Journal of Machine Learning Research 24 (2023) 1-62 Submitted 11/22; Revised 12/23; Published 12/23

Scalable PAC-Bayesian Meta-Learning via the PAC-Optimal Hyper-Posterior: From Theory to Practice

Jonas Rothfuss JONAS.ROTHFUSS@INF.ETHZ.CH
ETH Zurich, Switzerland

Martin Josifoski MARTIN.JOSIFOSKI@EPFL.CH
EPFL Lausanne, Switzerland

Vincent Fortuin VINCENT.FORTUIN@HELMHOLTZ-MUNICH.DE
Helmholtz AI and TU Munich, Germany

Andreas Krause KRAUSEA@ETHZ.CH
ETH Zurich, Switzerland

Editor: Daniel Roy

Abstract

Meta-Learning aims to speed up the learning process on new tasks by acquiring useful inductive biases from datasets of related learning tasks. While, in practice, the number of related tasks available is often small, most of the existing approaches assume an abundance of tasks; making them unrealistic and prone to overfitting. A central question in the meta-learning literature is how to regularize to ensure generalization to unseen tasks. In this work, we provide a theoretical analysis using the PAC-Bayesian theory and present a generalization bound for meta-learning, which was first derived by Rothfuss et al. (2021a). Crucially, the bound allows us to derive the closed form of the optimal hyper-posterior, referred to as PACOH, which leads to the best performance guarantees. We provide a theoretical analysis and empirical case study under which conditions and to what extent these guarantees for meta-learning improve upon PAC-Bayesian per-task learning bounds. The closed-form PACOH inspires a practical meta-learning approach that avoids the reliance on bi-level optimization, giving rise to a stochastic optimization problem that is amenable to standard variational methods that scale well. Our experiments show that, when instantiating the PACOH with Gaussian processes and Bayesian Neural Networks models, the resulting methods are more scalable, and yield state-of-the-art performance, both in terms of predictive accuracy and the quality of uncertainty estimates.

Keywords: Meta-Learning, Transfer-Learning, PAC-Bayes, Learning Theory, Bayesian Neural Networks

1. Introduction

Learning new concepts and skills from a small number of examples as well as adapting them quickly in the face of changing circumstances is a key aspect of human intelligence. While modern machine learning systems are remarkably successful in learning complex patterns

from vast quantities of data, they lack such adaptive capabilities under limited data. As a result, one often has to train our machine-learning models from scratch even though we have previously solved similar learning problems/tasks.

Meta-Learning (Thrun and Pratt, 1998; Schmidhuber, 1987) has emerged as a promising avenue towards alleviating this issue by facilitating transfer across learning tasks, allowing us to harness related data sources or previous experience. In particular, meta-learning aims to do so by extracting prior knowledge (i.e., inductive bias) from a set of learning tasks, so that inference on a new learning task of interest is accelerated. For example, meta-learning can be instrumental in making medical diagnoses based on imaging data such as MRI or X-ray scans where obtaining large-scale, disease-specific datasets is challenging. Here, meta-learning can be used to learn the statistical properties of the medical imaging domain from a variety of datasets. The gained knowledge about the problem domain is typically represented as some form of prior. Such a domain-specific prior would then enable us to train reliable prediction models for new diagnoses with much less data.

The majority of existing meta-learning approaches rely on settings with an abundance of related tasks/datasets that are available for meta-learning (e.g., Finn et al., 2017; Garnelo et al., 2018). However, in most practical settings, the number of tasks that are available for meta-learning is small. In such settings, we face the issue of *overfitting on the meta-level,* i.e., overfitting to the tasks used during the meta-learning stage (cf. Qin et al., 2018; Yin et al., 2020). This would impair our learning performance on yet unseen target tasks. Thus, a key question is*, how to regularize meta-learning so that it does not overfit and generalizes well to unseen tasks.*

PAC-Bayesian learning theory gives us a rigorous framework for reasoning about the generalization of learners (McAllester, 1999). However, initial PAC-Bayesian analyses of meta-learning (Pentina and Lampert, 2014; Amit and Meir, 2018) only consider *bounded* loss functions, which excludes important applications such as regression or probabilistic inference, where losses are typically unbounded. More importantly, their generalization bounds involve PAC-Bayesian posterior distributions for each task as well as a hyper-posterior—a distribution over meta-learning hypotheses. Obtaining each posterior in itself is a challenging stochastic optimization problem whose solution, in turn, depends on the hyper-posterior. Hence, the resulting meta-learning approaches that minimize the corresponding generalization bounds rely on solving a challenging bi-level optimization problem. This makes them computationally much more expensive and unstable than standard meta-learning approaches.

This manuscript constitutes a significantly extended version of Rothfuss et al. (2021a) that aims to address the above-mentioned issues. First, we derive a *PAC-Bayesian bound* for meta-learning that also holds for *unbounded loss* functions. Hence, the corresponding learning guarantees apply to a much larger set of problems. For Bayesian learners, we further tighten our PAC-Bayesian bounds, relating them directly to the generalized marginal log-likelihoods of the Bayesian model instead of the posteriors. This allows us to avoid the difficult bi-level optimization. Going one step further, we present the *PAC-optimal hyper-posterior (PACOH)*—the closed form of the PAC-Bayesian meta-learning problem. In particular, the PACOH minimizes the upper bounds on the generalization error of meta-learning. Thus, it promises strong performance guarantees and comes with principled meta-level regularization which alleviates the aforementioned problem of overfitting. Importantly, the *PACOH* can be approximated using standard variational methods (Blei et al., 2017). This gives rise

to a range of *scalable* meta-learning algorithms which we explain and compare in depth. Furthermore, we analyze and discuss the improvement of meta-learning over per-task learning within the PAC-Bayesian framework. From this, we gain useful insights about factors that determine how much we can benefit from meta-learning. In a detailed case study on linear and logistic regression, we validate the tightness of our meta-learning bounds and empirically compare them to per-task learning bounds.

In our experiments, which are an extension of those in Rothfuss et al. (2021a), we instantiate our framework with Gaussian Process (GP) and Bayesian Neural Network (BNN) models and empirically compare different methods for approximating the hyper-posterior. Across several regression and classification environments, our proposed approach achieves *state-of-the-art* predictive *accuracy*, while also improving the *calibration* of the uncertainty estimates. Moreover, we demonstrate that, through its principled regularization on the meta-level, *PACOH* effectively *alleviates the problem of overfitting on the meta-level.* This allows us to successfully extract inductive bias from as little as five tasks while reliably reasoning about the learner's epistemic uncertainty. Thanks to these properties, *PACOH* can also be employed in a broad range of *sequential decision problems*, which we showcase through a real-world Bayesian optimization task concerning the development of vaccines. The promising experimental results suggest that many other challenging real-world problems such as molecular biology or medical imaging may benefit from our approach as well.

2. Related work

In this section, we review the relevant literature and its connections to our work. First, we discuss the field of meta-learning followed by a second subsection on learning theory. Third, we draw connections to kernel and multi-task learning, as well as hierarchical Bayesian methods. Finally, we discuss how this paper relates to Rothfuss et al. (2021a) and follow-up work.

2.1 Meta-Learning

Meta-learning aims to extract inductive bias from a set of related tasks so that inference on a new learning task is accelerated (Schmidhuber, 1987; Thrun and Pratt, 1998). For instance, a popular approach is to learn an embedding space shared across tasks (Baxter, 2000; Vinyals et al., 2016; Snell et al., 2017; Goldblum et al., 2020; Xu et al., 2020). Another class of meta-learning methods learns to update the model parameters (Bengio et al., 1991; Hochreiter et al., 2001; Andrychowicz et al., 2016; Ravi and Larochelle, 2017; Chen et al., 2017). Going one step further, Santoro et al. (2016); Mishra et al. (2018); Kim et al. (2019) train a recurrent or attention-based model to learn the entire learning and inference process. An alternative popular approach is to learn the initialization of a neural network so it can be quickly adapted to new tasks (Finn et al., 2017; Li et al., 2017; Nichol et al., 2018; Rothfuss et al., 2019).

Recent methods also use probabilistic modeling adaptation to enable uncertainty quantification (Yoon et al., 2018; Finn et al., 2018; Garnelo et al., 2018; Kim et al., 2019). Such approaches partially or fully amortize the training/inference on a target task which is prone to failure cases and unpredictable behavior. In contrast, our approach learns a prior and relies on standard methods for (PAC-)Bayesian inference on the target task.

Although the above-mentioned approaches are all able to learn complex inference patterns, they rely on the abundance of meta-training tasks and fall short of providing performance guarantees. The issue of over-fitting on the meta-level has previously been noted (Qin et al., 2018; Fortuin and Rätsch, 2019; Yin et al., 2020). However, it still lacks a rigorous formal analysis under realistic assumptions (e.g., unbounded loss functions). Addressing this shortcoming, we study the generalization properties of meta-learners within the PAC-Bayesian framework and, based on that, contribute a novel meta-learning approach with principled meta-level regularization.

2.2 Learning Theory

Our work builds on PAC-Bayesian learning theory, a framework for deriving generalization bounds for randomized predictors (McAllester, 1999; Seeger, 2002; Maurer, 2004; Catoni, 2007; Alquier, 2008; Germain et al., 2016; Alquier et al., 2016). Such bounds typically require that a prior distribution over hypotheses is given exogenously. Here, we refer to the PAC-Bayesian prior which differs from Bayesian priors insofar that it does not have to reflect the data-generating process. The prior has considerable influence on the tightness of PAC-Bayesian bounds. Hence, a range of works study distribution-dependent (Lever et al., 2013; Oneto et al., 2016; Rivasplata et al., 2018) and data-dependent priors (Parrado-Hernandez et al., 2012; Dziugaite and Roy, 2018; Dziugaite et al., 2021; Pérez-Ortiz et al., 2021).

In this paper, we also study a setting where the prior is acquired in a data-driven manner. However, while data-dependent priors are typcially adjusted to the current learning task, we consider priors that are meta-learned from a set of related learning tasks. In that, we build on previous work that studies meta-learning in the PAC-Bayesian framework (Pentina and Lampert, 2014; Amit and Meir, 2018; Farid and Majumdar, 2021; Liu et al., 2021). However, their PAC-Bayesian generalization bounds for meta-learning only consider bounded loss functions. More importantly, they are hard to optimize as they leave both the hyper-posterior and posterior unspecified, leading to difficult bi-level optimization problems. In contrast, our bounds also hold for unbounded losses and yield a tractable meta-learning objective without the reliance on bi-level optimization.

Ding et al. (2021) tailor our bounds (originally introduced in Rothfuss et al. (2021a)) to the few-shot meta-learning setting where the number of samples per task during the meta-learning stage is much larger than for inference on the target task. Thereby, they are able to connect popular meta-learning approaches such as MAML (Finn et al., 2017) and Reptile (Nichol et al., 2018) to the PAC-Bayesian meta-learning setting which is discussed in this paper. Further extension work provides tighter bounds based on the proof techniques of Catoni (2007). However, these bounds no longer admit a closed-form hyper-posterior and do not translate into improved algorithms. Finally, Guan and Lu (2022) present a generic PAC-Bayesian meta-learning bound that unifies many of the mentioned meta-learning bounds, including ours.

Other work that theoretically studies generalization in meta-learning uses covering number arguments to obtain uniform generalization bounds for meta-learning over families of hypothesis spaces (Baxter, 2000; Maurer and Jaakkola, 2005). Since such approaches translate into meta-learning hypothesis spaces, it is hard to convey probabilities and uncertainties from the meta-level learner to the base learner. In contrast, our PAC-Bayesian approach

learns prior distributions over learning hypotheses, thus, giving us a natural way to also improve the uncertainty estimates of downstream predictions.

A recent line of work presents information-theoretic generalization bounds for meta-learning (Chen et al., 2021; Jose and Simeone, 2021; Rezazadeh et al., 2021; Jose et al., 2022). Unlike our PAC-Bayesian bounds which are high-probability worst-case guarantees over meta-learning tasks and data, such information-theoretic arguments depend on the task and per-task data distributions as well as the particularities of the meta-learning algorithm.

2.3 Connections to kernel learning, multi-task learning and hierarchical Bayes

Similar to the GP variant of our proposed method, Ong et al. (2005); Zien and Ong (2007); Gönen and Alpaydın (2011); Wilson et al. (2016); Reeb et al. (2018) learn kernels. However, while we meta-learn a kernel from multiple related tasks, such works focus on kernel learning on a single target task. More recent works study kernel learning in the meta-learning setting with guarantees (Cella and Pontil, 2021; Kassraie et al., 2022; Cella et al., 2022; Schur et al., 2023), but only considered linear combinations of known base kernels, restricting its generality.

Similar to our problem setting, multi-task learning aims to transfer knowledge across tasks (e.g. Micchelli and Pontil, 2004; Yu et al., 2005; Bonilla et al., 2008; Parameswaran and Weinberger, 2010; Sener and Koltun, 2018). Such methods perform transduction on the meta-level, typically requiring a form of task similarity. In contrast, our approach performs induction on the meta-level, trying to find a global meta-learning hypothesis, i.e., a prior that works well for all tasks from the distribution over tasks. Finally, our setting closely resembles hierarchical Bayesian models (e.g. Salakhutdinov et al., 2012; Grant et al., 2018; Yoon et al., 2018; Ravi and Beatson, 2018). However, compared to such hierarchical Bayesian meta-learners which assume exact knowledge of the data-generating process, our PAC-Bayesian model makes much weaker assumptions about the loss function and hyper-prior. In addition, we provide generalization guarantees which the above-mentioned works lack.

2.4 Relation to Rothfuss et al. (2021a) and follow up work.

This paper is a substantially extended version of Rothfuss et al. (2021a), now including a theoretical comparison to single-task learning (Section 5), in-depth case studies for linear and logistic regression (Section 6), algorithms with alternative variational approximations (Section 7), and additional baselines in the empirical benchmark study (Section 8). Follow-up work of Rothfuss et al. (2021b) extends the PACOH approach to stochastic processes as priors on the meta-level (i.e., hyper-priors) and employs the resulting method towards facilitating lifelong Bayesian Optimization. The work of Rothfuss et al. (2022) discusses how to ensure that the corresponding meta-learned GP priors satisfy the strict calibration requirements necessary for interactive learning under safety constraints.

3. Background: PAC-Bayesian Framework

In this section, we explain the relevant background upon which the remainder of the paper builds. In particular, we first define basic notation and elementary concepts. Then, we briefly discuss cumulant-generating functions and provide an introduction to PAC-Bayesian learning

theory. Finally, we discuss the connections and differences between the PAC-Bayesian framework and classical Bayesian inference. The expositions follow previous work such as Baxter (2000); Alquier et al. (2016); Germain et al. (2016) .

3.1 Preliminaries and Notation

A learning task is characterized by an unknown data distribution $\mathcal{D}$ over a domain $\mathcal{Z}$ from which we are given a set of m observations $S = \{z_i\}_{i=1}^m$, $z_i \sim \mathcal{D}$. By $S \sim \mathcal{D}^m$ we denote the independent and identically distributed (i.i.d.) sampling of m data points. In supervised learning, we are typically concerned with pairs $z_i = (x_i, y_i)$, where $x_i \in \mathcal{X}$ are observed input features and $y_i \in \mathcal{Y}$ are target labels. Given a sample S, the goal is to find a hypothesis $h \in \mathcal{H}$, typically a function $h : \mathcal{X} \to \mathcal{Y}$ in some hypothesis space $\mathcal{H}$, that enables us to make predictions for new inputs $x^* \sim \mathcal{D}_x$. The quality of the predictions is measured by a *loss function* $l : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}$. Accordingly, we want to minimize the *expected error* under the data distribution, that is, $\mathcal{L}(h, \mathcal{D}) = \mathbb{E}_{z^* \sim \mathcal{D}}\, l(h, z^*)$. Since $\mathcal{D}$ is unknown, we typically use the *empirical error* $\hat{\mathcal{L}}(h, S) = \frac{1}{m}\sum_{i=1}^m l(h, z_i)$ instead.

In the PAC-Bayesian framework, we are concerned with *randomized predictors*, that is, probability measures on the hypothesis space $\mathcal{H}$. This allows us to give performance guarantees for machine learning models that can also reason about the epistemic uncertainty associated with their predictions. We consider two such probability measures, the *prior* $P \in \mathcal{M}(\mathcal{H})$ and the *posterior* $Q \in \mathcal{M}(\mathcal{H})$. Here, $\mathcal{M}(\mathcal{H})$ denotes the set of all probability measures on $\mathcal{H}$. Note that in Bayesian inference, the prior and posterior are assumed to be tightly connected through the likelihood factor, by Bayes' theorem. In contrast, the PAC-Bayesian framework makes fewer assumptions and only requires the prior to be independent of the observed data, while the posterior may depend on it. For a detailed discussion of the literature on PAC-Bayesian learning theory, we refer to Alquier (2021). In the following, we overload the notation by also denoting their probability densities as Q and P and assume that the Kullback-Leibler (KL) divergence $D_{KL}\left(Q \| P\right)$ exists. Based on the error definitions above, we can define the so-called *Gibbs error* for a randomized predictor Q as $\mathcal{L}(Q, \mathcal{D}) = \mathbb{E}_{h \sim Q}\, \mathcal{L}(h, \mathcal{D})$ and its empirical counterpart as $\hat{\mathcal{L}}(Q, S) = \mathbb{E}_{h \sim Q}\, \hat{\mathcal{L}}(h, S)$.

3.2 Cumulant-generating functions and concentration inequalities

Before we proceed to PAC-Bayesian theory, we recall the concept of the centered cumulant-generating function (CGF) which is an essential part of many relevant concentration inequalities (cf. Boucheron et al., 2013) that are employed in learning theory.

Definition 1 ***(Centered cumulant)*** *For a random variable $X \in A$ with distribution $\nu \in \mathcal{M}(A)$ and a real-valued function $f : A \mapsto \mathbb{R}$, the centered cumulant-generating function is defined as*

$$\Psi_{\nu, f(\cdot)}(t) = \log \mathbb{E}_{X \sim \nu}\left[e^{t(f(X) - \mathbb{E}[f(X)])}\right]. \tag{1}$$

Centered CGFs are defined as the logarithm of the centered moment generating function and quantify how much the random variable $f(X)$ deviates from its mean. They are a central tool for obtaining concentration inequalities and are particularly useful when $f(X)$ is

unbounded. To derive meaningful statistical concentration results for unbounded $f(X)$, we need to make additional assumptions to ensure that the tails of the probability distribution of $f(X)$ decay sufficiently fast. Such assumptions can be conveniently expressed in terms of bounds on the CGF. We will give specific examples of such assumptions in Section 3.3.

3.3 PAC-Bayesian Bounds

In practice, the generalization error $\mathcal{L}(Q, \mathcal{D})$ is unknown. Thus, one typically resorts to empirical risk minimization (ERM), that is, optimizing $\hat{\mathcal{L}}(Q, S)$ instead. However, pure ERM often results in overfitting on the training data set S and poor generalization to the actual data distribution $\mathcal{D}$ (Shalev-Shwartz and Ben-David, 2014). Naturally, we would like to understand factors that influence the severity of over-fitting and provide generalization guarantees. PAC-Bayesian learning theory helps us do so by bounding the unknown generalization error based on its empirical estimate:

Theorem 2 *(Germain et al., 2016, Theorem 3) Given a data distribution $\mathcal{D}$, hypothesis space $\mathcal{H}$, loss function $l(h, z)$, prior P, confidence level $\delta \in (0, 1]$, and $\beta > 0$, with probability at least $1 - \delta$ over samples $S \sim \mathcal{D}^m$, we have for all $Q \in \mathcal{M}(\mathcal{H})$:*

$$\mathcal{L}(Q, \mathcal{D}) \leq \hat{\mathcal{L}}(Q, S) + \frac{1}{\beta}\left[D_{KL}(Q||P) + \log\frac{1}{\delta} + \Psi(\beta, m)\right] , \tag{2}$$

with $\Psi(\beta, m) = \log \mathbb{E}_P \mathbb{E}_{\mathcal{D}^m} \exp\left[\beta\left(\mathcal{L}(h, \mathcal{D}) - \hat{\mathcal{L}}(h, S)\right)\right]$.

Note that the prior distribution P must be independent of the data S that is used to evaluate the empirical risk $\hat{\mathcal{L}}(Q, S)$. $\Psi(\beta, m)$ is the centered cumulant-generating function of the negative empirical loss $\hat{\mathcal{L}}(h, S)$ under the prior P and data distribution $\mathcal{D}$. Typically, β is chosen as $\beta = \sqrt{m}$ or $\beta = m$, such that the influence of the KL-complexity term $D_{KL}(Q||P)$ decreases as the number of training points m increases (Germain et al., 2016). However, note that, at the same time, increasing β comes at the cost of a larger CGF $\Psi(\beta, m)$.

Since $\Psi(\beta, n)$ contains $\mathcal{L}(h, \mathcal{D})$ which is unknown in practice, Theorem 2, as it is, does not yet provide a tractable bound. However, if we make additional assumptions about the loss function l, we can bound $\Psi(\beta, m)$ and thereby obtain useful PAC-Bayesian bounds. Common assumptions that have been used in the PAC-Bayesian literature are: Bounded loss functions (McAllester, 1999; Maurer, 2004), assumptions on tail behavior (Alquier et al., 2016; Germain et al., 2016), and moment assumptions (Alquier and Guedj, 2018; Holland, 2019). In the following, we briefly discuss how to bound the CGF for bounded losses and under tail assumptions.

Bounded loss. When the loss function is bounded, that is, $l : \mathcal{H} \times \mathcal{Z} \to [a, b]$, we can use Hoeffding's lemma to bound $\Psi(\beta, m)$. In particular, we define the random variable $l_j = \mathcal{L}(h, \mathcal{D}) - l(h, z_j)$ and write

$$\Psi(\beta, m) = \sum_{j=1}^{m} \log \mathbb{E} \exp\left(\frac{\beta}{m} l_j\right) \leq \sum_{j=1}^{m} \log \mathbb{E} \exp\left(\frac{\beta^2 (b-a)^2}{8m^2}\right) = \frac{\beta^2 (b-a)^2}{8m} . \tag{3}$$

Sub-gamma loss. A loss function l is considered *sub-gamma* with variance factor s^2 and scale parameter c, under a prior P and data distribution $\mathcal{D}$, if it can be described by a sub-gamma random variable $V := \mathcal{L}(h, \mathcal{D}) - l(h, z)$, i.e., its moment generating function is upper bounded by that of a Gamma distribution $\Gamma(s, c)$: $\log \mathbb{E}_{h\sim P}\mathbb{E}_{z\sim\mathcal{D}}\left[e^{\lambda V}\right] \leq \frac{\lambda^2 s^2}{2(1-c\lambda)} \quad \forall \lambda \in (0, 1/c)$. For details, see Germain et al. (2016) and Boucheron et al. (2013). Note that the sub-gamma assumption here is referred to as "sub-gamma on the right tail" in Boucheron et al. (2013, chapter 2.4). We can use the sub-gamma assumption with $V = l_j$ for $j = 1, ..., m$ and $\lambda = \beta/m$ to bound $\Psi(\beta, m)$ as follows

$$\Psi(\beta, m) = \sum_{j=1}^{m} \log \mathbb{E} \exp\left(\frac{\beta}{m} l_j\right) \leq \frac{\beta^2 s^2}{2m(1 - \frac{c\beta}{m})} \,. \tag{4}$$

Sub-gaussian loss. A *sub-gaussian* loss function with variance s^2 can be considered as a limit case of the previously discussed sub-gamma assumption when $c \to 0^+$. As direct consequence, $\Psi(\beta, m)$ can be bounded by $\Psi(\beta, m) \leq \frac{\beta^2 s^2}{2m}$.

3.4 Connection between the PAC-Bayesian framework and Bayesian Inference

Typically, we are interested in a posterior distribution Q that promises us the lowest generalization error $\mathcal{L}(\mathcal{Q}, \mathcal{D})$. Thus, it is natural to use the $Q \in \mathcal{M}(\mathcal{H})$ that minimizes the upper bound in (2). The following lemma gives us the closed-form solution to such a minimization problem over $\mathcal{M}(\mathcal{H})$:

Lemma 3 *(Catoni, 2007) Let $\mathcal{H}$ be a set, $g : \mathcal{H} \to \mathbb{R}$ a (loss) function, $Q \in \mathcal{M}(\mathcal{H})$ and $P \in \mathcal{M}(\mathcal{H})$ probability densities over $\mathcal{H}$. Then, for any $\beta > 0$ and $h \in \mathcal{H}$,*

$$Q^*(h) := \frac{P(h)e^{-\beta g(h)}}{Z} = \frac{P(h)e^{-\beta g(h)}}{\mathbb{E}_{h\sim P}\left[e^{-\beta g(h)}\right]} \tag{5}$$

is the solution of $\arg\min_{Q\in\mathcal{M}(\mathcal{H})} \; \beta\mathbb{E}_{h\sim Q}\left[g(h)\right] + D_{KL}(Q||P)$.

This distribution is known as the *Gibbs posterior* Q^* (Catoni, 2007; Lever et al., 2013). As a direct consequence of Lemma 3, for fixed P, S, m, δ, we can write the minimizer of the upper bound given in (2) as

$$Q^*(h) := \; \underset{Q\in\mathcal{M}(\mathcal{H})}{\arg\min}\, \beta\hat{\mathcal{L}}(Q, S) + D_{KL}(Q||P) = \; \frac{P(h)e^{-\beta\hat{\mathcal{L}}(h,S)}}{Z_\beta(S, P)} \tag{6}$$

where $Z_\beta(S, P) = \int_{\mathcal{H}} P(h)e^{-\beta\hat{\mathcal{L}}(h,S)}dh$ is a normalization constant. In a probabilistic setting, the loss function is the negative log-likelihood of the data, that is, $l(h, z_i) := -\log p(z_i|h)$. As has been pointed out by Germain et al. (2016), in this case, the optimal Gibbs posterior coincides with the *generalized Bayesian posterior* $Q^*(h; P, S) = \frac{P(h)\, p(S\,|\,h)^{\beta/m}}{Z_\beta(S,P)}$ where $Z_\beta(S, P) = \int_{\mathcal{H}} P(h) \left(\prod_{j=1}^{m} p(z_j|h)\right)^{\beta/m} dh$ is called the *generalized marginal likelihood* of the sample S (Guedj, 2019). For $\beta = m$ we recover the standard Bayesian posterior.

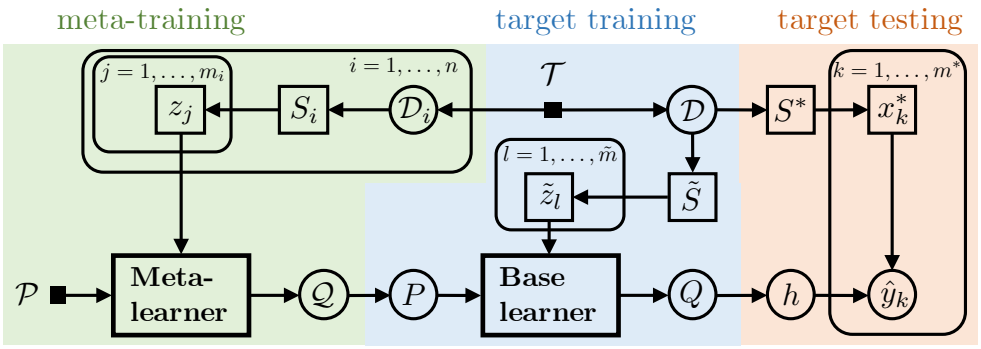


Figure 1: Overview of our meta-learning framework with environment $\mathcal{T}$, task distributions $\mathcal{D}_i$, target task distribution $\mathcal{D}$, hyper-prior $\mathcal{P}$, hyper-posterior $\mathcal{Q}$, target prior P, target posterior Q, dataset S, and data points $z = (x, y)$.

4. PAC-Bayesian Bounds for Meta-Learning

In this section, we present our main theoretical contributions. First, we describe and formalize meta-learning in the PAC-Bayesian setting, following previous work (Pentina and Lampert, 2014; Amit and Meir, 2018). The framework is illustrated in Figure 1. Then, we present our PAC-Bayesian meta-learning bound and discuss how, under certain instantiations, this bound can be transformed into a useful meta-learning objective. Finally, we introduce the *PAC-optimal Hyper-Posterior*, the closed-form solution of our PAC-Bayesian meta-learning problem. The corresponding proofs can be found in Appendix A.

4.1 Meta-Learning

In the standard learning setting (cf., Section 3), we assumed that the learner has prior knowledge in the form of a prior distribution P. When the learner experiences a task, in the form of a dataset S, then the data are used to update the prior into a posterior Q. A *base learner* $Q(S, P)$ can be formalized 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 (Amit and Meir, 2018). Note that the number of samples m may vary between datasets. Since we have not specified the base learner in more detail, the posterior that it produces can in principle be independent of the prior P. However, in practice, we typically employ base learners such as Bayesian inference, or more generally, Gibbs learners that output a Gibbs posterior as in (5).

How well such a learner generalizes from a limited dataset S to the entire data distribution $\mathcal{D}$ typically strongly depends on the prior. For instance, if the prior puts a high probability on hypotheses h that, a priori, describe the data well, the resulting posterior typically has a much smaller generalization error $\mathcal{L}(Q, \mathcal{D})$. Hence, choosing good priors for our learning task is of great importance. However, doing so by hand is often extremely challenging—especially when dealing with complex and uninterpretable hypothesis spaces such as those of neural networks

Meta-learning offers a solution to this problem by acquiring priors P in a *data-driven manner*, that is, by consulting a set of n statistically related learning tasks $\{\tau_1, ..., \tau_n\}$. We follow the setting of Baxter (2000) in which all tasks $\tau_i := (\mathcal{D}_i, S_i)$ share the same data

domain $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$, hypothesis space $\mathcal{H}$ and loss function $l(h, z)$, but may differ in their (unknown) data distributions $\mathcal{D}_i \in \mathcal{M}(\mathcal{Z})$ and the number of points m_i in the corresponding dataset $S_i \sim \mathcal{D}_i^{m_i}$.[1] To simplify our theoretical exposition, we assume that $m_i = m \ \forall i$.

Hence, we can employ meta-learning whenever we have access to multiple related datasets, e.g., MRI images labeled for different diagnoses or robotic data collected under different configurations of the robot (e.g., different tools/payloads). Given such a set of datasets $S_1, ..., S_n$, the goal is to *learn a prior P with which the base learner generalizes well on learning tasks of the particular problem setting* we are concerned with (e.g., MRI imaging). For this to work, we need to assume that the datasets $S_1, ..., S_n$ are representative of our problem setting which we will henceforth refer to as the *environment*. Formally, we assume that each task τ_i is generated i.i.d. by hierarchical sampling: 1) by sampling a data distribution $\mathcal{D}_i \sim \mathcal{T}$ from the *environment* $\mathcal{T} \in \mathcal{M}(\mathcal{M}(\mathcal{Z}))$, a distribution over data distributions, and, 2) by sampling a corresponding dataset $S_i \sim \mathcal{D}_i^m$. For brevity, we also refer to the distribution of this hierarchical sampling as $\mathcal{T}_h \in \mathcal{M}(\mathcal{M}(\mathcal{Z}) \times \mathcal{Z}^m)$ such that $(\mathcal{D}_i, S_i) \sim \mathcal{T}_h$. Formally, we want to learn a prior P that leads to small generalization error on new target tasks $\tau \sim \mathcal{T}_h$ (Pentina and Lampert, 2014).

Popular meta-learning approaches phrase this as a bi-level optimization problem (e.g., Finn et al., 2017; Amit and Meir, 2018). That is, they optimize a prior towards yielding a small empirical error when given to the base learner. This results in a bi-level optimization since the output of the base learner is typically the solution of an optimization problem in itself that depends on the prior.

To extend the PAC-Bayesian analysis to the meta-learning setting, we again consider the notion of probability distributions over hypotheses. The object of learning on a task has previously been a hypothesis $h \in \mathcal{H}$ over which one presumes a prior distribution $P \in \mathcal{M}(\mathcal{H})$ that is updated into a posterior $Q \in \mathcal{M}(\mathcal{H})$ based on observed data. In meta-learning, in turn, one presumes a *hyper-prior* $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$, i.e., a distribution over priors P. Then, combining the hyper-prior $\mathcal{P}$ with the datasets $S_1, ..., S_n$ from multiple tasks, the aim is to output a *hyper-posterior* $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ which can then be used to draw a prior for learning a new task. Accordingly, the hyper-posterior's performance is measured via the expected Gibbs error when sampling priors P from $\mathcal{Q}$ and applying the base learner, the so-called *transfer-error:*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) := \mathbb{E}_{P \sim \mathcal{Q}} \mathbb{E}_{(\mathcal{D}, S) \sim \mathcal{T}_h} \left[\mathcal{L}(Q(S, P), \mathcal{D}) \right] \tag{7}$$

While the transfer error is unknown in practice, we can estimate it using the *empirical multi-task error*

$$\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) := \mathbb{E}_{P \sim \mathcal{Q}} \left[\frac{1}{n} \sum_{i=1}^{n} \hat{\mathcal{L}} \left(Q(S_i, P), S_i \right) \right] . \tag{8}$$

4.2 PAC-Bayesian Meta-Learning bounds

We present our first main result: An upper bound on the true transfer error $\mathcal{L}(\mathcal{Q}, \mathcal{T})$, in terms of the empirical multi-task error $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$ plus several tractable complexity terms.

1. Note that, unlike previous work on data-dependent PAC-Bayesian priors (e.g., in Parrado-Hernandez et al., 2012; Dziugaite and Roy, 2018; Pérez-Ortiz et al., 2021), our approach uses data from different tasks rather than from the same data distribution.

Theorem 4 *Let $Q : \mathcal{Z}^m \times \mathcal{M}(\mathcal{H}) \to \mathcal{M}(\mathcal{H})$ be a base learner, $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ some fixed hyper-prior and $\lambda \geq \sqrt{n}, \beta \geq \sqrt{m}$. For any confidence level $\delta \in (0, 1]$ the inequality*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \left(\frac{1}{\lambda} + \frac{1}{n\beta}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta}\mathbb{E}_{P\sim\mathcal{Q}}\left[D_{KL}(Q(S_i, P)||P)\right] + \underbrace{\bar{\Psi}^I(\beta) + \Psi^{II}(\lambda) + \frac{1}{\sqrt{n}} \log \frac{1}{\delta}}_{:= C(\delta,\lambda,\beta)} \tag{9}$$

holds uniformly over all hyper-posteriors $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ with probability $1 - \delta$. Here, $\bar{\Psi}^I(\beta)$ is an upper bound on the CGF $\Psi^I(\beta) = \frac{\beta}{m} \log \mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}}\left[e^{\frac{\beta}{m}(\mathcal{L}(h,\mathcal{D})-l(h,z))}\right] \ \forall \ \mathcal{D}$ in the support of $\mathcal{T}$, and $\Psi^{II}(\lambda) = \frac{n}{\lambda} \log \mathbb{E}_{\mathcal{P}}\mathbb{E}_{\mathcal{T}_h}\left[e^{\frac{\lambda}{n}\mathbb{E}_{\mathcal{T}_h}\mathbb{E}_{\mathcal{P}}[\mathcal{L}(Q(P,S),\mathcal{D})]-\mathcal{L}(Q(P,S),\mathcal{D})}\right] \leq \bar{\Psi}^{II}(\lambda)$. Additionally, we require that the expectation $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ exists and is finite.

Next, we provide bounds on the CGFs $\Psi^{\mathrm{I}}(\beta)$ and $\Psi^{\mathrm{II}}(\lambda)$ under specific assumptions about the loss. For that, we denote corresponding upper bounds as $\bar{\Psi}^{\mathrm{I}}(\beta)$ and $\bar{\Psi}^{\mathrm{II}}(\lambda)$. If the loss function is bounded, an upper bound on the CGFs can be obtained via Hoeffding's inequality and follows Pentina and Lampert (2014); Amit and Meir (2018):

Corollary 5 ***(Bounded loss)*** *If the loss is bounded in $[a, b]$, Theorem 4 holds with*

$$\bar{\Psi}^I(\beta) + \bar{\Psi}^{II}(\lambda) = \left(\frac{\lambda}{8n} + \frac{\beta}{8m}\right)(b - a)^2 \ . \tag{10}$$

as upper bounds for the CGFs.

In the case of unbounded loss functions, we require additional assumptions. Inspired by Germain et al. (2016), but extended to the meta-learning setting, the following corollary provides a version of Theorem 4 that holds under sub-gamma tail assumptions:

Corollary 6 ***(Sub-gamma loss)*** *If the loss is sub-gamma with variance factor s_I^2 and scale parameter c_I under the data distributions $\mathcal{D}$ and hyper-prior $\mathcal{P}$, and sub-gamma with s_{II}^2, c_{II} under the task distribution $\mathcal{T}$ and hyper-prior $\mathcal{P}$ (see Appendix A.1, Step 3 for details) then, Theorem 4 holds with*

$$\bar{\Psi}^I(\beta) + \bar{\Psi}^{II}(\lambda) = \frac{\beta s_I^2}{2m(1 - (c_I\beta)/m)} + \frac{\lambda s_{II}^2}{2n(1 - (c_{II}\lambda)/n)} \ . \tag{11}$$

as upper bounds for the CGFs if $\lambda < n/c_{II}$ and $\beta < m/c_I$.

For bounded losses, Theorem 4 together with Corollary 5 provides a structurally similar, but tighter bound than Pentina and Lampert (2014). In particular, by using an improved proof technique, we are able to omit a union-bound argument, allowing us to reduce the negative influence of the confidence parameter δ. In contrast to Pentina and Lampert (2014) and Amit and Meir (2018), our theorem together with Corollary 6 also provides guarantees for *unbounded* loss functions under moment constraints (see Appendix A.1 for details). This

makes Corollary 6 particularly relevant for probabilistic models in which the loss function coincides with the inherently unbounded negative log-likelihood.

Common choices for λ and β are either 1) $\lambda = \sqrt{n}$, $\beta = \sqrt{m}$ or 2) $\lambda = n$, $\beta = m$ (Germain et al., 2016). If we choose $\lambda = \sqrt{n}$, $\beta = \sqrt{m}$, we obtain consistent bounds, meaning that the gap between the transfer error and the bound vanishes as $n, m \to \infty$. In the second case ($\lambda = n$, $\beta = m$), the bound always maintains a gap since $C(\delta, n, m)$ does not converge to zero. However, the KL-divergence terms decay faster, which can be advantageous for smaller sample sizes. For instance, despite their lack of consistency, sub-gamma bounds with $\beta = m$ have been shown to be much tighter in simple Bayesian linear regression scenarios with limited data ($m \lesssim 10^4$) (Germain et al., 2016).

Previous work (Pentina and Lampert, 2014; Amit and Meir, 2018) proposes meta-learning algorithms that minimize uniform generalization bounds like the one in (9). However, such bounds explicitly depend on the posterior $Q(S_i, P)$ which is often intractable and the solution of a non-trivial numerical optimization problem; e.g., variational inference in case of Bayesian Neural Networks. Critically, the solution of such an optimization depends on P which changes during the course of meta-learning. Thus, employing such bounds as a meta-learning objective typically results in a challenging two-level optimization problem wherein n posteriors Q_i and the hyper-posterior $\mathcal{Q}$ need to be optimized in an interdependent manner. This becomes highly intractable to solve for rich hypothesis spaces such as neural networks.

While Theorem 4 holds for any base learner $Q(S, P)$, we would preferably want to use a base learner that gives us *optimal performance guarantees*. As discussed in Section 3, the Gibbs posterior not only minimizes PAC-Bayesian error bounds, but also generalizes the Bayesian posterior. Assuming a Gibbs posterior as base learner, the bound in (9) can be re-stated in terms of the partition function $Z_\beta(S_i, P)$:

Corollary 7 *When using a Gibbs posterior $Q^*(S_i, P) := P(h)\exp(-\beta\hat{\mathcal{L}}(S_i, h))/Z_\beta(S_i, P)$ as a base learner, under the assumptions of Theorem 4, we have with probability at least $1-\delta$*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \; -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\beta}\mathbb{E}_{P\sim\mathcal{Q}}\left[\log Z_\beta(S_i, P)\right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta, \lambda, \beta)\,. \quad (12)$$

Remark 8 *Among all base learners, the Gibbs posterior Q^* achieves the smallest possible value of the bound in (9), i.e., the RHS of (12) is smaller or equal than the RHS of (9).*

Since this bound assumes a *PAC-optimal base learner*, it is at least as small as the bound in (9), which holds for any, potentially sub-optimal, $Q \in \mathcal{M}(\mathcal{H})$. More importantly, (12) avoids the explicit dependence on $Q(S_i, P)$, *turning the previously mentioned bi-level optimization problem into a standard stochastic optimization problem.* Moreover, if we choose the negative log-likelihood as the loss function and $\lambda = n, \beta = m$, then $\log Z_\beta(S_i, P)$ *coincides with the marginal log-likelihood (MLL)*, which is tractable for various popular learning models, such as GPs.

The bound in (12) consists of the expected generalized marginal log-likelihood under the hyper-posterior $\mathcal{Q}$ as well as the KL-divergence term which serves as a *regularizer on the meta-level*. As the number of training tasks n grows, the relative weighting of the KL term in (12) shrinks. This is consistent with the general notion that regularization should be strong if only little data is available and vanish asymptotically as $n, m \to \infty$.

Finally, Corollary 7 assumes that the same loss function is used for constructing Gibbs posterior and evaluating the bound. While this is generally the case in regression, in classification, we may want to obtain a bound on the misclassification error while using the negative cross-entropy loss to form a posterior. Hence, to evaluate the bound under a different loss than used for training, we have to resort to the bound in Theorem 4 that holds for arbitrary posteriors.

4.3 The PAC-Optimal Hyper-Posterior

A natural way to obtain a PAC-Bayesian meta-learning algorithm could be to minimize (12) with respect to $\mathcal{Q}$. However, we can go one step further and derive the closed-form solution of the PAC-Bayesian meta-learning problem, i.e., the minimizing hyper-posterior $\mathcal{Q}^*$. For that, we exploit once more the insight that the minimizer of (12) can be written as Gibbs distribution (cf., Lemma 3), giving us the following result:

Proposition 9 ***(PAC-Optimal Hyper-Posterior)*** *Given a hyper-prior $\mathcal{P}$ and datasets $S_1, ..., S_n$, the hyper-posterior minimizing the meta-learning bound in (12) is given by*

$$\mathcal{Q}^*(P) = \frac{\mathcal{P}(P) \exp\left(\frac{\lambda}{n\beta+\lambda} \sum_{i=1}^n \log Z_\beta(S_i, P)\right)}{Z^{II}(S_1, ..., S_n, \mathcal{P})} \tag{13}$$

with $Z^{II}(S_1, ..., S_n, \mathcal{P}) = \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{\lambda}{n\beta+\lambda} \sum_{i=1}^n \log Z_\beta(S_i, P)\right)\right]$.

This gives us the *closed-form solution of the PAC-Bayesian meta-learning problem*, that is, the *PAC-Optimal Hyper-Posterior (PACOH)* $\mathcal{Q}^*(P)$. In particular, we have a tractable expression for $\mathcal{Q}^*(P)$ up to the (level-II) partition function Z^{II}, which is constant with respect to P. We refer to $\mathcal{Q}^*$ as PAC-optimal, as it provides the best possible meta-generalization guarantees among all meta-learners in the sense of Theorem 4. These best possible guarantees can be stated as follows:

Corollary 10 *If we use the PACOH $\mathcal{Q}^*$ in (13) as hyper-posterior, under the same assumptions as in Corollary 7, with probability $\geq 1-\delta$, the transfer error $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ is bounded by*

$$\mathcal{L}(\mathcal{Q}^*, \mathcal{T}) \leq -\left(\frac{1}{\lambda} + \frac{1}{n\beta}\right) \log Z^{II}(S_1, ..., S_n, \mathcal{P}) + C(\delta, \lambda, \beta) \ . \tag{14}$$

5. Meta-Learning vs. Per-Task Learning

In this section, we aim to understand under which conditions and to what extent meta-learning improves upon per-task learning,that is, simply using the base learner on tasks individually without attempting to transfer knowledge across tasks.

A key challenge in this endeavor is that standard (per-task) learners typically use different assumptions than meta-learners. For instance, Theorem 2 presumes an exogenously given prior $P \in \mathcal{M}(\mathcal{H})$ whereas in meta-learning (cf. Section 4.1) we infer such prior in an endogenous manner from a set of datasets $S_1, ..., S_n$. Suppose we were to assume the best possible prior for per-task learning, i.e., the prior corresponding to the environment's data-generating process $\mathcal{T}$. In that case, no meta-learner can possibly improve upon per-task

learning. Hence, to be instructive, we must construct our comparison in such a way that both the meta-learner and the learner start with the same prior knowledge.

Instead of assuming a single prior P as in Theorem 2, we now assume that the per-task learner is given a distribution over priors, i.e., a hyper-prior $\mathcal{P}$. Given priors, sampled from $\mathcal{P}$, we use the PAC-optimal base learner $Q^*(S, P) := P(h)\exp(-\beta\hat{\mathcal{L}}(S)i, h))/Z(S_i, P)$ to infer the posterior for a given dataset S. How well this per-task PAC-Bayesian learning approach performs can be quantified by the expected generalization error on (unseen) tasks $\tau \sim \mathcal{T}_h$, i.e.,

$$\mathcal{L}(\mathcal{P}, \mathcal{T}) := \mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}_h}\left[\mathcal{L}(Q(S, P), \mathcal{D})\right] \ . \tag{15}$$

Note that (15) is similar to the transfer-error in (7) except that the priors are sampled from the hyper-prior $\mathcal{P}$ rather than a meta-learned hyper-posterior $\mathcal{Q}$. We can bound $\mathcal{L}(\mathcal{P}, \mathcal{T})$ using similar proof techniques as before to obtain the following environment-level generalization bound for per-task learning:

Theorem 11 ***(Per-Task Learning Bound)*** *Let the base learner be the Gibbs posterior $Q^*(S_i, P) := P(h)\exp(-\beta\hat{\mathcal{L}}(S_i, h))/Z_\beta(S_i, P)$, $\mathcal{P}$ some hyper-prior and $\lambda \geq \sqrt{n}, \beta \geq \sqrt{m}$. If we perform per-task learning with priors sampled from $\mathcal{P}$, then the expected generalization error on tasks $\tau \sim \mathcal{T}_h$ can be bounded by*

$$\mathcal{L}(\mathcal{P}, \mathcal{T}) \leq \ -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\beta}\mathbb{E}_{P\sim\mathcal{P}}\left[\log Z_\beta(S_i, P)\right] + C(\delta, \lambda, \beta) \tag{16}$$

which holds with probability at least $1 - \delta$.

Now, we can compare the generalization bounds for meta-learning with the per-task learning bound in Theorem 11. For such a comparison to be insightful, the respective bounds should be asymptotically consistent. Thus, we use $\lambda = \sqrt{n}$ and $\beta = \sqrt{m}$ henceforth in this chapter. At first glance, we observe that, unlike Corollary 7, the per-task learning bound no longer has a KL divergence term on the meta-level. This reflects the absence of meta-learning, i.e., there is no more potential overfitting of the hyper-posterior to the meta-learning tasks, which must be compensated. On the other hand, the first term in (16), i.e., the generalized MLL, is now in expectation under the fixed hyper-prior and thus cannot be reduced. Despite this intricate trade-off, we can derive the gap between the PACOH meta-learning bound in Corollary 10 and the per-task learning bound in Theorem 11 in closed form:

Proposition 12 ***(Meta-learning vs. per-task learning)*** *Let $\mathcal{P}$ be a hyper-prior and $S_1, ..., S_n$ datasets corresponding to tasks sampled from $\mathcal{T}_h$. We write $Z(S_i, P)$ short for $Z_{n^{1/2}}(S_i, P)$. The PACOH provides smaller generalization bound values than per-task learning with the Gibbs posterior (cf., Theorem 4). In particular, the improvement is given by*

$$\begin{aligned}\Delta = (16) - (14) &= \frac{1}{n}\sum_{i=1}^{n}\frac{\sqrt{nm}+1}{\sqrt{m}}\ \Psi_{\mathcal{P},\sum_i \log Z(S_i,\cdot)}\left(\frac{1}{\sqrt{nm}+1}\right) \\ &= \left(\frac{1}{\sqrt{n}} + \frac{1}{n\sqrt{m}}\right)\log\mathbb{E}_{P\sim\mathcal{P}}\left[e^{\frac{1}{\sqrt{nm}+1}\sum_{i=1}^{n}(\log Z(S_i,P) - \mathbb{E}_{P\sim\mathcal{P}}[\log Z(S_i,P)])}\right]\end{aligned} \tag{17}$$

and is always non-negative, i.e., $\Delta \geq 0$.

Proposition 12 suggests that the PACOH meta-learning bound always improves upon the per-task learning bound. This is natural since meta-learning with the PACOH adapts the hyper-prior into a hyper-posterior so that the corresponding PAC-Bayesian bound is minimized. To what extent the PACOH meta-learning bounds improves upon per-task learning depends on the following key factors:

- Hyper-prior informativeness: The CGF in (17) can be understood as quantifying how much information the hyper-prior already conveys about the tasks $S_1, ..., S_n$, and thus about the environment $\mathcal{T}$. For instance, if the hyper-prior $\mathcal{P}$ has a high variance, i.e., is relatively uninformative, then the CGF will be large, suggesting that meta-learning can significantly improve upon per-task learning. On the other extreme, if the hyper-prior is a Dirac measure on a single prior, the CGF will be zero, suggesting that no improvement upon per-task learning is possible. Generally, the more uninformative the hyper-prior $\mathcal{P}$, the larger the improvement Δ.

- Number of tasks n: Δ grows with the number of tasks n. This reflects that, as the meta-learning algorithm gets more training tasks, it can improve its hyper-posterior and thus improve its generalization on unseen tasks. In contrast, per-task learning benefits from additional training tasks through transfer.

- Number of samples per task m: While the CGF stays roughly constant with m[2], the pre-factor outside the CGF shrinks with m. This reflects that, the more training samples are available per task, the smaller the relative influence of the prior on the posteriors per task. Hence, it becomes increasingly hard for meta-learning to significantly improve the generalization error.

In Section 6, we provide an empirical evaluation of how some of these factors affect the improvement of meta-learning over per-task learning.

Finally, we want to emphasize that Proposition 12 compares upper bounds. Hence, it does not formally guarantee the improvement of meta-learning over per-task learning in all instances. However, since both bounds were obtained with the same proof techniques, we believe that the comparison is instructive and demonstrates how meta-learning can give us better PAC-Bayesian generalization guarantees. Moreover, the fact that the gap Δ exhibits the general behavior one would expect from a meta-learner (three factors above), suggests that it is not a mere difference of vacous bounds, but rather correlates well with the actual empirical improvement.

6. Case Study: Binary Classification and Linear Regression

In this section, we further examine the theoretical results from Sections 4 and 5 in the setting of binary classification and linear regression. First, we numerically compare our bound against previous meta-learning bounds. Since the previous bounds only hold for bounded losses, we compare them in a binary classification setting where the quantity for which we want to provide guarantees is the misclassification error. Second, we demonstrate how we can

2. $\log Z(S, P)$ grows at the order of $\mathcal{O}(\sqrt{n})$ which cancels with $\mathcal{O}(1/\sqrt{m})$ shrinkage of the pre-factor in the exponent

apply our PAC-Bayesian analysis from Corollary 10 to unbounded loss functions such as the negative log-likelihood. Inspired by Germain et al. (2016); Shalaeva et al. (2020), we choose a simple linear regression setting so that we can derive bounds for cumulant-generating functions and easily compute the bounds' values.

6.1 Binary classification

Model. For the binary classification setting, we consider linear classifiers. Hence, the output space $\mathcal{Y} = \{0, 1\}$ consists of the binary labels and our hypothesis space is $\mathcal{H} = \{h_{\mathbf{w}}(\mathbf{x}) = \mathbb{1}(\mathbf{w}^\top \mathbf{x} \leq 0) \mid \mathbf{w} \in \mathbb{R}^d\}$. The loss we aim to bound is the misclassification error $\tilde{l}(\mathbf{w}, \mathbf{x}, y) = \mathbb{1}(h_{\mathbf{w}}(\mathbf{x}) \neq y) \in [0, 1]$. As prior over weights $\mathbf{w}$, we use a Gaussian $P_{\mu_P, \sigma_P^2}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mu_P, \sigma_P^2)$ with (meta-learnable) mean vector $\mu_P \in \mathbb{R}^2$ and fixed variance σ_P^2. As hyper-prior we use a zero-centered Gaussian, i.e., $\mathcal{P}(\mu_P) = \mathcal{N}(\mu_P|\mathbf{0}, \sigma_{\mathcal{P}}^2)$ with variance $\sigma_{\mathcal{P}}^2$. To construct posteriors $Q_i = Q(S_i, P)$ over the weight vectors $\mathbf{w}$, we use a logistic regression loss $l(\mathbf{w}, \mathbf{x}, y) = -y \log g(\mathbf{w}^\top \mathbf{x}) - (1 - y) \log(1 - g(\mathbf{w}^\top \mathbf{x}))$ where $g(\mathbf{z}) := 1/(1 + \exp(-\mathbf{z}))$ is the sigmoid function. We use the corresponding Gibbs posteriors with $\beta = \sqrt{m}$:

$$\log Q_i(\mathbf{w}) = \log P_{\mu_P, \sigma_P^2}(\mathbf{w}) + \frac{1}{\sqrt{m}} \sum_{j=1}^{m} l(\mathbf{w}, \mathbf{x}_{ij}, y_{ij}) + \text{const.} \tag{18}$$

Similarly, we use the PACOH in (13) with $\lambda = \sqrt{n}$ as hyper-posterior. Note that, to reflect the standard practice in classification, we use the negative log-likelihood loss for training and the misclassification error to evaluate the bound (i.e., Theorem 4).

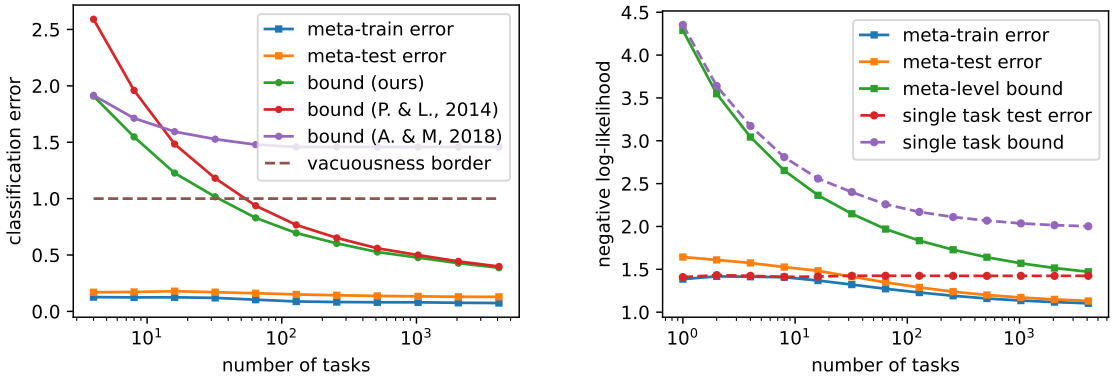
Data-generating process. Each task corresponds to a vector $\mathbf{w}_i^* \in \mathbb{R}^d$ with $d = 2$ that is sampled i.i.d. from the task distribution $\mathcal{T} = \mathcal{N}(\mu_{\mathcal{T}}, \sigma_{\mathcal{T}}^2 \mathbf{I})$ with $\mu_{\mathcal{T}} = 10 \cdot \mathbf{1}$ and $\sigma_{\mathcal{T}} = 3 \cdot \mathbf{1}$. The inputs $\mathbf{x}$ are sampled i.i.d. from $p(\mathbf{x}) = \mathcal{U}([-1, 1]^d)$ and the labels are conditionally independent draws from the Bernoulli distribution $p(y = 1|\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x})$. For the evaluation of the bounds we use $m = 5$ data points per task.

Comparison of the empirical bounds. We empirically evaluate the meta-learning bounds as well as the meta-train and -test error in the classification setting with $\sigma_P = 10 \cdot \mathbf{1}$ and $\sigma_{\mathcal{P}} = 20 \cdot \mathbf{1}$. Figure 2a displays the comparison of the bounds and errors across a varying number of tasks n. In particular, we compare the PACOH bound in Corollary 10 with the meta-learning bounds of Pentina and Lampert (2014) as well as Amit and Meir (2018). In the classification setting, the misclassification loss is bounded in $[0, 1]$. Thus, a trivial upper bound is 1 and any bound value larger than 1 is vacuous. For our setting with $m = 5$ number of data points per task, the bound of Amit and Meir (2018) is always vacuous. In contrast, our bound, as well as the one of Pentina and Lampert (2014), provide non-trivial bounds (< 1) for more than 30 meta-training tasks. Our PACOH bound is always tighter than Pentina and Lampert (2014) which is due to our slightly improved proof technique.

6.2 Linear regression

6.2.1 Setting

Model. For linear regression, the hypothesis space is the family of linear predictors $\mathcal{H} = \{h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} \mid \mathbf{w} \in \mathbb{R}^d\}$, mapping from the input space $\mathcal{X} = \mathbb{R}^d$ to the output



(a) PAC-Bayesian Logistic Regression

(b) PAC-Bayesian Linear Regression

Figure 2: Meta-train and meta-test error as well as corresponding worst-case bounds (e.g., Corollary 10) for $\delta = 0.1$, as functions of the number of meta-training tasks n.Left: Comparison of our bound with previous PAC-Bayesian meta-learning bounds for classification. Compared to Pentina and Lampert (2014) and Amit and Meir (2018), the presented PACOH bound is tighter and gives non-vacuous worst-case guarantees for $n > 32$. Right: Meta-level errors and bounds for PAC-Bayesian linear regression as well as the error for single-task learning and the bound of Theorem 11. The PACOH meta-learning bound offers meaningful worst-case guarantees for the transfer error which, as n grows, become tighter than bounds for per-task learning.

space $\mathcal{Y} = \mathbb{R}$. Given $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ and the model parameters $\mathbf{w}$, we consider a Gaussian likelihood with observation variance $\sigma^2 \in \mathbb{R}^+$, i.e., $p(y|\mathbf{x}, \mathbf{w}) = \mathcal{N}(y|\mathbf{w}^\top \mathbf{x}, \sigma^2)$. Our loss function, the negative log-likelihood, is $l(\mathbf{w}, \mathbf{x}, y) = \frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}(y - \mathbf{w}^\top \mathbf{x})^2$. As for the classification setting, we use a Gaussian prior $P_{\mu_P, \sigma_P^2}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mu_P, \sigma_P^2)$ with learnable mean μ_P and fixed variance σ_P^2 and a Gaussian hyper-prior, $\mathcal{P}(\mu_P) = \mathcal{N}(\mu_\mathcal{P}, \sigma_\mathcal{P}^2 \mathbf{I})$.

Data-generating process. We consider a synthetic meta-learning environment where each task τ_i corresponds to a vector $\mathbf{w}_i^* \in \mathbb{R}^d$ that is sampled i.i.d. from the task distribution $\mathcal{T} = \mathcal{N}(\mu_\mathcal{T}, \sigma_\mathcal{T}^2 \mathbf{I})$ with $\mu_\mathcal{T} = \frac{1}{5} \cdot \mathbf{1}$ and $\sigma_\mathcal{T} = \frac{1}{10} \cdot \mathbf{1}$. Each data point corresponding to the i-th task is generated as follows: The input is sampled from a Gaussian $\mathbf{x} \sim p(\mathbf{x}) = \mathcal{N}(0, \sigma_\mathbf{x}^2 \mathbf{I})$ with $\sigma_\mathbf{x}^2 = 1$ and the associated output is given by $y = (\mathbf{w}_i^*)^\top \mathbf{x} + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is observation noise standard deviation $\sigma_\epsilon = \frac{1}{3}$. Thus, we can write the conditional label distribution as $p_{\mathbf{w}_i^*}(y|\mathbf{x}) = \mathcal{N}((\mathbf{w}_i^*)^\top \mathbf{x}, \sigma_\epsilon^2)$. Consequently, the data distribution follows as $\mathcal{D}_i = p(\mathbf{x})\, p_{\mathbf{w}_i^*}(y|\mathbf{x})$.

6.2.2 Bounding the cumulant-generating functions for linear regression

To compute the bounds in Theorem 4 and Corollary 10, we need to bound the cumulant-generating functions (CGFs) $\Psi^\mathrm{I}(\beta)$ and $\Psi^\mathrm{II}(\gamma)$. In the classification setting, this was straightforward since the loss function is bounded in $[0, 1]$ and we could use (10). However, in the linear regression setting, the loss function is typically unbounded. In particular, we use negative log-likelihood with an i.i.d. Gaussian likelihood function, constituting a

generalization of the squared loss. Hence, we need to use the particularities of our data-generating process in Section 6.2.1 to derive a bound for the corresponding CGFs. To give an intuition, the CGFs quantify the difficulty of the learning problem: $\Psi^{\mathrm{I}}(\beta)$ quantifies the average difficulty of the learning tasks $\tau_1, ..., \tau_n$ with respect to the prior knowledge embedded in the two-level prior, i.e., the hierarchical model over hypotheses comprised of the hyper-prior and prior. $\Psi^{\mathrm{II}}(\beta)$ can be understood as quantifying the difficulty of the meta-learning environment $\mathcal{T}$ with respect to the hyper-prior $\mathcal{P}$. We use the same prior and hyper-prior as in the linear regression case above.

Bounding $\Psi^{\mathbf{I}}(\beta)$: First, we aim to bound $\Psi^{\mathrm{I}}(\beta)$ which is defined as

$$\Psi^{\mathrm{I}}(\beta) = \frac{1}{n\beta} \sum_{i=1}^{n} \sum_{j=1}^{m} \underbrace{\log \mathbb{E}_{\mathcal{P}} \mathbb{E}_{P} \mathbb{E}_{\mathcal{D}_i} \left[e^{\frac{\beta}{m} V_{ij}^{\mathrm{I}}} \right]}_{:=\Gamma_i^{\mathrm{I}}(\beta/m)} = \frac{m}{n\beta} \sum_{i=1}^{n} \Gamma_i^{\mathrm{I}}(\underbrace{\beta/m}_{\gamma}) \,, \tag{19}$$

wherein $V_{ij}^{\mathrm{I}} = \mathcal{L}(h, \mathcal{D}_i) - l(h, z_{ij}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_i} \left[l(\mathbf{w}, \mathbf{x}, y) \right] - l(\mathbf{w}, \mathbf{x}_{ij}, y_{ij})$. Note that, when defining Γ_i, we omit the index j since $V_{ij}^{\mathrm{I}}, j = 1, ..., m$ are distributed identically, and, thus, $\Gamma_i = \Gamma_{ij} = \Gamma_{ij'}$. Writing $\gamma := \beta/m$, we show in Appendix B that the cumulant-generating function can be bounded as follows:

$$\Gamma_i^{\mathrm{I}}(\gamma) = \log \mathbb{E}_{\mathcal{P}} \mathbb{E}_{P} \mathbb{E}_{\mathcal{D}_i} e^{\gamma V_{ij}^{\mathrm{I}}} \leq \frac{\gamma^2 s_i^2}{2(1 - \gamma c_i)} \quad \forall \gamma \in (0, 1/c_i) \tag{20}$$

with $s_i^2 = \frac{\vartheta_i}{\sigma^2}(\frac{1}{\gamma} - c_i) + \frac{c_i}{\gamma}$, $c_i = \frac{d}{\sigma^2}\sigma_{\mathbf{x}}^2(\sigma_P^2 + \sigma_{\mathcal{P}}^2) + \frac{\gamma}{\sigma^4} d\sigma_{\mathbf{x}}^2(\sigma_P^2 + \sigma_{\mathcal{P}}^2)\vartheta_i - \frac{\vartheta_i}{\sigma^2}$ and $\vartheta_i = \sigma_{\mathbf{x}}^2 ||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2$. Thus, for $\beta = \sqrt{m}$, we have that $\Phi^{\mathrm{I}}(1/\sqrt{m}) \leq \frac{1}{n} \sum_{i=1}^{n} \frac{s_i^2}{2(\sqrt{m} - c_i)}$ which is a monotonically decreasing positive function of m.

Bounding $\Psi^{\mathbf{II}}(\lambda)$: As shown in Appendix A.1,

$$\Psi^{\mathrm{II}}(\lambda) = \frac{1}{\lambda} \sum_{i=1}^{n} \log \mathbb{E}_{\mathcal{P}} \mathbb{E}_{\mathcal{T}} \left[e^{\frac{\lambda}{n} V_i^{\mathrm{II}}} \right] = \frac{n}{\lambda} \underbrace{\log \mathbb{E}_{\mathcal{P}} \mathbb{E}_{\mathcal{T}} \left[e^{\frac{\lambda}{n} V^{\mathrm{II}}} \right]}_{\Gamma^{\mathrm{II}}(\lambda/n)} \tag{21}$$

is defined as the sum of the cumulant generating functions of the random variable $V^{\mathrm{II}} := \mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}_h} \mathbb{E}_{P\sim\mathcal{P}} \left[\mathcal{L}(Q(P,S),\mathcal{D}) \right] - \mathcal{L}(Q(P,S),\mathcal{D})$. To simplify the notation, we define $\kappa := \lambda/n$. The resulting CGF $\Gamma^{\mathrm{II}}(\kappa)$ is sub-gamma, i.e.,

$$\Gamma^{\mathrm{II}}(\kappa) = \log \mathbb{E}_{\mathcal{T}} \mathbb{E}_{\mathcal{P}} \left[e^{\kappa V^{\mathrm{II}}} \right] \leq \frac{\kappa^2 s_{\mathrm{II}}^2}{2(1 - \kappa c_{\mathrm{II}})} \quad \forall \kappa \in (0, 1/c_{\mathrm{II}}) \tag{22}$$

with $c_{\mathrm{II}} = \frac{\sigma_{\mathbf{x}}^2}{\sigma^2}(\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)$ and $s_{\mathrm{II}}^2 = \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} c_{\mathrm{II}} ||\mu_{\mathcal{T}}||^2 + d c_{\mathrm{II}}^2$. Thus, for $\lambda = \sqrt{n}$ we have $\Gamma^{\mathrm{II}}(1/\sqrt{n}) \leq \frac{s_{\mathrm{II}}^2}{2(\sqrt{n} - c_{\mathrm{II}})}$, a monotonically decreasing positive function of n.

6.2.3 Empirical evaluation of the bound values

We now aim to inspect the bound values of the *PACOH* $\mathcal{Q}^*$. First, we focus on the linear regression setup. We generate our synthetic meta-learning environments as described in

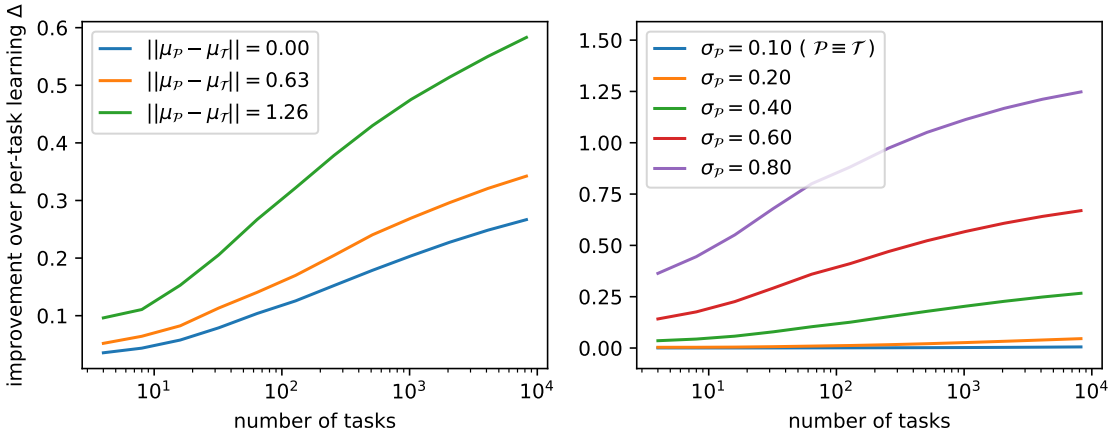


Figure 3: Improvement of the meta-learning bound over the per-task bound (i.e., Δ in Proposition 12) in response to the number of meta-training tasks n. Left: Different degrees of misspecification of the hyper-prior mean compared to the true task distribution mean, fixed $\sigma_\mathcal{P} = 0.4$. Right: Differing hyper-prior variances, fixed $\mu_\mathcal{P} = 0.2$. The smaller the hyper-prior informativeness, i.e., larger $\sigma_\mathcal{P}$ and/or larger mean misspecification $||\mu_\mathcal{P} - \mu_\mathcal{T}||$, the bigger the improvement of meta-learning over the per-task bound.

Section 6.2.1 with $d = 5$ dimensional features and $m = 5$ data points per task. Moreover, we set $\mu_\mathcal{P} = 0$, $\sigma^2_\mathcal{P} = \frac{1}{4}$, $\sigma^2_P = \frac{1}{25}$ and $\sigma^2_\mathbf{x} = 1$. Figure 2b plots the meta-train and meta-test (transfer) error together with our PAC-Bayesian transfer error bound of Corollary 10. In addition, we also display the test error for per-task learning and its corresponding bound from Theorem 11. Note that the displayed bounds are worst-case bounds, i.e., hold with high probability over the sampled tasks, whereas the plotted errors are means. The bounds can be considered fairly tight, giving values in the value range of the generalization error they bound. As discussed in Section 5, for a small number of tasks n, the meta-learning bound is almost identical to the per-task learner bound. However, with growing n, it quickly improves upon the per-task learner bound and offers much better performance guarantees which reflect the benefits of meta-learning.

Note that the linear and logistic regression settings which we consider here are very simplistic. The main goal of this section is to provide empirical evidence that the presented bounds are correct and behave as expected.

6.3 Improvement of meta-learning over per-task learning bounds

Finally, we provide numerical examples of the improvement of the meta-learning over single-task learning bounds, discussed in detail in Section 5. Again, we consider the linear regression setting from Section 6.2.1 with $d = 10$ and $m = 5$ data points per task. In Figure 3, we plot the difference between meta-learning and single-task learning bounds, i.e., Δ as defined in (17), in response to the number of tasks n. In the left sub-figure, we vary the degree of misspecification in the hyper-prior mean relative to the true task distribution mean, i.e., $||\mu_\mathcal{P} - \mu_\mathcal{T}||$. As one would expect, the larger the misspecification of the hyper-prior, the more we can gain from meta-learning which is able to correct such misspecification by moving the

hyper-posterior mean towards the true task distribution when given enough meta-training tasks. In the right sub-figure, we set $\mu_{\mathcal{P}} = \mu_{\mathcal{T}}$, i.e., no misspecification, but vary the hyper-prior standard deviation $\sigma_{\mathcal{P}}$. Similarly, we observe that, the higher $\sigma_{\mathcal{P}}$, the more uninformative is the hyper-prior which leads to a bigger improvement of the meta-learning bound over the per-task learning bound. If the variance of the hyper-prior is high, then the meta-learner already provably improves over the single task learner with only 10–20 meta-training tasks. On the other hand, if the hyper-prior is equivalent to the true task distribution, i.e., $\mathcal{P} \equiv \mathcal{T}$, then it is impossible to improve upon $\mathcal{P}$ by meta-learning a better $\mathcal{Q}$.

7. Meta-Learning Algorithms based on the PACOH

After having introduced the closed-form solution of the PAC-Bayesian meta-learning problem in Section 4, we now discuss how to translate the *PACOH* into a practical meta-learning algorithm when employing GPs and BNNs as base learners. For that, we assume a parametric family of priors $\{P_\phi | \phi \in \Phi\}$ wherein priors P_ϕ are governed by a parameter vector $\phi \in \Phi$. Accordingly, we can express the hyper-posterior and hyper-prior as distributions over prior parameters, i.e., $\mathcal{Q}(\phi) \in \mathcal{M}(\Phi)$ and $\mathcal{P}(\phi) \in \mathcal{M}(\Phi)$ respectively. Later, we will discuss in further detail our particular choice P_ϕ for GPs and BNNs, respectively.

7.1 Approximating the PACOH

Given the hyper-prior and generalized marginal log-likelihood (MLL) function $\log Z_\beta(S_i, P)$, we can compute the PACOH $\mathcal{Q}^*$ up to the normalization constant Z^{II}. Such setup lends itself to classical approximate inference methods (Blei et al., 2017; Liu and Wang, 2016). Thus, Proposition 9 yields an entire class of possible meta-learning methods. We now briefly discuss three tractable approximations of $\mathcal{Q}^*$ which we evaluate empirically in Section 8. The corresponding approximate distributions and approximate inference updates are summarized in Table 1.

Maximum A Posteriori (MAP). This is the simplest and most crude method, which approximates $\mathcal{Q}^*(\phi)$ by a Dirac measure $\delta_P(\phi^*)$ at the mode of $\mathcal{Q}^*$, i.e., $\phi^* = \arg\max_{\phi \in \Phi} \mathcal{Q}^*(\phi)$.

Variational Inference (VI). In the case of VI (Blei et al., 2017), we restrict the space of considered hyper-posteriors to a parametric variational family $\mathcal{F} = \{\tilde{\mathcal{Q}}_\upsilon | \upsilon \in \mathcal{U}\} \subset \mathcal{M}(\Phi)$ wherein the variational posterior $\tilde{\mathcal{Q}}_\upsilon$ is governed by a parameter vector $\upsilon \in \mathcal{U}$. Then, we aim to find a $\tilde{\mathcal{Q}}_\upsilon$ that minimizes the KL-divergence to $\mathcal{Q}^*$, that is,

$$\upsilon^* = \arg\min_{\upsilon \in \mathcal{U}} D_{KL}(\tilde{\mathcal{Q}}_\upsilon || \mathcal{Q}^*) \,. \tag{23}$$

In fact, it can be shown that the minimizing variational distribution $\tilde{\mathcal{Q}}_\upsilon$ of (23) is the same as the minimizer of the bound in (12) under the constraint $\mathcal{Q} \in \mathcal{F}$ (see Appendix A.9 for proof). Consequently, we can directly use (12) as an optimization objective.

Stein Variational Gradient Descent (SVGD). SVGD (Liu and Wang, 2016) approximates $\mathcal{Q}^*$ as a set of K particles, that is, $\tilde{\mathcal{Q}} = \frac{1}{K} \sum_{k=1}^{K} \delta(\phi_k)$. Initially, we sample K particles $\phi_k \sim \mathcal{P}$ from the hyper-prior. For notational brevity, we stack the particles into a $K \times \dim(\phi)$ matrix $\boldsymbol{\phi} := [\phi_1, ..., \phi_K]^\top$. Then, the method iteratively transports the set of

	Approximating Distribution	`Init_Approx_Inference`
MAP	$\tilde{\mathcal{Q}} = \delta(\tilde{\phi})$	$\tilde{\phi} \sim \mathcal{P}$
SVGD	$\tilde{\mathcal{Q}} = \frac{1}{K}\sum_{k=1}^{K} \delta(\tilde{\phi}_k),\ \tilde{\boldsymbol{\phi}} = \left[\tilde{\phi}_1, ..., \tilde{\phi}_K\right]^\top$	$\tilde{\phi}_k \sim \mathcal{P},\ k = 1, ..., K$
VI	$\tilde{\mathcal{Q}}_\upsilon = \mathcal{N}(\mu_\mathcal{Q}, \sigma^2_\mathcal{Q}),\ \upsilon = (\mu_\mathcal{Q}, \sigma^2_\mathcal{Q})$	$\upsilon = (\mu_\mathcal{P}, \sigma^2_\mathcal{P})$

	`Sample_Prior_Params`	`Approx_Inference_Update`
MAP	$\phi \leftarrow \tilde{\phi}$	$\tilde{\phi} \leftarrow \tilde{\phi} + \eta \nabla_\phi \log \mathcal{Q}^*(\tilde{\phi})$
SVGD	$\phi_k \leftarrow \tilde{\phi}_k$	$\tilde{\boldsymbol{\phi}} \leftarrow \tilde{\boldsymbol{\phi}} + \eta\ \mathbf{K}\ \nabla_{\boldsymbol{\phi}} \mathbf{ln} \tilde{\mathcal{Q}}^* + \nabla_{\phi} \mathbf{K}$
VI	$\phi_k \leftarrow \mu_\mathcal{Q} + \sigma_\mathcal{Q} \odot \epsilon, \epsilon \sim \mathcal{N}(0, \mathbf{I})$	$\upsilon \leftarrow \upsilon + \frac{\eta}{K}\sum_{k=1}^{K} \nabla_\upsilon \left[\log \mathcal{Q}^*(\phi_k) - \log \tilde{\mathcal{Q}}_\upsilon(\phi_k)\right]$

Table 1: Summary of approximations of the *PACOH* $\mathcal{Q}^*$

particles to match $\mathcal{Q}^*$, by applying a form of functional gradient descent that minimizes $D_{KL}(\tilde{\mathcal{Q}}|\mathcal{Q}^*)$ in the reproducing kernel Hilbert space induced by a kernel function $k(\cdot, \cdot)$. In particular, in each iteration, we update the particle matrix using the SVGD update rule:

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \eta\ \mathbf{K}\ \nabla_{\boldsymbol{\phi}} \mathbf{ln} \mathcal{Q}^* + \nabla_{\boldsymbol{\phi}} \mathbf{K} \tag{24}$$

where $\nabla_{\boldsymbol{\phi}} \mathbf{ln} \mathcal{Q}^* := [\nabla_{\phi_1} \log \mathcal{Q}^*(\phi_1), ..., \nabla_{\phi_K} \log \mathcal{Q}^*(\phi_K)]^\top$ denotes the matrix of stacked score gradients, $\mathbf{K} := [k(\phi_k, \phi_{k'})]_{k,k'}$ the kernel matrix induced by a kernel function $k(\cdot, \cdot)$ and η the step size for the SVGD updates.

7.2 Meta-Learning Gaussian Process Priors

Setup. In GP regression, each data point corresponds to a tuple $z_{i,j} = (x_{i,j}, y_{i,j})$. For the i-th dataset, we write $S_i = (\mathbf{X}_i, \mathbf{y}_i)$, where $\mathbf{X}_i = (x_{i,1}, ..., x_{i,m_i})^\top$ and $\mathbf{y}_i = (y_{i,1}, ..., y_{i,m_i})^\top$. GPs are a Bayesian method in which the prior $P_\phi(h) = \mathcal{GP}\left(h|m_\phi(x), k_\phi(x, x')\right)$ is specified by a kernel $k_\phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and a mean function $m_\phi : \mathcal{X} \to \mathbb{R}$.

Since we are interested in meta-learning such prior, we define the mean and kernel function both as parametric functions. Similar to Wilson et al. (2016) and Fortuin and Rätsch (2019), we instantiate m_ϕ and k_ϕ as neural networks, and meta-learn the parameter vector ϕ. To ensure the positive-definiteness of the kernel, we use the neural network as feature map $\varphi_\phi(x)$ on top of which we apply a squared exponential (SE) kernel. Accordingly, the parametric kernel reads as $k_\phi(x, x') = \frac{1}{2}\exp\left(-||\varphi_\phi(x) - \varphi_\phi(x')||_2^2\right)$.

As typical for GPs, we assume a Gaussian likelihood $p(\mathbf{y}|\mathbf{h}) = \mathcal{N}(\mathbf{y}; h(\mathbf{x}), \sigma^2 I)$ where σ^2 is observation noise variance. Moreover, we choose $\lambda = n$, $\beta = m$, so that the closed-form GP posterior coincides with the PAC-optimal posterior Q^*. In this case, the empirical loss under the GP posterior Q^* coincides with the negative log-likelihood of regression targets $\mathbf{y}_i$, that is, $\hat{\mathcal{L}}(Q^*, S_i) = -\frac{1}{m_i}\log p(\mathbf{y}_i|\mathbf{X}_i)$. Finally, we use a Gaussian hyper-prior $\mathcal{P} = \mathcal{N}(0, \sigma^2_\mathcal{P} I)$ over the GP prior parameters ϕ.

Algorithm. Depending on which approximate inference method from Section 7.1 we choose, the resulting meta-learning algorithms differ slightly. For this reason, we describe the algorithm in terms of generic operations that are specified in Table 1.

Algorithm 1 PACOH-GP - Approximate inference of $\mathcal{Q}^*$

Input: hyper-prior $\mathcal{P}$, datasets $S_1, ..., S_n$, step size η
$\tilde{\mathcal{Q}} \leftarrow$ `Init_Approx_Inference`()
while not converged **do**
  $\{\phi_1, ..., \phi_K\} \leftarrow$ `Sample_Prior_Params`$(\tilde{\mathcal{Q}})$
  for $k = 1, ..., K$ **do**
    for $i = 1, ..., n$ **do**
      $\log Z_{m_i}(S_i, P_{\phi_k}) \leftarrow \log p(\mathbf{y}_i|\mathbf{X}_i, \phi_k)$
    end for
    $\nabla_{\phi_k} \log \mathcal{Q}^* \leftarrow \nabla_{\phi_k} \log \mathcal{P} + \sum_{i=1}^{n} \frac{1}{m_i+1} \nabla_{\phi_k} \log Z_{m_i}(S_i, P_{\phi_k})$
  end for
  $\tilde{\mathcal{Q}} \leftarrow$ `Approx_Inference_Update`$(\tilde{\mathcal{Q}}, \nabla_{\phi_k} \log \mathcal{Q}^*, \eta)$
end while
Output: Approximate hyper-posterior $\tilde{\mathcal{Q}}$

First, we initialize the approximate hyper-posterior $\tilde{\mathcal{Q}}$. Then, in each iteration, we sample K prior parameters[3] ϕ_k from $\tilde{\mathcal{Q}}$, and compute the corresponding score gradients of $\mathcal{Q}^*$,

$$\nabla_{\phi_k} \log \mathcal{Q}^*(\phi_k) = \nabla_{\phi_k} \log \mathcal{P}(\phi_k) + \sum_{i=1}^{n} \frac{1}{m_i + 1} \nabla_{\phi_k} \log Z_{m_i}(S_i, P_{\phi_k}) \,. \tag{25}$$

In our setup, $\log Z_{m_i}(S_i, P_\phi) = \log p(\mathbf{y}_i|\mathbf{X}_i, \phi)$ is the MLL of the GP which can be computed in closed form, in particular,

$$\log p(\mathbf{y}|\mathbf{X}, \phi) = -\frac{1}{2} \left(\mathbf{y} - m_{\mathbf{X},\phi}\right))^\top \tilde{K}_{\mathbf{X},\phi}^{-1} \left(\mathbf{y} - m_{\mathbf{X},\phi}\right) - \frac{1}{2} \log |\tilde{K}_{\mathbf{X},\phi}| - \frac{m_i}{2} \log 2\pi \,, \tag{26}$$

where $\tilde{K}_{\mathbf{X},\phi} = K_{\mathbf{X},\phi} + \sigma^2 I$, with the kernel matrix $K_{\mathbf{X},\phi} = (k_\phi(x_l, x_k))_{l,k=1}^{m_i}$ and mean vector $m_{\mathbf{X},\phi} = (m_\phi(x_1), ..., m_\phi(x_{m_i}))^\top$. Based on the score gradients $\nabla_{\phi_k} \log \mathcal{Q}^*(\phi_k)$, we can then update the approximate hyper-posterior $\tilde{Q}$, using the corresponding approximate inference method. Algorithm 1 summarizes the resulting generic meta-learning procedure for GPs which we henceforth refer to as *PACOH-GP*.

7.3 Meta-Learning Bayesian Neural Network Priors

Setup. Let $h_\theta : \mathcal{X} \to \mathcal{Y}$ be a function parametrized by a neural network (NN) with weights $\theta \in \Theta$. Using the NN mapping, we define a conditional distribution $p(y|x, \theta)$. For regression, we may set $p(y|x, \theta) = \mathcal{N}(y|h_\theta(x), \sigma^2)$, where σ^2 is the observation noise variance. We treat $\log \sigma$ as a learnable parameter similar to the neural network weights θ such that a hypothesis coincides with a tuple $h = (\theta, \log \sigma)$. For classification, we choose $p(y|x, \theta) = \text{Categorical}(\text{softmax}(h_\theta(x)))$. Our loss function is the negative log-likelihood $l(\theta, z) = -\log p(y|x, \theta)$.

Next, we define a family of priors $\{P_\phi : \phi \in \Phi\}$ over the NN parameters θ. For computational convenience, we employ diagonal Gaussian priors, that is, $P_{\phi_l} = \mathcal{N}(\mu_{P_k}, \text{diag}(\sigma^2_{P_k}))$

3. Note that for MAP inference, we always have $K = 1$.

with $\phi := (\mu_{P_k}, \log \sigma_{P_k})$. Note that we represent σ_{P_k} in the log-space to avoid additional positivity constraints. In fact, any parametric distribution that supports re-parametrized sampling and has a tractable log-density (e.g., normalizing flows (cf., Rezende and Mohamed, 2015)) could be used. Moreover, we use a zero-centered, spherical Gaussian hyper-prior $\mathcal{P} := \mathcal{N}(0, \sigma_{\mathcal{P}}^2 I)$ over the prior parameters ϕ.

Approximating the marginal log-likelihood. Unlike for GPs, the (generalized) MLL $\log Z_\beta(S_i, P_\phi) = \log \mathbb{E}_{\theta \sim P_\phi} e^{-\beta_i \hat{\mathcal{L}}(\theta, S_i)}$ is intractable for BNNs. Estimating and optimizing $\log Z(S_i, P_\phi)$ is not only challenging due to the high-dimensional expectation over Θ but also due to numerical instabilities inherent in computing $e^{-\beta_i \hat{\mathcal{L}}(\theta, S_i)}$ when m_i and, thus, β_i is large. Aiming to overcome these issues, we compute numerically stable Monte Carlo estimates of $\nabla_\phi \log Z_{\beta_i}(S_i, P_{\phi_k})$ by combining the LogSumExp (LSE) with the re-parametrization trick (Kingma and Welling, 2014). In particular, we draw L samples $\theta_l := f(\phi_k, \epsilon_l) = \mu_{P_k} + \sigma_{P_k} \odot \epsilon_l,\ \epsilon_l \sim N(0, I)$ and compute the generalized MLL estimate as

$$\log \hat{Z}_{\beta_i}(S_i, P_\phi) := \ \mathrm{LSE}_{l=1}^{L} \left(-\beta_i \hat{\mathcal{L}}(\theta_l, S_i) \right) - \log L \ . \tag{27}$$

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

$$\nabla_\phi \log \hat{Z}_{\beta_i}(S_i, P_\phi) = -\beta_i \sum_{l=1}^{L} \underbrace{\frac{e^{-\beta_i \hat{\mathcal{L}}(\theta_l, S_i)}}{\sum_{l=1}^{L} e^{-\beta_i \hat{\mathcal{L}}(\theta_l, S_i)}}}_{\text{softmax}} \underbrace{\nabla_\phi f(\phi, \epsilon_l)^\top}_{\substack{\text{re-param.} \\ \text{Jacobian}}} \underbrace{\nabla_{\theta_l} \hat{\mathcal{L}}(\theta_l, S_i)}_{\text{score}} \tag{28}$$

Note that $\log \hat{Z}(S_i, P_\phi)$ is a consistent but not an unbiased estimator of $\log Z(S_i, P_\phi)$. The following proposition ensures us that we still minimize a valid upper bound:

Proposition 13 *In expectation, replacing* $\log Z(S_i, P_\phi)$ *in (12) with the Monte Carlo estimate* $\log \hat{Z}(S_i, P_\phi)$ *still yields a valid upper bound of the transfer error* $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ *for any* $L \in \mathbb{N}$*, i.e.,*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq (12) \leq -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{\mathcal{Q}} \left[\mathbb{E}_{\theta_1, \ldots, \theta_L \sim P} \left[\log \hat{Z}_\beta(S_i, P_\phi) \right] \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C \ .$$

Moreover, by the law of large numbers, we have that $\log \hat{Z}(S_i, P) \xrightarrow{\text{a.s.}} \log Z(S_i, P)$ as $L \to \infty$, that is, for large sample sizes L, we recover the original PAC-Bayesian bound in (12). In the opposite edge case, i.e., $L = 1$, the boundaries between tasks vanish meaning that the meta-training data $\{S_1, ..., S_n\}$ is treated as if it were one large dataset $\bigcup_i S_i$ (see Appendix D.3 for further discussion). As an alternative to our proposed MLL estimator in (27), we could use a Russian roulette estimator (Kahn, 1955; Luo et al., 2020) which would allows us to construct unbiased Monte Carlo estimates of the generalized MLL. However, albeit unbiased, Russian roulette estimators suffer from high variance and are hard to implement efficiently in parallel (Pharr et al., 2016). Thus, we found (27) to be the more favorable choice in practice.

Algorithm. Algorithm 2 summarizes the proposed meta-learning method, henceforth referred to as *PACOH-NN*. It follows similar steps as *PACOH-GP* but uses the proposed generalized MLL estimator $\log \hat{Z}_{\beta_i}(S_i, P_\theta)$ instead of a closed-form solution. To estimate the

Algorithm 2 PACOH-NN - Approximate inference of $\mathcal{Q}^*$

Input: hyper-prior $\mathcal{P}$, datasets $S_1, ..., S_n$, step size η
$\tilde{\mathcal{Q}} \leftarrow$ `Init_Approx_Inference`()
while not converged **do**
  $\{\phi_1, ..., \phi_K\} \leftarrow$ `Sample_Prior_Params`($\tilde{\mathcal{Q}}$)
  for $k = 1, ..., K$ **do**
    $\{\theta_1, ..., \theta_L\} \sim P_{\phi_k}$ // sample NN-parameters from prior
    for $i = 1, ..., n$ **do**
      $\log \hat{Z}(S_i, P_{\phi_k}) \leftarrow \text{LSE}_{l=1}^{L} \left(-\beta_i \hat{\mathcal{L}}(\theta_l, S_i)\right) - \log L$ // estimate generalized MLL
    end for
    $\nabla_{\phi_k} \log \hat{\mathcal{Q}}^*(\phi_k) \leftarrow \nabla_{\phi_k} \log \mathcal{P}(\phi_k) + \sum_{i=1}^{n} \frac{\lambda}{n\beta_i + \lambda} \nabla_{\phi_k} \log \hat{Z}_{\beta_i}(S_i, P_{\phi_k})$ // compute score
  end for
  $\tilde{\mathcal{Q}} \leftarrow$ `Approx_Inference_Update`($\tilde{\mathcal{Q}}, \nabla_{\phi_k} \log \mathcal{Q}^*(\phi_k), \eta$)
end while
Output: Approximate hyper-posterior $\tilde{\mathcal{Q}}$

hyper-posterior score $\nabla_{\phi_{k'}} \log \hat{\mathcal{Q}}^*(\phi_{k'}) = \nabla_{\phi_k} \log \mathcal{P}(\phi_k) + \sum_{i=1}^{n} \frac{\lambda}{n\beta_i + \lambda} \nabla_{\phi_k} \log \hat{Z}_{\beta_i}(S_i, P_{\phi_k})$, we can use mini-batching over tasks, only estimating the generalized MLL for a random subset of $n_{bs} \leq n$ tasks and adjusting the (truncated) sum of MLL gradients by the factor $\frac{n}{n_{bs}}$.

If we use the mini-batched version in conjunction with the SVGD approximate inference, the resulting algorithm (see Algorithm 3 in Appendix D) maintains K particles to approximate the hyper-posterior, and in each forward step samples L NN-parameters (of dimensionaly $|\Theta|$) per prior particle that are deployed on a mini-batch of n_{bs} tasks to estimate the score of $\mathcal{Q}^*$. As a result, the total space complexity is on the order of $\mathcal{O}(|\Theta|K + L)$ and the computational complexity of the algorithm for a single iteration is $\mathcal{O}(K^2 + KLn_{bs})$.

A key advantage of *PACOH-NN* over previous methods for meta-learning BNN priors (e.g. Pentina and Lampert, 2014; Amit and Meir, 2018) is that it turns the previously nested optimization problem into a much simpler stochastic optimization problem. This makes meta-learning not only much more stable but also more scalable. In particular, we do not need to explicitly compute the task posteriors Q_i and can use mini-batching over tasks. As a result, the space and compute complexity do not depend on the number of tasks n. In contrast, *MLAP* (Amit and Meir, 2018) has a memory footprint of $\mathcal{O}(|\Theta|n)$ making meta-learning prohibitive for more than 50 tasks.

A central feature of *PACOH-NN* is that is comes with principled meta-level regularization in form of the hyper-prior $\mathcal{P}$, which combats overfitting to the meta-training tasks (Qin et al., 2018). As we show in our experiments, this allows us to successfully perform meta-learning with as little as 5 tasks. This is unlike the majority of popular meta-learners (Finn et al., 2017; Yoon et al., 2018; Garnelo et al., 2018, e.g.), which rely on a large number of tasks to generalize well on the meta-level (Qin et al., 2018; Rothfuss et al., 2021a).

8. Experiments

We now empirically evaluate the *PACOH* algorithms introduced in Section 7. In particular, we evaluate the *PACOH* approach with GPs and NNs as base learners, i.e., *PACOH-GP* and *PACOH-NN*, as well as the different variational approximations of $\mathcal{Q}$, i.e., *MAP, SVGD* and, in case of PACOH-GP, also *VI*. Comparing them to existing meta-learning approaches on various regression and classification environments, we demonstrate that our *PACOH*-based methods: (i) outperform previous meta-learning algorithms in *predictive accuracy*, (ii) improve the calibration of *uncertainty estimates*, (iii) are much more *scalable* than previous PAC-Bayesian meta-learners, and (iv) effectively combat *meta-overfitting*. Finally, we showcase how meta-learned *PACOH-NN* priors can be harnessed in a real-world *sequential decision making* task concerning peptide-based vaccine development.

8.1 Experiment Setup

Baselines. We use a *Vanilla GP* with squared exponential kernel and a *Vanilla BNN* with a zero-centered, spherical Gaussian prior and SVGD posterior inference (Liu and Wang, 2016) as baselines. Moreover, we compare our proposed approach against various popular meta-learning algorithms, including model-agnostic meta-learning (*MAML*) (Finn et al., 2017), Bayesian MAML (*BMAML*) (Yoon et al., 2018) and the PAC-Bayesian approach by Amit and Meir (2018) (*MLAP*). For the regression task experiments, we also compare with two versions of *MR-MAML* (Yin et al., 2020) which add an information bottleneck regularization to the NN weights (W) or the activations (A) to prevent meta-overfitting in MAML. Additionally, we consider neural processes (*NPs*) (Garnelo et al., 2018) and a GP with neural-network-based mean and kernel function, meta-learned by maximizing the marginal log-likelihood (*MLL-GP*) (Fortuin and Rätsch, 2019). Among all, *MLAP* is the most similar to our approach as it is neural-network-based and minimizes PAC-Bayesian bounds on the transfer error. However, unlike *PACOH-NN*, it relies on nested optimization of the task posteriors Q_i and the hyper-posterior $\mathcal{Q}$. *MLL-GP* is similar to *PACOH-GP* insofar that it also maximizes the sum of marginal log-likelihoods $\log Z_m(S_i, P_\phi)$ across tasks. However, unlike *PACOH-GP*, it lacks any form of meta-level regularization.

Regression environments. We consider two synthetic and four real-world meta-learning environments for *regression*. As synthetic environments, we employ *Sinusoids* of varying amplitude, phase, and slope as well as a 2-dimensional mixture of *Cauchy* distributions plus random GP functions. As real-world environments, we use datasets corresponding to different calibration sessions of the Swiss Free Electron Laser (*SwissFEL*) (Milne et al., 2017; Kirschner et al., 2019b), as well as data from the *PhysioNet* 2012 challenge, which consists of time series of electronic health measurements from intensive care patients (Silva et al., 2012), in particular, the Glasgow Coma Scale (*GCS*) and the hematocrit value (*HCT*). Here, the different tasks correspond to different patients. Moreover, we employ the Intel Berkeley Research Lab temperature sensor dataset (*Berkeley-Sensor*) (Madden, 2004) where the tasks require auto-regressive prediction of temperature measurements corresponding to sensors installed in different locations of the building. Further details can be found in Appendix E.1.

Classification environments. We conduct experiments with the multi-task *classification* environment *Omniglot* (Lake et al., 2015), consisting of handwritten letters across 50 alpha-

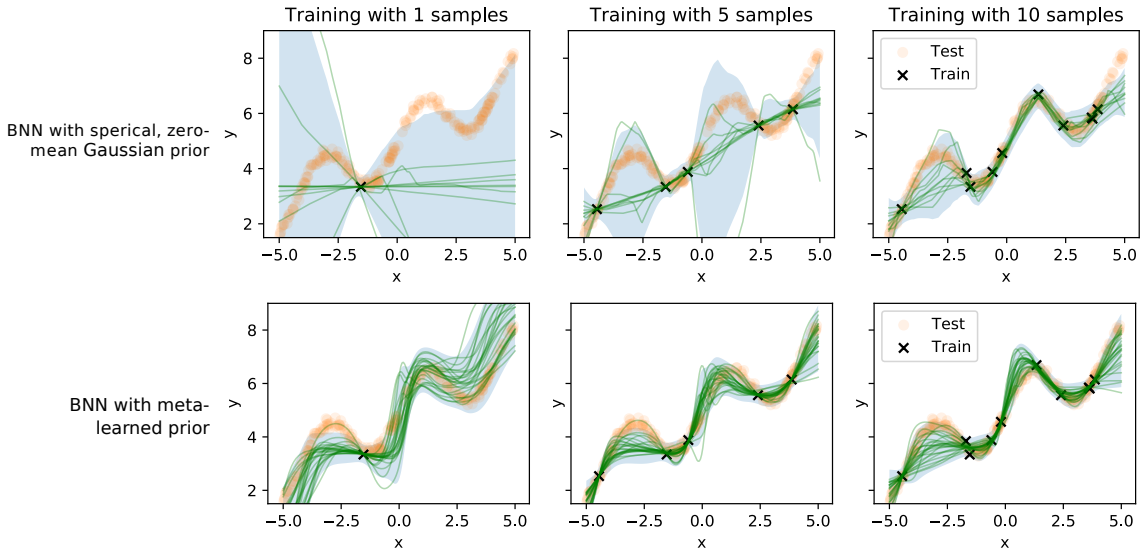


Figure 4: BNN posterior predictions with (top) standard Gaussian prior vs. (bottom) meta-learned prior. Meta-learning conducted on the *Sinusoids* environment. The meta-learned PACOH-NN-SVGD prior conveys useful inductive bias, improving the BNN predictions.

bets. Unlike previous work (e.g., Finn et al., 2017), we do not perform data augmentation and do not recombine letters of different alphabets. This preserves the data's original structure, where each task corresponds to classifying letters within an alphabet. In particular, one task corresponds to 5-way 5-shot classification of letters within an alphabet. This leaves us with much fewer tasks (30 train, 20 test tasks), making the environment more challenging and interesting for uncertainty quantification. This also allows us to include *MLAP* in the experiment, which hardly scales to more than 50 tasks.

8.2 Experiment Results

Qualitative example. Figure 4 illustrates *BNN* predictions on a sinusoidal regression task with a standard Gaussian prior as well as a *PACOH-NN* prior meta-learned with 20 tasks from the *Sinusoids* environment. We observe that the standard Gaussian prior provides poor inductive bias, not only leading to bad mean predictions away from the test points but also to poor 95% confidence intervals (blue shaded areas). In contrast, the meta-learned *PACOH-NN* prior encodes useful inductive bias towards sinusoidal function shapes, leading to better predictions and uncertainty estimates, even with minimal training data.

PACOH improves the predictive accuracy. Using the meta-learning environments and baseline methods that we introduced in Section 8.1, we perform a comprehensive benchmark study. Table 2 reports the results on the regression environments in terms of the root mean squared error (RMSE) on unseen test tasks. Table 4 reports the classification accuracy for the Omniglot classification environments. For the image classification problems, we only consider the neural network-based methods, since GPs generally perform poorly on high-dimensional problems. When comparing the different variational approximations of the PACOH, the SVGD approximation gives the lowest meta-test errors in the majority of

	Cauchy	SwissFel	Physionet-GCS	Physionet-HCT	Berkeley-Sensor
Vanilla GP	0.275 ± 0.000	0.876 ± 0.000	2.240 ± 0.000	2.768 ± 0.000	0.276 ± 0.000
Vanilla BNN (Liu and Wang, 2016)	0.327 ± 0.008	0.529 ± 0.022	2.664 ± 0.274	3.938 ± 0.869	0.109 ± 0.004
MLL-GP (Fortuin and Rätsch, 2019)	0.216 ± 0.003	0.974 ± 0.093	1.654 ± 0.094	2.634 ± 0.144	0.058 ± 0.002
MLAP (Amit and Meir, 2018)	0.219 ± 0.004	0.486 ± 0.026	2.009 ± 0.248	2.470 ± 0.039	0.050 ± 0.005
MAML (Finn et al., 2017)	0.219 ± 0.004	0.730 ± 0.057	1.895 ± 0.141	2.413 ± 0.113	$\mathbf{0.045 \pm 0.003}$
MR-MAML (W) (Yin et al., 2020)	0.227 ± 0.002	0.483 ± 0.021	1.643 ± 0.174	$\mathbf{2.306 \pm 0.162}$	0.059 ± 0.004
MR-MAML (A) (Yin et al., 2020)	0.228 ± 0.003	0.555 ± 0.045	1.556 ± 0.179	$\mathbf{2.275 \pm 0.155}$	0.066 ± 0.006
BMAML (Yoon et al., 2018)	0.225 ± 0.004	0.577 ± 0.044	1.894 ± 0.062	2.500 ± 0.002	0.073 ± 0.014
NP (Garnelo et al., 2018)	0.224 ± 0.008	0.471 ± 0.053	2.056 ± 0.209	2.594 ± 0.107	0.079 ± 0.007
PACOH-GP-MAP (ours)	0.213 ± 0.003	0.486 ± 0.055	1.492 ± 0.091	2.574 ± 0.058	0.056 ± 0.001
PACOH-GP-SVGD (ours)	0.209 ± 0.008	0.376 ± 0.024	1.498 ± 0.081	$\mathbf{2.361 \pm 0.047}$	0.065 ± 0.005
PACOH-GP-VI (ours)	0.218 ± 0.003	0.387 ± 0.019	$\mathbf{1.472 \pm 0.012}$	2.695 ± 0.040	0.095 ± 0.010
PACOH-NN-MAP (ours)	0.202 ± 0.003	0.375 ± 0.004	1.564 ± 0.200	2.480 ± 0.042	0.047 ± 0.001
PACOH-NN-SVGD (ours)	$\mathbf{0.195 \pm 0.001}$	$\mathbf{0.372 \pm 0.002}$	1.561 ± 0.061	$\mathbf{2.405 \pm 0.017}$	$\mathbf{0.043 \pm 0.001}$

Table 2: Comparison of standard and meta-learning algorithms in terms of test RMSE in 5 meta-learning environments for regression. We report the mean and standard deviation across 5 random seeds. *PACOH* achieves the best performance across the environments.

	Cauchy	SwissFel	Physionet-GCS	Physionet-HCT	Berkeley-Sensor
Vanilla GP	0.087 ± 0.000	0.135 ± 0.000	0.268 ± 0.000	$\mathbf{0.277 \pm 0.000}$	0.119 ± 0.000
Vanilla BNN (Liu and Wang, 2016)	0.055 ± 0.006	0.085 ± 0.008	0.277 ± 0.013	0.307 ± 0.009	0.179 ± 0.002
MLL-GP (Fortuin and Rätsch, 2019)	0.059 ± 0.003	0.096 ± 0.009	0.277 ± 0.009	0.305 ± 0.014	0.153 ± 0.007
MLAP (Amit and Meir, 2018)	0.086 ± 0.015	0.090 ± 0.021	0.343 ± 0.017	0.344 ± 0.016	0.108 ± 0.024
BMAML (Yoon et al., 2018)	0.061 ± 0.007	0.115 ± 0.036	0.279 ± 0.010	0.423 ± 0.106	0.161 ± 0.013
NP (Garnelo et al., 2018)	0.057 ± 0.009	0.131 ± 0.056	0.299 ± 0.012	0.319 ± 0.004	0.210 ± 0.007
PACOH-GP-MAP (ours)	0.058 ± 0.006	0.055 ± 0.005	0.267 ± 0.010	0.298 ± 0.003	0.155 ± 0.004
PACOH-GP-SVGD (ours)	0.056 ± 0.004	0.038 ± 0.006	$\mathbf{0.262 \pm 0.004}$	0.296 ± 0.003	0.098 ± 0.005
PACOH-GP-VI (ours)	0.057 ± 0.002	0.046 ± 0.016	0.265 ± 0.002	0.311 ± 0.005	0.089 ± 0.003
PACOH-NN-MAP (ours)	0.051 ± 0.002	0.031 ± 0.003	0.268 ± 0.015	0.306 ± 0.003	$\mathbf{0.063 \pm 0.016}$
PACOH-NN-SVGD (ours)	$\mathbf{0.046 \pm 0.001}$	$\mathbf{0.027 \pm 0.003}$	0.267 ± 0.005	0.302 ± 0.003	$\mathbf{0.067 \pm 0.005}$

Table 3: Comparison of standard and meta-learning methods in terms of the test calibration error in 5 regression environments. We report the mean and standard deviation across 5 random seeds. *PACOH* yields the best uncertainty calibration in the majority of environments.

environments. Throughout the benchmark study, *PACOH-NN* and *PACOH-GP consistently perform best or are among the best methods*. Similarly, *PACOH-NN* achieves the *highest accuracy* in the Omniglot classification environment (see Table 4). Overall, this demonstrates that the introduced meta-learning framework is not only theoretically sound, but also yields state-of-the-art empirical performance in practice.

PACOH improves the predictive uncertainty. We hypothesize that by acquiring the prior in a principled data-driven manner (e.g., with *PACOH*), we can improve the quality of the GP's and BNN's uncertainty estimates. To investigate the effect of meta-learned priors on the uncertainty estimates of the base learners, we compute the probabilistic predictors' calibration errors, reported in Table 3 and 4. In regression, the *calibration error* measures the discrepancy between the predicted confidence regions and the actual frequencies of test data points in the respective areas (Kuleshov et al., 2018). In the case of classification, it measures how well the probability of the predicted class reflects the corresponding misclassification rate (Guo et al., 2017). 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 Vanilla BNN's uncertainty estimates. Similarly, *PACOH-GP* yields a lower calibration error than the Vanilla GP in the majority of the environments. For meta-

	Accuracy		Calibration error	
	Omniglot 2-shot	Omniglot 5-shot	Omniglot 2-shot	Omniglot 5-shot
BNN (Liu and Wang, 2016)	0.6709 ± 0.006	0.795 ± 0.006	0.173 ± 0.009	0.135 ± 0.009
MLAP-M (Amit and Meir, 2018)	0.635 ± 0.015	0.804 ± 0.0168	0.108 ± 0.008	0.119 ± 0.019
MLAP-S (Amit and Meir, 2018)	0.615 ± 0.037	0.700 ± 0.0135	0.129 ± 0.018	0.108 ± 0.010
FO-MAML (Nichol et al., 2018)	0.429 ± 0.047	0.590 ± 0.010	N/A	N/A
MAML (Finn et al., 2017)	0.571 ± 0.018	0.693 ± 0.013	N/A	N/A
BMAML (Yoon et al., 2018)	0.651 ± 0.008	0.764 ± 0.025	0.132 ± 0.007	0.191 ± 0.018
PACOH-NN-SVGD (ours)	0.733 ± 0.009	$\mathbf{0.885 \pm 0.090}$	$\mathbf{0.094 \pm 0.004}$	0.091 ± 0.010
PACOH-NN-MAP (ours)	$\mathbf{0.735 \pm 0.010}$	0.866 ± 0.005	0.099 ± 0.009	$\mathbf{0.075 \pm 0.006}$

Table 4: Comparison of meta-learning algorithms in terms of test accuracy and calibration error on the *Omniglot* environment with 2-shot and 5-shot 5-way-classification tasks. We report the mean and standard deviation across 5 random seeds. *PACOH* achieves the best performance, while yielding the best uncertainty calibration.

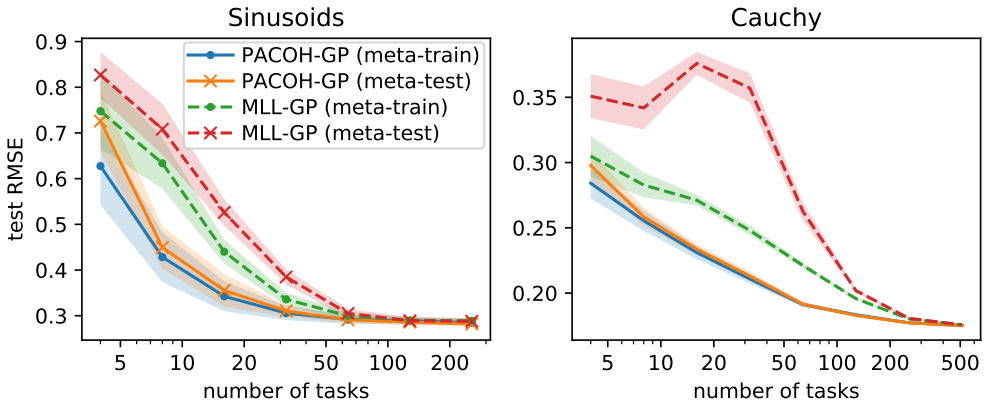


Figure 5: Test RMSE on meta-training and meta-testing tasks as a function of the number of meta-training tasks for PACOH-GP-MAP and MLL-GP. The performance gap between the train and test tasks demonstrates overfitting in the MLL method, while PACOH performs consistently better and barely overfits.

learning environments where the task similarity is high, like *SwissFEL* and *Berkeley-Sensor*, the improvement is substantial.

PACOH combats meta-overfitting. As Qin et al. (2018) and Yin et al. (2020) point out, many popular meta-learners (e.g., Finn et al., 2017; Garnelo et al., 2018) require a large number of meta-training tasks to generalize well. When presented with only a limited number of tasks, such algorithms suffer from severe meta-overfitting, adversely impacting their performance on unseen tasks from $\mathcal{T}$. This can even lead to *negative transfer*, such that meta-learning actually hurts the performance when compared to standard learning. In our experiments, we also observe such failure cases: For instance, in the classification environment (Table 4), *MAML* fails to improve upon the Vanilla BNN. Similarly, in the regression environments (Table 3) we find that *NPs*, *BMAML*, and *MLL-GP* often yield worse-calibrated predictive distributions than the Vanilla BNN and GP, respectively. In contrast, thanks to its theoretically principled construction, *PACOH-NN* is able to achieve positive transfer even when the tasks are diverse and small in number. In particular, the

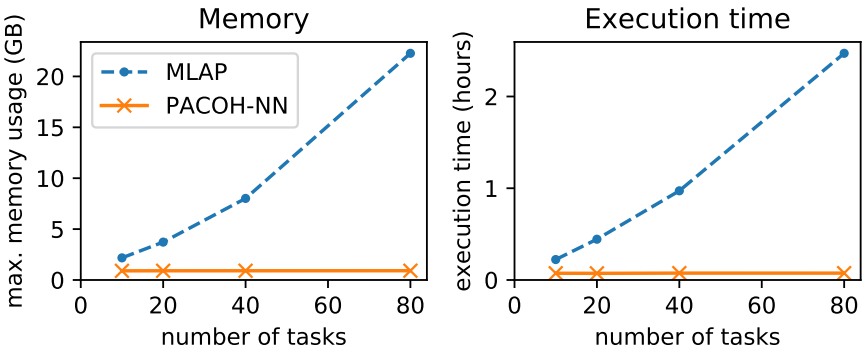


Figure 6: Comparison of *PACOH-NN* and *MLAP* in memory footprint and compute time, as the number of meta-training task grows. *PACOH-NN* scales much better in the number of tasks than *MLAP*.

hyper-prior acts as a meta-level regularizer by penalizing complex priors that are unlikely to convey useful inductive bias for unseen learning tasks.

To investigate the importance of meta-level regularization through the hyper-prior in more detail, we compare the performance of our proposed method *PACOH-GP* to *MLL-GP* (Fortuin and Rätsch, 2019), which also maximizes the sum of GP marginal log-likelihoods across tasks but has no hyper-prior nor meta-level regularization. Figure 5 shows that MLL-GP performs significantly better on the meta-training tasks than on the meta-test tasks in both of our synthetic regression environments. This gap between meta-train performance and meta-test performance signifies overfitting on the meta-level. In contrast, our method hardly exhibits this gap and consistently outperforms MLL-GP. As expected, this effect is particularly pronounced when the number of meta-training tasks is small (i.e., less than 100) and vanishes as n becomes large. Once more, this demonstrates the importance of meta-level regularization, and shows that our proposed framework effectively addresses the problem of meta-overfitting.

PACOH is scalable. A key feature of the proposed approach is that it drastically simplifies PAC-Bayesian meta-learning by converting the bi-level minimization of a PAC-Bayesian bound (cf., Eq. 9) into a variational approximation of the closed-form *PACOH*. Hence, unlike *MLAP* (Amit and Meir, 2018), *PACOH-NN* does not need to maintain posteriors Q_i for the meta-training tasks and can use mini-batching on the task level. Thus, it is *computationally much faster and more scalable* than previous PAC-Bayesian meta-learners. This is reflected in its computation and memory complexity, discussed in Section 5. Figure 6 showcases this computational advantage during meta-training with *PACOH-NN* and *MLAP* on the *Sinusoids* environment with varying number of tasks, reporting the maximum memory requirements, as well as the training time. While *MLAP's* memory consumption and compute time grow linearly and become prohibitively large even for less than 100 tasks, *PACOH-NN* maintains a constant memory and compute load as the number of tasks grow.

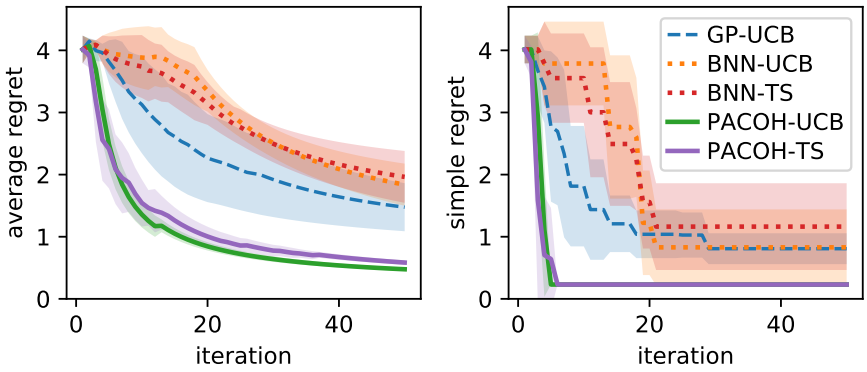


Figure 7: MHC-I peptide design task: Regret for different priors (GP, standard BNN prior and meta-learned *PACOH-NN* prior) and bandit algorithms (UCB and TS). A meta-learned *PACOH-NN* prior substantially improves the regret, compared to a standard BNN/GP prior.

8.3 Meta-Learning for Sequential Decision Making

In Section 8.2, we have considered supervised learning problems. Now, we go one step further and study our PACOH approach in the context of sequential decision-making where the predictions and uncertainty estimates are used to interactively collect the training data.

We consider a Bayesian Optimization (BO) problem where the goal is to optimize a (black-box) target function with as few function queries as possible (see, e.g., Shahriari et al., 2015). To do so efficiently, BO approaches typically use a probabilistic model of the objective function together with an uncertainty-aware acquisition function to decide where to sequentially query the objective function.

In particular, we study a problem from molecular biology: The goal is to discover peptides that bind to major histocompatibility complex class-I molecules (MHC-I). MHC-I molecules present fragments of proteins from within a cell to T-cells, allowing the immune system to distinguish between healthy and infected cells. Following the setup of Krause and Ong (2011), the considered BO problem corresponds to searching for maximally binding peptides, a vital step in the design of peptide-based vaccines. In each iteration, the experimenter (i.e., the BO algorithm) chooses to test one peptide among the pool of more than 800 candidates and receives its binding affinity as evaluation of the objective function.

We have a number of such BO tasks that differ in their targeted MHC-I allele, corresponding to different genetic variants of the MHC-I protein. We use data from Widmer et al. (2010), which contains the standardized binding affinities (IC_{50} values) of different peptide candidates (encoded as 45-dimensional feature vectors) to the MHC-I alleles. Since the feature space is high-dimensional, this BO problem is very challenging. We aim to investigate whether PACOH allows us to transfer useful knowledge across alleles and, thus, accelerate the optimization for new alleles.

In our experiment, we use the datasets for 5 alleles (tasks) to meta-learn a BNN prior with *PACOH-NN* and leave the most genetically dissimilar allele (A-6901) as the test BO task. In particular, we use a BNN with meta-learned prior as a probabilistic model of the objective function. To pick the next protein candidates to evaluate based on the BNN's predictions, we employ the UCB (Auer, 2002) and Thompson-Sampling (TS) (Thompson, 1933) BO algorithms. We refer to the respective approaches as *PACOH-UCB* and *PACOH-TS*. As

baselines, we compare against a BNN with zero-centered Gaussian prior (*BNN-UCB/TS*) and a Gaussian process as dynamics model (*GP-UCB*) (Srinivas et al., 2009). For further details we refer to Appendix E.4.

Figure 7 reports the respective average regret and simple 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. While the majority of meta-learning methods rely on a large number of meta-training tasks (Qin et al., 2018), *PACOH-NN* allows us to achieve promising positive transfer, even in complex real-world scenarios with only a few of tasks (in this case 5).

9. Summary and Critical Discussion

This paper provides a theoretical analysis of generalization in meta-learning, studying the factors that drive positive transfer and improvement over single-task learning. We present novel PAC-Bayesian generalization bounds on the transfer error and derive the PAC-optimal hyper-posterior (PACOH) in closed form. This transforms PAC-Bayesian meta-learning from a previously difficult bi-level optimization problem into simple approximate inference on the PACOH.

However, this comes at a price. While Theorem 4 holds for arbitrary choices of base learners and hyper-posteriors, the bound for Gibbs learners (Corollary 7) and for the PACOH (Corollary 10) only hold when we employ the exact Gibbs posterior and PAC-optimal hyper-posterior. In most practical settings, however, we can only approximate these distributions. Hence, the respective bounds formally do not hold for the algorithms proposed in Section 7. Additionally, for complex and over-parametrized hypothesis spaces such as those of neural networks, PAC-Bayesian bounds are well-known to be loose or even vacuous. This is most likely also the case for the bounds in this paper.

Nonetheless, the presented PACOH framework gives rise to a range of meta-learning algorithms that are practical and scalable. As demonstrated in the experiments, the proposed algorithms achieve state-of-the-art performance in terms of predictive accuracy and the quality of uncertainty estimates across both regression and classification tasks. Thanks to their foundation in learning theory, our meta-learning methods are equipped with principled meta-level regularization which allows them to achieve positive transfer with as little as five learning tasks. As shown by our experiments on vaccine design, our meta-learned priors can be effectively employed in realistic sequential decision-making problems. Overall, we believe that our approach provides an important step towards understanding generalization in meta-learning and, in practice, reliably learning useful inductive biases when meta-training data is scarce.

Acknowledgements

This research was supported by the European Research Council (ERC) under the EU's Horizon 2020 research and innovation program grant agreement no. 815943. Jonas Rothfuss was supported by an Apple Scholars in AI/ML fellowship. Vincent Fortuin was supported by a PhD Fellowship from the Swiss Data Science Center, a Postdoc.Mobility Fellowship from the Swiss National Science Foundation, a Research Fellowship from St John's College Cambridge, and a Branco Weiss Fellowship. We thank Parnian Kassraie for her valuable feedback.

References

Pierre Alquier. PAC-Bayesian bounds for randomized empirical risk minimizers. *Mathematical Methods of Statistics*, 17:279–304, 2008.

Pierre Alquier. User-friendly introduction to PAC-Bayes bounds. *arXiv preprint arXiv:2110.11216*, 2021.

Pierre Alquier and Benjamin Guedj. Simpler PAC-Bayesian bounds for hostile data. *Machine Learning*, 107:887–902, 2018.

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

Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended PAC-Bayes theory. In *International Conference on Machine Learning*, 2018.

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. In *Advances in Neural Information Processing Systems*, 2016.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.

Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 2000.

Y. Bengio, S. Bengio, and J. Cloutier. Learning a synaptic learning rule. In *International Joint Conference on Neural Networks*, 1991.

James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning*, 2013.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112:859–877, 2017.

Edwin V. Bonilla, Kian M. Chai, and Christopher Williams. Multi-task Gaussian Process prediction. In *Advances in Neural Information Processing Systems*, 2008.

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

Olivier Catoni. PAC-Bayesian supervised classification: the thermodynamics of statistical learning. *IMS Lecture Notes Monograph Series*, 56, 2007.

Leonardo Cella and Massimiliano Pontil. Multi-task and meta-learning with sparse linear bandits. In *Uncertainty in Artificial Intelligence*, 2021.

Leonardo Cella, Karim Lounici, and Massimiliano Pontil. Meta representation learning with contextual linear bandits. *arXiv preprint arXiv:2205.15100*, 2022.

Qi Chen, Changjian Shui, and Mario Marchand. Generalization bounds for meta-learning: An information-theoretic analysis. *Advances in Neural Information Processing Systems*, 2021.

Yutian Chen, Matthew W. Hoffman, Sergio Gómez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando De Freitas. Learning to Learn without Gradient Descent by Gradient Descent. In *International Conference on Machine Learning*, 2017.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3:146–158, 1975.

Nan Ding, Xi Chen, Tomer Levinboim, Sebastian Goodman, and Radu Soricut. Bridging the Gap Between Practice and PAC-Bayes Theory in Few-Shot Meta-Learning. In *Advances in Neural Information Processing Systems*, 2021.

M.D. Donsker and S.R.S. Varadhan. Large deviations for Markov processes and the asymptotic evaluation of certain Markov process expectations for large times. In *Probabilistic Methods in Differential Equations*, 1975.

Gintare Karolina Dziugaite and Daniel M Roy. Data-dependent PAC-Bayes priors via differential privacy. *Advances in Neural Information Processing Systems*, 2018.

Gintare Karolina Dziugaite, Kyle Hsu, Waseem Gharbieh, Gabriel Arpino, and Daniel Roy. On the role of data in PAC-Bayes bounds. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Alec Farid and Anirudha Majumdar. Generalization bounds for meta-learning via PAC-Bayes and uniform stability. In *Advances in Neural Information Processing Systems*, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

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

Vincent Fortuin and Gunnar Rätsch. Deep Mean Functions for Meta-Learning in Gaussian Processes. *arXiv preprint arXiv:1901.08098*, 2019.

Vincent Fortuin, Adrià Garriga-Alonso, Florian Wenzel, Gunnar Rätsch, Richard Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. In *International Conference on Learning Representations*, 2022.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. In *ICML 2018 workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018.

Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. PAC-Bayesian theory meets Bayesian inference. In *Advances in Neural Information Processing Systems*, 2016.

Micah Goldblum, Steven Reich, Liam Fowl, Renkun Ni, Valeriia Cherepanova, and Tom Goldstein. Unraveling meta-learning: Understanding feature representations for few-shot tasks. In *International Conference on Machine Learning*, 2020.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011.

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical Bayes. In *International Conference on Learning Representations*, 2018.

Jiechao Guan and Zhiwu Lu. Fast-rate PAC-Bayesian generalization bounds for meta-learning. In *International Conference on Machine Learning*, 2022.

Benjamin Guedj. A primer on PAC-Bayesian learning. In *2nd Congress of the French Mathematical Society*, 2019.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Sepp Hochreiter, A. Steven Younger, and Peter R. Conwell. Learning To Learn Using Gradient Descent. In *International Conference on Artificial Neural Networks*, 2001.

Matthew Holland. PAC-Bayes under potentially heavy tails. *Advances in Neural Information Processing Systems*, 2019.

Sharu Theresa Jose and Osvaldo Simeone. Information-theoretic generalization bounds for meta-learning and applications. *Entropy*, 23:126, 2021.

Sharu Theresa Jose, Sangwook Park, and Osvaldo Simeone. Information-Theoretic Analysis of Epistemic Uncertainty in Bayesian Meta-learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Herman Kahn. *Use of Different Monte Carlo Sampling Techniques*. RAND Corporation, 1955.

Parnian Kassraie, Jonas Rothfuss, and Andreas Krause. Meta-learning hypothesis spaces for sequential decision-making. In *International Conference on Machine Learning*, 2022.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *International Conference on Learning Representations*, 2019.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

Johannes Kirschner, Mojmir Mutný, Nicole Hiller, Rasmus Ischebeck, and Andreas Krause. Adaptive and Safe Bayesian Optimization in High Dimensions via One-Dimensional Subspaces. In *International Conference on Machine Learning*, 2019a.

Johannes Kirschner, Manuel Nonnenmacher, Mojmír Mutnỳ, Andreas Krause, Nicole Hiller, Rasmus Ischebeck, and Andreas Adelmann. Bayesian Optimization for Fast and Safe Parameter Tuning of SwissFEL. In *International Free-Electron Laser Conference*, 2019b.

Andreas Krause and Cheng S. Ong. Contextual Gaussian Process bandit optimization. In *Advances in Neural Information Processing Systems*, 2011.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International Conference on Machine Learning*, 2018.

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

Guy Lever, François Laviolette, and John Shawe-Taylor. Tighter PAC-Bayes bounds through distribution-dependent priors. *Theoretical Computer Science*, 473:4–28, 2013.

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

Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. In *Advances in Neural Information Processing Systems*, 2016.

Tianyu Liu, Jie Lu, Zheng Yan, and Guangquan Zhang. Statistical generalization performance guarantee for meta-learning with data-dependent prior. *Neurocomputing*, 465:391–405, 2021.

Yucen Luo, Alex Beatson, Mohammad Norouzi, Jun Zhu, David Duvenaud, Ryan P. Adams, and Ricky T. Q. Chen. SUMO: Unbiased Estimation of Log Marginal Probability for Latent Variable Models. In *International Conference on Learning Representations*, 2020.

Samuel Madden. Intel lab data. `http://db.csail.mit.edu/labdata/labdata.html`, 2004. Accessed: Sep 8, 2020.

Andreas Maurer. A note on the PAC Bayesian theorem. *arXiv preprint arXiv:0411099*, 2004.

Andreas Maurer and Tommi Jaakkola. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 6:967–994, 2005.

David A McAllester. Some PAC-Bayesian theorems. *Machine Learning*, 1999.

Charles Micchelli and Massimiliano Pontil. Kernels for Multi-task Learning. *Advances in Neural Information Processing Systems*, 2004.

Christopher J Milne, Thomas Schietinger, Masamitsu Aiba, et al. SwissFEL: the Swiss X-ray Free Electron Laser. *Applied Sciences*, 2017.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations*, 2018.

Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms. *arXiv*, 2018.

Luca Oneto, Davide Anguita, and Sandro Ridella. PAC-Bayesian analysis of distribution dependent priors: Tighter risk bounds and stability analysis. *Pattern Recognition Letters*, 80:200–207, 2016.

Cheng S. Ong, Alexander J. Smola, and Robert C. Williamson. Learning the Kernel with Hyperkernels. *Journal of Machine Learning Research*, 3:1043–1071, 2005.

Shibin Parameswaran and Kilian Q Weinberger. Large margin multi-task metric learning. *Advances in Neural Information Processing Systems*, 2010.

Emilio Parrado-Hernandez, Amiran Ambroladze, John Shawe-Taylor, and Shiliang Sun. PAC-Bayes Bounds with Data Dependent Priors. *Journal of Machine Learning Research*, 13:3507–3531, 2012.

Anastasia Pentina and Christoph Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, 2014.

María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári. Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22:10326–10365, 2021.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*, chapter 13.7. Morgan Kaufmann, 3rd edition, 2016.

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.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes in machine learning*. MIT Press, 2006.

Sachin Ravi and Alex Beatson. Amortized Bayesian meta-learning. In *International Conference on Learning Representations*, 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a Model for Few-Shot Learning. In *International Conference on Learning Representations*, 2017.

David Reeb, Andreas Doerr, Sebastian Gerwinn, and Barbara Rakitsch. Learning Gaussian Processes by Minimizing PAC-Bayesian Generalization Bounds. In *Advances in Neural Information Processing Systems*, 2018.

Arezou Rezazadeh, Sharu Theresa Jose, Giuseppe Durisi, and Osvaldo Simeone. Conditional mutual information-based generalization bound for meta learning. In *International Symposium on Information Theory*, 2021.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *International Conference on Machine Learning*, 2015.

Omar Rivasplata, Emilio Parrado-Hernández, John S. Shawe-Taylor, Shiliang Sun, and Csaba Szepesvári. PAC-Bayes bounds for stable algorithms with instance-dependent priors. *Advances in Neural Information Processing Systems*, 2018.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. ProMP: Proximal Meta-Policy Search. In *International Conference on Learning Representations*, 2019.

Jonas Rothfuss, Vincent Fortuin, Martin Josifoski, and Andreas Krause. PACOH: Bayes-Optimal Meta-Learning with PAC-Guarantees. In *International Conference on Machine Learning*, 2021a.

Jonas Rothfuss, Dominique Heyn, Jinfan Chen, and Andreas Krause. Meta-learning Reliable Priors in the Function Space. In *Advances in Neural Information Processing Systems*, 2021b.

Jonas Rothfuss, Christopher Koenig, Alisa Rupenyan, and Andreas Krause. Meta-Learning Priors for Safe Bayesian Optimization. In *Conference on Robot Learning*, 2022.

Ruslan Salakhutdinov, Joshua Tenenbaum, and Antonio Torralba. One-shot learning with a hierarchical nonparametric Bayesian model. In *ICML Workshop on Unsupervised and Transfer Learning*, 2012.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, 2016.

Juergen Schmidhuber. *Evolutionary principles in self-referential learning.* PhD thesis, Technische Universitaet Munchen, 1987.

Felix Schur, Parnian Kassraie, Jonas Rothfuss, and Andreas Krause. Lifelong bandit optimization: no prior and no regret. In *Uncertainty in Artificial Intelligence*, 2023.

Matthias Seeger. PAC-Bayesian generalisation error bounds for Gaussian process classification. *Journal of Machine Learning Research*, 3:233–269, 2002.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

Vera Shalaeva, Alireza Fakhrizadeh Esfahani, Pascal Germain, and Mihaly Petreczky. Improved PAC-Bayesian bounds for linear regression. In *AAAI Conference on Artificial Intelligence*, 2020.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

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 *Computing in Cardiology*, 2012.

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

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, 2009.

Sanjay Thakur, Herke Van Hoof, Gunshi Gupta, and David Meger. Unifying variational inference and pac-bayes for supervised learning that scales. *arXiv preprint arXiv:1910.10367*, 2019.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285—294, 1933.

Sebastian Thrun and Lorien Pratt. *Learning to Learn*. Springer, 1998.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, 2016.

Christian Widmer, Nora C. Toussaint, Yasemin Altun, and Gunnar Rätsch. Inferring latent task structure for multitask learning by multiple kernel learning. *Bioinformatics*, 2010.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *International Conference on Artificial Intelligence and Statistics*, 2016.

Jin Xu, Jean-Francois Ton, Hyunjik Kim, Adam Kosiorek, and Yee Whye Teh. Metafun: Meta-learning with iterative functional updates. In *International Conference on Machine Learning*, 2020.

Zhen-Hang Yang and Yu-Ming Chu. On approximating the modified bessel function of the second kind. *Journal of Inequalities and Applications*, 41:1–8, 2017.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. In *International Conference on Learning Representations*, 2020.

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, 2018.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian processes from multiple tasks. In *International Conference on Machine Learning*, 2005.

Alexander Zien and Cheng Soon Ong. Multiclass Multiple Kernel Learning. In *International Conference on Machine Learning*, 2007.

Appendix

Appendix A. Proofs and Derivations

Various proofs in this section follow along with Rothfuss et al. (2021a), i.e., the short conference version of this paper. In particular, the proofs of Corollary 7 and Proposition 9 are almost identical to Rothfuss et al. (2021a). The proofs of Theorem 4, Corollary 5, and Corollary 6 have seen significant revisions to make them easier to follow and more rigorous. The proofs of Corollary 10, Theorem 11, and Proposition 12 are new additions.

A.1 Proof of Theorem 4

A key tool in the PAC-Bayesian framework which we also use in our proofs is the change of measure inequality:

Lemma 14 ***(Csiszár (1975); Donsker and Varadhan (1975))*** *Let f be a random variable taking values in a set A and $g : A \to \mathbb{R}$ a function. For distributions $\pi, \rho \in \mathcal{M}(A)$ and any $\lambda > 0$, if $\mathbb{E}_{f\sim\rho}[g(f)]$ exists and is finite, we have that*

$$\mathbb{E}_{f\sim\rho}[g(f)] \leq \frac{1}{\lambda}\left(D_{KL}(\rho||\pi) + \log \mathbb{E}_{f\sim\pi}\left[e^{\lambda g(f)}\right]\right). \tag{29}$$

Lemma 14 follows from the positivity of the KL-divergence and is a special case of the convex duality of the KL divergence (see e.g., Seeger (2002, Appendix A) for a proof). Hence, it is fairly general and only requires the integrability of $g(f)$ under ρ. Importantly, it also holds for random functions g, a property we will use in our proof.

To prove Theorem 4, 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 make use of an intermediate quantity, the *expected multi-task error*:

$$\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n) = \mathbb{E}_{P\sim\mathcal{Q}}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(Q(S_i, P), \mathcal{D}_i)\right] \tag{30}$$

In the following, we invoke Lemma 14 twice. First, in step 1, we bound the difference between $\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n)$ and $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$, then, in step 2, the difference between $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ and $\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n)$. Finally, in step 3, we combine both results and bound the cumulant-generating functions that arise from applying (29).

Step 1 (Task-specific generalization) First, we bound the expected generalization error across the observed tasks $\tau_i = (\mathcal{D}_i, S_i)$, $i = 1, ..., n$, when using a hyper-posterior $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$ and a base learner $Q : \mathcal{Z}^m \times \mathcal{M}(\mathcal{H}) \to \mathcal{M}(\mathcal{H})$. Remember from Section 4.1 that the base learner takes as an input a prior distribution $P \in \mathcal{M}(\mathcal{H})$ as well as a dataset $S_i \sim \mathcal{D}_i^m$ of size m and outputs a posterior distribution $Q_i = Q(S_i, P)$ over hypotheses $h \in \mathcal{H}$. For brevity, we also write Q_i short for $Q(S_i, P)$ in the following.

Instantiations for Lemma 14: To use Lemma 14, we set $A = \mathcal{M}(\mathcal{H}) \times \mathcal{H}^n$, and, correspondingly, $f = (P, h_1, ..., h_n) \in A$. We can interpret this as a joint two-level hypothesis of a prior P (i.e., the hypothesis of the meta-learner) and n base hypotheses h_i, one for each task. Denoting P^n as product distribution of n prior distributions P, we set $\pi = (\mathcal{P}, P^n)$ which is a distribution over the joint two-level hypotheses corresponding to hierarchical sampling via $P \sim \mathcal{P}$ and

$h_i \sim P \; \forall i = 1, ..., n$. Furthermore, we denote $Q^n = Q_1 \cdots Q_n$ as product distribution of the task-specific posteriors and set $\rho = (\mathcal{Q}, Q^n)$ as a distribution over the joint two-level hypotheses corresponding to hierarchical sampling via $P \sim \mathcal{Q}$ and $h_i \sim Q_i = Q(S_i, P) \; \forall i = 1, ..., n$. Finally, we set $g(f) = g(P, h_1, ..., h_n) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h_i, \mathcal{D}_i) - \hat{\mathcal{L}}(h_i, S_i)$ and $\lambda = \gamma$.
Integrability conditions for Lemma 14: Finally, we check whether the integrability conditions for the application of Lemma 14 are satisfied. By assumption, the expectation

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) = \mathbb{E}_{\mathcal{D} \sim \mathcal{T}} \mathbb{E}_{S \sim \mathcal{D}^m} \mathbb{E}_{P \sim \mathcal{Q}} \left[\mathcal{L}(Q(S, P), \mathcal{D}) \right] \tag{31}$$

is finite. Thus, the probability that we sample a $(\mathcal{D}, S) \sim \mathcal{T}_h$ so that either $\mathbb{E}_{P \sim \mathcal{Q}} \left[\mathcal{L}(Q(S, P), \mathcal{D}) \right]$ or $\mathbb{E}_{P \sim \mathcal{Q}} \left[\hat{\mathcal{L}}(Q(S, P), S) \right]$ are infinite or undefined must be zero. Hence, we have with probability 1 that every summand in

$$\mathbb{E}_{f \sim \rho}[g(f)] = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{P \sim \mathcal{Q}} \left[\mathcal{L}(Q(S_i, P), \mathcal{D}_i) \right] - \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{P \sim \mathcal{Q}} \left[\hat{\mathcal{L}}(Q(S_i, P), S_i) \right] \tag{32}$$

is finite, and, thus that the overall sum, i.e., $\mathbb{E}_{f \sim \rho}[g(f)]$ is finite.
Application of Lemma 14: Finally, we can invoke Lemma 14 to obtain that, with probability 1, for any $\gamma > 0$,

$$\begin{aligned} \mathbb{E}_{(P, h_1, ..., h_n) \sim (\mathcal{Q}, Q^n)} \left[\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h_i, \mathcal{D}_i) - \hat{\mathcal{L}}(h_i, S_i) \right] \leq \frac{1}{\gamma} \Bigg(& D_{KL}((\mathcal{Q}, Q^n) || (\mathcal{P}, P^n)) \\ & + \log \mathbb{E}_{(P, h_1, ..., h_n) \sim (\mathcal{P}, P^n)} \left[e^{\gamma \left(\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h_i, \mathcal{D}_i) - \hat{\mathcal{L}}(h_i, S_i) \right)} \right] \Bigg) . \end{aligned} \tag{33}$$

By using the definitions of $\mathcal{L}(Q_i, \mathcal{D}_i)$ and $\hat{\mathcal{L}}(Q_i, S_i)$, and that $h_i \sim P$ are i.i.d., we can write (33) as

$$\begin{aligned} \mathbb{E}_{P \sim \mathcal{Q}} \left[\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(Q_i, \mathcal{D}_i) - \hat{\mathcal{L}}(Q_i, S_i) \right] \leq \frac{1}{\gamma} & D_{KL}((\mathcal{Q}, Q^n) || (\mathcal{P}, P^n)) \\ & + \frac{1}{\gamma} \log \mathbb{E}_{P \sim \mathcal{P}} \mathbb{E}_{h \sim P} \left[e^{\frac{\gamma}{n} \sum_{i=1}^{n} \mathcal{L}(h, \mathcal{D}_i) - \hat{\mathcal{L}}(h, S_i)} \right] \Bigg) \end{aligned} \tag{34}$$

Next we follow the arguments of Pentina and Lampert (2014) and use the above definitions to rewrite the KL-divergence term as follows:

$$D_{KL}\left[(\mathcal{Q}, Q^n) || (\mathcal{P}, P^n)\right] = \mathbb{E}_{P \sim \mathcal{Q}} \left[\mathbb{E}_{h_1 \sim Q_1} ... \mathbb{E}_{h_n \sim Q_n} \left[\log \frac{\mathcal{Q}(P) \prod_{i=1}^{n} Q_i(h_i)}{\mathcal{P}(P) \prod_{i=1}^{n} P(h_i)} \right] \right] \tag{35}$$

$$= \mathbb{E}_{P \sim \mathcal{Q}} \left[\log \frac{\mathcal{Q}(P)}{\mathcal{P}(P)} \right] + \sum_{i=1}^{n} \mathbb{E}_{P \sim \mathcal{Q}} \left[\mathbb{E}_{h \sim Q_i} \left[\log \frac{Q_i(h)}{P(h)} \right] \right] \tag{36}$$

$$= D_{KL}(\mathcal{Q} || \mathcal{P}) + \sum_{i=1}^{n} \mathbb{E}_{P \sim \mathcal{Q}} \left[D_{KL}(Q_i || P) \right] \tag{37}$$

By inserting (37) into (34) and using the definitions of $\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n)$ and $\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n)$ we obtain a bound on the expected multi-task error:

$$\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \frac{1}{\gamma} D_{KL}(\mathcal{Q}||\mathcal{P}) + \frac{1}{\gamma} \sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{Q}} \left[D_{KL}(Q_i||P) \right] + \frac{1}{\gamma} \underbrace{\log \mathbb{E}_{P\sim\mathcal{P}} \mathbb{E}_{h\sim P} \left[e^{\frac{\gamma}{n} \sum_{i=1}^{n} \mathcal{L}(h,\mathcal{D}_i) - \hat{\mathcal{L}}(h,S_i)} \right]}_{:=\Upsilon^{\mathrm{I}}(\gamma)} \tag{38}$$

Step 2 (Task environment generalization) Now, we apply Lemma 14 on the meta-level to bound the difference between the transfer error $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ and the expected multi-task error $\tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n)$. For that, we use the following instantiations for Lemma 14: $A = \mathcal{M}(\mathcal{H})$, $f = P$, $\rho = \mathcal{Q}$, $\pi = \mathcal{P}$ and $g(f) = g(P) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\mathcal{D}\sim\mathcal{T}} \mathbb{E}_{S\sim\mathcal{D}^m} \left[\mathcal{L}(Q(S,P), \mathcal{D}) \right] - \mathcal{L}(Q(S_i,P), \mathcal{D}_i)$. Again, we have to check the integrability conditions: By assumption, the expectation $\mathcal{L}(\mathcal{Q}, \mathcal{T})$ is finite and, thus, $\mathbb{E}_{f\sim\rho}[g(f)]$ must be finite with probability 1 over sampled tasks $\tau \sim \mathcal{T}_h$ (proof analogous to Step 1). Hence, we obtain that, with probability 1, for all $\lambda > 0$,

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \tilde{\mathcal{L}}(\mathcal{Q}, \tau_1, ..., \tau_n) + \frac{1}{\lambda} D_{KL}(\mathcal{Q}||\mathcal{P}) + \frac{1}{\lambda} \underbrace{\log \mathbb{E}_{P\sim\mathcal{P}} \left[e^{\frac{\lambda}{n} \sum_{i=1}^{n} \mathbb{E}_{\mathcal{D}\sim\mathcal{T}} \mathbb{E}_{S\sim\mathcal{D}^m} [\mathcal{L}(Q(S,P),\mathcal{D})] - \mathcal{L}(Q(S_i,P),\mathcal{D}_i)} \right]}_{:=\Upsilon^{\mathrm{II}}(\lambda)}. \tag{39}$$

Combining (38) with (39), we obtain

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \left(\frac{1}{\lambda} + \frac{1}{\gamma} \right) D_{KL}(\mathcal{Q}||\mathcal{P}) + \frac{1}{\gamma} \sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{Q}} \left[D_{KL}(Q_i||P) \right] + \frac{1}{\gamma} \Upsilon^{\mathrm{I}}(\gamma) + \frac{1}{\lambda} \Upsilon^{\mathrm{II}}(\lambda) \tag{40}$$

Step 3 (Bounding the cumulant-generating functions) Finally, we aim to bound the random quantity $\frac{1}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{1}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)$. Note that the randomness of $\Upsilon^{\mathrm{I}}(\gamma)$ is governed by the random data points $z_{i,j}$ sampled i.i.d. from the respective data distribution $\mathcal{D}_i$, while $\Upsilon^{\mathrm{II}}(\lambda)$ is governed by random tasks sampled from the environment $\mathcal{T}$.
First, we factor out $\sqrt{n}$ from γ and λ, obtaining

$$\frac{1}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{1}{\lambda}\Upsilon^{\mathrm{II}}(\lambda) = \frac{1}{\sqrt{n}} \left(\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda) \right) \tag{41}$$

Next, we proceed by bounding the inner part on the RHS, i.e., $\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)$. Using Markov's inequality, we have

$$e^{\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)} \leq \frac{\mathbb{E}\left[e^{\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)} \right]}{\delta} \tag{42}$$

and thus

$$\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda) \leq \log \mathbb{E}\left[e^{\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)} \right] - \log \delta \tag{43}$$

with probability at least $1-\delta$. Next, we bound the expectation, i.e.,

$$\mathbb{E}\left[e^{\frac{\sqrt{n}}{\gamma}\Upsilon^{\mathrm{I}}(\gamma)+\frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)}\right] = \mathbb{E}_{\mathcal{T}}\left[e^{\frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)}\,\mathbb{E}_{\mathcal{D}_1}...\mathbb{E}_{\mathcal{D}_1}\left[\left(e^{\Upsilon^{\mathrm{I}}(\gamma)}\right)^{\frac{\sqrt{n}}{\gamma}}\right]\right]. \tag{44}$$

If $\gamma \geq \sqrt{n}$ and $x \mapsto x^{\sqrt{n}/\gamma}$ is a concave function, and we can obtain an upper bound on (44) by using Jensen's inequality to move the exponent in (44) outside the inner expectation. Further, we denote $V_i^{\mathrm{II}} := \mathbb{E}_{\mathcal{T}}\left[\mathcal{L}(Q_i,\mathcal{D})\right] - \mathcal{L}(Q_i,\mathcal{D}_i)$ as i.i.d. realizations of the random variable under the task distribution. Similarly, for each $i = 1,....,n$ and $j = 1,....,m$, we denote the independent realizations of $\mathcal{L}(h,\mathcal{D}_i) - l(h,z_{ij})$ as V_{ij}^{I}. Hence, we can write

$$(44) \leq \mathbb{E}_{\mathcal{T}}\left[e^{\frac{\sqrt{n}}{\lambda}\Upsilon^{\mathrm{II}}(\lambda)}\,\mathbb{E}_{\mathcal{D}_1}...\mathbb{E}_{\mathcal{D}_n}\left[e^{\Upsilon^{\mathrm{I}}(\gamma)}\right]^{\frac{\sqrt{n}}{\gamma}}\right] \tag{45}$$

$$= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{\mathcal{P}}\left[e^{\frac{\lambda}{n}\sum_{i=1}^{n}V_i^{\mathrm{II}}}\right]^{\sqrt{n}/\lambda}\cdot\mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}_1}...\;\mathbb{E}_{\mathcal{D}_n}\left[e^{\frac{\gamma}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}V_{ij}^{\mathrm{I}}}\right]^{\sqrt{n}/\gamma}\right] \tag{46}$$

$$= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{\mathcal{P}}\left[\prod_{i=1}^{n}e^{\frac{\lambda}{n}V_i^{\mathrm{II}}}\right]^{\sqrt{n}/\lambda}\cdot\mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}_1}...\;\mathbb{E}_{\mathcal{D}_n}\left[\prod_{i=1}^{n}\prod_{j=1}^{m}e^{\frac{\gamma}{nm}V_{ij}^{\mathrm{I}}}\right]^{\sqrt{n}/\gamma}\right]. \tag{47}$$

Following the arguments of Pentina and Lampert (2014, Proof of Lemma 2) and Germain et al. (2016, Proof of Corollary 4), given a fixed h sampled via $h \sim P$, $P \sim \mathcal{P}$, the V_{ij}^{I} are independent. Similarly, given a P sampled via $P \sim \mathcal{P}$, the V_i^{II} are independent. Hence, we can write (47) as

$$(44) \leq \mathbb{E}_{\mathcal{T}}\left[\prod_{i=1}^{n}\mathbb{E}_{\mathcal{P}}\left[e^{\frac{\lambda}{n}V_i^{\mathrm{II}}}\right]^{\sqrt{n}/\lambda}\left(\prod_{i=1}^{n}\prod_{j=1}^{m}\mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}_i}e^{\frac{\gamma}{nm}V_{ij}^{\mathrm{I}}}\right)^{\sqrt{n}/\gamma}\right]. \tag{48}$$

Next, we set $\gamma = \beta n$ and use the uniform bound $\bar{\Psi}^{\mathrm{I}}(\beta) \geq \Psi^{\mathrm{I}}(\beta) = \frac{m}{\beta}\log\mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}}\left[e^{\frac{\beta}{m}V^{\mathrm{I}}}\right]\;\forall\,\mathcal{D}$ in the support of $\mathcal{T}$. Crucially, if we upper bound $\prod_{i=1}^{n}\prod_{j=1}^{m}\mathbb{E}_{\mathcal{P}}\mathbb{E}_P\mathbb{E}_{\mathcal{D}_i}e^{\frac{\gamma}{nm}V_{ij}^{\mathrm{I}}}$ by $e^{\beta n\bar{\Psi}^{\mathrm{I}}(\beta)}$, the upper bound no longer depends on the tasks sampled from $\mathcal{T}$, allowing us to move it outside the expectation. As a result, we obtain

$$(44) \leq \mathbb{E}_{\mathcal{T}}\left[\prod_{i=1}^{n}\mathbb{E}_{\mathcal{P}}\left[e^{\frac{\lambda}{n}V_i^{\mathrm{II}}}\right]^{\sqrt{n}/\lambda}e^{\sqrt{n}\bar{\Psi}^{\mathrm{I}}(\beta)}\right] \tag{49}$$

$$= \prod_{i=1}^{n}\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\left[e^{\frac{\lambda}{n}V_i^{\mathrm{II}}}\right]^{\sqrt{n}/\lambda}e^{\sqrt{n}\bar{\Psi}^{\mathrm{I}}(\beta)}. \tag{50}$$

Further, with $\Psi^{\mathrm{II}}(\lambda) = \frac{n}{\lambda}\log\mathbb{E}_{\mathcal{P}}\mathbb{E}_{\mathcal{T}}\left[e^{\frac{\lambda}{n}V_i^{\mathrm{II}}}\right]$, we have that

$$(44) \leq e^{\sqrt{n}\Psi^{\mathrm{II}}(\lambda)}\cdot e^{\sqrt{n}\bar{\Psi}^{\mathrm{I}}(\beta)}. \tag{51}$$

Finally, we insert (51) into (43) to obtain that

$$\frac{1}{\gamma}\Upsilon^{\mathrm{I}}(\gamma) + \frac{1}{\lambda}\Upsilon^{\mathrm{II}}(\lambda) \leq \bar{\Psi}^{\mathrm{I}}(\beta) + \Psi^{\mathrm{II}}(\lambda) - \frac{1}{\sqrt{n}}\log\delta \tag{52}$$

with probability at least $1-\delta$ which concludes our high-probability bound.

A.2 Proof of Corollary 5

If the loss function $l(h_i, z_{ij})$ is bounded in $[a, b]$, we can apply Hoeffding's lemma to $\Psi^{\mathrm{I}}(\beta)$ and $\Psi^{\mathrm{II}}(\lambda)$. In particular, we have that

$$\Psi^{\mathrm{I}}(\beta) = \frac{m}{\beta}\log\mathbb{E}_{\mathcal{P}}\mathbb{E}_{P}\mathbb{E}_{\mathcal{D}}\left[e^{\frac{\beta}{m}V^{\mathrm{I}}}\right] \leq \frac{\beta^2(b-a)^2}{8m^2} = \bar{\Psi}^{\mathrm{I}}(\gamma) \quad \forall\ \mathcal{D}\ . \tag{53}$$

Similarly, we can bound $\Psi^{\mathrm{II}}(\lambda)$ as

$$\Psi^{\mathrm{II}}(\lambda) \leq \frac{\lambda^2(b-a)^2}{8n^2} = \bar{\Psi}^{\mathrm{II}}(\lambda)\ . \tag{54}$$

When unserting these bounds on the CGFs into (52), we have that Theorem 4 holds with

$$C(\beta, \lambda, \delta) = \frac{\lambda(b-a)^2}{8n} + \frac{\beta(b-a)^2}{8m} - \frac{1}{\sqrt{n}}\log\delta\ . \tag{55}$$

A.3 Proof of Corollary 6

First, we bound the CGF corresponding to $\bar{\Psi}^{\mathrm{I}}(\beta)$. If the loss is sub-gamma with variance factor s_{I}^2 and scale parameter c_{I} under the two-level prior and any data distribution $\mathcal{D}$ in the support of $\mathcal{T}$, i.e.,

$$\mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{h\sim P}\mathbb{E}_{z\sim\mathcal{D}}\left[e^{\nu(\mathcal{L}(\mathcal{D},h)-l(h,z))}\right] \leq \exp\left(\frac{\nu^2 s_{\mathrm{I}}^2}{2(1-\nu c_{\mathrm{I}})}\right) \quad \forall\ \nu\in(0, 1/c_{\mathrm{I}}) \tag{56}$$

then, with $\nu := \beta/m$, we have that, if $\beta < \frac{m}{c_{\mathrm{I}}}$,

$$\frac{m}{\beta}\log\mathbb{E}_{\mathcal{P}}\mathbb{E}_{P}\mathbb{E}_{\mathcal{D}}\left[e^{\frac{\beta}{m}V^{\mathrm{I}}}\right] \leq \frac{m}{\beta}\frac{\beta^2 s_{\mathrm{I}}^2}{2m^2(1-\frac{c_{\mathrm{I}}\beta}{m})} = \frac{\beta s_{\mathrm{I}}^2}{2m(1-\frac{c_{\mathrm{I}}\beta}{m})} = \bar{\Psi}^{\mathrm{I}}(\beta) \quad \forall\ \mathcal{D}\ . \tag{57}$$

Second, we bound the remaining CGF corresponding to $\Psi^{\mathrm{II}}(\lambda)$. For that, we use the assumption that the random variable $V_i^{\mathrm{II}} := \mathbb{E}_{\mathcal{T}_h}\left[\mathcal{L}(Q_i, \mathcal{D})\right] - \mathcal{L}(Q_i, \mathcal{D}_i)$ is sub-gamma with variance factor s_{II}^2 and scale parameter c_{II} under the hyper-prior $\mathcal{P}$ and the task distribution $\mathcal{T}_h$. That is, its moment generating function can be bounded by that of a Gamma distribution $\Gamma(s_{\mathrm{II}}^2, c_{\mathrm{II}})$:

$$\mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}_h}\mathbb{E}_{P\sim\mathcal{P}}\left[e^{\kappa\mathbb{E}_{(D,S)\sim\mathcal{T}_h}[\mathcal{L}(Q(P,S),\mathcal{D})]-\mathcal{L}(Q(P,S),\mathcal{D})}\right] \leq \exp\left(\frac{\kappa^2 s_{\mathrm{II}}^2}{2(1-c_{\mathrm{II}}\kappa)}\right)\ \forall\kappa\in(0, 1/c_{\mathrm{II}})\ . \tag{58}$$

Using the sub-gamma assumption with $\kappa := \lambda/n$, we obtain that

$$\Psi^{\mathrm{II}}(\lambda) \leq \frac{\lambda^2 s_{\mathrm{II}}^2}{2n^2(1-\frac{\lambda c_{\mathrm{II}}}{n})} \ , \tag{59}$$

under the condition that $\lambda < \frac{n}{c_{\mathrm{II}}}$.
Finally, we insert (59) and (57) into (52) to obtain that, if $\beta < m/c_{\mathrm{I}}$ and $\lambda \leq n/c_{\mathrm{II}}$, Theorem 4 holds with

$$C(\lambda, \beta, \delta) = \frac{\beta s_{\mathrm{I}}^2}{2m(1-\frac{c_{\mathrm{I}}\beta}{m})} + \frac{\lambda s_{\mathrm{II}}^2}{2n(1-\frac{\lambda c_{\mathrm{II}}}{n})} - \frac{1}{\sqrt{n}} \log \delta \tag{60}$$

with probability at least $1-\delta$. This concludes our high-probability bound on the CGFs in the case of sub-gamma tail assumptions.

A.4 Proof of Corollary 7

The proof of Corollary 7 is inspired by Germain et al. (2016). When we choose the posterior Q as the optimal Gibbs posterior $Q_i^* := Q^*(S_i, P)$, we have that

$$\hat{\mathcal{L}}(\mathcal{Q}, S_1, ..., S_n) + \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{P\sim\mathcal{Q}} \left[D_{KL}(Q_i^* || P) \right] \tag{61}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \left(\mathbb{E}_{P\sim\mathcal{Q}} \mathbb{E}_{h\sim Q_i^*} \left[\hat{\mathcal{L}}(h, S_i) \right] + \frac{1}{\beta} \left(\mathbb{E}_{P\sim\mathcal{Q}} \left[D_{KL}(Q_i^* || P) \right] \right) \right) \tag{62}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta} \left(\mathbb{E}_{P\sim\mathcal{Q}} \mathbb{E}_{h\sim Q_i^*} \left[\beta \hat{\mathcal{L}}(h, S_i) + \log \frac{P(h) e^{-\beta \hat{\mathcal{L}}(h, S_i)}}{P(h) Z_\beta(S_i, P)} \right] \right) \tag{63}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta} \left(-\mathbb{E}_{P\sim\mathcal{Q}} \left[\log Z_\beta(S_i, P) \right] \right) \ . \tag{64}$$

This allows us to write the inequality in (9) as

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \ -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{P\sim\mathcal{Q}} \left[\log Z_\beta(S_i, P) \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C(\delta, \lambda, \beta) \ . \tag{65}$$

According to Lemma 3, the Gibbs posterior $Q^*(S_i, P)$ is the minimizer of (62), in particular $\forall P \in \mathcal{M}(\mathcal{H}), \forall i = 1, ..., n$:

$$Q^*(S_i, P) = \frac{P(h) e^{-\beta \hat{\mathcal{L}}(h, S_i)}}{Z_\beta(S_i, P)} = \underset{Q\in\mathcal{M}(\mathcal{H})}{\arg\min}\, \mathbb{E}_{h\sim Q} \left[\hat{\mathcal{L}}(h, S_i) \right] + \frac{1}{\beta} D_{KL}(Q || P) \ . \tag{66}$$

Hence, we can write

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{P\sim\mathcal{Q}} \left[\log Z_\beta(S_i, P) \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C(\delta, \lambda, \beta)$$

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

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

which proves that the bound for Gibbs-optimal base learners in (65) and (12) is smaller than the bound in Theorem 4 which holds uniformly for all $Q \in \mathcal{M}(\mathcal{H})$.

A.5 Proof of Proposition 9: PAC-Optimal Hyper-Posterior

An objective function corresponding to (12) reads as

$$J(\mathcal{Q}) = -\mathbb{E}_{\mathcal{Q}}\left[\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z(S_i,P)\right] + D_{KL}(\mathcal{Q}||\mathcal{P})\,. \tag{67}$$

To obtain $J(\mathcal{Q})$, we omit all additive terms from (12) that do not depend on $\mathcal{Q}$ and multiply by the scaling factor $\frac{\lambda n\beta}{n\beta+\lambda}$. Since the described transformations are monotone, the minimizing distribution of $J(\mathcal{Q})$, that is,

$$\mathcal{Q}^* = \operatorname*{arg\,min}_{\mathcal{Q}\in\mathcal{M}(\mathcal{M}(\mathcal{H}))} J(\mathcal{Q})\,, \tag{68}$$

is also the minimizer of (12). More importantly, $J(\mathcal{Q})$ is structurally similar to the generic minimization problem in Lemma 3. Hence, we can invoke Lemma 3 with $A = \mathcal{M}(\mathcal{H})$, $g(a) = -\sum_{i=1}^{n}\log Z(S_i,P)$, $\beta = \frac{1}{\sqrt{n\tilde{m}}+1}$, to show that the optimal hyper-posterior is

$$\mathcal{Q}^*(P) = \frac{\mathcal{P}(P)\exp\left(\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z_\beta(S_i,P)\right)}{Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P})}\,, \tag{69}$$

wherein $Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P}) = \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z_\beta(S_i,P)\right)\right]$. □

A.6 Proof of Corollary 10

The proof follows the same scheme as the proof of Corollary 7. However, for completeness, we'll state it the following: If we choose $\mathcal{Q} = \mathcal{Q}^*$, the first two terms of the PAC-Bayes bound in (12) can be re-arranged as follows:

$$-\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\beta}\mathbb{E}_{P\sim\mathcal{Q}}\left[\log Z_\beta(S_i,P)\right] + \left(\frac{1}{\lambda}+\frac{1}{n\beta}\right) D_{KL}(\mathcal{Q}^*||\mathcal{P}) \tag{70}$$

$$= \mathbb{E}_{P\sim\mathcal{Q}}\left[-\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\beta}\log Z_\beta(S_i,P) + \left(\frac{1}{\lambda}+\frac{1}{n\beta}\right)\log\frac{\mathcal{P}(P)\exp\left(\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z_\beta(S_i,P)\right)}{\mathcal{P}(P)Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P})}\right] \tag{71}$$

$$= \mathbb{E}_{P\sim\mathcal{Q}}\left[-\left(\frac{1}{\lambda}+\frac{1}{n\beta}\right)\log Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P})\right] = -\left(\frac{1}{\lambda}+\frac{1}{n\beta}\right)\log Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P}) \tag{72}$$

Hence, if we insert (72) in (12), we obtain

$$\mathcal{L}(\mathcal{Q},\mathcal{T}) \leq -\left(\frac{1}{\lambda}+\frac{1}{n\beta}\right)\log Z^{\mathrm{II}}(S_1,...,S_n,\mathcal{P}) + C(\delta,\lambda,\beta)\ . \tag{73}$$

which concludes the proof. □

A.7 Proof of Theorem 11

Step 1: Uniform bound for any $Q_i \in \mathcal{M}(\mathcal{H})$. In the following, we bound the generalization error of a base learner $Q(S,P)$ with priors $P \sim \mathcal{P}$ when applied to a new task $\tau \sim \mathcal{T}_h$, i.e.,

$$\mathcal{L}(\mathcal{P},\mathcal{T}) = \mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}_h}\left[\mathcal{L}(Q(S,P),\mathcal{D})\right]\ . \tag{74}$$

Similar to the proof of Theorem 4, we first bound the intermediary average generalization error across a set of given tasks $\{\tau_1,...,\tau_n\}$, i.e.,

$$\mathcal{L}(\mathcal{P},\tau_1,...,\tau_n) = \mathbb{E}_{P\sim\mathcal{P}}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(Q(P,S_i),\mathcal{D}_i)\right].$$

However, unlike in the meta-learning case, the priors are directly sampled from the hyper-prior $\mathcal{P}$ instead of a data-dependent hyper-posterior $\mathcal{Q}$. Thus, we only need to apply the change-of-measure inequality once. In particular, we apply Lemma 14 with the following instantiations. We set $A = \mathcal{M}(\mathcal{H})\times\mathcal{H}^n$, $f=(P,h_1,...,h_n)\in A$, $\pi=(\mathcal{P},P^n)$ and $\rho=(\mathcal{P},Q^n)$ as a distribution over the joint two-level hypotheses corresponding to hierarchical sampling via $P\sim\mathcal{P}$ and $h_i\sim Q_i = Q(S_i,P)\ \forall i=1,...,n$. Furthermore, we set $g(f)=g(P,h_1,...,h_n) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(h_i,\mathcal{D}_i)-\hat{\mathcal{L}}(h_i,S_i)$ and $\lambda=\gamma$. Hence, we obtain that, for $\gamma>0$,

$$\begin{aligned}\mathcal{L}(\mathcal{P},\tau_1,...,\tau_n) \leq\ & \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{P}}\left[\mathcal{L}(Q_i,S_i)\right] + \frac{1}{\gamma}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{P}}\left[D_{KL}(Q_i||P)\right]\\ &+\frac{1}{\gamma}\underbrace{\log\mathbb{E}_{P\sim\mathcal{P}}\mathbb{E}_{h\sim P}\left[e^{\frac{\gamma}{n}\sum_{i=1}^{n}(\mathcal{L}(h,\mathcal{D}_i)-\hat{\mathcal{L}}(h,S_i))}\right]}_{\Upsilon^{\mathrm{I}}(\gamma)}\ .\end{aligned} \tag{75}$$

Next, we bound the difference between $\mathcal{L}(\mathcal{P},\mathcal{T}_h)$ and $\mathcal{L}(\mathcal{P},\tau_1,...,\tau_n)$. Unlike the meta-learning bound, we do not require the change of measure lemma since there is no meta-learned hyper-posterior. Instead, we can simply use Jensen's inequality to obtain that

$$\mathcal{L}(\mathcal{P},\mathcal{T}) - \mathcal{L}(\mathcal{P},\tau_1,...,\tau_n) \leq \frac{1}{\lambda}\underbrace{\log\mathbb{E}_{P\sim\mathcal{P}}\left[e^{\frac{\lambda}{n}\sum_{i=1}^{n}\mathbb{E}_{\mathcal{D}\sim\mathcal{T}}\mathbb{E}_{S\sim\mathcal{D}^m}[\mathcal{L}(Q(S,P),\mathcal{D})]-\mathcal{L}(Q(S_i,P),\mathcal{D}_i)}\right]}_{\Upsilon^{\mathrm{II}}(\lambda)}\ . \tag{76}$$

Combining (76) with (75) we have that

$$\mathcal{L}(\mathcal{P},\mathcal{T}) \leq \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{P}}\left[\mathcal{L}(Q_i,S_i)\right] + \frac{1}{\gamma}\sum_{i=1}^{n}\mathbb{E}_{P\sim\mathcal{P}}\left[D_{KL}(Q_i||P)\right] + \Upsilon^{\mathrm{I}}(\gamma) + \Upsilon^{\mathrm{II}}(\lambda)\ . \tag{77}$$

Now, it remains to bound (with high probability) the CGFs $\Upsilon^{\mathrm{I}}(\gamma) + \Upsilon^{\mathrm{II}}(\lambda)$ which are identical to those in the proof of Theorem 4. Using the results from there and $\gamma := n\beta$ we have with probability at least $1-\delta$ that

$$\mathcal{L}(\mathcal{P}, \mathcal{T}) \leq \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{P}}\left[\mathcal{L}(Q_i, S_i)\right] + \frac{1}{n\beta}\sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{P}}\left[D_{KL}(Q_i||P)\right] \tag{78}$$

$$+ \underbrace{\bar{\Psi}^{\mathrm{I}}(\beta) + \Psi^{\mathrm{II}}(\lambda) + \frac{1}{\sqrt{n}}\log\frac{1}{\delta}}_{=C(\beta,\lambda,\delta)}. \tag{79}$$

Step 2: Bound for Gibbs learner. Now, we assume a Gibbs distribution as base learner $Q^*(S_i, P) := \frac{P(h)e^{-\beta\hat{\mathcal{L}}(h,S_i)}}{Z_\beta(S_i,P)}$. Following the same steps as in Appendix 7, we obtain

$$\mathcal{L}(\mathcal{P}, \mathcal{T}) \leq \; -\frac{1}{n\beta}\sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{P}}\left[\log Z_\beta(S_i, P)\right] + C(\beta, \lambda, \delta) \tag{80}$$

which concludes the proof of the bound. □

A.8 Proof of Proposition 12

To show that meta-learning improves upon single-task learning, we study the difference between (14) and (16), i.e., $\Delta = (16) - (14)$. After removing terms that cancel out, we have that

$$\Delta = \left(\frac{1}{\lambda} + \frac{1}{n\beta}\right)\log Z^{\mathrm{II}}(S_1, ..., S_n, \mathcal{P}) - \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\beta}\mathbb{E}_{P\sim\mathcal{P}}\left[\log Z_\beta(S_i, P)\right] \tag{81}$$

$$= \left(\frac{1}{\lambda} + \frac{1}{n\beta}\right)\left[\log \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z_\beta(S_i, P)\right)\right]\right. \tag{82}$$

$$\left. - \mathbb{E}_{P\sim\mathcal{P}}\left[\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\log Z_\beta(S_i, P)\right]\right] \tag{83}$$

$$= \left(\frac{1}{\lambda} + \frac{1}{n\beta}\right)\log \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{\lambda}{n\beta+\lambda}\sum_{i=1}^{n}\left(\log Z_\beta(S_i, P) - \mathbb{E}_{P\sim\mathcal{P}}\left[\log Z_\beta(S_i, P)\right]\right)\right)\right] \tag{84}$$

Using $\lambda = \sqrt{m}$ and $\beta = \sqrt{m}$, we can re-write this as

$$\Delta = \left(\frac{1}{\sqrt{m}} + \frac{1}{n\sqrt{m}}\right)\log \mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\frac{1}{\sqrt{mn}+1}\sum_{i=1}^{n}\left(\log Z(S_i, P) - \mathbb{E}_{P\sim\mathcal{P}}\left[\log Z(S_i, P)\right]\right)\right)\right] \tag{85}$$

where we have abbreviated $Z_{\sqrt{n}}(S_i, P)$ as $Z(S_i, P)$.
Finally, we show that Δ is non-negative via Jensen's inequality. In particular, the non-negativity follows from

$$\log \mathbb{E}_{P\sim\mathcal{P}}\left[e^{\frac{1}{\sqrt{nm}+1}\sum_{i=1}^{n}\left(\log Z_\beta(S_i,P) - \mathbb{E}_{P\sim\mathcal{P}}\left[\log Z_\beta(S_i,P)\right]\right)}\right] \tag{86}$$

$$\geq \frac{1}{\sqrt{nm}+1} \sum_{i=1}^{n} \mathbb{E}_{P\sim\mathcal{P}} \left[(\log Z_\beta(S_i, P) - \mathbb{E}_{P\sim\mathcal{P}} \left[\log Z_\beta(S_i, P) \right]) \right] \tag{87}$$

$$= 0 \ . \tag{88}$$

A.9 Proof of the Equivalence of Variational Inference and Minimization of the PAC-Bayes Meta-Learning Bound

Here, we show that performing variational inference (Blei et al., 2017) on the PACOH $\mathcal{Q}^*$ is equivalent to minizing the PAC-Bayesian meta-learning bound in (12). A similar connection for per-task learning has previously been pointed out by Thakur et al. (2019). We can write the optimal variational distribution $\tilde{\mathcal{Q}}$ with respect to $\mathcal{Q}^*$ as

$$\tilde{\mathcal{Q}} = \arg\min_{\mathcal{Q}\in\mathcal{F}} \ D_{KL}(\mathcal{Q}||\mathcal{Q}^*) = \arg\min_{\mathcal{Q}\in\mathcal{F}} \ \mathbb{E}_{P\sim\mathcal{Q}} \left[\log \mathcal{Q}(P) - \log \mathcal{Q}^*(P) \right] \tag{89}$$

$$= \arg\min_{\mathcal{Q}\in\mathcal{F}} \ \mathbb{E}_{P\sim\mathcal{Q}} \left[\log \mathcal{Q}(P) - \log \mathcal{P}(P) - \left(\frac{\beta}{\beta+\lambda} \sum_{i=1}^{n} \frac{1}{\beta_i} \log Z(S_i, P) \right) + \log Z^{\mathrm{II}} \right] \tag{90}$$

$$= \arg\min_{\mathcal{Q}\in\mathcal{F}} \ -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\beta_i} \mathbb{E}_{P\sim\mathcal{Q}} \left[\log Z(S_i, P) \right] + \left(\frac{1}{\lambda} + \frac{1}{\beta} \right) D_{KL}(\mathcal{Q}||\mathcal{P}) \ . \tag{91}$$

Now it is straightforward to see that (91) is the same as the meta-learning PAC-Bayes bound in (12) up to the constant $C(\delta, n, \beta)$. Hence, we can conclude that variational inference with respect to $\mathcal{Q}^*$ is equivalent to minimizing (12) over the same variational family $\mathcal{F}$.

Appendix B. Bounding the CGF for linear regression

The setting and analysis in this section is inspired by Germain et al. (2016) who provide CGF bounds for per-task learning with linear regression. However, we bound the CGFs for meta-learning which is significantly more involved due to the hierarchical nature of the priors and data-generating process.

Assumptions: Regression problem with $\mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^d \times \mathbb{R}$, family of linear predictors: $\mathcal{H} = \{h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} \mid \mathbf{w} \in \mathbb{R}^d\}$, Gaussian prior $P_{\mu_P, \sigma_P^2} = \mathcal{N}(\mu_P, \sigma_P^2 \mathbf{I})$, Gaussian hyper-prior $\mathcal{P}(\mu_p) = \mathcal{N}(0, \sigma_{\mathcal{P}}^2 \mathbf{I})$, Loss function $l(\mathbf{w}, \mathbf{x}, y) = \frac{1}{2}\ln(2\pi\sigma^2) + \frac{1}{2\sigma^2}(\mathbf{w}^\top \mathbf{x} - y)^2$.

Data generating process: $\mathbf{w}_i^* \sim \mathcal{N}(\mu_{\mathcal{T}}, \sigma_{\mathcal{T}}^2 \mathbf{I})$, $\mathbf{x} \sim p(\mathbf{x}) = \mathcal{N}(0, \sigma_{\mathbf{x}}^2)$, $y = \mathbf{w}_i^{*\top} \mathbf{x} + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. Thus, we can write the conditional label distribution as $p_{\mathbf{w}_i^*}(y|\mathbf{x}) = \mathcal{N}(\mathbf{w}_i^{*\top} \mathbf{x}, \sigma_\epsilon^2)$. Accordingly, the data distribution follows as $\mathcal{D}_i = p(\mathbf{x})\, p_{\mathbf{w}_i^*}(y|\mathbf{x})$.

Bounding $\mathbf{\Psi}^{\mathrm{I}}(\gamma)$: We aim to bound

$$\Phi^{\mathrm{I}}(\beta) = \frac{1}{n\beta} \sum_{i=1}^{n} \sum_{j=1}^{m} \ln \mathbb{E}_{\mathcal{P}} \mathbb{E}_P \mathbb{E}_{\mathcal{D}_i} \left[e^{\frac{\beta}{m} V_{ij}^{\mathrm{I}}} \right] \tag{92}$$

For that, we focus on the cumulant-generating function of $V_{ij} = \mathcal{L}(h, \mathcal{D}_i) - l(h, z) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_i} \left[l(\mathbf{w}, \mathbf{x}, y) \right] - l(\mathbf{w}, \mathbf{x}, y)$, i.e.,

$$\Gamma^{\mathrm{I}}(\gamma) = \ln \mathbb{E}_{\mathcal{P}} \mathbb{E}_P \mathbb{E}_{\mathcal{D}_i} e^{\gamma V_i} \leq \frac{\gamma^2 s^2}{2(1-\gamma c)} \ , \quad \forall \gamma \in (0, 1/c) \ . \tag{93}$$

$$\Gamma^{\mathrm{I}}(\gamma) = \ln \mathbb{E}_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \exp\left(\frac{\gamma}{2\sigma^2}\left(\mathbb{E}_{(\mathbf{x},y)}\left[l(\mathbf{w},\mathbf{x},y)\right] - l(\mathbf{w},\mathbf{x},y)\right)\right) \tag{94}$$

$$= \ln \mathbb{E}_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \exp\left(\frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w} - \mathbf{w}_i^*||^2 + \sigma_\epsilon^2 - (\mathbf{w}^\top \mathbf{x} - y)^2\right)\right) \tag{95}$$

$$= \ln \mathbb{E}_{\mathbf{w}} \mathbb{E}_{\mathbf{x}} \mathbb{E}_{\epsilon} \exp\left(\frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w} - \mathbf{w}_i^*||^2 + \sigma_\epsilon^2 - ((\mathbf{w} - \mathbf{w}_i^*)^\top \mathbf{x} + \epsilon)^2\right)\right) \tag{96}$$

$$= \ln \mathbb{E}_{\mathbf{w}} \left[\exp\left(\frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w} - \mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right)\right) \underbrace{\mathbb{E}_{\mathbf{x}} \mathbb{E}_{\epsilon} \exp\left(-\frac{\gamma}{2da\sigma^2}((\mathbf{w} - \mathbf{w}_i^*)^\top \mathbf{x} + \epsilon)^2\right)}_{(*)} \right] \tag{97}$$

$$(*) = \mathbb{E}_{\mathbf{x}} \mathbb{E}_{\epsilon} \exp\left(-\frac{\gamma}{2\sigma^2}((\mathbf{w} - \mathbf{w}_i^*)^\top \mathbf{x} + \epsilon)^2\right) \tag{98}$$

$$= \left(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2}\right)^{-\frac{1}{2}} \mathbb{E}_{\mathbf{x}} \exp\left(-\frac{\frac{\gamma}{2\sigma^2}}{1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2}}((\mathbf{w} - \mathbf{w}_i^*)^\top \mathbf{x})^2\right) \tag{99}$$

$$= \left(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2}\right)^{-\frac{1}{2}} \left(1 + \frac{\frac{\gamma}{\sigma^2}}{1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2}} ||\mathbf{w} - \mathbf{w}_i^*||^2 \sigma_{\mathbf{x}}^2\right)^{-\frac{1}{2}} \tag{100}$$

$$= \left(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2} + \frac{\gamma}{\sigma^2} ||\mathbf{w} - \mathbf{w}_i^*||^2 \sigma_{\mathbf{x}}^2\right)^{-\frac{1}{2}} \tag{101}$$

Now, we can insert (101) into (97):

$$\Gamma^{\mathrm{I}}(\gamma) = \ln \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(0, \sigma_P^2 I_d)} \left[\frac{\exp\left(\frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w} - \mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right)\right)}{(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2} + \frac{\gamma}{\sigma^2} ||\mathbf{w} - \mathbf{w}_i^*||^2 \sigma_{\mathbf{x}}^2)^{\frac{1}{2}}} \right] \tag{102}$$

$$\leq \frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right) + \ln \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(0, \sigma_P^2 \mathbf{I}_d)} \left[\frac{\exp\left(\frac{\gamma}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mathbf{w}||^2\right)}{(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2} + \gamma \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} ||\mathbf{w} - \mathbf{w}_i^*||^2)^{\frac{1}{2}}} \right] \tag{103}$$

$$\leq \frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2 ||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right) + \ln \mathbb{E}_{w \sim \mathcal{N}(0, \sigma_P^2)} \left[\frac{\exp\left(\frac{\gamma}{2\sigma^2} \sigma_{\mathbf{x}}^2 dw^2\right)}{(1 + \gamma \frac{\sigma_\epsilon^2}{\sigma^2} + \gamma \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} ||\mathbf{w}_i^*||^2 + \gamma \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} dw^2)^{\frac{1}{2}}} \right] \tag{104}$$

Lemma 15 *Let $a, b \in \mathbb{R}^+$ and $x \sim \mathcal{N}(0, \sigma^2)$ be a Gaussian random variable with mean 0 and variance σ^2. Then, we have*

$$\mathbb{E}_{x \sim \mathcal{N}(0,\sigma^2)} \left[\frac{e^{ax^2}}{(b + 2ax^2)^{\frac{1}{2}}} \right] < \frac{1}{(b - 2ba\sigma^2))^{\frac{1}{2}}} \tag{105}$$

Proof

$$E_{x \sim \mathcal{N}(0,\sigma^2)} \left[\frac{e^{ax^2}}{(b + 2ax^2)^{\frac{1}{2}}} \right] = \frac{e^{\frac{b}{8}\left(\frac{1}{a\sigma^2} - 2\right)} K_0\left(\frac{b}{8}\left(\frac{1}{a\sigma^2} - 2\right)\right)}{\sqrt{4a\pi\sigma^2}} \tag{106}$$

wherein $K_v(y)$ is the modified Bessel function of the second kind. In particular, for $v = 0$ we have

$$K_0(y) = \int_0^\infty e^{-y\cosh(t)}dt < \frac{\sqrt{\pi}e^{-y}}{\sqrt{2y}} \tag{107}$$

For a proof of the inequality in (107), we refer to Yang and Chu (2017). Using (107) to upper-bound (106), we obtain

$$E_{x\sim\mathcal{N}(0,\sigma^2)}\left[\frac{e^{ax^2}}{(b+2ax^2)^{\frac{1}{2}}}\right] < \frac{1}{\sqrt{b(1-2a\sigma^2)}} \tag{108}$$

which concludes the proof. ■

Next, we invoke Lemma 15 with $a = \frac{\gamma}{2\sigma^2}\sigma_{\mathbf{x}}^2 d$ and $b = 1 + \frac{\gamma}{\sigma^2}\sigma_\epsilon^2 + \frac{\gamma}{\sigma^2}\sigma_{\mathbf{x}}^2||\mathbf{w}_i^*||^2$, such that

$$(104) \leq \frac{\gamma}{2\sigma^2}\left(\sigma_{\mathbf{x}}^2||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right) - \frac{1}{2}\ln\left(b(1-2a(\sigma_P^2+\sigma_{\mathcal{P}}^2))\right) \tag{109}$$

$$= \frac{\gamma}{2\sigma^2}\underbrace{\left(\sigma_{\mathbf{x}}^2||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right)}_{\vartheta_i} - \frac{1}{2}\ln\left(\left(1+\frac{\gamma}{\sigma^2}\underbrace{\left(\sigma_{\mathbf{x}}^2||\mathbf{w}_i^*||^2 + \sigma_\epsilon^2\right)}_{\vartheta_i}\right)\left(1-\frac{\gamma}{\sigma^2}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)\right)\right) \tag{110}$$

$$= \frac{\gamma}{2\sigma^2}\vartheta_i - \frac{1}{2}\ln\left((1+\frac{\gamma}{\sigma^2}\vartheta_i - \frac{\gamma}{\sigma^2}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2) - \frac{\gamma^2}{\sigma^4}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)\vartheta_i\right) \tag{111}$$

$$\leq \frac{\gamma}{2\sigma^2}\vartheta_i + \frac{\frac{\gamma}{2\sigma^2}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2) + \frac{\gamma^2}{2\sigma^4}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)\vartheta_i - \frac{\gamma}{2\sigma^2}\vartheta_i}{1+\frac{\gamma}{\sigma^2}\vartheta_i - \frac{\gamma}{\sigma^2}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2) - \frac{\gamma^2}{\sigma^4}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)\vartheta_i} \tag{112}$$

$$= \frac{\gamma}{2\sigma^2}\vartheta_i + \frac{\frac{\gamma}{2}c_i}{1-\gamma c_i} = \frac{\gamma^2\left(\frac{\vartheta_i}{\sigma^2}(\frac{1}{\gamma}-c_i)+\frac{c_i}{\gamma}\right)}{2(1-\gamma c_i)} = \frac{\gamma^2 s_i^2}{2(1-\gamma c_i)} \tag{113}$$

with $s_i^2 = \frac{\vartheta_i}{\sigma^2}(\frac{1}{\gamma}-c_i)+\frac{c_i}{\gamma}$, $c_i = \frac{d}{\sigma^2}\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)+\frac{\gamma}{\sigma^4}d\sigma_{\mathbf{x}}^2(\sigma_P^2+\sigma_{\mathcal{P}}^2)\vartheta_i - \frac{\vartheta_i}{\sigma^2}$ and $\vartheta_i = \sigma_{\mathbf{x}}^2||\mathbf{w}_i^*||^2+\sigma_\epsilon^2$.

Bounding $\Psi^{\mathrm{II}}(\gamma)$: To show that the cumulant-generating function of the random variable $V^{\mathrm{II}} := \mathbb{E}_{(D,S)\sim\mathcal{T}}\mathbb{E}_{P\sim\mathcal{P}}\left[\mathcal{L}(Q(P,S),\mathcal{D})\right] - \mathcal{L}(Q(P,S_i),\mathcal{D}_i)$ is *sub-gamma*, we aim to find values $s_{\mathrm{II}}^2 > 0$ and parameter $c_{\mathrm{II}} > 0$ such that

$$\Psi^{\mathrm{II}}(\gamma) = \ln\mathbb{E}_{(\mathcal{D},S)\sim\mathcal{T}}\mathbb{E}_{P\sim\mathcal{P}}\left[\exp\left(\gamma V^{\mathrm{II}}\right)\right] \leq \exp\left(\frac{\gamma^2 s_{\mathrm{II}}^2}{2(1-c_{\mathrm{II}}\gamma)}\right) \quad \forall\gamma\in(0,1/c_{\mathrm{II}}) \tag{114}$$

$$\begin{aligned}
\Psi^{\mathrm{II}}(\lambda) &= \ln\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\exp\left(\frac{\lambda}{2\sigma^2}\left(\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\left[\mathcal{L}(Q(P,\mathcal{S}),\mathcal{D})\right] - \mathcal{L}(Q(P,\mathcal{S}),\mathcal{D})\right)\right) \\
&= \ln\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\exp\left(\frac{\lambda}{2\sigma^2}\left(\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\left[E_{\mathbf{w}\sim Q}\left[\sigma_{\mathbf{x}}^2||\mathbf{w}^*-\mathbf{w}||^2\right]\right] - \mathbb{E}_{\mathbf{w}\sim Q}\left[\sigma_{\mathbf{x}}^2||\mathbf{w}^*-\mathbf{w}||^2\right]\right)\right) \\
&\leq \ln\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\exp\left(\frac{\lambda}{2\sigma^2}\left(\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\mathcal{P}}\left[E_{\mathbf{w}\sim P}\left[\sigma_{\mathbf{x}}^2||\mathbf{w}^*-\mathbf{w}||^2\right]\right] - \mathbb{E}_{\mathbf{w}\sim P}\left[\sigma_{\mathbf{x}}^2||\mathbf{w}^*-\mathbf{w}||^2\right]\right)\right)
\end{aligned}$$

$$
\begin{aligned}
&= \ln \mathbb{E}_{\mathcal{T}} \mathbb{E}_{\mathcal{P}} \exp\left(\frac{\lambda}{2\sigma^2} \left(\left[\sigma_{\mathbf{x}}^2 (||\mu_{\mathcal{T}}||^2 + d\sigma_{\mathcal{T}}^2 + d\sigma_{\mathcal{P}}^2) \right] - (\sigma_{\mathbf{x}}^2 ||\mathbf{w}^* - \mu_P||^2) \right) \right) \\
&= \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 (||\mu_{\mathcal{T}}||^2 + d\sigma_{\mathcal{T}}^2 + d\sigma_{\mathcal{P}}^2) + \ln \mathbb{E}_{\mathcal{T}} \mathbb{E}_{\mathcal{P}} \exp\left(- \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mathbf{w}^* - \mu_P||^2 \right) \\
&= \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 (||\mu_{\mathcal{T}}||^2 + d\sigma_{\mathcal{T}}^2 + d\sigma_{\mathcal{P}}^2) + \ln \left(1 + \frac{\lambda}{\sigma^2} \sigma_{\mathbf{x}}^2 \sigma_{\mathcal{P}}^2 \right)^{-\frac{d}{2}} \mathbb{E}_{\mathcal{T}} \exp\left(- \frac{\frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mathbf{w}^*||^2}{1 + \frac{\lambda}{\sigma^2} \sigma_{\mathbf{x}}^2 \sigma_{\mathcal{P}}^2} \right) \\
&= \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 (||\mu_{\mathcal{T}}||^2 + d\sigma_{\mathcal{T}}^2 + d\sigma_{\mathcal{P}}^2) - \frac{d}{2} \ln \left(1 + \lambda \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} (\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2) \right) - \frac{\frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mu_{\mathcal{T}}||^2}{1 + \lambda \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} (\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)} \\
&\leq \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 (||\mu_{\mathcal{T}}||^2 + d\sigma_{\mathcal{T}}^2 + d\sigma_{\mathcal{P}}^2) - \frac{d \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 (\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)}{1 + \lambda \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} (\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)} - \frac{\frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mu_{\mathcal{T}}||^2}{1 + \lambda \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} (\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)} \\
&= \frac{(\frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mu_{\mathcal{T}}||^2 + \lambda \frac{d}{2} c_{\mathrm{II}})(1 + \lambda c_{\mathrm{II}}) - \lambda \frac{d}{2} c_{\mathrm{II}} - \frac{\lambda}{2\sigma^2} \sigma_{\mathbf{x}}^2 ||\mu_{\mathcal{T}}||^2}{1 + \lambda c_{\mathrm{II}}} \\
&= \frac{\lambda^2 (\frac{\sigma_{\mathbf{x}}^2}{\sigma^2} c_{\mathrm{II}} ||\mu_{\mathcal{T}}||^2 + d c_{\mathrm{II}}^2)}{2(1 + \lambda c_{\mathrm{II}})} = \frac{\lambda^2 s_{\mathrm{II}}^2}{2(1 + \lambda c_{\mathrm{II}})}
\end{aligned}
$$

with $c_{\mathrm{II}} = \frac{\sigma_{\mathbf{x}}^2}{\sigma^2}(\sigma_{\mathcal{P}}^2 + \sigma_{\mathcal{T}}^2)$ and $s_{\mathrm{II}}^2 = \frac{\sigma_{\mathbf{x}}^2}{\sigma^2} c_{\mathrm{II}} ||\mu_{\mathcal{T}}||^2 + d c_{\mathrm{II}}^2$.

Appendix C. PACOH-GP algorithm details

C.1 Meta-training with PACOH-GP

Prior parametrization. When meta-learning a GP prior, we instantiate the GP's mean m_ϕ and kernel function k_ϕ as neural networks, where the parameter vector ϕ can be meta-learned. To ensure the positive-definiteness of the kernel, we use the neural network as feature map $\varphi_\phi(x)$ on top of which we apply a squared exponential (SE) kernel. Accordingly, the parametric kernel reads as $k_\phi(x, x') = \frac{1}{2} \exp\left(-||\varphi_\phi(x) - \varphi_\phi(x')||_2^2\right)$. Both $m_\phi(x)$ and $\varphi_\phi(x)$ are fully-connected neural networks with 4 layers with each 32 neurons and tanh non-linearities. The parameter vector ϕ represents the weights and biases of both neural networks. As hyper-prior we choose a zero-mean isotropic Gaussian $\mathcal{P}(\phi) = \mathcal{N}(0, \sigma_{\mathcal{P}}^2 I)$.

Estimating the hyper-posterior score. To estimate $\nabla_\phi \mathcal{Q}^*(\phi)$, we use mini-batching on the task level. In each iteration, we sample a mini-batch of $H \leq n$ datasets $S_1, ..., S_H$ and form an unbiased estimate of the hyper-posterior score as follows:

$$
\tilde{\nabla}_\phi \log \mathcal{Q}^*(\phi) = \frac{n}{H} \cdot \sum_{h=1}^{H} \frac{1}{m_h + 1} \nabla_\phi \log Z(S_h, P_\phi) + \nabla_\phi \log \mathcal{P}(\phi) \; . \tag{115}
$$

Here, $\nabla_\phi \log Z(S_h, P_\phi)$ is the derivative of the closed-form marginal log-likelihood (see Eq. 26) w.r.t. the prior parameters ϕ.

MAP approximation. A maximum a-posteriori (MAP) approximation of $\mathcal{Q}^*$ is the simplest way to obtain a practical meta-learning algorithm from our PAC-Bayesian theory. In particular, it approximates the $\mathcal{Q}^*(\phi)$ by a Dirac measure $\delta_\phi(\phi^*)$ on the prior parameter vector ϕ^* that maximizes $\mathcal{Q}^*$, i.e., $\phi^* = \arg\max_\phi \mathcal{Q}^*(\phi)$. To find ϕ^*, we initially randomly initialize ϕ, and then optimize ϕ by performing gradient descent on $\tilde{\nabla}_\phi \log \mathcal{Q}^*(\phi)$.

SVGD approximation. SVGD (Liu and Wang, 2016) approximates $\mathcal{Q}^*$ as a set of particles $\hat{\mathcal{Q}} = \{P_1, ..., P_K\}$. In our setup, each particle corresponds to the parameters of the GP prior, i.e., $\hat{\mathcal{Q}} = \{\phi_1, ..., \phi_K\}$. Initially, we sample random priors $\phi_k \sim \mathcal{P}$ from our hyper-prior. Then, the SVGD iteratively transports the set of particles to match $\mathcal{Q}^*$, by applying a form of functional gradient descent that minimizes $D_{KL}(\hat{\mathcal{Q}}|\mathcal{Q}^*)$ in the reproducing kernel Hilbert space induced by $k(\cdot, \cdot)$. We choose a squared exponential kernel with length scale (hyper-)parameter ℓ, i.e., $k(\phi, \phi') = \exp\left(-\frac{||\phi-\phi'||_2^2}{2\ell}\right)$. In each iteration, the particles are updated by

$$\phi_k \leftarrow \phi_k + \eta_t \psi^*(\phi_k) \ , \quad \text{with} \quad \psi^*(\phi) = \frac{1}{K} \sum_{l=1}^{K} \left[k(\phi_l, \phi) \nabla_{\phi_l} \log \mathcal{Q}^*(\phi_l) + \nabla_{\phi_l} k(\phi_l, \phi) \right] \ .$$

VI approximation. When aiming to approximate $\mathcal{Q}^*$ via variational inference, we consider the variational family of Gaussians with diagonal covariance matrices over our prior parameters ϕ. In particular, we have variational hyper-posteriors of the form

$$\tilde{\mathcal{Q}}_\upsilon(\phi) = \mathcal{N}(\phi; \mu_\mathcal{Q}, \sigma_\mathcal{Q}^2), \text{ with } \upsilon = (\mu_\mathcal{Q}, \log \sigma_\mathcal{Q})$$

where we parameterize the variance of $\tilde{\mathcal{Q}}$ in the log-space to avoid a positivity constraint. The resulting VI loss follows as

$$J^{\text{VI}}(\upsilon) = -\mathbb{E}_{\phi \sim \mathcal{Q}_\upsilon} \left[\frac{\tilde{m}}{\tilde{m}+1} \sum_{i=1}^{n} \frac{1}{m_i} \log Z(S_i, P_\phi) + \log \mathcal{P}(\phi) - \log \mathcal{Q}_\upsilon(\phi) \right] . \tag{116}$$

Here, $\tilde{m}$ is the harmonic mean of dataset sizes $m_1, ..., m_n$. To estimate the gradients of $J^{\text{VI}}(\upsilon)$ w.r.t. υ, we employ a pathwise gradient estimator, also known as reparametrization trick. That is, we sample a set of K prior parameters $\phi_k := \mu_\mathcal{Q} + \sigma_\mathcal{Q} \epsilon_k$, $\epsilon_k \sim \mathcal{N}(0, I)$ as well as a mini-batch of H datasets $S_1, ..., S_H$ and compute an unbiased gradient estimate of (116) as follows:

$$\nabla_\upsilon J^{\text{VI}}(\upsilon) \approx -\frac{1}{K} \sum_{k=1}^{L} \nabla_{\mu_\mathcal{Q}, \sigma_\mathcal{Q}} \left(\frac{n}{H} \cdot \frac{\tilde{m}}{\tilde{m}+1} \sum_{h=1}^{H} \frac{1}{m_h} \log Z(S_h, P_{\phi_k}) + \log \mathcal{P}(\phi_k) - \log \mathcal{Q}_\upsilon(\phi_k) \right) . \tag{117}$$

During gradient descent with $\nabla_\upsilon J^{\text{VI}}(\upsilon)$, we employ the adaptive learning rate method Adam. Due to the double stochasticity (mini-batches of tasks and mini-batches of $\phi_k \sim \mathcal{Q}_\upsilon$), we found that in practice the gradient estimates of the marginal log-likelihood term in (117) are very noisy whereas the second and third term (meta-level KL-divergence) are subject to less variance. As a result, the less noisy gradients of the KL-divergence dominate during gradient-descent, pushing the VI posterior towards the prior which in turn leads to a higher entropy of $\mathcal{Q}_\upsilon$ and even noisier gradient estimates for the marginal log-likelihood term. To counteract this explosion in hyper-posterior entropy, we add a weight $0 < \eta < 1$ in front of $\log \mathcal{P}(\phi) - \log \mathcal{Q}_\upsilon(\upsilon)$ which effectively down-scales the effect of $D_{KL}(\mathcal{Q}_\phi || \mathcal{P})$ and improves results significantly. Such tempering of the prior often help when the Gaussian prior (here hyper-prior) is misspecified and has been studied in e.g. Fortuin et al. (2022).

C.2 Meta-Testing / target training with PACOH-GP

Meta-learning with PACOH gives us an approximation of $\mathcal{Q}^*$. In target-testing (see Figure 1), the base learner is instantiated with the meta-learned prior P_ϕ, receives a dataset $\tilde{S} = (\tilde{\mathbf{X}}, \tilde{\mathbf{y}})$

from an unseen task $\mathcal{D} \sim \mathcal{T}$ and outputs a posterior Q as product of its inference. In our GP setup, Q is the GP posterior and the predictive distribution $\hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \phi)$ is a Gaussian (for details, see Rasmussen and Williams, 2006). Since the meta-learner outputs $\mathcal{Q}$, a distribution over priors, we may obtain different predictions for different priors $P_\phi \sim \mathcal{Q}$, sampled from the hyper-posterior. To obtain a predictive distribution we empirically marginalize $\mathcal{Q}$. That is, we draw a set of prior parameters $\phi_1, ..., \phi_K \sim \mathcal{Q}$ from the hyper-posterior, compute their respective predictive distributions $\hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \phi_k)$ and form an equally weighted mixture:

$$\hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \mathcal{Q}) = \mathbb{E}_{\phi\sim\mathcal{Q}}\left[\hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \phi)\right] \approx \frac{1}{K}\sum_{k=0}^{K} \hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \phi_k) \ , \quad \phi_k \sim \mathcal{Q} \quad (118)$$

Since we are concerned with GPs, (118) coincides with a mixture of Gaussians. As one would expect, the mean prediction under $\mathcal{Q}$ (i.e., the expectation of (118)), is the average of the mean predictions corresponding to the sampled prior parameters $\phi_1, ..., \phi_K$. In case of PACOH-GP-VI, we sample $K = 100$ priors from the variational hyper-posterior $\tilde{\mathcal{Q}}$. For PACOH-GP-SVGD, samples from the hyper-posterior correspond to the $K = 10$ particles. PACOH-GP-MAP can be viewed as a special case of SVGD with $K = 1$, that is, only one particle. Thus, $\hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \mathcal{Q}) \approx \hat{p}(y^*|x^*, \tilde{\mathbf{X}}, \tilde{\mathbf{y}}, \phi^{MAP})$ is a single Gaussian.

Appendix D. PACOH-NN algorithm details

Here, we summarize and further discuss our proposed meta-learning algorithm *PACOH-NN*. An overview of our proposed framework is illustrated in Figure 1. Overall, it consists of the two stages *meta-training* and *meta-testing*, which we explain in more details in the following.

D.1 Meta-training with PACOH-NN

The hyper-posterior distribution $\mathcal{Q}$ that minimizes the upper bound on the transfer error is given by $\mathcal{Q}^*(P) \propto \mathcal{P}(P) \exp\left(\sum_{i=1}^{n} \frac{\lambda}{n\beta_i+\lambda} \log \tilde{Z}(S_i, P)\right)$. Here, we no longer assume that $m = m_i \ \forall i = 1, ..., n$, which was done in the theory to maintain notational brevity. Thus, we use a different β_i for each task as we want to set $\beta_i = m_i$ or $\beta_i = \sqrt{m_i}$. Provided with a set of datasets $S_1, ..., S_n$, the meta-learner minimizes the respective meta-objective, in the case of *PACOH-NN-SVGD*, by performing SVGD on the $\mathcal{Q}^*$. *PACOH-NN-MAP* can be considered a special case of the SVGD-based approximation of $\mathcal{Q}^*$ with only one particle, i.e., $K = 1$. Algorithm 2 outlines the required steps in more detail. Alternatively, to estimate the score of $\nabla_{\phi_k} \tilde{\mathcal{Q}}^*(\phi_k)$, we can use mini-batching at both the task and the dataset level. Specifically, for a given meta-batch size of n_{bs} and a batch size of m_{bs}, we get Algorithm 3.

D.2 Meta-testing / target training with PACOH-NN

The result of meta-training with *PACOH-NN* is a set of neural network priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$. To understand how good these meta-learned priors are, we need to instantiate our base learner with these priors and evaluate its performance on an unseen learning task $\tau = (\mathcal{D}, m) \sim \mathcal{T}$ when given a corresponding training dataset $\tilde{S} \sim \mathcal{D}^m$. In case of neural networks, our base learner forms a generalized Bayesian posterior $Q^*(S, P_\phi)$ over neural networks parameters ϕ. Since this $Q^*(S, P_\phi)$ is intractable for neural networks, we employ SVGD to approximate it—a

Algorithm 3 PACOH-NN-SVGD: mini-batched 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**
  $\{T_1, ..., T_{n_{bs}}\} \subseteq [n]$ // sample n_{bs} tasks uniformly at random
  for $i = 1, ..., n_{bs}$ **do**
    $\tilde{S}_i \leftarrow \{z_1, ..., z_{m_{bs}}\} \subseteq S_{T_i}$ // sample m_{bs} datapoints from S_{T_i} uniformly at random
  end for
  for $k = 1, ..., K$ **do**
    $\{\theta_1, ..., \theta_L\} \sim P_{\phi_k}$ // sample NN-parameters from prior
    for $i = 1, ..., n_{bs}$ **do**
      $\log \tilde{Z}(\tilde{S}_i, P_{\phi_k}) \leftarrow \text{LSE}_{l=1}^{L} \left(-\beta_i \hat{\mathcal{L}}(\theta_l, \tilde{S}_i)\right) - \log L$ // estimate generalized MLL
    end for
    $\nabla_{\phi_k} \log \tilde{\mathcal{Q}}^*(\phi_k) \leftarrow \nabla_{\phi_k} \log \mathcal{P}(\phi_k) + \frac{n}{n_{bs}} \sum_{i=1}^{n_{bs}} \frac{\lambda}{n\beta_i + \lambda} \nabla_{\phi_k} \log \tilde{Z}(S_i, P_{\phi_k})$ // compute score
  end for
  $\phi_k \leftarrow \phi_k + \frac{\eta}{K} \sum_{k'=1}^{K} \left[k(\phi_{k'}, \phi_k) \nabla_{\phi_{k'}} \log \tilde{\mathcal{Q}}^*(\phi_{k'}) + \nabla_{\phi_{k'}} k(\phi_{k'}, \phi_k) \right] \; \forall k \in [K]$ // SVGD
end while
Output: set of priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$

standard procedure in the context of Bayesian Neural Networks. Algorithm 4 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 E.2).

Algorithm 4 PACOH-NN: meta-testing

Input: set of priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$, target training dataset $\tilde{S}$
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**
      $\nabla_{\theta_l^k} Q^*(\theta_l^k)) \leftarrow \nabla_{\theta_l^k} \log P_{\phi_k}(\theta_l^k)) + \beta \; \nabla_{\theta_l^k} \mathcal{L}(l, \tilde{S})$ // compute score
    end for
    $\theta_l^k \leftarrow \theta_l^k + \frac{\nu}{L} \sum_{l'=1}^{L} \left[k(\theta_{l'}^k, \theta_l^k) \nabla_{\theta_{l'}^k} \log Q^*(\theta_{l'}^k) + \nabla_{\theta_{l'}^k} k(\theta_{l'}^k, \theta_l^k) \right] \forall l \in [L]$ // update
  end while
end for
Output: a set of NN parameters $\bigcup_{k=1}^{K} \{\theta_1^k ..., \theta_L^k\}$

D.3 Properties of the score estimator

Since the marginal log-likelihood of BNNs is intractable, we have replaced it by a numerically stable Monte Carlo estimator $\log \tilde{Z}_\beta(S_i, P_\phi)$ in (27), in particular

$$\log \tilde{Z}_\beta(S_i, P_\phi) := \log \frac{1}{L} \sum_{l=1}^{L} e^{-\beta \hat{\mathcal{L}}(\theta_l, S_i)} = \text{LSE}_{l=1}^{L} \left(-\beta \hat{\mathcal{L}}(\theta_l, S_i) \right) - \log L \ , \quad \theta_l \sim P_\phi \ . \tag{119}$$

Since the Monte Carlo estimator involves approximating an expectation of an exponential, it is not unbiased. However, we can show that replacing $\log Z_\beta(S_i, P_\phi)$ by the estimator $\log \tilde{Z}_\beta(S_i, P_\phi)$, we still minimize a valid upper bound on the transfer error (see Proposition 13).

Proposition 16 *In expectation, replacing* $\log Z_\beta(S_i, P_\phi)$ *in (12) by the Monte Carlo estimate* $\log \tilde{Z}_\beta(S_i, P) := \log \frac{1}{L} \sum_{l=1}^{L} e^{-\beta \hat{\mathcal{L}}(\theta_l, S_i)}$, $\theta_l \sim P$ *still yields a valid upper bound of the transfer error. In particular, it holds that*

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{P \sim \mathcal{Q}} \left[\log Z(S_i, P) \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C \tag{120}$$

$$\leq -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\beta} \mathbb{E}_{P \sim \mathcal{Q}} \left[\mathbb{E}_{\theta_1, \ldots, \theta_L \sim P} \left[\log \tilde{Z}(S_i, P) \right] \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C. \tag{121}$$

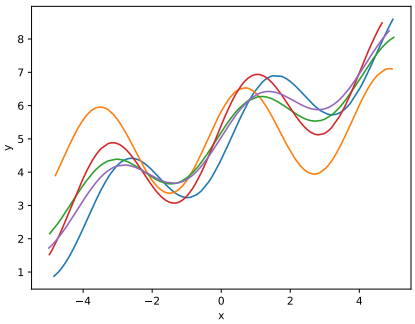
Proof First, we show that:

$$\begin{aligned} \mathbb{E}_{\theta_1, \ldots, \theta_L \sim P} \left[\log \tilde{Z}_\beta(S_i, P) \right] &= \mathbb{E}_{\theta_1, \ldots, \theta_L \sim P} \left[\log \frac{1}{L} \sum_{l=1}^{L} e^{-\beta \hat{\mathcal{L}}(\theta_l, S_i)} \right] \\ &\leq \log \frac{1}{L} \sum_{l=1}^{L} \mathbb{E}_{\theta_l \sim P} \left[e^{-\beta \hat{\mathcal{L}}(\theta_l, S_i)} \right] = \log Z_\beta(S_i, P) \end{aligned} \tag{122}$$

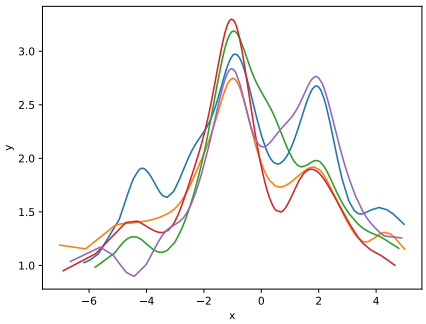
which follows directly from Jensen's inequality and the concavity of the logarithm. Now, Proposition 16 follows directly from (122). ■

By the law of large numbers, it is straightforward to show that as $L \to \infty$, the $\log \tilde{Z}(S_i, P) \xrightarrow{\text{a.s.}} \log Z(S_i, P)$, i.e., the estimator becomes asymptotically unbiased and we recover the original PAC-Bayesian bound (i.e., (121) $\xrightarrow{\text{a.s.}}$ (120)). Also, it is noteworthy that the bound in (121) we get by our estimator is, in expectation, tighter than the upper bound when using the naïve estimator $\log \hat{Z}_\beta(S_i, P) := -\beta \ \frac{1}{L} \sum_{l=1}^{L} \hat{\mathcal{L}}(\theta_l, S_i) \quad \theta_l \sim P_\phi$ which can be obtained by applying Jensen's inequality to $\log \mathbb{E}_{\theta \sim P_\phi} \left[e^{-\beta \hat{\mathcal{L}}(\theta, S_i)} \right]$. In the edge case $L = 1$, our LSE estimator $\log \tilde{Z}_\beta(S_i, P)$ falls back to this naïve estimator and coincides in expectation with $\mathbb{E}[\log \hat{Z}_\beta(S_i, P)] = -\beta \ \mathbb{E}_{\theta \sim P} \hat{\mathcal{L}}(\theta_l, S_i)$. As a result, we effectively minimize the looser upper bound

$$\mathcal{L}(\mathcal{Q}, \mathcal{T}) \leq \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\theta \sim P} \left[\hat{\mathcal{L}}(\theta, S_i) \right] + \left(\frac{1}{\lambda} + \frac{1}{n\beta} \right) D_{KL}(\mathcal{Q} || \mathcal{P}) + C(\delta, n, \tilde{m}).$$



(a) Sinusoid tasks



(b) Cauchy tasks

Figure S1: Depiction of tasks (i.e., functions) sampled from the Sinusoid and Cauchy task environment, respectively. Note that the Cauchy task environment is two-dimensional ($\dim(\mathcal{X}) = 2$), while (b) displays a one-dimensional projection.

$$= \mathbb{E}_{\theta \sim P}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i} -\log p(y_{ij}|x_{ij},\theta)\right] + \left(\frac{1}{\lambda}+\frac{1}{n\beta}\right) D_{KL}(\mathcal{Q}||\mathcal{P}) + C(\delta, n, \tilde{m})$$

Since the boundaries between the tasks vanish in the edge case of $L = 1$, i.e., all datapoints are treated as if they would belong to one dataset, we should choose $L > 1$. In our experiments, we used $L = 5$ and found the corresponding approximation to be sufficient.

Appendix E. Experiments

E.1 Meta-Learning Environments

We provide further details on the meta-learning environments used in Section 8. Information about the numbers of tasks and samples per environments can be found in Table S1.

	Sinusoid	Cauchy	SwissFEL	Physionet	Berkeley
n	20	20	5	100	36
m_i	5	20	200	4 - 24	288

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

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

$$f_{a,b,c,\beta}(x) = \beta \cdot x + a \cdot \sin(1.5 \cdot (x - b)) + c\,, \tag{123}$$

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:

$$a \sim \mathcal{U}(0.7, 1.3), \quad b \sim \mathcal{N}(0, 0.1^2), \quad c \sim \mathcal{N}(5.0, 0.1^2), \quad \beta \sim \mathcal{N}(0.5, 0.2^2)\,. \tag{124}$$

Figure S1a depicts functions $f_{a,b,c,\beta}$ with parameters sampled according to (124). 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 $f(x)$:

$$x \sim \mathcal{U}(-5, 5) \ , \qquad y \sim \mathcal{N}(f_{a,b,c,\beta}(x), 0.1^2) \ . \tag{125}$$

Cauchy. Each task of the Cauchy environment can be interpreted as a two-dimensional mixture of Cauchy distributions plus a function sampled from a Gaussian process prior with zero mean and SE kernel function $k(x, x') = \exp\left(\frac{||x-x'||_2^2}{2l}\right)$ with $l = 0.2$. The (unnormalized) mixture of Cauchy densities is defined as:

$$m(x) = \frac{6}{\pi \cdot (1 + ||x - \mu_1||_2^2)} + \frac{3}{\pi \cdot (1 + ||x - \mu_2||_2^2)} \ , \tag{126}$$

with $\mu_1 = (-1, -1)^\top$ and $\mu_2 = (2, 2)^\top$. Functions from the environment are sampled as

$$f(x) = m(x) + g(x) \ , \qquad g \sim \mathcal{GP}(0, k(x, x')) \ . \tag{127}$$

Figure S1b depicts a one-dimensional projection of functions sampled according to (127). 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 $f(x)$:

$$x := \min\{\max\{\tilde{x}, 2\}, -3\} \ , \quad \tilde{x} \sim \mathcal{N}(0, 2.5^2) \ , \qquad y \sim \mathcal{N}(f(x), 0.05^2) \ . \tag{128}$$

SwissFEL. Free-electron lasers (FELs) accelerate electrons to generate pulsed laser beams in the X-ray spectrum. They can be used to map nanometer-scale structures, e.g., in molecular biology and material science. The accelerator and the electron beam line of a FEL consist of multiple undulators whose parameters can be adjusted to maximize the pulse energy (Kirschner et al., 2019a). Due to different operational modes, parameter drift, and changing (latent) conditions, the laser's pulse energy function changes across time. Hence, 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 (Milne et al., 2017). The input space, corresponding to the laser's parameters, has 12 dimensions. The scalar regression target is the pulse energy. For details on the individual parameters, we refer to Kirschner et al. (2019b). 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 diverse throughout the optimization. Hence, 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 (Silva et al., 2012) published a 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 (ca. 80 % of features), the dataset is also popular as a test bed for time series prediction. 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.

Berkeley-Sensor. We use data from 46 sensors deployed in different locations at the Intel Research lab in Berkeley (Madden, 2004). The dataset contains 4 days of data, sampled at 10 minute intervals. Each task corresponds to one of the 46 sensors and requires auto-regressive prediction, in particular, predicting the next temperature measurement given the last 10 measurement values. 36 sensors (tasks) with data for the first two days are used for meta-training, whereas the remaining 10 sensors with data for the last two days are employed for meta-testing. Note that we separate meta-training and -testing data both temporally and spatially, since the data is non-i.i.d. For the meta-testing, we use the 3rd day as context data, i.e., for target training, and the remaining data for target testing.

E.2 Experimental Methodology

In the following, we describe our experimental methodology and provide details on how the empirical results reported in Section 8 were generated. Overall, evaluating a meta-learner consists of two phases, *meta-training* and *meta-testing*, outlined in Appendix D. The latter can be further sub-divided into *target training* and *target testing*. Figure 1 illustrates these different stages for our PAC-Bayesian meta-learning framework.

The outcome of the training procedure is an approximation of the generalized Bayesian posterior $Q^*(S, P)$ (see Appendix D), pertaining to an unseen task $\tau = (\mathcal{D}, m) \sim \mathcal{T}$ from which we observe a dataset $\tilde{S} \sim \mathcal{D}^m$. 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)* and the *calibration error* (see Appendix E.2.1). Rather than reporting the test log-likelihood, this allows us to measure the quality of mean predictions and the quality of uncertainty estimates separately. The meta-training and meta-testing procedure is repeated for five random seeds that influence both the initialization and gradient-estimates of the algorithms. The reported averages and standard deviations are based on the results obtained for different seeds.

E.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.

Calibration error for regression. 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

$$\hat{q}_h = \frac{|\{y_j \mid \hat{F}(y_j|x_j) \leq q_h, j = 1, ..., m\}|}{m} , \tag{129}$$

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 (Kuleshov et al., 2018), we can define the calibration error as a function of residuals $\hat{q}_h - q_h$, in particular,

$$\text{calib-err} = \frac{1}{H} \sum_{h=1}^{H} |\hat{q}_h - q_h| . \tag{130}$$

Note that while Kuleshov et al. (2018) report 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 (130) with $M = 20$ equally spaced confidence levels between 0 and 1.

Calibration error for classification. Our classifiers output a categorical probability distribution $\hat{p}(y = k|x)$ for $k = 1, ..., C$ where $\mathcal{Y} = \{1, ..., C\}$ with C denoting the number of classes. The prediction of the classifier is the most probable class label, i.e., $\hat{y}_j = \arg\max_k \hat{p}(y_j = k|x_j)$. Correspondingly, we denote the classifiers confidence in the prediction for the input x_j as $\hat{p}_j := \hat{p}(y_j = \hat{y}_j|x_j)$. Following the calibration error definition of Guo et al. (2017), we group the predictions into $H = 20$ interval bins of size $1/H$ depending on their prediction confidence. In particular, let $B_h = \{j \mid p_j \in \left(\frac{h-1}{H}, \frac{h}{H}\right]\}$ be the set of indices of test points $\{(x_j, y_j)\}_{j=1}^m$ whose prediction fall into the interval $\left(\frac{h-1}{H}, \frac{h}{H}\right] \subseteq (0, 1]$. Formally, we define the accuracy of within a bin B_h as $\text{acc}(B_h) = \frac{1}{|B_h|} \sum_{j \in B_h} \mathbf{1}(\hat{y}_i = y_j)$ and the average confidence within a bin as $\text{conf}(B_h) = \frac{1}{|B_h|} \sum_{j \in B_h} \hat{p}_j$. If the classifier is calibrated, we expect that the confidence of the classifier reflects its accuracy on unseen test data, that is, $\text{acc}(B_h) = \text{conf}(B_h)\ \forall h = 1, ..., H$. As proposed by Guo et al. (2017), we use the expected calibration error (ECE) to quantify how much the classifier deviates from this criterion: More precisely, in Table 4, we report the ECE with the following definition:

$$\text{calib-err} = \text{ECE} = \sum_{h=1}^{H} \frac{|B_h|}{m} \big|\text{acc}(B_h) - \text{conf}(B_h)\big| \tag{131}$$

with m denoting the overall number of test points.

E.3 Open Source Code and Hyper-Parameter Selection

We provide open-source implementations of the PACOH method. In particular, the PACOH-GP variants are implemented in PyTorch and available in the PACOH-GP code repository[4]. The PACOH-NN variants are implemented in Tensorflow and available here[5].

For each of the meta-environments and algorithms, we ran a separate hyper-parameter search to select the hyper-parameters. In particular, we used the `hyperopt`[6] package (Bergstra et al.,

4. `https://github.com/jonasrothfuss/meta_learning_pacoh`
5. `https://github.com/jonasrothfuss/pacoh_nn`
6. http://hyperopt.github.io/hyperopt/

Allele	A-0202	A-0203	A-0201	A-2301	A-2402
m_i	1446	1442	3088	103	196

Table S2: MHC-I alleles used for meta-training and corresponding number of samples m_i.

2013) which performs Bayesian optimization based on regression trees. As the optimization metric, we employed the average log-likelihood, evaluated on a separate validation set of tasks. The scripts for reproducing the hyper-parameter search are included in the PACOH-GP code repository. For the reported results, we provide the selected hyper-parameters and detailed evaluation results under `https://tinyurl.com/s48p76x`.

E.4 Meta-Learning for Bayesian Optimization: Vaccine Development

Here, we provide additional details on the experiment in Section 8.3.

Data and BO task setup. We use data from Widmer et al. (2010), which contains the binding affinities (IC_{50} values) of many peptide candidates to seven different MHC-I alleles. Following Krause and Ong (2011), we convert the IC_{50} values into negative log-scale and normalize them such that 500nM corresponds to zero, i.e., $r := -\log_{10}(IC_{50}) + \log_{10}(500)$, which is used as the reward signal (i.e. objective function) for our Bayesian Optimization . We use 5 alleles (A-0202, A-0203, A-0201, A-2301, A-2402) to meta-learn a BNN prior. The alleles and the corresponding number of data points, available for meta-training, are listed in Table S2. The most genetically dissimilar allele (A-6901) is used for our BO task. In each iteration, the experimenter (i.e., BO algorithm) chooses to test one peptide among the pool of 813 candidates and receives r as the reward feedback. Hence, we are concerned with an 813-arm bandit wherein the action $a_t \in \{1, ..., 813\} = \mathcal{A}$ in iteration t corresponds to testing the a_t-th peptide candidate. In response, the algorithm receives the respective negative log-IC_{50} as reward $r(a_t)$.
As metrics, we report the *average regret*

$$R_T^{avg.} := \max_{a \in \mathcal{A}} r(a) - \frac{1}{T} \sum_{t=1}^{T} r(a_t)$$

and the *simple regret*

$$R_T^{simple} := \max_{a \in \mathcal{A}} r(a) - \max_{t=1,...,T} r(a_t) \ .$$

The PACOH-UCB/TS approach. First, we perform meta-learning with PACOH-NN-SVGD on the five datasets $S_1, ..., S_5$ which correspond to the five alleles A-0202, A-0203, A-0201, A-2301, A-2402. As a result, we obtain an approximate hyper-posterior, i.e., a set of priors $\{P_{\phi_1}, ..., P_{\phi_K}\}$.
After the meta-learning stage, we perform BO on the test allele (A-6901). Initially, we query a random action $a^0 \in \mathcal{A}$. From then on, actions a_t ($t > 0$) are chosen either via UCB or Thompson Sampling (TS). In every iteration $t = 1, ..., T$, we have access to the previously queried data $\tilde{S}_{<t} = \{(s_{a_{t'}}, y_{a_{t'}})\}_{t'=0}^{t-1}$. As described in Appendix D.2, we use this data in combination with the meta-learned prior to obtain an approximate posterior which is represented as a set of NN parameters $\bigcup_{k=1}^{K} \{\theta_1^k, \ldots, \theta_L^k\}$ (cf. Algorithm 4). In the case

of the Upper-Confidence Bound (UCB) acquisition algorithm, we choose the action that maximizes the UCB, i.e.,

$$a_t = \arg\max_{a \in \mathcal{A}} \tilde{\mu}(y_a | x_a, \tilde{S}_{<t}) + \beta \tilde{\sigma}(y_a | x_a, \tilde{S}_{<t}) \ , \tag{132}$$

where $\tilde{\mu}(y|x, \tilde{S}_{<t}) = \frac{1}{K \cdot L} \sum_{k=1}^{K} \sum_{l=1}^{L} h_{\theta_l^k}(x)$ is the predictive mean and $\tilde{\sigma}^2(y|x\tilde{S}_{<t}) = \frac{1}{K \cdot L} \sum_{k=1}^{K} \sum_{l=1}^{L} (h_{\theta_l^k}(x) - \tilde{\mu}(y|x, \tilde{S}_{<t}))^2$ the epistemic variance of the BNN trained with $\tilde{S}_{<t}$. We use $\beta = 2$ for the BO experiments. In the case of Thompson Sampling (TS), the next action is chosen by sampling a function from the posterior and picking its maximum. In our case, this is done by uniformly sampling one of the NN particles/parameters and taking the action a that maximizes the corresponding NN function $h_{\theta_l^k}(x_a)$

$$a_t = \arg\max_{a \in \mathcal{A}} h_{\theta_l^k}(x_a) \quad \text{with} \quad k \sim \mathcal{U}(1, ..., L), l \sim \mathcal{U}(1, ..., L) \ . \tag{133}$$

Baselines. The BNN-UCB and BNN-TS baselines work analogously except that instead of meta-learned priors, we use a zero-centered Gaussian prior over the NN parameters. In the case of GP-UCB (Srinivas et al., 2009), a Gaussian Process (GP) instead of a BNN is used as a surrogate model of the objective function. To ensure a fair comparison, the prior parameters of the GP are meta-learned by minimizing the GP's marginal log-likelihood on the five meta-training tasks. For the prior, we use a constant mean function and tried various kernel functions (linear, SE, Matern). Due to the 45-dimensional feature space, we found the linear kernel to work best. So, the constant mean and the variance parameter of the linear kernel are meta-learned.