from a small number of examples.

3. Methodology

In this section, we first define and formulate the few-shot meta-learning problem. We then describe MAML, derive our proposed algorithm, and mention the similarities and differences between our method and recently proposed meta-learning methods that are relevant to our proposal.

3.1. Few-shot Learning Problem Setup

While conventional machine learning paradigm is designed to optimise the performance on a single task, few-shot learning is trained on a set of conditional independent and identically distributed (i.i.d.) tasks given meta-parameters. We employ the notation of "task environment" [32], where tasks are sampled from an unknown task distribution $\mathcal{D}$ over a family of tasks. Each task $\mathcal{T}_i$ in this family is indexed by $i \in \{1, ..., T\}$ and consists of a support set $\{\mathcal{X}_i^{(t)}, \mathcal{Y}_i^{(t)}\}$ and a query set $\{\mathcal{X}_i^{(v)}, \mathcal{Y}_i^{(v)}\}$, with

[1] ABML [28] and VERSA [29] have been developed in parallel to our proposed VAMPIRE.
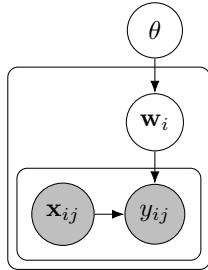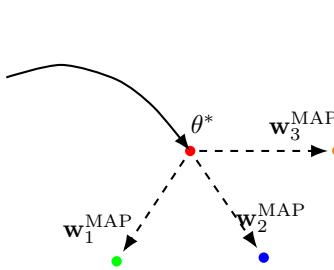
(a) Hierarchical graphical model   (b) MAML (reproduced from [19])   (c) VAMPIRE
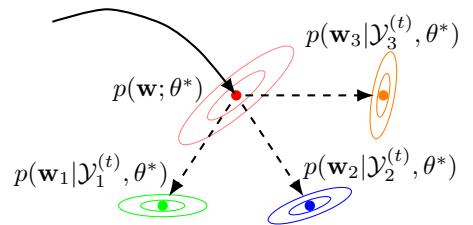
Figure 1: (a) Hierarchical graphical model of the few-shot meta-learning, where a prior parameterised by θ is shared across many tasks; (b) and (c) Visualisation between MAML and VAMPIRE, respectively, where VAMPIRE extends both the deterministic prior $p(\mathbf{w}_i; \theta)$ and posterior $p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)$ in MAML by using probabilistic distributions.

$\mathcal{X}_i^{(t)} = \{\mathbf{x}_{ij}^{(t)}\}_{j=1}^M$ and $\mathcal{Y}_i^{(t)} = \{y_{ij}^{(t)}\}_{j=1}^M$ ($\mathcal{X}_i^{(v)}$ and $\mathcal{Y}_i^{(v)}$ are similarly defined). The aim of few-shot learning is to predict the output $y_{ij}^{(v)}$ of the query input $\mathbf{x}_{ij}^{(v)}$ given the small support set for task $\mathcal{T}_i$ (e.g. $M \leq 20$). We rely on a Bayesian hierarchical model [24] to model the few-shot meta-learning problem. In the graphical model shown in Figure 1a, θ denotes the meta-parameters of interest, and $\mathbf{w}_i$ represents the task-specific parameters for task $\mathcal{T}_i$. One typical example of this modelling approach is MAML [19], where $\mathbf{w}_i$ are the neural network weights adapted to task $\mathcal{T}_i$ by performing truncated gradient descent using the data from the support set $\{\mathcal{X}_i^{(t)}, \mathcal{Y}_i^{(t)}\}$ and the initialisation θ.

The objective function of few-shot learning is, therefore, to find a meta-learner, parameterised by θ, across tasks sampled from $\mathcal{D}$, as follows:

$$\theta^* = \arg\min_\theta -\frac{1}{T}\sum_{i=1}^T \ln p(\mathcal{Y}_i^{(v)}|\mathcal{Y}_i^{(t)}, \theta), \quad (1)$$

where T denotes the number of tasks, and, hereafter, we simplify the notation by dropping the explicit dependence on $\mathcal{X}_i^{(t)}$ and $\mathcal{X}_i^{(v)}$ from the set of conditioning variables. Each term of the predictive probability on the right hand side of (1) can be expanded by applying the sum rule of probability and lower-bounded by Jensen's inequality:

$$\ln p(\mathcal{Y}_i^{(v)}|\mathcal{Y}_i^{(t)}, \theta) = \ln \mathbb{E}_{p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)}\left[p(\mathcal{Y}_i^{(v)}|\mathbf{w}_i)\right] \geq \mathcal{L}_i^{(v)},$$

where:

$$\mathcal{L}_i^{(v)}(\theta) = \mathbb{E}_{p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)}\left[\ln p(\mathcal{Y}_i^{(v)}|\mathbf{w}_i)\right]. \quad (2)$$

Hence, instead of minimising the negative log-likelihood in (1), we minimise the upper-bound of the corresponding negative log-likelihood which can be presented as:

$$\mathcal{L}^{(v)}(\theta) = -\frac{1}{T}\sum_{i=1}^T \mathcal{L}_i^{(v)}. \quad (3)$$

If each task-specific posterior, $p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)$, is well-behaved, we can apply Monte Carlo to approximate the expectation in (3) by sampling model parameters $\mathbf{w}_i$ from $p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)$. Thus, depending on the formulation of the task-specific posterior $p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)$, we can formulate different algorithms to solve the problem of few-shot learning. We review a deterministic method widely used in the literature in subsection 3.2, and present our proposed approach in subsection 3.3.

3.2. Point Estimate - MAML

A simple way is to approximate $p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta)$ by a Dirac delta function at its local mode:

$$p(\mathbf{w}_i|\mathcal{Y}_i^{(t)}, \theta) = \delta(\mathbf{w}_i - \mathbf{w}_i^{\mathrm{MAP}}), \quad (4)$$

where the local mode $\mathbf{w}_i^{\mathrm{MAP}}$ can be obtained by using maximum a posterior (MAP):

$$\mathbf{w}_i^{\mathrm{MAP}} = \arg\max_{\mathbf{w}_i} \ln p(\mathcal{Y}^{(t)}|\mathbf{w}_i) + \ln p(\mathbf{w}_i; \theta). \quad (5)$$

In the simplest case where the prior is also assumed to be a Dirac delta function: $p(\mathbf{w}_i; \theta) = \delta(\mathrm{w}_i - \theta)$, and gradient descent is used, the local mode can be determined as:

$$\mathbf{w}_i^{\mathrm{MAP}} = \theta - \alpha\nabla_{\mathbf{w}_i}\left[-\ln p(\mathcal{Y}_i^{(t)}|\mathbf{w}_i)\right], \quad (6)$$

where α is the learning rate, and the truncated gradient descent consists of a single step of (6) (the extension to a larger number of steps is trivial). Given the point estimate assumption in (4), the upper-bound of the negative log-likelihood in (3) can be simplified to:

$$\mathcal{L}^{(v)}(\theta) = \frac{1}{T}\sum_{i=1}^T -\ln p(\mathcal{Y}_i^{(v)}|\mathbf{w}_i^{\mathrm{MAP}}). \quad (7)$$

Minimising the upper-bound of the negative log-likelihood in (7) w.r.t. θ represents the MAML algorithm [19]. This derivation also explains the intuition behind MAML, which finds a good initialisation of model parameters as illustrated in Figure 1b.

3.3. Gradient-based Variational Inference

In contrast to the deterministic method presented in subsection 3.2, we use a variational distribution $q(\mathbf{w}_i; \lambda_i)$, parameterized by λ_i – a function of $\mathcal{Y}_i^{(t)}$ and θ), to approximate the task-specific posterior $p(\mathbf{w}_i | \mathcal{Y}_i^{(t)}, \theta)$. In variational inference, $q(\mathbf{w}_i; \lambda_i)$ can be obtained by minimising the following Kullback-Leibler (KL) divergence:

$$
\begin{aligned}
\lambda_i^* &= \arg \min_{\lambda_i} \text{KL} \left[q(\mathbf{w}_i; \lambda_i) \| p(\mathbf{w}_i | \mathcal{Y}_i^{(t)}, \theta) \right] \\
&= \arg \min_{\lambda_i} \int q(\mathbf{w}_i; \lambda_i) \ln \frac{q(\mathbf{w}_i; \lambda_i) p(\mathcal{Y}_i^{(t)} | \theta)}{p(\mathcal{Y}_i^{(t)} | \mathbf{w}_i) p(\mathbf{w}_i; \theta)} d\mathbf{w}_i \\
&= \arg \min_{\lambda_i} \mathcal{L}_i^{(t)}(\lambda_i, \theta) + \underbrace{\ln p(\mathcal{Y}^{(t)} | \theta)}_{\text{const. wrt } \lambda_i}.
\end{aligned}
\tag{8}
$$

where:

$$
\begin{aligned}
\mathcal{L}_i^{(t)}(\lambda_i, \theta) = {} & \text{KL} \left[q(\mathbf{w}_i; \lambda_i) \| p(\mathbf{w}_i; \theta) \right] \\
& + \mathbb{E}_{q(\mathbf{w}_i; \lambda_i)} \left[-\ln p(\mathcal{Y}_i^{(t)} | \mathbf{w}_i) \right].
\end{aligned}
\tag{9}
$$

The resulting cost function (excluding the constant term) $\mathcal{L}_i^{(t)}$ is often known as the variational free energy (VFE). The first term of VFE can be considered as a regularisation that penalises the difference between the prior $p(\mathbf{w}_i; \theta)$ and the approximated posterior $q(\mathbf{w}_i; \lambda_i)$, while the second term is referred as likelihood cost. Exactly minimising the cost function in (9) is computationally challenging, so gradient descent is used with θ as the initialisation of λ_i:

$$
\lambda_i \leftarrow \theta - \alpha \nabla_{\lambda_i} \mathcal{L}_i^{(t)}(\lambda_i, \theta),
\tag{10}
$$

where α is the learning rate.

Given the approximated posterior $q(\mathbf{w}_i; \lambda_i)$ with parameter λ_i updated according to (10), we can calculate and optimise the upper-bound in (3) to find a local-optimal meta-parameter θ.

In Bayesian statistics, the prior $p(\mathbf{w}_i | \theta)$ represents a modelling assumption, and the approximated posterior $q(\mathbf{w}_i; \lambda_i)$ is a flexible function that can be adjusted to achieve a good trade-off between performance and complexity. For simplicity, we assume that both $p(\mathbf{w}_i; \theta)$ and $q(\mathbf{w}_i; \lambda_i)$ are Gaussian distributions with diagonal covariance matrices:

$$
\begin{cases}
p(\mathbf{w}_i; \theta) &= \mathcal{N} \left[\mathbf{w}_i | \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta = \text{diag}(\boldsymbol{\sigma}_\theta^2) \right] \\
q(\mathbf{w}_i; \lambda_i) &= \mathcal{N} \left[\mathbf{w}_i | \boldsymbol{\mu}_{\lambda_i}, \boldsymbol{\Sigma}_{\lambda_i} = \text{diag}(\boldsymbol{\sigma}_{\lambda_i}^2) \right],
\end{cases}
\tag{11}
$$

where $\boldsymbol{\mu}_\theta, \boldsymbol{\mu}_{\lambda_i}, \boldsymbol{\sigma}_\theta, \boldsymbol{\sigma}_{\lambda_i} \in \mathbb{R}^d$, with d denoting the number of model parameters, and the operator $\text{diag}(.)$ returns a diagonal matrix of the vector in its input parameter.

Given the prior $p(\mathbf{w}_i | \theta)$ and the posterior $q(\mathbf{w}_i; \lambda_i)$ in (11), we can compute the KL divergence of VFE shown

---

Algorithm 1 VAMPIRE training

Require: task distribution $\mathcal{D}$
Require: Hyper-parameters: T, L_t, L_v, α and γ
1: initialise θ
2: **while** θ not converged **do**
3:     sample a mini-batch of tasks $\mathcal{T}_i \sim \mathcal{D}, i = 1 : T$
4:     **for** each task $\mathcal{T}_i$ **do**
5:         $\lambda_i \leftarrow \theta$
6:         draw L_t samples $\hat{\mathbf{w}}_i^{(l_t)} \sim q(\mathbf{w}_i; \lambda_i), l_t = 1 : L_t$
7:         update: $\lambda_i \leftarrow \lambda_i - \frac{\alpha}{L_t} \nabla_{\lambda_i} \mathcal{L}_i^{(t)}(\lambda_i, \theta)$ {Eq (10)}
8:         draw L_v samples $\hat{\mathbf{w}}_i^{(l_v)} \sim q(\mathbf{w}_i; \lambda_i), l_v = 1 : L_v$
9:         $\mathcal{L}_i^{(v)}(\theta) = \frac{1}{L_v} \sum_{l_v=1}^{L_v} \ln p \left(\mathcal{Y}_i^{(v)} | \hat{\mathbf{w}}_i^{(l_v)} \right)$ {Eq. (2)}
10:     **end for**
11:     meta-update: $\theta \leftarrow \theta + \frac{\gamma}{T} \nabla_\theta \sum_{i=1}^{T} \mathcal{L}_i^{(v)}(\theta)$
12: **end while**

---

in (9) by using either Monte Carlo sampling or a closed-form solution. According to [33], sampling model parameters from the approximated posterior $q(\mathbf{w}_i; \lambda_i)$ to compute the KL divergence term and optimise the cost function in (9) does not perform better or worse than using the closed-form of the KL divergence between two Gaussian distributions. Therefore, we employ the closed-form formula of the KL divergence to speed up the training process.

For numerical stability, we parameterise the standard deviation point-wisely as $\sigma = \exp(\rho)$ when performing gradient update for the standard deviations of model parameters. The meta-parameters $\theta = (\boldsymbol{\mu}_\theta, \exp(\boldsymbol{\rho}_\theta))$ are the initial mean and standard deviation of neural network weights, and the variational parameters $\lambda_i = (\boldsymbol{\mu}_{\lambda_i}, \exp(\boldsymbol{\rho}_{\lambda_i}))$ are the optimised mean and standard deviation of those network weights adapted to task $\mathcal{T}_i$. We also implement the re-parameterisation trick [34] when sampling the network weights from the approximated posterior to compute the expectation of the data log-likelihood in (9):

$$
\mathbf{w}_i = \boldsymbol{\mu}_{\lambda_i} + \epsilon \odot \exp(\boldsymbol{\rho}_{\lambda_i}),
\tag{12}
$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I}_d)$, and $\odot$ is the element-wise multiplication. Given this direct dependency, the gradients of the cost function $\mathcal{L}_i^{(t)}$ in (9) with respect to λ_i can be derived as:

$$
\begin{cases}
\nabla_{\mu_{\lambda_i}} \mathcal{L}_i^{(t)} &= \dfrac{\partial \mathcal{L}_i^{(t)}}{\partial \mathbf{w}_i} + \dfrac{\partial \mathcal{L}_i^{(t)}}{\partial \boldsymbol{\mu}_{\lambda_i}} \\[2ex]
\nabla_{\rho_{\lambda_i}} \mathcal{L}_i^{(t)} &= \dfrac{\partial \mathcal{L}_i^{(t)}}{\partial \mathbf{w}_i} \epsilon \odot \exp(\boldsymbol{\rho}_{\lambda_i}) + \dfrac{\partial \mathcal{L}_i^{(t)}}{\partial \boldsymbol{\rho}_{\lambda_i}}.
\end{cases}
\tag{13}
$$

After obtaining the variational parameters λ_i in (10), we can apply Monte Carlo approximation by sampling L_v sets of model parameters from the approximated posterior

$q(\mathbf{w}_i; \lambda_i)$ to calculate and optimise the upper-bound in (3) w.r.t. θ. This approach leads to the general form of our proposed algorithm, named Variational Agnostic Modelling that Performs Inference for Robust Estimation (VAMPIRE), shown in Algorithm 1.

3.4. Differentiating VAMPIRE and Other Bayesian Meta-learning Methods

VAMPIRE is different from the "probabilistic MAML" - PLATIPUS [25] in several ways. First, PLATIPUS uses VI to approximate the joint distribution $p(\mathbf{w}_i, \theta | \mathcal{Y}_i^{(t)}, \mathcal{Y}_i^{(v)})$, while VAMPIRE uses VI to approximate the task-specific posterior $p(\mathbf{w}_i | \mathcal{Y}_i^{(t)}, \theta)$. To handle the complexity of sampling from a joint distribution, PLATIPUS relies on the same point estimate of the task-specific posterior as MAML, as shown in (4). Second, to adapt to task $\mathcal{T}_i$, PLATIPUS learns only the mean, without varying the variance. In contrast, VAMPIRE learns both $\boldsymbol{\mu}_\theta$ and Σ_θ for each task $\mathcal{T}_i$. Lastly, when adapting to a task, PLATIPUS requires 2 additional gradient update steps, corresponding to steps 7 and 10 of Algorithm 1 in [25], while VAMPIRE needs only 1 gradient update step as shown in step 7 of Algorithm 1. Hence, VAMPIRE is based on a simpler formulation that does not rely on any point estimate, and it is also more flexible and efficient because it allows all meta-parameters to be learnt while performing less gradient update steps.

VAMPIRE is also different from the PAC-Bayes meta-learning method designed for multi-task learning [35] at the relation between the shared prior $p(\mathbf{w}_i; \theta)$ and the variational task-specific posterior $q(\mathbf{w}_i; \lambda_i)$. While the PAC-Bayes meta-learning method does not relate the "posterior" to the "prior" as in the standard Bayesian analysis, VAMPIRE relates these two probabilities through a likelihood function by performing a fixed number of gradient updates as shown in (10). Due to this discrepancy, the PAC-Bayes meta-learning needs to maintain all the task-specific posteriors, requiring more memory storage, consequently resulting in an un-scalable approach, especially when the number of tasks is very large. In contrast, VAMPIRE learns only the shared prior, and hence, is a more favourable method for large-scaled applications, such as few-shot learning.

Our proposed algorithm is different from BMAML [20] at the methods used to approximate task-specific posterior $p(\mathbf{w}_i | \mathcal{Y}_i^{(t)}, \theta)$: BMAML is based on SVGD, while VAMPIRE is based on a variant of amortised inference. Although SVGD is a non-parametric approach that allows a flexible variational approximation, its downsides are the computational complexity of kernel matrix, and high memory usage when increasing the number of particles. In contrast, our approach uses a straightforward VI using parametric functions, resulting in a simpler computational and memory-efficient approach. One advantage of BMAML compared

to our method in Algorithm 1 is the use of Chaser Loss, which may be an effective way of preventing overfitting. Nevertheless, in principle, we can also implement the same loss for our proposed algorithm.

VAMPIRE is different from ABML [28] at the data subset used to update the meta-parameters θ: whole data set of task $\mathcal{T}_i$ in ABML versus only the query subset $\{\mathcal{X}_i^{(v)}, \mathcal{Y}_i^{(v)}\}$ in VAMPIRE. This discrepancy is due to the differences in the objective function. In particular, ABML maximises the lower bound of marginal likelihood, while VAMPIRE maximises the predictive probability in (1). Moreover, when deriving a lower bound of marginal log-likelihood using VI [28, Derivation right before Eq. (1)], the variational distribution q must be strictly greater than zero for all θ and variational parameters. The assumption that approximates the variational distribution $q(\theta; \psi)$ by a Dirac delta function made in Amortised ML [28, Eq. (4)] is, therefore, arguable.

Another Bayesian meta-learning approach similar to VAMPIRE is VERSA [29]. The two methods are different at the methods modelling the parameters of interest θ. VAMPIRE relies on gradient update to relate the prior and posterior through likelihood function, while VERSA is based on an amortisation network to output the parameters of the variational distributions. To scale up to deep neural network models, VERSA models only the parameters of the last fully connected layer, while leaving other parameters as point estimates that are shared across all tasks. As a result, VAMPIRE is more flexible since it does not need to define which parameters are shared or not shared, nor does it require any additional network.

4. Experimental Evaluation

The goal of our experiments is to present empirical evaluation of VAMPIRE compared to state-of-art meta-learning approaches. Our experiments include both regression and few-shot classification problems. The experiments are carried out using the training procedure shown in Algorithm 1. All implementations of VAMPIRE use PyTorch [36][2].

4.1. Regression

We evaluate VAMPIRE using a multi-modal task distribution where half of the data is generated from sinusoidal functions, while the other half is from linear functions [25]. A detailed configuration of the problem setup, the model used, as well as additional visualisation results, can be referred to Supplementary Material A.

The results in Figures 2a and 2b show that VAMPIRE can effectively reason which underlying function generates the training data points as the predictions are all sinusoidal or linear. In addition, VAMPIRE is able to vary the predic-

---

[2]Code can be found at
https://github.com/cnguyen10/few_shot_meta_learning

(a) Sinusoidal data   (b) Linear data

(c) Reliability diagram   (d) ECE and MCE
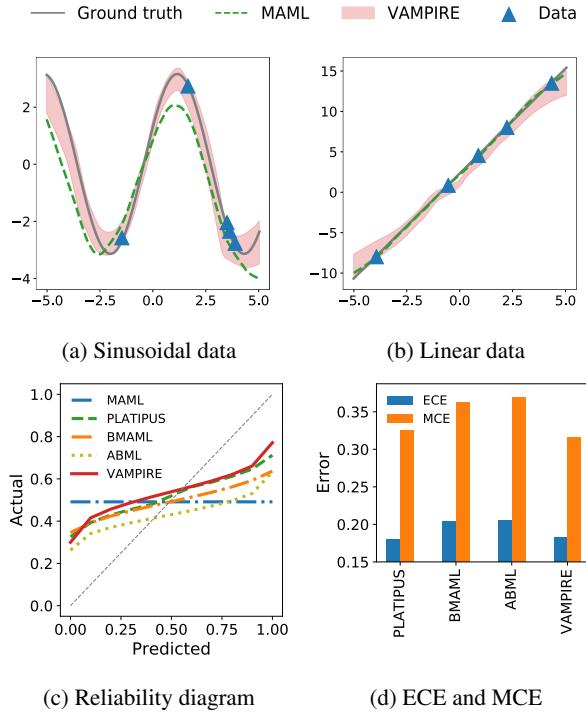
Figure 2: Qualitative and quantitative results on multi-modal data – half of the tasks are generated from sinusoidal functions, and the other half are from linear functions: (a) and (b) visualisation of MAML and VAMPIRE, where the shaded area is the prediction made by VAMPIRE $\pm 2\times$ standard deviation, (c) reliability diagram of various meta-learning methods averaged over 1000 tasks, and (d) ECE and MCE of the Bayesian meta-learning methods.

tion variance, especially when there is more uncertainty in the training data. In contrast, due to the deterministic nature, MAML can only output a single value at each input.

To quantitatively compare the performance between VAMPIRE and other few-shot meta-learning methods, we use the reliability diagram which is based on the quantile calibration for regression [37]. A model is perfectly calibrated when its predicted probability equals to the actual probability, resulting in a curve that is well-aligned with the diagonal $y = x$. We re-implement some few-shot meta-learning methods, train until convergence, and plot their reliability diagram for 1000 tasks in Figure 2c. To have a fair comparison, BMAML is trained without Chaser Loss, and ABML is trained with a uniform hyper-posterior. Due to the deterministic nature, the performance curve of MAML is presented as a horizontal line. For a further quantitative comparison, we also plot the expected calibration error (ECE), which averages the absolute errors measuring from the diagonal, and the maximum calibration error (MCE), which returns the maximum of absolute errors in Figure 2d.

Overall, in terms of ECE and MCE, the model trained with VAMPIRE is better than BMAML and ABML, while competitive with PLATIPUS. The performance of BMAML could be higher if more particles and Chaser Loss are used. Another observation is that ABML has slightly lower performance than MAML, although the training procedures of the two methods are very similar. We hypothesise that this is due to overfitting induced by using the whole training data subset that includes $\{\mathcal{X}_i^{(t)}, \mathcal{Y}_i^{(t)}\}$, while MAML and VAMPIRE use only the query data subset $\{\mathcal{X}_i^{(v)}, \mathcal{Y}_i^{(v)}\}$ to train the meta-parameters, which is consistent between the training and testing scenarios.

4.2. Few-shot Classification

The experiments in this sub-section are based on the N-way k-shot learning task, where a meta learner is trained on many related tasks containing N classes and small training sets of k samples for each class (i.e., this is the size of $\mathcal{Y}_i^{(t)}$). We benchmark our results against the state of the art on the data sets Omniglot [8], mini-ImageNet [16, 23] and tiered-ImageNet [38].

Omniglot contains 1623 different handwritten characters from 50 different alphabets, where each one of the characters was drawn online via Amazon's Mechanical Turk by 20 different people [8]. Omniglot is often split by randomly picking 1200 characters for training and the remaining for testing [16, 18, 19]. However, for language character classification, this random split may be unfair since knowing a character of an alphabet may facilitate the learning of other characters in the same alphabet. The original train-test split [8] suggests 30 alphabets for training and 20 alphabets for testing – such split clearly avoids potential information leakage from the training set to the testing set. We run experiments using both splits to compare with state-of-the-art methods and to perform testing without any potential data leakage. As standardly done in the literature, our training includes a data augmentation based on rotating the samples by multiples of 90 degrees [15]. Before performing experiments, all Omniglot images are down-sampled to 28-by-28 pixels to be consistent with the reported works in the meta-learning literature [16, 19, 23].

Mini-ImageNet [23] was proposed as an evaluation for few-shot learning. It consists of 100 different classes, each having 600 colour images taken from the original ImageNet data set [42]. We use the common train-test split that consists of 64 classes for training, 16 for validation, and 20 for testing [16]. Similarly to Omniglot, the examples in mini-ImageNet are pre-processed by down-sampling the images to 84-by-84 pixels to be consistent with previous works in the literature.

Tiered-ImageNet [38] is a larger subset of ImageNet that has 608 classes grouped into 34 high-level categories. We use the standard train-test split that consists of 20, 6, and 8

	5-WAY		20-WAY	
	1-SHOT	5-SHOT	1-SHOT	5-SHOT
OMNIGLOT [8] - ORIGINAL SPLIT, STANDARD 4-LAYER CNN				
MAML	**96.68 ± 0.57**	98.33 ± 0.22	84.38 ± 0.64	**96.32 ± 0.17**
VAMPIRE	96.27 ± 0.38	**98.77 ± 0.27**	**86.60 ± 0.24**	96.14 ± 0.10
OMNIGLOT [8] - RANDOM SPLIT, STANDARD 4-LAYER CNN				
MATCHING NETS [23]	98.1	98.9	93.8	98.5
PROTO. NETS [18] [2]	98.8	99.7	96.0	**98.9**
MAML [19]	**98.7 ± 0.4**	**99.9 ± 0.1**	**95.8 ± 0.3**	**98.9 ± 0.2**
VAMPIRE	98.43 ± 0.19	99.56 ± 0.08	93.20 ± 0.28	98.52 ± 0.13
OMNIGLOT [8] - RANDOM SPLIT, NON-STANDARD CNNS				
SIAMESE NETS [39]	97.3	98.4	88.2	97.0
NEURAL STATISTICIAN [31]	98.1	99.5	93.2	98.1
MEMORY MODULE [40]	98.4	99.6	95.0	98.6
RELATION NETS [41]	99.6 ± 0.2	**99.8 ± 0.1**	97.6 ± 0.2	**99.1 ± 0.1**
VERSA [29]	**99.70 ± 0.20**	99.75 ± 0.13	**97.66 ± 0.29**	98.77 ± 0.18

Table 1: Few-shot classification accuracy (in percentage) on Omniglot, tested on 1000 tasks and reported with 95% confidence intervals. The results of VAMPIRE are competitive to the state-of-the-art baselines which are carried out on a standard 4-convolution-layer neural networks. The top of the table contains methods trained on the original split defined in [8], while the middle part contains methods using a standard 4-layer CNN trained on random train-test split. The bottom part presents results of different methods using different network architectures, or requiring external modules and additional parameters trained on random split. Note that the Omniglot results on random split cannot be fairly compared.

categories for training, validation and testing. The experiments on tiered-ImageNet is carried with input as features extracted by a residual network that was pre-trained on data and classes from training meta-set [22, Section 4.2.2].

For Omniglot and mini-ImageNet, we use the same network architecture of state-of-the-art methods [16, 19, 23]. The network consists of 4 hidden convolution modules, each containing 64 3-by-3 filters, followed by batch normalisation [43], ReLU activation, and a 2-by-2 strided convolution. For the mini-ImageNet, the strided convolution is replaced by a 2-by-2 max-pooling layer, and only 32 filters are used on each convolution layer to avoid over-fitting [16, 19]. For tiered-ImageNet, we use a 2-hidden-layer fully-connected network with 128 and 32 hidden units. Please refer to Supplementary Material B for detailed description on the configuration and the hyperparameters used.

The N-way k-shot classification accuracy measured on Omniglot, and mini-ImageNet, tiered-ImageNet data sets are shown in Tables 1 and 2, respectively. Overall, the results of VAMPIRE are competitive to the state-of-the-art methods that use the same network architecture [16, 19, 23].

On Omniglot, our results on a random train-test split are competitive in most scenarios. Our proposed method outperforms some previous works in few-shot learning, such as

siamese networks [39], matching networks [23] and memory models [40], although they are designed with a focus on few-shot classification. Our result on the 20-way 1-shot is slightly lower than prototypical networks [18] and VERSA [29], but prototypical networks need more classes (higher "way") per training episode to obtain advantageous results and VERSA requires an additional amortised networks to learn the parameters of variational distributions. Our results are also slightly lower than MAML, potentially due to the difference of train-test split. To obtain a fair comparison, we run the public code provided by MAML's authors, and measure its accuracy on the original split suggested in [8]. Using this split, VAMPIRE achieves competitive performance, and outperforms MAML in some cases.

On mini-ImageNet, VAMPIRE outperforms all reported methods that use the standard 4-layer CNN architecture on the 1-shot tests, while being competitive on the 5-shot episodes. Prototypical Networks achieve a higher accuracy on the 5-shot tests due to, again, the use of extra classes during training. Although our work does not aim to achieve the state-of-the-art results in few-shot learning, we also run an experiment using input as features extracted by a residual network that was pre-trained on data and classes from training meta-set [22, Sect. 4.2.2]. The results, including the state-of-the-art methods that employ much deeper networks with various architectures, are presented in Supplementary

[2]Trained with 60-way episodes.

| | MINI-IMAGENET [16] | |
	1-SHOT	5-SHOT
STANDARD 4-BLOCK CNN		
MATCHING NETS [23]	43.56 ± 0.84	55.31 ± 0.73
META-LEARNER LSTM [16]	43.44 ± 0.77	60.60 ± 0.71
MAML [19]	48.70 ± 1.84	63.15 ± 0.91
PROTO. NETS [18] [3]	49.42 ± 0.78	$\mathbf{68.20 \pm 0.66}$
LLAMA [24]	49.40 ± 1.83	_
PLATIPUS [25]	50.13 ± 1.86	_
BMAML [20][4]	49.17 ± 0.87	64.23 ± 0.69
AMORTISED ML [28]	45.00 ± 0.60	_
VAMPIRE	$\mathbf{51.54 \pm 0.74}$	64.31 ± 0.74
	TIERED-IMAGENET [38]	
	1-SHOT	5-SHOT
DIFFERENT SETTINGS AND NETWORK ARCHITECTURES		
MAML [45]	51.67 ± 1.81	70.30 ± 0.08
PROTO. NETS [38]	53.31 ± 0.89	72.69 ± 0.74
RELATION NET [45]	54.48 ± 0.93	71.32 ± 0.78
TRNS. PRP. NETS [45]	57.41 ± 0.94	71.55 ± 0.74
LEO [22]	66.33 ± 0.05	81.44 ± 0.09
METAOPTNET [46]	65.81 ± 0.74	81.75 ± 0.53
VAMPIRE	$\mathbf{69.87 \pm 0.29}$	$\mathbf{82.70 \pm 0.21}$

Table 2: The few-shot 5-way classification accuracy results (in percentage) of VAMPIRE averaged over 600 mini-ImageNet tasks and 5000 tiered-ImageNet tasks are competitive to the state-of-the-art methods.

Material B. Note that deeper networks tend to reduce intra-class variation, resulting in a smaller gap of performance among many meta-learning methods [44].

On tiered-ImageNet, VAMPIRE outperforms many methods published previously by a large margin on both 1- and 5-shot settings.

To evaluate the predictive uncertainty of the models trained with different meta-learning methods, we show in Figure 3a the "normalised" reliability diagrams [47] which presents the absolute errors averaged over many unseen tasks. A perfectly calibrated model will have its "normalised" values overlapped with the y-axis, indicating that the probability associated with the label prediction is the same as the true probability. To have a fair comparison, we train all the methods of interest under the same configuration, e.g. network architecture, number of gradient updates, while keeping all method-specific hyper-parameters the same as the reported values. Due to the constrain of GPU memory, BMAML is trained with only 8 particles,

---

[3]Trained with 30-way episodes for 1-shot classification and 20-way episodes for 5-shot classification

[4]Produced locally
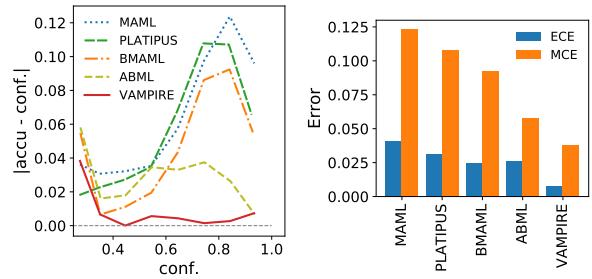
(a) "Normalised" reliability diagram

(b) ECE and MCE

Figure 3: (a) Uncertainty evaluation between different meta-learning methods using reliability diagrams, and (b) expected calibration error (ECE) and maximum calibration error (MCE), in which the evaluation is carried out on 5-way 1-shot setting for $\binom{20}{5} = 15504$ unseen tasks sampled from mini-ImageNet dataset.

while PLATIPUS, Amortimised Meta-learner and VAMPIRE are trained with 10 Monte Carlo samples. According to the reliability graphs, the model trained with VAMPIRE shows a much better calibration than the ones trained with the other methods used in the comparison. To further evaluate, we compute the expected calibration error (ECE) and maximum calibration error (MCE) [47] of each models trained with these methods. Intuitively, ECE is the weighted average error, while MCE is the largest error. The results plotted in Figure 3b show that the model trained with VAMPIRE has smaller ECE and MCE compared to all the state-of-the-art meta-learning methods. The slightly low performance of ABML might be due to the usage of the whole task-specific dataset, potentially overfitting to the training data. Another factor contributed might be the arguable Dirac-delta hyper-prior used, which can be also the cause for the low prediction accuracy shown in Table 2.

5. Conclusion

We introduce and formulate a new Bayesian algorithm used for few-shot meta-learning. The proposed algorithm, VAMPIRE, employs variational inference to optimise a well-defined cost function to learn a distribution of model parameters. The uncertainty, in the form of the learnt distribution, can introduce more variability into the decision made by the model, resulting in well-calibrated and highly-accurate prediction. The algorithm can be combined with different models that are trainable with gradient-based optimisation, and is applicable in regression and classification. We demonstrate that the algorithm can make reasonable predictions about unseen data in a multi-modal 5-shot learning regression problem, and achieve state-of-the-art calibration and classification results with only 1 or 5 training examples per class on public image data sets.

References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013*, IEEE, 2013, pp. 6645–6649.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[5] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle, "Brain tumor segmentation with deep neural networks," *Medical image analysis*, vol. 35, pp. 18–31, 2017.

[6] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[7] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *NIPS 2005 workshop on transfer learning*, vol. 898, 2005, pp. 1–4.

[8] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[9] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[10] J. S. Bridle and S. J. Cox, "Recnorm: Simultaneous normalisation and classification applied to speech recognition," in *Advances in Neural Information Processing Systems*, 1991, pp. 234–240.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[12] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[13] J. Schmidhuber, "Evolutionary principles in self-referential learning (on learning how to learn: The meta-meta-... hook)," Diploma thesis, Technische Universität München, 1987.

[14] S. Thrun and L. Pratt, *Learning to learn*. Springer Science & Business Media, 1998.

[15] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International Conference on Machine Learning*, 2016, pp. 1842–1850.

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

[17] T. Munkhdalai and H. Yu, "Meta networks," in *International Conference on Machine Learning*, 2017.

[18] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4077–4087.

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

[20] J. Yoon, T. Kim, O. Dia, S. Kim, Y. Bengio, and S. Ahn, "Bayesian model-agnostic meta-learning," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 7343–7353.

[21] R. Zhang, T. Che, Z. Ghahramani, Y. Bengio, and Y. Song, "Metagan: An adversarial approach to few-shot learning," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 2371–2380.

[22] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *International Conference on Learning Representations*, 2019.

[23] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 3630–3638.

[24] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths, "Recasting gradient-based meta-learning as hierarchical bayes," in *International Conference on Learning Representations*, 2018.

[25] C. Finn, K. Xu, and S. Levine, "Probabilistic model-agnostic meta-learning," in *Conference on Neural Information Processing Systems*, 2018, pp. 9537–9548.

[26] D. K. Naik and R. Mammone, "Meta-neural networks that learn by learning," in *International Joint Conference on Neural Networks*, IEEE, vol. 1, 1992, pp. 437–442.

[27] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," in *Advances in neural information processing systems*, 2016, pp. 3981–3989.

[28] S. Ravi and A. Beatson, "Amortized bayesian meta-learning," in *International Conference on Learning Representations*, 2019.

[29] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, and R. Turner, "Meta-learning probabilistic inference for prediction," in *International Conference on Learning Representations*, 2019.

[30] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh, "Neural processes," in *ICML workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018.

[31] H. Edwards and A. Storkey, "Towards a neural statistician," in *International Conference on Learning Representation*, 2017.

[32] J. Baxter, "A model of inductive bias learning," *Journal of Artificial Intelligence Research*, vol. 12, pp. 149–198, 2000.

[33] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *International Conference on Machine Learning*, 2015, pp. 1613–1622.

[34] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[35] R. Amit and R. Meir, "Meta-learning by adjusting priors based on extended PAC-Bayes theory," in *International Conference on Machine Learning*, 2018, pp. 205–214.

[36] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS Workshop on Automatic Differentiation*, 2017.

[37] H. Song, T. Diethe, M. Kull, and P. Flach, "Distribution calibration for regression," in *International Conference on Machine Learning*, Jun. 2019, pp. 5897–5906.

[38] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel, "Meta-learning for semi-supervised few-shot classification," in *International Conference on Learning Representation*, 2018.

[39] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, vol. 2, 2015.

[40] Ł. Kaiser, O. Nachum, A. Roy, and S. Bengio, "Learning to remember rare events," in *International Conference on Learning Representations*, 2017.

[41] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Conference on Computer Vision and Pattern Recognition*, 2018.

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Conference on Computer Vision and Pattern Recognition*, 2009.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

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

[45] Y. Liu, J. Lee, M. Park, S. Kim, E. Yang, S. J. Hwang, and Y. Yang, "Transductive propagation network for few-shot learning," in *International Conference on Learning Representation*, 2018.

[46] K. Lee, S. Maji, A. Ravichandran, and S. Soatto, "Meta-learning with differentiable convex optimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 657–10 665.

[47] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*, 2017.

[48] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," in *International Conference on Learning Representations*, 2018.

[49] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler, "Rapid adaptation with conditionally shifted neurons," in *International Conference on Machine Learning*, 2018, pp. 3661–3670.

[50] B. N. Oreshkin, A. Lacoste, and P. Rodriguez, "Tadam: Task dependent adaptive metric for improved few-shot learning," in *Conference on Neural Information Processing Systems*, 2018, pp. 719–729.