Meta-Learning Bayesian Neural Network Priors Based on PAC-Bayesian Theory

Anonymous Author(s)
Affiliation
Address
email

Abstract

Bayesian Neural Networks (BNNs) are a promising approach towards improved uncertainty quantification and sample efficiency. Due to their complex parameter space, choosing informative priors for BNNs is challenging. Thus, often a naive, zero-centered Gaussian is used, resulting both in bad generalization and poor uncertainty estimates when training data is scarce. In contrast, *meta-learning* aims to extract such prior knowledge from a set of related learning tasks. We propose a principled and scalable algorithm for meta-learning BNN priors based on PAC-Bayesian bounds. Whereas previous approaches require optimizing the prior and multiple variational posteriors in an interdependent manner, our method does not rely on difficult nested optimization problems. Our experiments show that the proposed method is not only computationally more efficient but also yields better predictions and uncertainty estimates when compared to previous meta-learning methods and BNNs with standard priors.

1 Introduction

Bayesian Neural Networks (BNNs) offer a probabilistic interpretation of deep learning by inferring distributions over the model's weights [30]. With the potential of combining the scalability and performance of neural networks (NNs) with a framework for uncertainty quantification, BNNs have lately received increased attention [7, 14]. However, BNNs face two major issues: 1) the intractability of posterior inference and 2) the difficulty of choosing good Bayesian priors. While the former has been addressed in an extensive body of literature on variation inference [e.g. 6, 7, 26, 29], the latter has only received limited attention [17, 42]. Choosing an informative prior for BNNs is particularly difficult due to the high-dimensional and hardly interpretable parameter space of NNs. Due to the lack of good alternatives, often a zero-centered, isotropic Gaussian is used, reflecting (almost) no a priori knowledge about the problem at hand. This does not only lead to poor generalization when data is scarce, but also renders the Bayesian uncertainty estimates poorly calibrated [23].

Meta-learning [36, 41] acquires inductive bias in a data-driven way, constituting an alternative route to addressing this issue. In particular, meta-learners attempt to extract shared (prior) knowledge from a set of learning tasks (i.e., datasets), aiming to learn in the face of a new, related task [3, 11, 15]. Our work develops a principled and scalable algorithm for meta-learning BNN priors. We build on the PAC-Bayesian framework, a methodology from statistical learning theory for deriving generalization bounds [1, 9, 16, 27]. Previous PAC-Bayesian bounds for meta-learners [2, 32] require solving a difficult optimization problem, involving the optimization of the prior as well as multiple variational posteriors in a nested manner. Aiming to overcome this issue, we present a *PAC-Bayesian bound that does not rely on nested optimization* and, unlike [35], can be *tractably optimized for BNNs*. This makes the resulting meta-learner, referred to as *PACOH-NN*, not only much more *computationally efficient and scalable* than previous approaches for meta-learning BNN priors [2], but also *agnostic to the choice of approximate posterior inference method* which allows us to combine it freely with recent advances in MCMC [e.g. 10] or variational inference [e.g. 43].

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

Our experiments demonstrate that the computational advantages of *PACOH-NN* do not result in degraded predictive performance. In fact, across several regression and classification environments, *PACOH-NN* achieves a *comparable or better predictive accuracy* than several popular meta-learning approaches, while *improving the quality of the uncertainty estimates*. Finally, we show how meta-learned *PACOH-NN* priors can be used in a real-world bandit task concerning vaccine development, suggesting that many other challenging real-world problems may benefit from our approach.

2 Background: The PAC-Bayesian Framework

Bayesian Neural Networks. Consider a supervised learning task with data $S = \{(x_j, y_j)\}_{j=1}^m$ drawn from the distribution $\mathcal{D}$. In that, $\mathbf{X} = \{x_j\}_{j=1}^m \in \mathcal{X}^m$ denotes training inputs and $\mathbf{Y} = \{y_j\}_{j=1}^m \in \mathcal{Y}^m$ the targets. For brevity, we also write $z_j := (x_j, y_j) \in \mathcal{Z}$. Let $h_\theta : \mathcal{X} \to \mathcal{Y}$ be a function parametrized by a NN with weights $\theta \in \Theta$. Using the NN mapping, we define a conditional distribution $p(y|x, \theta)$, usually referred to as likelihood. For Bayesian inference, one presumes a prior distribution $p(\theta)$ over the model parameters θ which is combined with the training data S into a posterior distribution $p(\theta|\mathbf{X}, \mathbf{Y}) \propto p(\theta)p(\mathbf{Y}|\mathbf{X}, \theta)$. For unseen test data points x^*, we form the predictive distribution as $p(y^*|x^*, \mathbf{X}, \mathbf{Y}) = \int p(y^*|x^*, \theta)p(\theta|\mathbf{X}, \mathbf{Y})d\theta$. Due to the difficulties in choosing an appropriate BNN prior, the prior is typically strongly misspecified [39]. As a result, modulating the influence of the prior relative to the likelihood during inference typically improves the empirical performance of BNNs and is thus a common practice [44]. Using such a "tempered" posterior $p_\tau(\theta|\mathbf{X}, \mathbf{Y}) \propto p(\theta)p(\mathbf{Y}|\mathbf{X}, \theta)^\tau$ with $\tau > 0$ is also referred to as *generalized Bayesian learning* [18].

The PAC-Bayesian Framework. Given a loss function $l : \theta \times \mathcal{Z} \to \mathbb{R}$, we typically want to minimize the *generalization error* $\mathcal{L}(\theta, \mathcal{D}) = \mathbb{E}_{z^* \sim \mathcal{D}}\ l(\theta, z^*)$. Since $\mathcal{D}$ is unknown, the *empirical error* $\hat{\mathcal{L}}(\theta, S) = \frac{1}{m}\sum_{i=1}^m l(\theta, z_i)$ is usually employed, instead. In the PAC-Bayesian framework, we are concerned with *randomized predictors*, i.e., probability measures on the parameter space Θ, allowing us to reason about *epistemic uncertainty*. In particular, we consider two such probability measures, a *prior* $P \in \mathcal{M}(\mathcal{H})$ and a *posterior* $Q \in \mathcal{M}(\Theta)$. In here, by $\mathcal{M}(\Theta)$, we denote the set of all probability measures on Θ. Unlike in Bayesian inference, where prior and posterior are tightly coupled, the PAC-Bayesian framework only requires the prior to be independent of S. Using the definitions above, the so-called *Gibbs error* for a randomized predictor Q is defined as $\mathcal{L}(Q, \mathcal{D}) = \mathbb{E}_{h \sim Q}\ \mathcal{L}(h, \mathcal{D})$. Similarly, we define its empirical counterpart as $\hat{\mathcal{L}}(Q, S) = \mathbb{E}_{h \sim Q}\ \hat{\mathcal{L}}(h, S)$. The PAC-Bayesian framework provides upper bounds for the unknown *Gibbs error* in the following form:

Theorem 1. *[1] Given a data distribution $\mathcal{D}$, a prior $P \in \mathcal{M}(\Theta)$, a confidence level $\delta \in (0, 1]$, with probability at least $1 - \delta$ over samples $S \sim \mathcal{D}^m$, we have:*

$$\forall Q \in \mathcal{M}(\Theta): \quad \mathcal{L}(Q, \mathcal{D}) \leq \hat{\mathcal{L}}(Q, S) + \frac{1}{\sqrt{m}}\left[D_{KL}(Q||P) + \ln\frac{1}{\delta} + \Psi(\sqrt{m})\right] \tag{1}$$

where $\Psi(\sqrt{m}) = \ln \mathbb{E}_{\theta \sim P}\mathbb{E}_{S \in \mathcal{D}^m} \exp\left[\sqrt{m}\left(\mathcal{L}(\theta, \mathcal{D}) - \hat{\mathcal{L}}(\theta, S)\right)\right]$.

With additional assumptions about the loss, we can bound $\Psi(\sqrt{m})$ and obtain tractable bounds. For unbounded loss functions like the negative log-likelihood, it is common to assume bounded moments. For instance, a loss is considered *sub-gamma* with variance factor s^2 and scale parameter c, under a prior P and data distribution $\mathcal{D}$, if its deviations from the mean can be characterized by random variable $V := \mathcal{L}(h, \mathcal{D}) - l(h, z)$ whose moment generating function is upper bounded by that of a Gamma distribution $\Gamma(s, c)$ [8]. In such case, we can bound $\Psi(\sqrt{m}) \leq s^2/(2 - \frac{2c}{\sqrt{m}}))$.

Connecting the PAC-Bayesian framework and generalized Bayesian learning. In PAC-Bayesian learning we aim to find the posterior that minimizes the bound in (1) which is in general a challenging optimization problem over the space of measures $\mathcal{M}(\Theta)$. However, to our benefit, it can be shown that the *Gibbs posterior* is the probability measure that minimizes (1). For details we refer to Lemma 2 in the Appendix or [9]. In particular, this gives us

$$Q^*(\theta) := \operatorname*{arg\,min}_{Q \in \mathcal{M}(\Theta)} \sqrt{m}\hat{\mathcal{L}}(Q, S) + D_{KL}(Q||P) = P(\theta)e^{-\sqrt{m}\hat{\mathcal{L}}(\theta, S)}/Z(S, P)\ , \tag{2}$$

where $Z(S, P)$ is a normalization constant. In a probabilistic setting, our loss function is the negative log-likelihood, i.e. $l(\theta, z_i) := -\log p(z_i|\theta)$. In this case, the optimal Gibbs posterior coincides with the *generalized Bayesian posterior* $Q^*(\theta; P, S) \propto P(\theta)p(S|\theta)^{1/\sqrt{m}}/Z(S, P)$ where $Z(S, P) = \int_\Theta P(\theta)p(S|\theta)^{1/\sqrt{m}}\, d\theta$ is the *generalized marginal likelihood* of the data sample S.

3 PAC-Bayesian bounds for Meta-Learning

In the standard learning setup (see Sec. 2), the learner has prior knowledge in form of a prior distribution P that is given exogenously. When the learner faces a new task, it uses the evidence, provided as a dataset S, to update the prior into a posterior Q. We formalize such a *base learner* $Q(S, P)$ as a mapping $Q : \mathcal{Z}^m \times \mathcal{M}(\mathcal{H}) \to \mathcal{M}(\mathcal{H})$ that takes in a dataset and prior and outputs a posterior.

In contrast, in meta-learning we acquire such prior P in a data-driven manner, i.e., by consulting a set of n statistically related learning tasks $\{\tau_1, ..., \tau_n\}$. We follow the setup of [4] in which all tasks $\tau_i := (\mathcal{D}_i, S_i)$ share the same data domain $\mathcal{Z}$, parameter space Θ and loss function $l(\theta, z)$, but may differ in their data distribution $\mathcal{D}_i$ and the number of points m_i in the corresponding dataset S_i. For our theoretical expositions, we assume w.l.o.g that $m = m_i \ \forall i$. Further, each task $\tau_i \sim \mathcal{T}$ is drawn i.i.d. from an *environment* $\mathcal{T}$, a probability distribution over data distributions and datasets. The goal is to extract knowledge from the observed datasets which can then be used as a prior for learning on new target tasks $\tau \sim \mathcal{T}$ [2]. To extend the PAC-Bayesian analysis to the meta-learning setting, we again consider the notion of probability distributions over hypotheses/parameters. However, while the object of learning has previously been the NN parameters θ, it is now the prior distribution $P \in \mathcal{M}(\Theta)$. Accordingly, the meta-learner presumes a *hyper-prior* $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$, i.e., a distribution over priors P. Combining the hyper-prior $\mathcal{P}$ with the data sets $S_1, ..., S_n$ from multiple tasks, the meta-learner then outputs a *hyper-posterior* $\mathcal{Q}$ over priors. The hyper-posterior's performance/quality is measured as the expected Gibbs error incurred when sampling priors P from $\mathcal{Q}$ and applying the base learner, the so-called transfer-error: $\mathcal{L}(\mathcal{Q}, \mathcal{T}) := \mathbb{E}_{P\sim\mathcal{Q}}\left[\mathbb{E}_{(\mathcal{D},m)\sim\mathcal{T}}\left[\mathbb{E}_{S\sim\mathcal{D}^m}\left[\mathcal{L}(Q(S, P), \mathcal{D})\right]\right]\right]$. While the transfer error is unknown in practice, we can estimate it using the *empirical multi-task error* $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) := \mathbb{E}_{P\sim\mathcal{Q}}\frac{1}{n}\sum_{i=1}^{n}\hat{\mathcal{L}}\left(Q(S_i, P), S_i\right)$. Similar to Theorem 1, we can bound the transfer error by its empirical counterpart $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$ plus several tractable complexity terms:

Theorem 2. *Let $Q : \mathcal{Z}^m \times \mathcal{M}(\mathcal{H}) \to \mathcal{M}(\mathcal{H})$ be a base learner and $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ some arbitrary but fixed hyper-prior. For all hyper-posteriors $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ and $\delta \in (0, 1]$,*

$$\begin{aligned}\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \left(\frac{1}{n\sqrt{m}} + \frac{1}{\sqrt{n}}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) \\ + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{m}}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q(S_i, P)||P)\right] + C(\delta, n, m)\end{aligned} \tag{3}$$

holds with probability $1 - \delta$. If the loss is bounded in $[a, b]$, we have $C(\delta, n, m) = \frac{(b_k-a_k)^2}{8\sqrt{m}} + \frac{(b_k-a_k)^2}{8\sqrt{n}} - \frac{1}{\sqrt{n}}\ln\delta$. If the loss is sub-gamma with variance factor s_I^2 and scale parameter c_I for $\mathcal{D}_i$ and s_{II}^2, c_{II} for $\mathcal{T}$, the inequality holds with $C(\delta, n, m) = \frac{s_I^2}{2(\sqrt{m}-c_I)} + \frac{s_{II}^2}{2(\sqrt{n}-c_{II})} - \frac{1}{\sqrt{n}}\ln\delta$.

Under bounded loss assumption, Theorem 2 provides a structurally similar, but slightly tighter bound than [32, 35]. Unlike [2, 32], the theorem also provides guarantees for *unbounded* loss functions under moment constraints (see Appendix A.1 for details). This makes Theorem 2 particularly relevant for probabilistic models such as BNNs in which the loss function coincides with the inherently unbounded negative log-likelihood. [2] propose to meta-learn NN priors by directly minimizing a bound similar to (3). However, the posterior inference for BNNs, i.e. obtaining $Q_i = Q(S_i, P)$, is an optimization problem in itself whose solution in turn depends on P. Hence, minimizing such meta-level bound w.r.t. P constitutes a computationally infeasible two-level optimization problem. To circumvent this issue, they jointly optimize P and n approximate posteriors Q_i that correspond to the different datasets S_i, leading to an unstable and poorly scalable meta-learning algorithm.

To overcome these issues, we employ the methodology of [35] and assume the Gibbs posterior $Q^*(S_i, P)$ as a base learner. Since $Q^*(S_i, P)$ minimizes PAC-Bayesian single task bound in Theorem 1, the resulting meta-level bound in Corollary 1 is tighter than (3) and can be stated in terms of the partition function $Z(S_i, P)$ which allows us to forgo the explicit reliance on the task posteriors Q_i. This makes the bound much easier to optimize as a meta-learning objective than previous bounds [e.g. 2, 32], since it no longer constitutes a two-level optimization problem.

Corollary 1. *When using the Gibbs posterior $Q^*(S_i, P) := P(\theta)\exp(-\sqrt{m}\hat{\mathcal{L}}(S_i, \theta))/Z(S_i, P)$ as a base learner, under the same assumptions as in Theorem 2, it holds with probability $1 - \delta$ that*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{m}}\mathbb{E}_{P\sim\mathcal{Q}}\left[\ln Z(S_i, P)\right] + \left(\frac{1}{\sqrt{n}} + \frac{1}{n\sqrt{m}}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta, n, m) \tag{4}$$

wherein $Z(S_i, P) = \mathbb{E}_{\theta\sim P}\left[\exp(-\sqrt{m}\hat{\mathcal{L}}(S_i,\theta))\right]$ *is the generalized marginal log-likelihood.*

A natural way to obtain a PAC-Bayesian meta-learning algorithm could be to minimize (4) w.r.t. $\mathcal{Q}$. Though, in general, this is a hard problem since it would require a minimization over $\mathcal{M}(\mathcal{M}(\mathcal{H}))$, the space of all probability measures over priors. Following [35], we exploit once more the insight that the minimizer of (4) can be written as Gibbs distribution (c.f. Lemma 2), allowing us to to derive such minimizing hyper-posterior $\mathcal{Q}^*$, i.e. the *PACOH*, in closed form:

Proposition 1. *(PAC-Optimal Hyper-Posterior) Given a hyper-prior $\mathcal{P}$ and data sets $S_1, ..., S_n$, the hyper-posterior $\mathcal{Q}$ that minimizes the PAC-Bayesian meta-learning bound in (4) is given by*

$$\mathcal{Q}^*(P) = \frac{\mathcal{P}(P)\exp\left(\frac{1}{\sqrt{nm}+1}\sum_{i=1}^n \ln Z(S_i,P)\right)}{Z^{II}(S_1,...,S_n,\mathcal{P})} \tag{5}$$

with the partition function defined as $Z^{II} = \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{1}{\sqrt{nm}+1}\sum_{i=1}^n \ln Z(S_i,P)\right)\right]$.

We refer to $\mathcal{Q}^*(P)$ as *PAC-optimal* since, among all meta-learners, it gives us the best possible PAC-Bayesian guarantees induced by Theorem 2.

4 PACOH-NN: A scalable algorithm for learning BNN priors

We discuss our main contribution, translating the *PACOH* (Prop. 1) into a practical algorithm for meta-learning BNN priors. To this end, we first specify various components of the generic meta-learning setup presented in Sec. 3 and then discuss how to obtain a tractable approximation of $\mathcal{Q}^*$.

First, we define a family of priors $\{P_\phi : \phi \in \Phi\}$ over the NN parameters θ. For computational convenience, we employ Gaussian priors with diagonal covariance matrix, i.e. $P_\phi = \mathcal{N}(\mu_P, \text{diag}(\sigma_P^2))$ with $\phi := (\mu_P, \log\sigma_P)$. Note that we represent σ_P in the log-space to avoid additional positivity constraints. In fact, any parametric distribution such as normalizing flows [34] that allows for re-parametrized sampling and has a tractable log-density could be used. As typical in the Bayesian framework, our loss function is the negative log-likelihood, i.e. $l(\theta, z) = -\ln p(y|x,\theta)$ for which we assume an additive Gaussian noise model $p(y|x,\theta) = \mathcal{N}(y; h_\theta(x), \sigma_y^2)$ in regression and a categorigal softmax distribution in case of classification. Moreover, we use a zero-centered, spherical Gaussian $\mathcal{P} := \mathcal{N}(0, \sigma_{\mathcal{P}}^2 I)$ as a hyper-prior over the parameters ϕ that specify the prior. In our setup, the hyper-prior acts a form of meta-level regularization that penalizes complex priors.

Given the hyper-prior and (level-I) log-partition function $\ln Z(S_i, P)$, we can compute the PACOH $\mathcal{Q}^*$ up to the normalization constant Z^{II}. Such a setup lends itself to approximate inference [6]. In particular, we employ Stein Variational Gradient Descent (SVGD) [26] which approximates $\mathcal{Q}^*$ as a set of particles $\hat{\mathcal{Q}} = \{\phi_1, ..., \phi_K\}$. Initially, the particles ϕ_K (prior parameters) are sampled randomly from $\mathcal{P}$. Then, the method iteratively transports the particles to match $\mathcal{Q}^*$, by applying a form of functional gradient descent, minimizing $D_{KL}(\hat{\mathcal{Q}}|\mathcal{Q}^*)$ in the reproducing kernel Hilbert space induced by a kernel $k(\cdot,\cdot)$. In each iteration, the particles are updated by $\phi_k \leftarrow \phi_k + \eta\psi^*(\phi_k)$ with

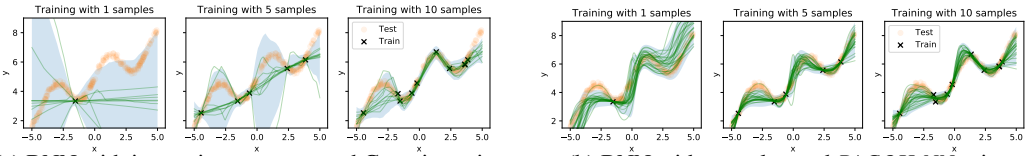
$$\psi^*(\phi) = \frac{1}{K}\sum_{k'=1}^K \left[k(\phi_{k'},\phi)\nabla_{\phi_{k'}}\ln\mathcal{Q}^*(\phi_{k'}) + \nabla_{\phi_{k'}}k(\phi_{k'},\phi)\right] \tag{6}$$

where η is the step size and $\nabla_{\phi_k}\ln\mathcal{Q}^*(\phi_k) = \nabla_{\phi_k}\ln\mathcal{P}(\phi_k) + \frac{1}{\sqrt{n\tilde{m}}+1}\sum_{i=1}^n \nabla_{\phi_k}\ln Z(S_i, P_{\phi_k})$ the score of $\mathcal{Q}^*$. The last remaining issue towards a viable meta-learning algorithm is the intractable generalized marginal likelihood in $\ln Z(S_i, P_\phi) = \ln\mathbb{E}_{\theta\sim P_\phi} e^{-\sqrt{m_i}\hat{\mathcal{L}}(\theta,S)}$. Estimating and optimizing $\ln Z(S_i, P_\phi)$ is not only challenging due to the high-dimensional expectation over Θ but also due to numerical instabilities inherent in computing $e^{-\sqrt{m_i}\hat{\mathcal{L}}(\theta,S)}$ when m is large. Aiming to overcome these issues, we compute numerically stable Monte Carlo estimates of $\nabla_\phi \ln Z(S_i, P_{\phi_k})$ by using the LogSumExp together with the re-parametrization trick [19]. In particular, we draw L samples $\theta_l := f(\phi, \epsilon_l) = \mu_P + \sigma_P \odot \epsilon_l,\ \epsilon_l j \sim N(0, I)$ and compute the forward pass as follows:

$$\ln\tilde{Z}(S_i, P_\phi) := \ln\frac{1}{L}\sum_{l=1}^L e^{-\sqrt{m_i}\hat{\mathcal{L}}(\theta_l,S_i)} = \text{LSE}_{l=1}^L\left(-\sqrt{m_i}\hat{\mathcal{L}}(\theta_l,S_i)\right) - \ln L\ ,\ \ \theta_l\sim P_\phi \tag{7}$$

The corresponding gradients follow as softmax-weighted average of score gradients:

$$\nabla_\phi \ln\tilde{Z}(S_i, P_\phi) = -\sqrt{m_i}\sum_{l=1}^L \underbrace{\frac{e^{-\sqrt{m_i}\hat{\mathcal{L}}(\theta_l,S_i)}}{\sum_{l=1}^L e^{-\sqrt{m_i}\hat{\mathcal{L}}(\theta_l,S_i)}}}_{\text{softmax}} \underbrace{\nabla_\phi f(\phi,\epsilon_l)^\top}_{\text{Jacobian}} \underbrace{\nabla_{\theta_l}\hat{\mathcal{L}}(\theta_l,S_i)}_{\text{score}} \tag{8}$$



(a) BNN with isotropic, zero-centered Gaussian prior (b) BNN with meta-learned *PACOH-NN* prior

Figure 1: BNN posterior predictions with (a) standard Gaussian prior vs. (b) meta-learned prior. Meta-learning with *PACOH-NN-SVGD* was conducted on the *Sinusoids* environment.

	Root-Mean-Squared Error (RMSE)		Accuracy	Calibration error		
	SwissFEL	Physionet	Omniglot	SwissFEL	Physionet	Omniglot
BNN [26]	0.529 ± 0.022	2.664 ± 0.274	0.795 ± 0.006	0.085 ± 0.008	0.277 ± 0.013	0.135 ± 0.009
PACOH-NN-SVGD (ours)	$\mathbf{0.372 \pm 0.002}$	$\mathbf{1.561 \pm 0.061}$	$\mathbf{0.885 \pm 0.090}$	$\mathbf{0.027 \pm 0.003}$	$\mathbf{0.267 \pm 0.005}$	0.091 ± 0.010
PACOH-NN-MAP (ours)	$\mathbf{0.375 \pm 0.004}$	$\mathbf{1.564 \pm 0.200}$	0.866 ± 0.005	0.031 ± 0.003	$\mathbf{0.268 \pm 0.015}$	$\mathbf{0.075 \pm 0.006}$
MLAP-M [2]	0.492 ± 0.009	2.232 ± 0.261	0.804 ± 0.0168	0.104 ± 0.015	0.339 ± 0.012	0.190 ± 0.019
MLAP-S [2]	0.486 ± 0.026	2.009 ± 0.248	0.700 ± 0.0135	0.090 ± 0.021	0.343 ± 0.017	0.108 ± 0.010
FO-MAML [31]	0.897 ± 0.071	2.545 ± 0.615	0.804 ± 0.0168	N/A	N/A	N/A
MAML [11]	0.730 ± 0.057	1.895 ± 0.141	0.693 ± 0.013	N/A	N/A	N/A
NPs [15]	0.471 ± 0.053	2.056 ± 0.209	N/A	0.131 ± 0.056	0.299 ± 0.012	N/A

Table 1: Comparison of meta-learning methods in terms of RMSE / accuracy and calibration error on the SwissFEL and Physionet regression as well as the Omniglot classification environment.

Note that $\ln \tilde{Z}(S_i, P_\phi)$ is not an unbiased estimator of $\ln Z(S_i, P_\phi)$. However, in expectation, the optimization procedure that uses the Monte Carlo estimate $\ln \tilde{Z}(S_i, P_\phi)$ still corresponds to minimization of a valid upper bound on the transfer error (see Appendix B). Algorithm 1 in Appendix B summarizes the proposed meta-learning method which we henceforth refer to as *PACOH-NN*.

5 Experiments

We empirically evaluate the method introduced in Section 4, in particular, two variants thereof: *PACOH-NN-SVGD* with $K = 5$ priors as particles and the edge case $K = 1$ which coincides with a maximum-a-posteriori (MAP) approximation of $\mathcal{Q}^*$, thus denoted as *PACOH-NN-MAP*. Comparing it to various NN-based meta-learning approaches on regression and classification environments, we demonstrate that *PACOH-NN* (i) outperforms previous meta-learning algorithms in terms of predictive accuracy, (ii) improves the quality of uncertainty estimates and (iii) is much more scalable than previous PAC-Bayesian meta-learners. Finally, we showcase how meta-learned *PACOH-NN* priors can be harnessed in a real-world bandit task concerning peptide-based vaccine development.

5.1 Meta-Learning BNN priors for regression and classification

Figure 1 illustrates *BNN* predictions on a sinusoidal regression task with a standard Gaussian prior as well as a *PACOH-NN* prior, meta-learned on various sinusoidal functions. In Figure 1a we can see that the standard Gaussian prior provides poor inductive bias, not only leading to bad mean predictions away from the testing points but also to poor confidence intervals. In contrast, the meta-learned *PACOH-NN* prior encodes useful inductive bias towards sinusoidal function shapes, leading to good predictions and uncertainty estimates, even in face of minimal training data (i.e. 1 training point).

Meta-learning benchmark. We present a comprehensive benchmark study. First, we use a *BNN* with a zero-centered, spherical Gaussian prior and SVGD inference [26] as a baseline. Second, we compare our proposed approach against popular meta-learning algorithms, including neural processes (*NPs*) [15], model-agnostic meta-learning (*MAML*) [11], its first-order version (*FOMAML*) [31], and two variants of the PAC-Bayesian approach by [2] (*MLAP*). In that, *MLAP* is the most similar to our approach as it also minimizes PAC-Bayesian bounds of the transfer error. However, unlike *PACOH-NN*, it relies on nested optimization of the task posteriors Q_i and the hyper-posterior $\mathcal{Q}$.

Meta-learning environments. In our experiments, we consider two real-world environments for regression. In particular, we use datasets corresponding to different calibration sessions of the Swiss Free Electron Laser (*SwissFEL*) [21, 28], as well as data from the *PhysioNet* 2012 challenge, which consists of time series of electronic health measurements from intensive care patients [37]. Here, the different tasks correspond to different patients. Further details can be found in Appendix C.1. Moreover, we conduct experiments with the multi-task *classification* environment *Omniglot* [24], consisting of handwritten letters across 50 alphabets. Unlike previous work [e.g. 11] we do not perform data-augmentation and do not re-combine letters of different alphabets, preserving the data's original structure. In particular, one task corresponds to 5-shot, 5-way classification of letters within an alphabet. This leaves us with much fewer tasks (i.e. 30 train / 20 test tasks), thus, making the environment more challenging and more interesting for uncertainty quantification. This also allows us to include *MLAP* in the experiment which hardly scales to more than 50 tasks.

Table 1 reports the results of our benchmark study in terms of the root mean squared error (RMSE) and classification accuracy on unseen test tasks. *PACOH-NN* consistently outperforms the other meta-learning approaches and the standard BNN, demonstrating that the introduced meta-learning framework is not only sound, but also endows us with an algorithm that works well in practice. To investigate the effect of meta-learned priors on the uncertainty estimates of the BNN, we compute the probabilistic predictors' calibration error [23] (see Table 1). Note that, since *MAML* only produces point predictions, the concept of calibration does not apply to it. We observe that meta-learning priors with *PACOH-NN* consistently improves the standard BNN's uncertainty estimates. Surprisingly, we find that *NPs* yields worse-calibrated predictions than the BNN while *MAML* fails to improve upon the standard BNN in terms of accuracy, i.e., demonstrating negative transfer. This is consistent with previous work [33, 35] raising concerns about overfitting to the meta-training tasks and observing that e.g. *MAML* requires a large number of tasks to generalize well. In contrast, by its very construction on meta-generalization bounds, *PACOH-NN* is able to achieve positive transfer even when the meta-training tasks are diverse and small in number.

Scalability. Unlike *MLAP* [2], *PACOH-NN* does not need to maintain posteriors Q_i for the meta-training tasks. To estimate the score of $\mathcal{Q}^*$ in (6), we can even use task mini-batching. Thus, *PACOH-NN* is computationally much faster and more salable than previous PAC-Bayesian meta-learners. Figure 2 shows this computational advantage during meta-training in the *Sinusoids* environment with varying number of tasks. While *MLAP's* memory consumption and compute time grows linearly, becoming prohibitively large even for less than 100 tasks, *PACOH-NN* maintains a nearly constant memory and compute load as the number of tasks grow.

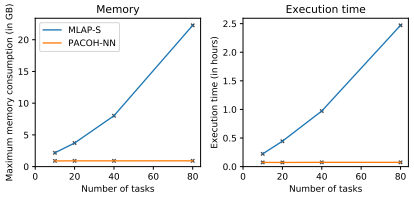


Figure 2: Scalability comparison of *PACOH-NN* and MLAP-S in memory footprint and compute time, as the number of meta-training task grows.

5.2 Meta-Learning for Bandits - Vaccine Development

We showcase how a relevant real-world application such as vaccine design can benefit from our proposed method. Following the bandit setup of [22], each task corresponds to searching for peptides that maximally bind to major histocompatibility complex class-I (MHC-I) molecules, a vital step in the design of peptide-based vaccines. The tasks differ in their targeted MHC-I allele, i.e., correspond to different genetic variants of the MHC-I protein. We use data from [45] which contains the binding affinities (IC_{50}values) of many peptide candidates (encoded as 45-dimensional feature vector) to the MHC-I alleles.

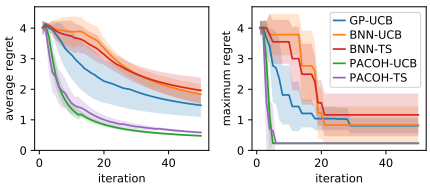


Figure 3: MHC-I peptide design task: Regret for different priors and bandit algorithms. A meta-learned *PACOH-NN* prior substantially improves the regret, compared to a standard BNN/GP prior.

We use 5 alleles (tasks) to meta-learn a BNN prior with *PACOH-NN* and leave one allele for our bandit task. In each iteration, the algorithm chooses to test one peptide among the more than 800 candidates, receiving its binding affinity as reward. We employ UCB [25] and Thompson-Sampling (TS) [40] as bandit algorithms, comparing the BNN-based bandits with meta-learned prior (*PACOH-UCB/TS*) against a standard Gaussian BNN prior (*BNN-UCB/TS*) and a Gaussian process (*GP-UCB*) [38]. Figure 3 reports the respective average regret and maximum regret over 50 iterations. Unlike the bandit algorithms with standard BNN/GP prior, *PACOH-UCB/TS* reaches near optimal regret within less than 10 iterations and after 50 iterations still maintains a significant performance advantage. This highlights the importance of *transfer (learning)* for solving real-world problems and demonstrates the effectiveness of *PACOH-NN* to this end.

6 Conclusion

Based on PAC-Bayesian theory, we present a scalable algorithm for meta-learning BNN priors that overcomes previous issues of nested optimization. Experiments show that *PACOH-NN* does not only come with computational advantages, but also achieves comparable or better predictive accuracy than several meta-learning approaches, while improving the quality of uncertainty estimates - a key aspect of our approach. The benefits of our principled treatment of uncertainty - showcased in the vaccine development bandit task – are particularly amenable to interactive machine learning systems.

References

[1] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of Gibbs posteriors. *Journal of Machine Learning Research*, 17(236):1–41, 2016.

[2] Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *ICML*, 2018.

[3] Marcin Andrychowicz, Misha Denil, Sergio Gómez Colmenarejo, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *arXix preprint arXix:1606.04474*, 2016.

[4] Jonathan Baxter. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198, 2000.

[5] James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Sanjoy Dasgupta and David McAllester, editors, *ICML*, volume 28 of *Proceedings of Machine Learning Research*, pages 115–123, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[6] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. jan 2016.

[7] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. 2015.

[8] Stephane Boucheron, Gabor Lugosi, and Pascal Massart. *Concentration inequalities : a nonasymptotic theory of independence*. Oxford University Press, 2013.

[9] Olivier Catoni. Pac-bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*, 2007.

[10] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *ICML*, pages 1683–1691, 2014.

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

[12] Vincent Fortuin and Gunnar Rätsch. Deep mean functions for meta-learning in gaussian processes. *arXiv preprint arXiv:1901.08098*, 2019.

[13] Vincent Fortuin, Gunnar Rätsch, and Stephan Mandt. Multivariate time series imputation with variational autoencoders. *arXiv preprint arXiv:1907.04155*, 2019.

[14] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, page 1050–1059, 2016.

[15] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

[16] Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. Pac-bayesian theory meets bayesian inference. In *Advances in Neural Information Processing Systems*, pages 1884–1892, 2016.

[17] Soumya Ghosh and Finale Doshi-Velez. Model selection in bayesian neural networks via horseshoe priors. *arXiv preprint arXiv:1705.10388*, 2017.

[18] Benjamin Guedj. A primer on PAC-Bayesian learning. *arXiv preprint arXiv:1901.05353*, 2019.

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[20] Johannes Kirschner, Mojmir Mutný, Nicole Hiller, Rasmus Ischebeck, and Andreas Krause. Adaptive and safe bayesian optimization in high dimensions via one-dimensional subspaces. In *Proc. International Conference for Machine Learning (ICML)*, June 2019.

[21] Johannes Kirschner, Manuel Nonnenmacher, Mojmir Mutný, Nicole Hiller, Andreas Adelmann, Rasmus Ischebeck, and Andreas Krause. Bayesian optimization for fast and safe parameter tuning of swissfel. In *Proc. International Free-Electron Laser Conference (FEL2019)*, June 2019.

[22] Andreas Krause and Cheng S Ong. Contextual gaussian process bandit optimization. In *NIPS*, pages 2447–2455, 2011.

[23] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *ICML*, 2018.

[24] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[25] Tor Lattimore and Csaba Szepesvari. *Bandit Algorithms*. Cambridge University Press, 2020.

[26] Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. aug 2016.

[27] David A McAllester. Some pac-bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.

[28] Christopher J Milne, Thomas Schietinger, Masamitsu Aiba, Arturo Alarcon, Jürgen Alex, Alexander Anghel, Vladimir Arsov, Carl Beard, Paul Beaud, Simona Bettoni, et al. Swissfel: the swiss x-ray free electron laser. *Applied Sciences*, 7(7):720, 2017.

[29] Aaron Mishkin, Frederik Kunstner, Didrik Nielsen, Mark Schmidt, and Mohammad Emtiyaz Khan. Slang: Fast structured covariance approximations for bayesian deep learning with natural gradient. In *NeurIPS*, pages 6245–6255, 2018.

[30] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, Berlin, Heidelberg, 1996.

[31] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

[32] Anastasia Pentina and Christoph Lampert. A pac-bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pages 991–999, 2014.

[33] Yunxiao Qin, Weiguo Zhang, Chenxu Zhao, Zezheng Wang, Hailin Shi, Guojun Qi, Jingping Shi, and Zhen Lei. Rethink and redesign meta learning. *arXiv preprint arXiv:1812.04955*, 2018.

[34] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *ICML*, 2015.

[35] Jonas Rothfuss, Vincent Fortuin, and Andreas Krause. Pacoh: Bayes-optimal meta-learning with pac-guarantees. *arXiv preprint arXiv:2002.05551*, 2020.

[36] Juergen Schmidhuber. *Evolutionary principles in self-referential learning. On learning how to learn: The meta-meta-... hook*. PhD thesis, Technische Universitaet Munchen, 1987.

[37] Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248. IEEE, 2012.

[38] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML*, 2009.

[39] Nicholas Syring and Ryan Martin. Calibrating general posterior credible regions. *Biometrika*, 106(2):479–486, 12 2018.

[40] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[41] Sebastian Thrun and Lorien Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, USA, 1998.

[42] Mariia Vladimirova, Jakob Verbeek, Pablo Mesejo, and Julyan Arbel. Understanding priors in Bayesian neural networks at the unit level. In *ICML*, pages 6458–6467, 2019.

[43] Ziyu Wang, Tongzheng Ren, Jun Zhu, and Bo Zhang. Function space particle optimization for bayesian neural networks. *arXiv preprint arXiv:1902.09754*, 2019.

[44] Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Swikatkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

[45] Christian Widmer, Nora C Toussaint, Yasemin Altun, and Gunnar Rätsch. Inferring latent task structure for multitask learning by multiple kernel learning. *BMC bioinformatics*, 11(S8):S5, 2010.

Appendix

A Proofs and Derivations

A.1 Proof of Theorem 2

Lemma 1. ***(Change of measure inequality)*** *Let f be a random variable taking values in a set A and let $X_1, ..., X_l$ be independent random variables, with $X_k \in A$ with distribution μ_k. For functions $g_k : A \times A \to \mathbb{R}, k = 1, ..., l$, let $\xi_k(f) = \mathbb{E}_{X_k \sim \mu_k}[g_k(f, X_k)]$ denote the expectation of g_k under μ_k for any fixed $f \in A$. Then for any fixed distribution π on A and any $\lambda > 0$, we have that*

$$\mathbb{E}_{f\sim\rho}\left[\sum_{k=1}^{l}\xi_k(f) \; - g_k(f, X_k)\right] \leq \frac{1}{\lambda}\left(D_{KL}(\rho||\pi) + \ln \mathbb{E}_{f\sim\pi}\left[e^{\lambda\left(\sum_{k=1}^{l}\xi_k(f) - g_k(f,X_k)\right)}\right]\right). \quad (9)$$

Proof of Theorem 2 To proof the Theorem, we need to bound the difference between *transfer error* $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ and the *empirical multi-task error* $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$. To this end, we introduce an intermediate quantity, the *expected multi-task error*:

$$\tilde{\mathcal{L}}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_n) = \mathbb{E}_{P\sim\mathcal{Q}}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{S\sim D_i^{m_i}}\left[\mathcal{L}(Q(S, P), \mathcal{D}_i)\right]\right] \quad (10)$$

In the following we invoke Lemma 1 twice. First, in step 1, we bound the difference between $\tilde{\mathcal{L}}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_n)$ and $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$, then, in step 2, the difference between $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ and $\tilde{\mathcal{L}}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_n)$. Finally, in step 3, we use a union bound argument to combine both results.

Step 1 (Task specific generalization) First, we bound the generalization error of the observed tasks $\tau_i = (\mathcal{D}_i, m_i)$, $i = 1, ..., n$, when using a learning algorithm $Q : \mathcal{M} \times \mathcal{Z}^{m_i} \to \mathcal{M}$, which outputs a posterior distribution $Q_i = Q(S_i, P)$ over hypotheses θ, given a prior distribution P and a dataset $S_i \sim \mathcal{D}_i^{m_i}$ of size m_i.

In particular, we apply Lemma 1 to the union of all training sets $S' = \bigcup_{i=1}^{n} S_i$ with $l = \sum_{i=1}^{n} m_i$. Hence, each X_k corresponds to one data point, i.e. $X_k = z_{ij}$ and $\mu_k = \mathcal{D}_i$. Further, we set $f = (P, h_1, ..., h_n)$ to be a tuple of one prior and n base hypotheses. This can be understood as a two-level hypothesis, wherein P constitutes a hypothesis of the meta-learning problem and h_i a hypothesis for solving the supervised task τ_i. Correspondingly, we take $\pi = (\mathcal{Q}, Q^n) = \mathcal{P} \cdot \prod_{i=1}^{n} P$ and $\rho = (\mathcal{Q}, Q^n) = \mathcal{Q} \cdot \prod_{i=1}^{n} Q_i$ as joint two-level distributions and $g_k(f, X_k) = \frac{1}{nm_i} l(h_i, z_{ij})$ as summand of the empirical multi-task error. We can now invoke Lemma 1 to obtain that (11) and (14)

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[\mathcal{L}(Q_i, D_i)\right] \leq & \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[\mathcal{L}(Q_i, S_i)\right] + \frac{1}{\lambda}\Bigg(D_{KL}\left[(\mathcal{Q}, Q^n)||(\mathcal{P}, P^n)\right] \\ & + \ln \mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{h\sim P}\left[e^{\frac{\lambda}{n}\sum_{i=1}^{n}(\mathcal{L}(h,\mathcal{D}_i) - \hat{\mathcal{L}}(h,S_i))}\right]\Bigg)\end{aligned} \quad (11)$$

Using the above definitions, the KL-divergence term can be re-written in the following way:

$$\begin{aligned} D_{KL}\left[(\mathcal{Q}, Q^n)||(\mathcal{P}, P^n)\right] &= \mathbb{E}_{P\sim\mathcal{Q}}\left[\mathbb{E}_{h\sim Q_i}\left[\ln \frac{\mathcal{Q}(P)\prod_{i=1}^{n}\prod Q_i(h)}{\mathcal{P}(P)\prod_{i=1}^{n}P(h)}\right]\right] & (12)\\ &= \mathbb{E}_{P\sim\mathcal{Q}}\left[\ln\frac{\mathcal{Q}(P)}{\mathcal{P}(P)}\right] + \sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[\mathbb{E}_{h\sim Q_i}\left[\ln\frac{Q_i(h)}{P(h)}\right]\right] & (13)\\ &= D_{KL}(\mathcal{Q}||\mathcal{P}) + \sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q_i||P)\right] & (14)\end{aligned}$$

Using (11) and (14) we can bound the expected multi-task error as follows:

$$\begin{aligned}\tilde{\mathcal{L}}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_n) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \frac{1}{\lambda}D_{KL}(\mathcal{Q}||\mathcal{P}) + \frac{1}{\lambda}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q_i||P)\right] \\ + \underbrace{\frac{1}{\lambda}\ln \mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{h\sim P}\left[e^{\frac{\lambda}{n}\sum_{i=1}^{n}(\mathcal{L}(h,\mathcal{D}_i) - \hat{\mathcal{L}}(h,S_i))}\right]}_{\Upsilon^{\mathrm{I}}(\lambda)}\end{aligned} \quad (15)$$

Step 2 (Task environment generalization) Now, we apply Lemma 1 on the meta-level. For that, we treat each task as random variable and instantiate the components as $X_k = \tau_i, l = n$ and $\mu_k = \mathcal{T}$. Furthermore, we set $\rho = \mathcal{Q}$, $\pi = \mathcal{P}$, $f = P$ and $g_k(f, X_k) = \frac{1}{n}\mathcal{L}(Q_i, \mathcal{D}_i)$. This allows us to bound the transfer error as

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \tilde{\mathcal{L}}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_n) + \frac{1}{\beta} D_{KL}(\rho||\pi) + \Upsilon^{\text{II}}(\beta) \tag{16}$$

wherein $\Upsilon^{\text{II}}(\beta) = \frac{1}{\beta} \ln \mathbb{E}_{P\sim\mathcal{P}} \left[e^{\frac{\beta}{n} \sum_{i=1}^{n} \mathbb{E}_{(D,S)\sim\mathcal{T}}[\mathcal{L}(Q(P,S),\mathcal{D})] - \mathcal{L}(Q(P,S_i),\mathcal{D}_i)} \right]$.

Combining (15) with (16), we obtain

$$\begin{aligned} \mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \left(\frac{1}{\beta} + \frac{1}{\lambda} \right) D_{KL}(\mathcal{Q}||\mathcal{P}) \\ + \frac{1}{\lambda} \sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{Q}} \left[D_{KL}(Q_i||P) \right] + \Upsilon^{\text{I}}(\lambda) + \Upsilon^{\text{II}}(\beta) \end{aligned} \tag{17}$$

Step 3 (Bounding the moment generating functions)

$$\begin{aligned} e^{(\Upsilon^{\text{I}}(\lambda) + \Upsilon^{\text{II}}(\beta))} =& \mathbb{E}_{P\sim\mathcal{P}} \left[e^{\frac{\beta}{n} \sum_{i=1}^{n} \mathbb{E}_{(D,S)\sim\mathcal{T}}[\mathcal{L}(Q(P,S),\mathcal{D})] - \mathcal{L}(Q(P,S_i),\mathcal{D}_i)} \right]^{1/\beta} \cdot \\ & \mathbb{E}_{P\sim\mathcal{P}} \mathbb{E}_{h\sim P} \left[e^{\frac{\lambda}{n} \sum_{i=1}^{n} (\mathcal{L}(h,\mathcal{D}_i) - \hat{\mathcal{L}}(h,S_i))} \right]^{1/\lambda} \\ =& \mathbb{E}_{P\sim\mathcal{P}} \left[\prod_{i=1}^{n} e^{\left(\frac{\beta}{n} \mathbb{E}_{(D,S)\sim\mathcal{T}}[\mathcal{L}(Q(P,S),\mathcal{D}))\right] - \mathcal{L}(Q(P,S_i),\mathcal{D}_i)} \right]^{1/\beta} \cdot \\ & \mathbb{E}_{P\sim\mathcal{P}} \mathbb{E}_{h\sim P} \left[\prod_{i}^{n} \prod_{i}^{m_i} e^{\frac{\lambda}{n m_i} (\mathcal{L}(h,\mathcal{D}_i) - l(h_i, z_{ij}))} \right]^{1/\lambda} \end{aligned} \tag{18}$$

Case I: bounded loss

If the loss function $l(h_i, z_{ij})$ is bounded in $[a_k, b_k]$, we can apply Hoeffding's lemma to each factor in (18), obtaining:

$$e^{\Upsilon^{\text{I}}(\lambda) + \Upsilon^{\text{II}}(\beta)} \leq \mathbb{E}_{P\sim\mathcal{P}} \left[e^{\frac{\beta^2}{8n}(b_k - a_k)^2} \right]^{1/\beta} \cdot \mathbb{E}_{P\sim\mathcal{P}} \mathbb{E}_{h\sim P} \left[e^{\frac{\lambda^2}{8nm}(b_k - a_k)^2} \right]^{1/\lambda} \tag{19}$$

$$= e^{\left(\frac{\beta}{8n} + \frac{\lambda}{8n\tilde{m}} \right)(b_k - a_k)^2} \tag{20}$$

Next, we factor out $\sqrt{n}$ from λ and β, obtaining

$$e^{\Upsilon^{\text{I}}(\lambda) + \Upsilon^{\text{II}}(\beta)} = \left(e^{\Upsilon^{\text{I}}(\lambda\sqrt{n}) + \Upsilon^{\text{II}}(\beta\sqrt{n})} \right)^{\frac{1}{\sqrt{n}}} \tag{21}$$

Using

$$\mathbb{E}_{\mathcal{T}} \mathbb{E}_{\mathcal{D}_1} ... \mathbb{E}_{\mathcal{D}_n} \left[e^{\Upsilon^{\text{I}}(\lambda\sqrt{n}) + \Upsilon^{\text{II}}(\beta\sqrt{n})} \right] \leq e^{\left(\frac{\beta}{8\sqrt{n}} + \frac{\lambda}{8\sqrt{n}\tilde{m}} \right)(b_k - a_k)^2} \tag{22}$$

we can apply Markov's inequality w.r.t. the expectations over the task distribution $\mathcal{T}$ and data distributions $\mathcal{D}_i$ to obtain that

$$\Upsilon^{\text{I}}(\lambda) + \Upsilon^{\text{II}}(\beta) \leq \underbrace{\frac{\beta}{8n}(b_k - a_k)^2}_{\Psi^{\text{I}}(\beta)} + \underbrace{\frac{\lambda}{8n\tilde{m}}(b_k - a_k)^2}_{\Psi^{\text{II}}(\lambda)} - \frac{1}{\sqrt{n}} \ln \delta \tag{23}$$

with probability at least $1 - \delta$.

Case II: sub-gamma loss

First, we assume that, $\forall i = 1, ..., n$ the random variables $V_i^{\text{I}} := \mathcal{L}(h, \mathcal{D}_i) - l(h_i, z_{i,j})$ are *sub-gamma* with variance factor s_{I}^2 and scale parameter c_{I} under the two-level prior $(\mathcal{P}, P)$ and the respective

data distribution $\mathcal{D}_i$. That is, their moment generating function can be bounded by that of a Gamma distribution $\Gamma(s_{\text{I}}^2, c_{\text{I}})$:

$$\mathbb{E}_{z\sim\mathcal{D}_i}\mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{h\sim P}\left[e^{\gamma(\mathcal{L}(h,\mathcal{D}_i)-l(h,z))}\right] \leq \exp\left(\frac{\gamma^2 s_{\text{I}}^2}{2(1-c_{\text{I}}\gamma)}\right) \quad \forall\gamma\in(0,1/c_{\text{I}}) \tag{24}$$

Second, we assume that, the random variable $V^{\text{II}} := \mathbb{E}_{(D,S)\sim\mathcal{T}}\left[\mathcal{L}(Q(P,S),\mathcal{D})\right] - \mathcal{L}(Q(P,S_i),\mathcal{D}_i)$ is *sub-gamma* with variance factor s_{II}^2 and scale parameter c_{II} under the hyper-prior $\mathcal{P}$ and the task distribution $\mathcal{T}$. That is, its moment generating function can be bounded by that of a Gamma distribution $\Gamma(s_{\text{II}}^2, c_{\text{II}})$:

$$\mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}}\mathbb{E}_{P\sim\mathcal{P}}\left[e^{\gamma\,\mathbb{E}_{(D,S)\sim\mathcal{T}}[\mathcal{L}(Q(P,S),\mathcal{D})]-\mathcal{L}(Q(P,S),\mathcal{D})}\right] \leq \exp\left(\frac{\gamma^2 s_{\text{II}}^2}{2(1-c_{\text{II}}\gamma)}\right) \quad \forall\gamma\in(0,1/c_{\text{II}}) \tag{25}$$

These two assumptions allow us to bound the expectation of (18) as follows:

$$\mathbb{E}\left[e^{\Upsilon^{\text{I}}(\lambda)+\Upsilon^{\text{II}}(\beta)}\right] \leq \exp\left(\frac{\lambda s_{\text{I}}^2}{2n\tilde{m}(1-c_{\text{I}}\lambda/(n\tilde{m})}\right)\cdot\exp\left(\frac{\beta s_{\text{II}}^2}{2n(1-c_{\text{II}}\beta/n)}\right) \tag{26}$$

Next, we factor out $\sqrt{n}$ from λ and β, obtaining

$$e^{\Upsilon^{\text{I}}(\lambda)+\Upsilon^{\text{II}}(\beta)} = \left(e^{\Upsilon^{\text{I}}(\lambda\sqrt{n})+\Upsilon^{\text{II}}(\beta\sqrt{n})}\right)^{\frac{1}{\sqrt{n}}} \tag{27}$$

Finally, by using Markov's inequality we obtain that

$$\Upsilon^{\text{I}}(\lambda)+\Upsilon^{\text{II}}(\beta) \leq \underbrace{\frac{\lambda s_{\text{I}}^2}{2n\tilde{m}(1-c_{\text{I}}\lambda/(n\tilde{m})}}_{\Psi^{\text{I}}(\beta)} + \underbrace{\frac{\beta s_{\text{II}}^2}{2n(1-c_{\text{II}}\beta/n)}}_{\Psi^{\text{II}}(\lambda)} - \frac{1}{\sqrt{n}}\ln\delta \tag{28}$$

with probability at least $1-\delta$.

To conclude the proof, we choose $\lambda = n\sqrt{\tilde{m}}$ and $\beta = \sqrt{n}$.

A.2 Proof of Corollary 1

Lemma 2. *[9] Let A be a set, $g : A \to \mathbb{R}$ a function, and $\rho \in \mathcal{M}(A)$ and $\pi \in \mathcal{M}(A)$ probability densities over A. Then for any $\beta > 0$ and $\forall a \in A$,*

$$\rho^*(a) := \frac{\pi(a)e^{-\beta g(a)}}{Z} = \frac{\pi(a)e^{-\beta g(a)}}{\mathbb{E}_{a\sim\pi}\left[e^{-\beta g(a)}\right]} \tag{29}$$

is the minimizing probability density $\arg\min_{\rho\in\mathcal{M}(A)}\ \beta\mathbb{E}_{a\sim\rho}\left[g(a)\right] + D_{KL}(\rho||\pi)$

Proof of Corollary 1 When we choose the posterior Q as the optimal Gibbs posterior $Q_i^* := Q^*(S_i, P)$, it follows that

$$\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q_i^*||P)\right] \tag{30}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(\mathbb{E}_{P\sim\mathcal{Q}}\mathbb{E}_{h\sim Q_i^*}\left[\hat{\mathcal{L}}(h,S_i)\right] + \frac{1}{\sqrt{\tilde{m}}}\left(\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q_i^*||P)\right]\right)\right) \tag{31}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\left(\mathbb{E}_{P\sim\mathcal{Q}}\mathbb{E}_{h\sim Q_i^*}\left[\sqrt{\tilde{m}}\hat{\mathcal{L}}(h,S_i) + \ln\frac{Q_i^*(h)}{P(h)}\right]\right) \tag{32}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\left(\mathbb{E}_{P\sim\mathcal{Q}}\mathbb{E}_{h\sim Q_i^*}\left[\frac{1}{\sqrt{\tilde{m}}}\sum_{j=1}^{m} l(h,z_i) + \ln\frac{P(h)e^{-\frac{1}{\sqrt{\tilde{m}}}\sum_{j=1}^{m} l(h,z_i)}}{P(h)Z(S_i,P)}\right]\right) \tag{33}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\left(-\mathbb{E}_{P\sim\mathcal{Q}}\left[\ln Z(S_i,P)\right]\right)\ . \tag{34}$$

This allows us to write the inequality in (3) as

$$\mathcal{L}(\mathcal{Q},\mathcal{T}) \leq \ -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{m_i}\mathbb{E}_{P\sim\mathcal{Q}}\left[\ln Z(S_i,P)\right] + \left(\frac{1}{\sqrt{n}}+\frac{1}{n\sqrt{\tilde{m}}}\right)D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta,n,\tilde{m})\ . \tag{35}$$

According to Lemma 2, the Gibbs posterior $Q^*(S_i,P)$ is the minimizer of (32), in particular $\forall P \in \mathcal{M}(\mathcal{H}), \forall i = 1,...,n$:

$$Q^*(S_i,P) = \frac{P(h)e^{-\sqrt{m}\hat{\mathcal{L}}(h,S_i)}}{Z(S_i,P)} = \underset{Q\in\mathcal{M}(\mathcal{H})}{\arg\min}\ \mathbb{E}_{h\sim Q}\left[\hat{\mathcal{L}}(h,S_i)\right] + \frac{1}{\sqrt{\tilde{m}}}D_{KL}(Q||P)\ . \tag{36}$$

Hence, we can write

$$\mathcal{L}(\mathcal{Q},\mathcal{T}) \leq -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\mathbb{E}_{P\sim\mathcal{Q}}\left[\ln Z(S_i,P)\right] + \left(\frac{1}{\sqrt{n}}+\frac{1}{n\sqrt{\tilde{m}}}\right)D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta,n,\tilde{m}) \tag{37}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[\min_{Q\in\mathcal{M}(\mathcal{H})}\hat{\mathcal{L}}(Q,S_i) + \frac{1}{\sqrt{\tilde{m}}}D_{KL}(Q||P)\right] \tag{38}$$

$$+\left(\frac{1}{\sqrt{n}}+\frac{1}{n\sqrt{\tilde{m}}}\right) + C(\delta,n,\tilde{m}) \tag{39}$$

$$\leq \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{Q}}\left[\hat{\mathcal{L}}(Q,S_i) + \frac{1}{\sqrt{\tilde{m}}}D_{KL}(Q||P)\right] \tag{40}$$

$$+\left(\frac{1}{\sqrt{n}}+\frac{1}{n\sqrt{\tilde{m}}}\right)D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta,n,\tilde{m}) \tag{41}$$

$$= \hat{\mathcal{L}}(\mathcal{Q},S_1,...,S_n) + \left(\frac{1}{\sqrt{n}}+\frac{1}{n\sqrt{\tilde{m}}}\right)D_{KL}(\mathcal{Q}||\mathcal{P}) \tag{42}$$

$$+\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{\tilde{m}}}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q_i||P)\right] + C(\delta,n,\tilde{m})\ , \tag{43}$$

which proves that the bound for Gibbs-optimal base learners in (35) and (4) is tighter than the bound in Theorem 2 which holds uniformly for all $Q \in \mathcal{M}(\mathcal{H})$.

A.3 Proof of Proposition 1: PAC-Optimal Hyper-Posterior

An objective function corresponding to (4) reads as

$$J(\mathcal{Q}) = -\mathbb{E}_{\mathcal{Q}}\left[\frac{1}{\sqrt{n\tilde{m}}+1}\sum_{i=1}^{n}\ln Z(S_i,P)\right] + D_{KL}(\mathcal{Q}||\mathcal{P})\ . \tag{44}$$

To obtain $J(\mathcal{Q})$, we omit all additive terms from (4) that do not depend on $\mathcal{Q}$ and multiply by the scaling factor $\frac{n\sqrt{\tilde{m}}}{\sqrt{\tilde{m}}n+1}$. Since the described transformations are monotone, the minimizing distribution of $J(\mathcal{Q})$, that is,

$$\mathcal{Q}^* = \underset{\mathcal{Q}\in\mathcal{M}(\mathcal{M}(\mathcal{H}))}{\arg\min}\ J(\mathcal{Q})\ , \tag{45}$$

is also the minimizer of (4). More importantly, $J(\mathcal{Q})$ is structurally similar to the generic minimization problem in Lemma 2. Hence, we can invoke Lemma 2 with $A = \mathcal{M}(\mathcal{H})$, $g(a) = -\sum_{i=1}^{n}\ln Z(S_i,P)$, $\beta = \frac{1}{\sqrt{n\tilde{m}}+1}$, to show that the optimal hyper-posterior is

$$\mathcal{Q}^*(P) = \frac{\mathcal{P}(P)\exp\left(\frac{1}{\sqrt{n\tilde{m}}+1}\sum_{i=1}^{n}\ln Z(S_i,P)\right)}{Z^{\text{II}}(S_1,...,S_n,\mathcal{P})}\ , \tag{46}$$

wherein

$$Z^{\text{II}}(S_1,...,S_n,\mathcal{P}) = \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{1}{\sqrt{n\tilde{m}}+1}\sum_{i=1}^{n}\ln Z(S_i,P)\right)\right]\ .$$

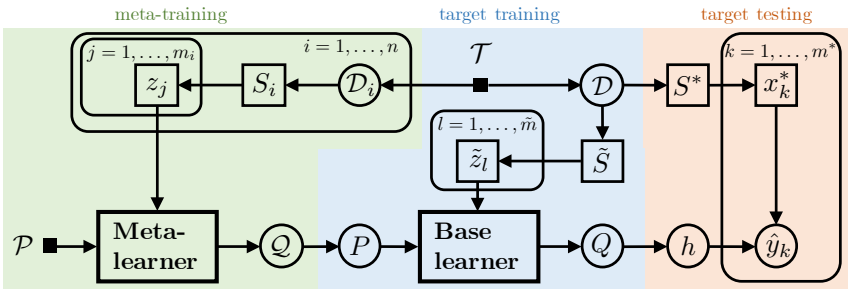


Figure S1: Overview of our meta-learning framework with environment $\mathcal{T}$, meta-task distributions $\mathcal{D}_i$, target task distribution $\mathcal{D}$, hyper-prior $\mathcal{P}$, hyper-posterior $\mathcal{Q}$, target prior P, target posterior Q, datasets S, and data points $z = (x, y)$.

□

Technically, this concludes the proof of Proposition 1. However, we want to remark the following result:

If we choose $\mathcal{Q} = \mathcal{Q}^*$, the PAC-Bayes bound in (4) can be expressed in terms of the meta-level partition function Z^{II}, that is,

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq -\left(\frac{1}{\sqrt{n}} + \frac{1}{n\sqrt{\tilde{m}}}\right) \ln Z^{\text{II}}(S_1, ..., S_n, \mathcal{P}) + C(\delta, n, \tilde{m}) \; . \tag{47}$$

We omit a detailed derivation of (47) since it is similar to the one for Corollary 1.

B PACOH-NN: A scalable algorithm for learning BNN priors

Proposition 2. *In expectation, replacing* $\ln Z(S_i, P_\phi)$ *in (4) by the Monte Carlo estimate* $\ln \tilde{Z}(S_i, P) := \ln \frac{1}{L} \sum_{l=1}^{L} e^{-\sqrt{m}\hat{\mathcal{L}}(\theta_l, S)}$, $\theta_l \sim P$ *still yields an valid upper bound of the transfer error. In particular, it holds that*

$$\begin{aligned}
\mathcal{L}(\mathcal{Q}, \mathcal{T}) &\leq -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{\tilde{m}}} \mathbb{E}_{P\sim\mathcal{Q}} \left[\ln Z(S_i, P)\right] + \left(\frac{1}{\sqrt{n}} + \frac{1}{n\sqrt{\tilde{m}}}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta, n, \tilde{m}) \\
&\leq -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{\tilde{m}}} \mathbb{E}_{P\sim\mathcal{Q}} \left[\mathbb{E}_{\theta_1,...,\theta_L\sim P} \left[\ln \tilde{Z}(S_i, P)\right]\right] + \left(\frac{1}{\sqrt{n}} + \frac{1}{n\sqrt{\tilde{m}}}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta, n, \tilde{m}).
\end{aligned}$$

Proof. Firsts, we show that:

$$\begin{aligned}
\mathbb{E}_{\theta_1,...,\theta_L\sim P} \left[\ln \tilde{Z}(S_i, P)\right] &= \mathbb{E}_{\theta_1,...,\theta_L\sim P} \left[\ln \frac{1}{L} \sum_{l=1}^{L} e^{-\sqrt{m}\hat{\mathcal{L}}(\theta_l, S_i)}\right] \\
&\leq \ln \frac{1}{L} \sum_{l=1}^{L} \mathbb{E}_{\theta_L\sim P} \left[e^{-\sqrt{m}\hat{\mathcal{L}}(\theta_l, S)}\right] \\
&= \ln \mathbb{E}_{\theta\sim P} \left[e^{-\sqrt{m}\hat{\mathcal{L}}(\theta, S_i)}\right] \\
&= \ln Z(S_i, P)
\end{aligned} \tag{48}$$

which follows directly from Jensen's inequality and the concavity of the logarithm. Now, Proposition 2 follows directly from (48). □

We now summarize the proposed meta-learning method, *PACOH-NN*, which is based on the result in Proposition 2. It consists of two stages.

Meta-training The hyper-posterior distribution $\mathcal{Q}$ that minimizes the upper bound on the transfer error is given by

$$\mathcal{Q}^*(P) = \frac{\mathcal{P}(P) \exp\left(\frac{1}{\sqrt{nm}+1} \sum_{i=1}^{n} \ln \tilde{Z}(S_i, P)\right)}{Z^{\text{II}}(S_1, ..., S_n, \mathcal{P})} \tag{49}$$

Provided with a set of datasets $S_1, ..., S_2$, the meta-learner minimizes the respective meta-objective, in the case of *PACOH-SVGD*, by performing SVGD on the $\mathcal{Q}^*$. Algorithm 1 outlines the required steps in more detail.

Algorithm 1 PACOH-NN-SVGD: meta-training

Input: hyper-prior $\mathcal{P}$, datasets $S_1, ..., S_n$
Input: kernel function $k(\cdot, \cdot)$, SVGD step size η, number of particles K
$\{\phi_1, ..., \phi_K\} \sim \mathcal{P}$ // Initialize prior particles
while not converged **do**
  for $k = 1, ..., K$ **do**
    $\{\theta_1, ..., \theta_L\} \sim P_{\phi_k}$ // sample NN-parameters from prior
    for $n = 1, ..., n$ **do**
      $\ln \tilde{Z}(S_i, P_{\phi_k}) \leftarrow \text{LSE}_{l=1}^{L}\left(-\sqrt{m}\hat{\mathcal{L}}(\theta_l, S_i)\right) - \ln L$ // estimate generalized MLL
    end for
    $\nabla_{\phi_k} \tilde{\mathcal{Q}}^*(\phi_k) \leftarrow \nabla_{\phi_k} \ln \mathcal{P}(\phi_k) + \frac{1}{\sqrt{n\tilde{m}}+1} \sum_{i=1}^{n} \nabla_{\phi_k} \ln \tilde{Z}(S_i, P_{\phi_k})$ // compute score
    $\phi_k \leftarrow \phi_k + \eta \frac{1}{K} \sum_{k'=1}^{K} \left[k(\phi_{k'}, \phi_k) \nabla_{\phi_{k'}} \ln \tilde{\mathcal{Q}}^*(\phi_{k'}) + \nabla_{\phi'_k} k(\phi_{k'}, \phi_k)\right]$ // update particles
  end for
end while
Output: set of priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$

Meta-testing The meta-learned prior knowledge is now deployed by a base learner. The base learner is given a training dataset $\tilde{S} \sim \mathcal{D}$ pertaining to an unseen task $\tau = (\mathcal{D}, m) \sim \mathcal{T}$. With the purpose of approximating the generalized Bayesian posterior $Q^*(S, P)$, the base learner performs (normal) posterior inference. Algorithm 2 details the steps of the approximating procedure – referred to as *target training* – when performed via SVGD. For a data point x^*, the respective predictor outputs a probability distribution given as $\tilde{p}(y^*|x^*, \tilde{S}) \leftarrow \frac{1}{K \cdot L} \sum_{k=1}^{K} \sum_{l=1}^{L} p(y^*|h_{\theta_l^k}(x^*))$. We evaluate the quality of the predictions on a held-out test dataset $\tilde{S}^* \sim \mathcal{D}$ from the same task, in a *target testing* phase (see Appendix C.2).

Algorithm 2 PACOH-NN-SVGD: meta-testing

Input: set of priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$, target training dataset $\tilde{S}$, evaluation point x^*
Input: kernel function $k(\cdot, \cdot)$, SVGD step size ν, number of particles L
for $k = 1, ..., K$ **do**
  $\{\theta_1^k, ..., \theta_L^k\} \sim P_{\phi_k}$ // initialize NN posterior particles from k-th prior
  while not converged **do**
    for $l = 1, ..., L$ **do**
      $\nabla_{\theta_l^k} Q^*(\theta_l^k)) \leftarrow \nabla_{\theta_l^k} \ln P_{\phi_k}(\theta_l^k)) + \sqrt{m}\, \nabla_{\theta_l^k} \mathcal{L}(l, \tilde{S})$ // compute score
      $\theta_l^k \leftarrow \theta_l^k + \frac{\nu}{L} \sum_{l'=1}^{L} \left[k(\theta_{l'}^k, \theta_l^k) \nabla_{\theta_{l'}^k} \ln Q^*(\theta_{l'}^k) + \nabla_{\theta_{l'}^k} k(\theta_{l'}^k, \theta_l^k)\right]$ // update particles
    end for
  end while
end for
Output: a set of NN parameters $\bigcup_{k=1}^{K} \{\theta_1^k ..., \theta_L^k\}$

	Sinusoid	SwissFEL	Physionet
n	20	5	100
m_i	5	200	4 - 24

Table S1: Number of tasks n and samples per task m_i for the different meta-learning environments.

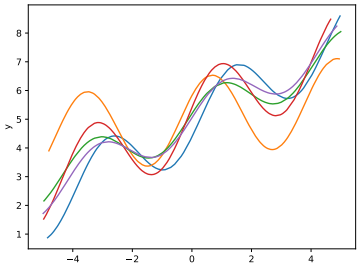


Figure S2: Depiction of tasks (i.e., functions) sampled from the Sinusoid environment.

C Experiments

C.1 Meta-Learning Environments

In this section, we provide further details on the meta-learning environments used in Section 5. Information about the numbers of tasks and samples in the respective environments can be found in Table S1.

Sinusoids. Each task of the sinusoid environment corresponds to a parametric function

$$f_{a,b,c,\beta}(x) = \beta * x + a * \sin(1.5 * (x - b)) + c \,, \tag{50}$$

which, in essence, consists of an affine as well as a sinusoid function. Tasks differ in the function parameters (a, b, c, β) that are sampled from the task environment $\mathcal{T}$ as follows:

$$a \sim \mathcal{U}(0.7, 1.3), \quad b \sim \mathcal{N}(0, 0.1^2), \quad c \sim \mathcal{N}(5.0, 0.1^2), \quad \beta \sim \mathcal{N}(0.5, 0.2^2) \,. \tag{51}$$

Figure **??** depicts functions $f_{a,b,c,\beta}$ with parameters sampled according to (51). To draw training samples from each task, we draw x uniformly from $\mathcal{U}(-5, 5)$ and add Gaussian noise with standard deviation 0.1 to the function values $f(x)$:

$$x \sim \mathcal{U}(-5, 5) \,, \qquad y \sim \mathcal{N}(f_{a,b,c,\beta}(x), 0.1^2) \,. \tag{52}$$

Figure **??** depicts a one-dimensional projection of functions sampled according to (**??**). To draw training samples from each task, we draw x from a truncated normal distribution and add Gaussian noise with standard deviation 0.05 to the function values $f(x)$:

$$x := \min\{\max\{\tilde{x}, 2\}, -3\} \,, \;\; \tilde{x} \sim \mathcal{N}(0, 2.5^2) \,, \qquad y \sim \mathcal{N}(f(x), 0.05^2) \,. \tag{53}$$

SwissFEL. Free-electron lasers (FELs) accelerate electrons to very high speed in order to generate shortly pulsed laser beams with wavelengths in the X-ray spectrum. These X-ray pulses can be used to map nanometer scale structures, thus facilitating experiments in molecular biology and material science. The accelerator and the electron beam line of a FEL consist of multiple magnets and other adjustable components, each of which has several parameters that experts adjust to maximize the pulse energy [20]. Due do different operational modes, parameter drift, and changing (latent) conditions, the laser's pulse energy function, in response to its parameters, changes across time. As a result, optimizing the laser's parameters is a recurrent task.

Overall, our meta-learning environment consists of different parameter optimization runs (i.e., tasks) on the SwissFEL, an 800 meter long laser located in Switzerland [28]. A picture of the SwissFEL is shown in Figure S3. The input space, corresponding to the laser's parameters, has 12 dimensions whereas the regression target is the pulse energy (1-dimensional). For details on the individual parameters, we refer to [21]. For each run, we have around 2000 data points. Since these data-points are generated with online optimization methods, the data are non-i.i.d. and get successively less

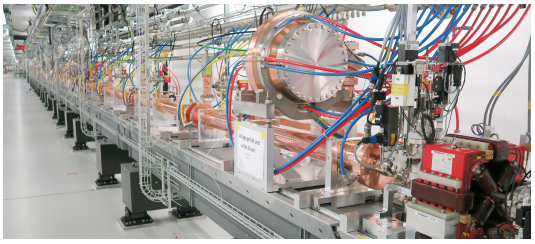


Figure S3: Accelerator of the Swiss Free-Electron Laser (SwissFEL).

diverse throughout the optimization. As we are concerned with meta-learning with limited data and want to avoid issues with highly dependent data points, we only take the first 400 data points per run and split them into training and test subsets of size 200. Overall, we have 9 runs (tasks) available. 5 of those runs are used for meta-training and the remaining 4 runs are used for meta-testing.

PhysioNet. The 2012 Physionet competition [37] published an open-access dataset of patient stays on the intensive care unit (ICU). Each patient stay consists of a time series over 48 hours, where up to 37 clinical variables are measured. The original task in the competition was binary classification of patient mortality, but due to the large number of missing values (around 80 % across all features), the dataset is also popular as a test bed for time series prediction methods, especially using Gaussian processes [12, 13]. In this work, we treat each patient as a separate task and the different clinical variables as different environments. We use the Glasgow coma scale (GCS) and hematocrit value (HCT) as environments for our study, since they are among the most frequently measured variables in this dataset. From the dataset, we remove all patients where less than four measurements of CGS (and HCT respectively) are available. From the remaining patients we use 100 patients for meta-training and 500 patients each for meta-validation and meta-testing. Here, each patient corresponds to a task. Since the number of available measurements differs across patients, the number of training points m_i ranges between 4 and 24.

C.2 Experimental Methodology

In the following, we describe our experimental methodology and provide details on how the empirical results reported in Section 5 were generated. Overall, evaluating a meta-learner consists of two phases, *meta-training* and *meta-testing*, outlined in Appendix B. The latter can be further subdivided into *target training* and *target testing*. Figure S1 illustrates these different stages for our PAC-Bayesian meta-learning framework.

The outcome of the training procedure is an approximation for the generalized Bayesian posterior $Q^*(S, P)$ (see Appendix), pertaining to an unseen task $\tau = (\mathcal{D}, m) \sim \mathcal{T}$ from which we observe a dataset $\tilde{S} \sim \mathcal{D}$. In *target-testing*, we evaluate its predictions on a held-out test dataset $\tilde{S}^* \sim \mathcal{D}$ from the same task. For PACOH-NN, NPs and MLAP the respective predictor outputs a probability distribution $\hat{p}(y^*|x^*, \tilde{S})$ for the x^* in $\tilde{S}^*$. The respective mean prediction corresponds to the expectation of $\hat{p}$, that is $\hat{y} = \hat{\mathbb{E}}(y^*|x^*, \tilde{S})$. In the case of MAML, only a mean prediction is available. Based on the mean predictions, we compute the *root mean-squared error (RMSE)*:

$$\text{RMSE} = \sqrt{\frac{1}{|\tilde{S}^*|} \sum_{(x^*, y^*) \in S^*} (y^* - \hat{y})^2} \,. \tag{54}$$

Similarly, we compute the *average log-likelihood*:

$$\text{LL} = \frac{1}{|\tilde{S}^*|} \sum_{(x^*, y^*) \in S^*} \ln \hat{p}(y^*|x^*, \tilde{S}) \,, \tag{55}$$

and the *calibration error* (see Appendix C.2.1).

The described meta-training and meta-testing procedure is repeated for five random seeds that influence both the initialization and gradient-estimates of the concerned algorithms. The reported averages and standard deviations are based on the results obtained for different seeds.

C.2.1 Calibration Error

The concept of calibration applies to probabilistic predictors that, given a new target input x_i, produce a probability distribution $\hat{p}(y_i|x_i)$ over predicted target values y_i. Corresponding to the predictive density, we denote a predictor's cumulative density function (CDF) as $\hat{F}(y_j|x_j) = \int_{-\infty}^{y_j} \hat{p}(y|x_i)dy$. For confidence levels $0 \leq q_h < ... < q_H \leq 1$, we can compute the corresponding empirical frequency

$$\hat{q}_h = \frac{|\{y_j \mid \hat{F}(y_j|x_j) \leq q_h, j = 1, ..., m\}|}{m} , \tag{56}$$

based on dataset $S = \{(x_i, y_i)\}_{i=1}^m$ of m samples. If we have calibrated predictions we would expect that $\hat{q}_h \to q_h$ as $m \to \infty$. Similar to [23], we can define the calibration error as a function of residuals $\hat{q}_h - q_h$, in particular,

$$\text{calib-err} = \frac{1}{H} \sum_{h=1}^{H} |\hat{q}_h - q_h| . \tag{57}$$

Note that we while [23] reports the average of squared residuals $|\hat{q}_h - q_h|^2$, we report the average of absolute residuals $|\hat{q}_h - q_h|$ in order to preserve the units and keep the calibration error easier to interpret. In our experiments, we compute (57) with $K = 20$ equally spaced confidence levels between 0 and 1.

C.3 Hyper-Parameter Selection

For each of the meta-environments and algorithms, we ran a separate hyper-parameter search to select the hyper-parameters. In particular, we use the `hyperopt`[1] package [5] which performs Bayesian optimization based on regression trees. As optimization metric, we employ the average log-likelihood, evaluated on a separate validation set of tasks. The scripts for reproducing the hyper-parameter search are included in our code repository[2] For the reported results, we provide the selected hyper-parameters and detailed evaluation results under `[Link will be made added upon acceptance]`

[1] http://hyperopt.github.io/hyperopt/

[2] [Link will be made added upon acceptance]