More Flexible PAC-Bayesian Meta-Learning by Learning Learning Algorithms

Hossein Zakerinia [1] **Amin Behjati** [2] **Christoph H. Lampert** [1]

Abstract

We introduce a new framework for studying meta-learning methods using PAC-Bayesian theory. Its main advantage over previous work is that it allows for more flexibility in how the transfer of knowledge between tasks is realized. For previous approaches, this could only happen indirectly, by means of learning *prior distributions over models*. In contrast, the new generalization bounds that we prove express the process of meta-learning much more directly as learning the *learning algorithm* that should be used for future tasks. The flexibility of our framework makes it suitable to analyze a wide range of meta-learning mechanisms and even design new mechanisms. Other than our theoretical contributions we also show empirically that our framework improves the prediction quality in practical meta-learning mechanisms.

1. Introduction

Machine learning systems have remarkable success in solving complex tasks when they are trained on large amounts of data. However, their success is still limited when only little data is available for a task. One reason for this is that common machine learning algorithms, such as minimizing training loss through gradient-based optimization, are very generic. Tailored to be applicable across a wide range of data sources and target tasks, the underlying models need to have many degrees of freedom, and they require a lot of training data to adjust these suitably. In contrast to this, natural learning systems can learn new tasks from little task-specific data. They achieve this by transferring and reusing information from their past experience to new tasks, instead of learning a new model from scratch every time.

Meta-learning (also called *learning-to-learn*) is a principled way of also giving machine learning systems the ability to share knowledge between related learning tasks (Schmidhuber, 1987; Thrun & Pratt, 1998). Instead of learning individual models for each task, a meta-learner learns a mechanism, a *learning algorithm*, that sets the model parameters given a (usually small) amount of training data.

In practice, there are numerous possibilities to realize this step. *Model-based* approaches to meta-learning learn prototypical models which can efficiently be adjusted (e.g. fine-tuned) to specific tasks (Finn et al., 2017; Nichol et al., 2018). Relatedly, *regularization-based* approaches learn regularization terms that prevent future tasks from overfitting even if trained on little data (Denevi et al., 2019). *Hypernetwork*-based approaches learn secondary networks that output the weights of task-specific models (Zhao et al., 2020; Scott et al., 2024). *Representation-based* approaches learn (often low-dimensional) feature representations in which learning can be performed with less training data than in the original input space (Maurer, 2009; Maurer et al., 2016; Lee et al., 2019). *Optimization-based* approaches learn the steps or (hyper)parameters of an optimization procedure (Hochreiter et al., 2001; Ravi & Larochelle, 2017; Li et al., 2017).

All meta-learning methods strive for *generalization* between previously seen tasks and future ones. Unfortunately, most of the above methods are only understood in terms of their empirical performance on example tasks, but they lack theoretical guarantees on their generalization abilities.

Meta-learning theory studies the theoretical properties of meta-learning method. In particular, it aims at providing quantitative generalization guarantees for them, in the form of high-probability upper bounds on the quality of models learned on future tasks, even before data for these tasks is available. Historically, the first attempts to do so had the form of classical PAC guarantees, which were data-independent and had to be derived individually for each method (Baxter, 2000; Maurer, 2009). However, starting with Pentina & Lampert (2014), most recent works exploit PAC-Bayesian theory, which proved to be more flexible and powerful. Not only does it allow deriving bounds that can be instantiated for different meta-learning methods, but the generalization guarantees are also data-dependent, and interpreting them as learning objectives can inspire new meta-learning methods (Amit & Meir, 2018; Rothfuss et al., 2021). Unfortunately, existing theoretical results still apply

[1]Institute of Science and Technology Austria (ISTA) [2]Sharif University of Technology. Correspondence to: Hossein Zakerinia <Hossein.Zakerinia@ist.ac.at>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

only to a small number of actual meta-learning methods, namely those in which the knowledge transfers from previous to future tasks can be expressed as a shift of prior distributions over models. Of the ones we mention above, that includes methods based on learning a model prototype or a regularization term, but not the more flexible ones based on learning representations, optimizers, or hypernetworks.

Our main contribution in this work is the introduction of new form of PAC-Bayesian generalization bounds that provides theoretical guarantees for a much broader class of meta-learning methods than previous ones. In particular, this includes *all* types of methods listed above. Specifically, the transfer of knowledge from previous to future tasks is modeled not indirectly, using distributions over models priors, but directly, using distributions over *learning algorithms*. This viewpoint reflects the *learning-to-learn* aspect of meta-learning much better than previous ones, as it allows the meta-learner to directly select which learning algorithms are meant to be executed on future tasks. In contrast, the prior-based transfer of previous bounds allows only an indirect influence by means of expressing a preference of some models over others.

Besides our theoretical contributions, we also report on experiments in two standard benchmark settings: we demonstrate that using our generalization bound as a learning objective yields a meta-learning algorithm of improved empirical performance compared to previous methods based on prior-based transfer, and we show that even in the case where knowledge transfer can actually be expressed as model priors, our bound is numerically tighter than previous ones.

2. Background

There are many ways how the sharing of information between learning tasks can be formalized. In this work, we adopt the meta-learning setting first proposed in Baxter (2000) under the name of *learning to learn*. We call a tuple $t = (D, S)$ a *task*, where D is a data distribution over a sample space $\mathcal{Z}$, and $S = \{z_1, \ldots, z_m\}$ is a dataset sampled i.i.d. according to this distribution. A *meta-learning method* (or *meta-learner*) has access to the training sets, $S_1, \ldots, S_n$ of a number of *training tasks*, $t_1, \ldots, t_n$, which themselves are i.i.d. sampled from an unknown distribution, τ, over a *task environment*. Let $\mathcal{F}$ be a hypothesis set of possible models, and $\ell : \mathcal{Z} \times \mathcal{F} \to [0, 1]$ a loss function. For any set X, we denote by $\mathcal{M}(X)$ the set of probability distributions over X, and by $\mathcal{P}(X)$ the power set of X, where for our purposes, only subsets of finite size matter.

The goal of meta-learning is to output (in some parameterized form) a *learning algorithm*, $A : \mathcal{P}(\mathcal{Z}) \to \mathcal{F}$, i.e., a mapping from the set of datasets to the set of models, with the goal to make the *risk* (expected loss) as small as possible in expectation when applying the algorithm A to a future task, i.e. minimize

$$\mathop{\mathbb{E}}_{(D,S)\sim\tau} \mathcal{R}(A(S)) \quad \text{for} \quad \mathcal{R}(f) = \mathop{\mathbb{E}}_{z\sim D} \ell(z, f). \tag{1}$$

Our main tool for analyzing such meta-learning methods theoretically will be PAC-Bayesian learning theory. Before we apply this in the meta-learning setting, we remind the reader of its main concepts in the standard setting.

2.1. PAC-Bayesian Learning

Classical PAC-Bayesian bounds (McAllester, 1998; Maurer, 2004) quantify the generalization properties of *stochastic models*. A stochastic model is parameterized by a distribution, $Q \in \mathcal{M}(\mathcal{F})$, over the model space. For any input $z \in \mathcal{Z}$ it makes a stochastic predictions by sampling $f \sim Q$ and outputting $f(z)$. We extend the loss function to this as

$$\ell(z, Q) := \mathop{\mathbb{E}}_{f\sim Q} \ell(z, f). \tag{2}$$

Now, let $t = (D, S)$ be the given task. The PAC-Bayesian framework provides an upper bound on the *expected risk* of a (stochastic) model, Q,

$$\mathcal{R}(Q) = \mathop{\mathbb{E}}_{z\sim D} \ell(z, Q) \tag{3}$$

in terms of its *empirical risk*

$$\widehat{\mathcal{R}}(Q) = \frac{1}{|S|} \sum_{z\in S} \ell(z, Q), \tag{4}$$

and some complexity terms.

For example, from Maurer (2004) it follows that for any fixed $\delta > 0$ and any fixed prior distribution over models $P \in \mathcal{M}(\mathcal{F})$, with probability at least $1-\delta$ over the sampling of the training dataset, S, it holds that for all $Q \in \mathcal{M}(\mathcal{F})$,

$$\mathcal{R}(Q) \leq \widehat{\mathcal{R}}(Q) + \sqrt{\frac{\mathbf{KL}(Q\|P) + \log(\frac{2\sqrt{m}}{\delta})}{2m}}, \tag{5}$$

where $\mathbf{KL}$ denotes the Kullback-Leibler divergence. In words, the stochastic model Q is guaranteed to generalize well, if it is chosen sufficiently close to the prior, P.

Since their introduction by McAllester (1998), many similar bounds have been derived that mostly differ the way they compare empirical and expected error, the specific form of the complexity term, and with further assumptions they make. See, , e.g., Guedj (2019); Alquier (2024); Hellström et al. (2023) for surveys. However, the bounds have in common that the size of the complexity term is mostly determined by the $\mathbf{KL}$ divergence between the posterior distribution Q and a fixed data-independent prior P, like it does in Equation (5). The bounds also have in common

that they hold uniformly with respect to Q. This means that one can use the right-hand side of the inequality as a training objective, and the guarantees will still hold for the (stochastic) model resulting from minimizing it.

2.2. PAC-Bayesian Meta-Learning

PAC-Bayesian bounds for meta-learning were pioneered by Pentina & Lampert (2014). They assumed a fixed learning procedure that outputs a posterior distribution over models, $Q(S, P) \in \mathcal{M}(\mathcal{F})$, depending on the training data, S, as well as on a prior distribution, $P \in \mathcal{M}(\mathcal{F})$. A canonical example of such a procedure would be to return the stochastic model that minimizes the right hand side of (5).

For any prior, P, the expected risk on a new task,

$$\mathcal{R}(P) = \mathop{\mathbb{E}}_{(S,D)\sim\tau} \mathop{\mathbb{E}}_{z\sim D} \ell(z, Q(S, P)), \tag{6}$$

provides a measure how suitable this choice of prior is for future tasks. However, (6) cannot be computed, because it depends on unobserved quantities. Under Baxter's assumptions that the available training tasks are sampled from the same task environment as future tasks, the *empirical multi-task risk*,

$$\widehat{\mathcal{R}}(P) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|S_i|} \sum_{z\in S_i} \ell(z, Q(S_i, P)), \tag{7}$$

can serve as empirical proxy for $\mathcal{R}(P)$. The difference between (6) and (7) can be bounded with PAC-Bayesian techniques, as long as the prior is not simply chosen in a deterministic way, but by means of specifying its posterior distribution, $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$, called the *hyper-posterior*. Overall, one obtains guarantees that $\mathcal{R}(\mathcal{Q}) = \mathbb{E}_{P\sim\mathcal{Q}} \mathcal{R}(P)$ is bounded by $\widehat{\mathcal{R}}(\mathcal{Q}) = \mathbb{E}_{P\sim\mathcal{Q}} \widehat{\mathcal{R}}(P)$ and some complexity terms that are increasing functions of $\mathbf{KL}(\mathcal{Q}\|\mathcal{P})$, where $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$ is a data-independent *hyper-prior*, and of $\mathbf{KL}(A(S_i, P)\|P)$ in expectation over $P \sim \mathcal{Q}$.

Numerous later works improved and extended these PAC-Bayesian meta-learning bounds: Amit & Meir (2018) designed an optimization algorithm based on this setup for neural networks, Liu et al. (2021) proved bounds with a different form, Guan & Lu (2022) and Riou et al. (2023) proved fast rate bounds for this setup, Friedman & Meir (2023) proved bounds based on data-dependent PAC-Bayes bounds, and Rezazadeh (2022) provided a general framework for proving several different form bounds. Rothfuss et al. (2021; 2023) generalized the setup to unbounded loss functions, and developed a new algorithm for estimating optimal hyper-posteriors. Ding et al. (2021); Tian & Yu (2023) studied this setup for few-shot meta-learning, and Farid & Majumdar (2021) studied the connection between PAC-Bayes and uniform stability in this setup.

Table 1. Notations

$P \in \mathcal{M}(\mathcal{F})$	Prior distribution
$Q \in \mathcal{M}(\mathcal{F})$	Posterior distribution
$\pi \in \mathcal{M}(\mathcal{A})$	Meta-Prior over algorithms
$\rho \in \mathcal{M}(\mathcal{A})$	Meta-Posterior over algorithms
$\mathcal{P}(A) \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$	Hyper-Prior over priors
$\mathcal{Q}(A) \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$	Hyper-Posterior over priors

While these works constitute substantial progress, all of them share a common limitation that they inherited from the setup originally defined in Pentina & Lampert (2014): **they only apply to meta-learning methods that are expressible as a single learning strategy parameterized by a prior distribution over models.** However, many practical meta-learning algorithms do not follow this pattern, thereby preventing the existing PAC-Bayesian frameworks from studying the generalization ability of these algorithms.

One exception is Pentina & Lampert (2015), which provided a bound over transformation operators between tasks, but this situation applies only under rather restrictive assumptions. Another one is the recent Scott et al. (2024), which proved a related PAC-Bayesian bound in the context of personalized federated learning. However, the result provides only rather weak guarantees, because it assumes fixed prior distributions that have to be chosen without any knowledge about the task environment instead of benefiting from environment-dependent priors as the works above. Moreover, Nguyen et al. (2022) provided a bound in an alternative meta-learning framework that uses both validation and training data, which also assumes fixed priors.

As an alternative framework, information-theoretic bounds have been derived (Chen et al., 2021; Hellström & Durisi, 2022; Hellström et al., 2023). These, however, typically provide bounds in expectation rather than with high probability over the training tasks, and they are harder to compute than the PAC-Bayesian ones. Additionally, these works use distribution-dependent priors while the works in the PAC-Bayesian framework use data-dependent priors (through a hyper-posterior). In Section 3.1, we describe the role of hyper-posteriors in more detail.

In this work, **we introduce a new form of PAC-Bayesian meta-learning bounds that overcomes the limitation of previous works.** It works in a more general setup that applies to any set of learning algorithms as well as allowing for algorithm-specific hyper-posteriors.

3. Main Results

In this section, we state and discuss our main result: a generalization bound that holds for any meta-learning method that is expressible as a way to *choose a learning algorithm*

for future tasks.

Formally, let $\mathcal{A} = \{A : \mathcal{P}(\mathcal{Z}) \to \mathcal{M}(\mathcal{F})\}$ be a *set of stochastic learning algorithms* that take as input a dataset and output a posterior distribution over models. Note that this algorithm set does not have to be homogeneous. For example, $\mathcal{A}$, could contain different architectures of neural networks, which are initialized in different ways and adjusted to the training data by different optimizers, or decision trees with different construction rules, or support vector machines with different kernels, or prototype-based classifiers with different distance measures, or all of the above together.

Given a set of training datasets, $S_1, \ldots, S_n$, the meta-learner outputs a posterior distribution over the algorithms, $\rho \in \mathcal{M}(\mathcal{A})$. We call ρ the *meta-posterior (distribution)*, and we define its risk on future tasks as

$$\mathcal{R}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \mathop{\mathbb{E}}_{(S,D)\sim\tau} \mathop{\mathbb{E}}_{z\sim D} \ell(z, A(S)). \tag{8}$$

If this value is small then the meta-learner has done a good job at identifying learning algorithms that work well on future tasks. As such, Equation (8) describes the actual quantity of interest. However, it is not a computable value. Therefore, we introduce the *empirical risk* of the meta-posterior, ρ, on the n training tasks as

$$\widehat{\mathcal{R}}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|S_i|} \sum_{z\in S_i} \ell(z, A(S_i)). \tag{9}$$

Our main results, Theorems 3.1 and 3.2 below, provide upper bounds on $\mathcal{R}(\rho)$ in terms of $\widehat{\mathcal{R}}(\rho)$ and suitable complexity terms.

Before stating them, we introduce one additional source of flexibility that our framework possesses. Remember that in the classical setting of Section 2.2, one fixed hyper-prior distribution over priors was given, and the meta-learner was meant to learn one hyper-posterior distribution over priors. In our setting, especially if the algorithm set is heterogeneous, it stands to reason that different algorithms might benefit from different choices of prior distributions. To express this, let $\mathcal{P} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ now be a fixed data-independent mapping of algorithms to hyper-priors, i.e. for each algorithm, A, one hyper-prior, $\mathcal{P}(A) \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$, is associated in a data-independent way. Analogously, denote by $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ a mapping of algorithms to hyper-posteriors. As part of the meta-learning process, the meta-learner constructs $\mathcal{Q}$ by specifying a hyper-posterior distribution, $\mathcal{Q}(A) \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$, for any learning algorithm $A \in \mathcal{A}$.

We now state our first main result: a generalization bound for $\mathcal{R}(\rho)$ in terms of $\widehat{\mathcal{R}}(\rho)$ that holds with high probability uniformly for all possible choices of ρ and $\mathcal{Q}$.

Theorem 3.1. *For any fixed meta-prior π, fixed hyper-prior mapping $\mathcal{P}$ and any $\delta > 0$, with probability at least $1-\delta$ over the sampling of the training tasks, for all distributions $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior mappings $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ it holds*

$$\mathcal{R}(\rho) \le \widehat{\mathcal{R}}(\rho) + \sqrt{\frac{\mathbf{KL}(\rho\|\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}} \tag{10}$$
$$+ \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \mathbb{E}_{A\sim\rho}[C_1(A,\mathcal{Q},\mathcal{P})] + \log(\frac{8mn}{\delta}) + 1}{2mn}},$$

with

$$C_1(A,\mathcal{Q},\mathcal{P}) = \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A)) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)} \sum_{i=1}^{n} \mathbf{KL}(A(S_i)||P). \tag{11}$$

Our second main result is a tightened variant of Theorem 3.1 that holds in the special case that all algorithms share the same hyper-prior, i.e. $\mathcal{P}$ is constant.

Theorem 3.2. *For any fixed meta-prior π, fixed hyper-prior $\mathcal{P}$ and any $\delta > 0$ with probability at least $1-\delta$ over the sampling of the datasets, for all distributions $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior functions $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ it holds*

$$\mathcal{R}(\rho) \le \widehat{\mathcal{R}}(\rho) + \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}} \tag{12}$$
$$+ \mathop{\mathbb{E}}_{A\sim\rho} \sqrt{\frac{C_2(A,\mathcal{Q},\mathcal{P}) + \log(\frac{8mn}{\delta}) + 1}{2mn}} \tag{13}$$

$$C_2(A,\mathcal{Q},\mathcal{P}) = \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)} \sum_{i=1}^{n} \mathbf{KL}(A(S_i)||P) \tag{14}$$

We provide proof sketches for both theorems in Section 4. For complete proofs, see Appendix A.

3.1. Discussion

In this section, we discuss the properties of the bounds and explain the role and benefits of different terms. We also highlight the differences between our general setup with the more narrow setups of previous works and explain how our algorithm applies to existing meta-learning methods.

Complexity terms As it is common for PAC-Bayesian meta-learning, the bounds (10) and (13) each contain two

complexity terms, which reflect the two types of generalization required for meta-learning guarantees: *task-level generalization*, i.e. generalization from the observed tasks to future tasks, and *within-task generalization* for all training tasks (also called *multi-task generalization*). In the following, we first discuss (10) in detail and afterwards discuss how (13) differs from it.

The first complexity term of (10) expresses the aspect of task-level generalization: it contains the $\mathbf{KL}$-divergence between the data-dependent meta-posterior and the data-independent meta-prior over algorithms. As such, it reflects directly how much the choice of learning algorithm is influenced by the data. In addition, it contains an additional logarithmic term that is small for all practical choices of n and δ. Both terms are divided by $2n$, meaning that the first complexity term decreases with the number of training tasks and vanishes (only) for $n \to \infty$. It is not affected by the number of training samples per task, m. Such a behavior makes sense: the uncertainty about the task environment, i.e. what kind of tasks will appear in the future, is reduced with each additional training task, but having more data points from the tasks available does not provide new information about this aspect.

The second complexity term of Equation (10) contains the same $\mathbf{KL}$-divergence term as well as two additional ones: in simplified form (ignoring the expectations over algorithms), the first term relates the algorithm's hyper-posterior, $\mathcal{Q}(A)$, to its hyper-prior, $\mathcal{P}(A)$. This term reflects the amount by which the meta-learner's choice of priors depends on the observed data. However, the denominator for this complexity term is $2nm$ instead of $2n$ for the first term, indicating that the hyper-posterior can be adjusted rather flexibly without increasing the size of this term too much. The final $\mathbf{KL}$-term relates each task's predicted models, $A(S_i)$, and its respective prior distributions, P. The sum of these n terms is divided by $2nm$, making the term decrease with m and vanish for $m \to \infty$. Once again, a term of this type makes sense. It reflects the average uncertainty about the true risk for models learned from finite data of each training task. When the number of samples, m, per task grows, the uncertainty about each task is reduced. When just the number, n, of training tasks grows, however, the amount of data per task remains the same, so no reduction of the average per-task uncertainty can be expected. The remaining terms in the numerator depend only logarithmically on this number and δ and are negligible in most practical settings.

More precisely, Equation (10) contains the expectations of these terms over the actually stochastic choice of algorithm and prior distribution.

Hyper-posteriors A basic PAC-Bayes bound with a fixed prior would result in separate and independent complexity terms for each task, independent of the environment, and will not take into account the relation between training tasks. Instead, we introduce algorithm-dependent hyper-posteriors, from which we sample priors, and are learned specifically for each learning algorithm, shared between all the tasks. Therefore, the n complexity terms for each A become $\mathbb{E}_{P\sim\mathcal{Q}(A)} \sum_{i=1}^{n} \mathbf{KL}(A(S_i)||P)$ with the additional cost of $\mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A))$, which improves n terms at the cost of one extra term.

In this formulation, $\mathcal{Q}(A)$ can be seen as a similarity measure for the output of the algorithm. The complexity measure is small if for the outputs of the algorithm for n tasks, there is a good hyper-posterior to generate priors close to all posteriors. Note that we can learn different hyper-posteriors for different algorithms, and capture these similarities specifically for the outputs of each algorithm.

Note that the hyper-posterior is a mathematical notion, and the bound holds for all hyper-posteriors at the same time (with high probability). Its role is to help better capture the relations between tasks when using a specific algorithm. For a future task, only the meta-posterior would apply and the role of the hyper-posterior by default is implicit.

Difference between the theorems The bound (13) differs from bound (10) most in the fact that the term $\mathbf{KL}(\rho\|\pi)$ does not appear in the second term, and ρ only appears in the second term as the distribution over algorithms (when we take the expectation of algorithms in $\mathbb{E}_{A\sim\rho}$). The expectation is also moved outside of the square root, which makes the bound tighter (Jensen's inequality). We attribute the differences mainly to the fact that for an algorithm-independent hyper-prior, some steps in the proof can be performed in a tighter way. Generally, both bounds agree in their main behavior with respect to the number of training tasks and samples.

Comparison with previous works The setting of Section 3 is a strict generalization of the setup from previous works where the meta-learner only learned priors. In fact, the latter setting can be recovered from ours as follows: let $Q(S, P)$ be the fixed learning rule of a prior-based meta-learning method. We then define a family of algorithms as $\mathcal{A} = \{Q(\cdot, P) : P \in \mathcal{M}(\mathcal{F})\}$. With each element of $\mathcal{A}$ uniquely determined by a prior P, choosing a distribution of algorithms is equivalent to choosing a distribution over priors. Now, by setting $\mathcal{Q}(A) = \mathcal{Q}(P) = \delta_P$ and $\mathcal{P}(A) = \mathcal{P}(P) = \delta_P$, Theorem 3.1 is applicable. In Section 5.1 we compare the resulting generalization bound numerically to prior ones in this setting.

Similarly, we can recover the bounds of Scott et al. (2024), which transfer algorithms but only allow for a single fixed choice of prior: for any $A \in \mathcal{A}$ we set $\mathcal{Q}(A) = \delta_{P_0}$ and $\mathcal{P}(A) = \delta_{P_0}$, where P_0 is the fixed prior. Again, Theorem 3.2 is now applicable. This construction also shows that

our result not only recovers the bound of Scott et al. (2024) but improves over it. The reason is that Theorem 3.2 holds uniformly over all (potentially data-dependent) choices of $\mathcal{Q}$, of which the construction described above is simply a single data-independent choice.

Recovering common meta-learning methods As discussed in the introduction, previous works on meta-learning that rely on transferring priors over models are not applicable to hypernetwork-based, representation-based, or optimization-based meta-learning methods because these require different parametrizations of their algorithm sets.

In our framework, expressing these methods is straightforward. For the hypernetwork-based methods (Zhao et al., 2020; Scott et al., 2024), the set of algorithms is parametrized by the set of hypernetwork weights. Consequently, learning the algorithm means training the hypernetwork. For representation-based methods (Maurer, 2009; Maurer et al., 2016; Lee et al., 2019) parametrizing each algorithm is a feature extractor, such as a linear projection or a convolutional network. For optimization-based meta-learning (Hochreiter et al., 2001; Ravi & Larochelle, 2017; Li et al., 2017), the algorithm set can contain all hyperparameters to be learned, or the set of all considered optimization procedures, e.g. in the form of the weights of a recurrent network. In all cases, our bound can directly be applicable. In fact, based on our bound one might even improve such methods by suggesting appropriate (meta-)regularization term.

Another observation is that our framework also allows combining different approaches. For example, the algorithm set could be parametrized by the starting point of an optimization step (e.g. the initialization of a network), as well as by a regularization term. The result is a hybrid of methods based on model prototypes and on methods based on learning a regularizer. Prior works were applicable to study either of these approaches individually, but not their combination. Nevertheless, we provide an experimental demonstration that such a hybrid approach can be beneficial in Section 5.

4. Proof Sketch

We provide the proof sketch of Theorem 3.1. The full proof and the proof of Theorem 3.2 are available in Appendix A.

For the proof we first define an intermediate objective that represents the true risk of the training tasks,

$$\widetilde{\mathcal{R}}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \frac{1}{n}\sum_{i=1}^{n} \mathop{\mathbb{E}}_{z\sim D_i} \ell(z, A(S_i)). \tag{15}$$

The proof is then divided into two parts. First, we bound the difference of the true risks between training tasks and future tasks $\mathcal{R}(\rho) - \widetilde{\mathcal{R}}(\rho)$. Second, we bound the difference between the true risk and empirical risk of training tasks $\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho)$. The final result follows by combining the two bounds.

Part I To bound $\mathcal{R}(\rho) - \widetilde{\mathcal{R}}(\rho)$ we use standard PAC-Bayesian arguments, specifically the following lemma:

Lemma 4.1. *For all $\delta > 0$ it holds with probability at least $1 - \frac{\delta}{2}$ over the sampling of tasks for all meta-posteriors $\rho \in \mathcal{M}(\mathcal{A})$:*

$$\mathcal{R}(\rho) - \widetilde{\mathcal{R}}(\rho) \leq \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}}. \tag{16}$$

This lemma is an application of applying the standard PAC-Bayesian bounds Maurer (2004); Pérez-Ortiz et al. (2021). For details, see Appendix A.

Part II We define the following two functions which produce distributions over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$, i.e. they assigns joint probabilities to tuples, $(A, P, f_1, ..., f_n)$, which contain an algorithm, a prior over models, and n models.

Posterior $\mathfrak{Q}(\rho, \mathcal{Q})$: given as input a meta-posterior ρ over algorithms and a hyper-posterior mapping $\mathcal{Q}$ as input, it outputs the distribution over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample an algorithm $A \sim \rho$, *ii)* sample a prior $P \sim \mathcal{Q}(A)$, *iii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim A(S_i)$.

Prior $\mathfrak{P}(\pi, \mathcal{P})$: given as input a meta-prior π over algorithms and a hyper-prior mapping $\mathcal{P}$ as input, it outputs the distribution over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample an algorithm $A \sim \pi$, *ii)* sample a prior $P \sim \mathcal{P}(A)$, *iii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim P$.

Note that the inputs to $\mathfrak{Q}(\rho, \mathcal{Q})$ are data-dependent and will be learned from data. In contrast, the input to $\mathfrak{P}(\pi, \mathcal{P})$ are data-independent and need to be fixed before seeing the data.

With these definitions, we state the following key lemma:

Lemma 4.2. *For any fixed meta-prior π, fixed hyper-prior function $\mathcal{P}$, any $\delta > 0$ and any $\lambda > 0$, it holds with probability at least $1 - \frac{\delta}{2}$ over the sampling of the training datasets that for all meta-posteriors $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior functions $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$:*

$$\begin{aligned}\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq\ & \frac{1}{\lambda}(\mathbf{KL}(\mathfrak{Q}(\rho, \mathcal{Q})\|\mathfrak{P}(\pi, \mathcal{P}))) \\ & + \frac{1}{\lambda}\log(\frac{2}{\delta}) + \frac{\lambda}{8nm}\end{aligned} \tag{17}$$

Proof. First for any task i and any model f_i we define:

$$\Delta_i(f_i) = \mathop{\mathbb{E}}_{z\sim D_i} \ell(z, f_i) - \frac{1}{|S_i|}\sum_{z\in S_i} \ell(z, f_i). \tag{18}$$

By this definition and the definitions of $\widetilde{\mathcal{R}}$ and $\widehat{\mathcal{R}}$ we have:

$$\mathop{\mathbb{E}}_{(A,P,f_1,\ldots,f_n)\sim\mathfrak{Q}(\rho,\mathcal{Q})}\Big[\frac{1}{n}\sum_{i=1}^{n}\Delta_i(f_i)\Big] = \widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \tag{19}$$

Using this equation and the *change of measure inequality* (Seldin et al., 2012) between the two distributions $\mathfrak{Q}(\rho,\mathcal{Q})$ and $\mathfrak{P}(\pi,\mathcal{P})$, for any $\lambda > 0$, any ρ and any $\mathcal{Q}$, we have:

$$\begin{aligned}&\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) - \frac{1}{\lambda}\,\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P}))\\ &\leq \frac{1}{\lambda}\log \mathop{\mathbb{E}}_{(A,P,f_1,\ldots,f_n)\sim\mathfrak{P}(\pi,\mathcal{P})}\prod_{i=1}^{n} e^{\frac{\lambda}{n}\Delta_i(f_i)}\end{aligned} \tag{20}$$

It remains to bound the right-hand of (20). Given that π and $\mathcal{P}$ are data-independent, standard tools (in particular Hoeffding's lemma and Markov's inequality) allow us to prove an upper bound that holds in high probability with respect to the randomness of training datasets, from which the statement in the lemma follows. Detailed steps are provided in Appendix A. □

The following lemma provides a split of the $\mathbf{KL}$-term from Lemma 4.2.

Lemma 4.3. *For the posterior and prior defined above we have:*

$$\begin{aligned}&\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})) = \mathbf{KL}(\rho\|\pi)\\ &+ \mathop{\mathbb{E}}_{A\sim\rho}\left[\mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A)) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P)\right]\end{aligned} \tag{21}$$

The proof makes use of the explicit construction of $\mathfrak{P}(\pi,\mathcal{P})$ and $\mathfrak{Q}(\rho,\mathcal{Q})$. It can be found in Appendix A.

Proof of Theorem 3.1 To get tight guarantees, we need to choose the value of λ in Lemma 4.2 an optimal way dependent on the data. As the statement of the Lemma is not uniform in λ, we do so approximately by allowing a fixed set of values in the range $\{1,...,4mn\}$ and applying a union-bound argument for values in this set. The theorem then follows by combining the result with Lemma 4.1 and using Lemma 4.3.

Proof sketch of Theorem 3.2 The proof is similar to the proof of Theorem 3.1. For the first part, we use the same Lemma 4.1. For the second part, we use the fact that we have the same data-independent prior for all algorithms. Due to this fact, we can remove ρ in the posterior function and prove a generalization bound that holds uniformly for all algorithms applied to the datasets. Therefore we can bound the multi-task generalization of all meta-posteriors ρ without the term $\mathbf{KL}(\rho\|\pi)$, and the result is Theorem 3.2. For the detailed proof, please see Appendix A.

5. Experimental Demonstration

While our main contribution in this work is theoretical, We also report on two experimental studies that allow us to better relate our results to prior work.

5.1. Numerical Evaluations of the Bound

In this section, we numerically compare the tightness of our bound to those from prior work, as far as this is possible. We adopt the same scenario as Rothfuss et al. (2023), in which the goal is to improve the learning of linear classifiers by means of meta-learning a distribution over priors.

In this experiment, each task is a binary classification task, which has a task parameter $\mathbf{w}^*$ and given an input $\mathbf{x} \sim \mathcal{U}([-1,1]^d)$ outputs $y = \mathbb{1}(\mathbf{w}^{*\top}\mathbf{x} \leq 0)$. The task environment is the set of vectors $\mathbf{w}^* \in \mathbb{R}^d$ with task distribution $\tau = \mathcal{N}(\mathbf{w}^*|\mu_\mathcal{T}, \sigma_\mathcal{T}^2 \cdot \mathrm{Id})$ for $\mu_\mathcal{T} = 10 \cdot \mathbf{1}$ and $\sigma_\mathcal{T} = 3$. The model set consists of linear classifiers, $\mathcal{F} = \{\mathbb{1}(\mathbf{w}^\top\mathbf{x} \leq 0) : \mathbf{w} \in \mathbb{R}^d\}$, and the priors and posteriors are Gaussian distributions over their weight vectors. Specifically, the priors have the form $\mathcal{N}(\mathbf{w}|\mu_P, \sigma_P^2 \cdot \mathrm{Id})$ with $\sigma_P = 10$, from which the posteriors are learned by minimizing a PAC-Bayes bound with the logistic regression loss. The meta-learner learns a Gaussian hyper-posterior over the mean of the priors (μ_P), based on the hyper-prior $\mathcal{P}(\mu_P) = \mathcal{N}(\mu_P|\mathbf{0}, \sigma_\mathcal{P}^2 \cdot \mathrm{Id})$ with $\sigma_\mathcal{P} = 20$. For background information on the experimental setting, please see the original reference Rothfuss et al. (2023).

In Figure 1 we plot the numeric values of the right-hand side of our bound (13), i.e. empirical loss plus the complexity terms, in the setting of $d = 2$, $m = 5$, and different values of n. We also plot the corresponding values for the bounds from Pentina & Lampert (2014); Amit & Meir (2018); Rothfuss et al. (2021); Rezazadeh (2022); Guan & Lu (2022), as well as the actual quantities of interest, the meta-test loss, and the meta-training loss. One can see that our bound has the smallest value, i.e. is the tightest. It becomes non-vacuous already for $n = 10$ tasks, while the other bounds are still vacuous for $n < 20$.

We would like to emphasize that this analysis compares the tightness of the bounds just in the setting of prior-based meta-learning, because this is the only setting in which previous works can provide a guarantee. The main advantage of our result, however, is its applicability to many more settings, where a numeric comparison is not possible, because previous works are not applicable.

5.2. Learning Initialization as well as Regularization

In Section 3.1 we argued that hybrid meta-learning scenarios can be beneficial, for example, learning the initialization of a network as well as a regularization term for its opti-

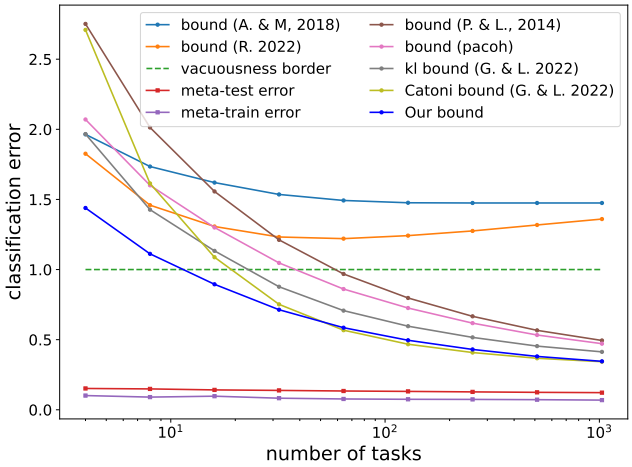

Figure 1. Numeric values of different meta-learning bounds (empirical loss plus complexity terms) for the binary classification task described in Section 5.1. Values below 1 are called *non-vacuous*.

mization. As empirical evidence for this setting, we report on the standard experiments in the literature suggested by Amit & Meir (2018), in which a stochastic neural network is learned for two exemplary meta-learning scenarios, *shuffled pixel* and *permuted labels*. Both are image classification tasks based on the MNIST dataset (LeCun & Cortes, 1998). In the former, tasks differ from each other by different permutations of the input pixels at a fixed subset of locations. In the latter, task differ from each other by different permutations of the label space. In these experiments, there are 10 training tasks with 600 samples per task. We evaluate the methods on 20 tasks with 100 samples per task. For experimental details see Appendix B.

Previous works used this setting as a benchmark for PAC-Bayesian meta-learning bounds in the following way: the learning rule, $Q(S, P)$, consists of first initializing a stochastic network at the mean of the prior, P, and then training the network by minimizing the right-hand side of a PAC-Bayes bound using prior P, where the $\mathbf{KL}$-divergence between the prior and the learned model acts as a regularizer towards the prior mean. This description shows that the prior is used in two different ways, for initialization and as regularizer, even though it is not a priori clear why the best choice for these two quantities would be to make them identical.

In our new framework the roles of initialization and regularization can easily be separated, thereby allowing us to assess the above question quantitatively. To explore this, we use a simple formalization, in which each algorithm consists of two distributions (P_0, P_1) over models. For learning a new task, the stochastic neural network is initialized from P_0, and then trained by minimizing a PAC-Bayes bound with P_1 as prior using gradient descent. Formally, for each

Table 2. Comparison between our mechanism with prior-based mechanisms. The separation of initialization and regularization improves the performance for *permuted labels (PL)*, and achieves similar performance for the *shuffled pixels (SP)*.

Meta-Learning PAC Bayes Bounds: Test Error (%)		
Bound	SP	PL
Independent learning	28.9 ± 1.7	19.6 ± 1.5
(Amit & Meir, 2018)	$\mathbf{9.9 \pm 0.9}$	13.7 ± 3.5
(Rezazadeh, 2022)	11.2 ± 1.0	90.1 ± 5.6
(Guan & Lu, 2022)	20.5 ± 1.1	89.9 ± 0.5
Ours	$\mathbf{9.9 \pm 1.1}$	$\mathbf{7.9 \pm 1.7}$

algorithm, we have $\mathcal{Q}(A) = \mathcal{P}(A) = \delta_{P_1}$, and (ρ_0, ρ_1) as meta-posterior, and (π_0, π_1) as meta-prior, where ρ_0, π_0 are distributions over P_0 and ρ_1, π_1 are distributions over P_1.

For simplicity, we work with Gaussian distributions, and we learn their mean and variance by the re-parametrization trick of Kingma et al. (2015). The optimization is performed as in Amit & Meir (2018): one approximately minimizes the meta-learning bound by optimizing for the hyper-posterior, ρ and for separate task posteriors, $Q_1, \ldots, Q_n$, for a fixed number of epochs (originally 200). For our setup, we use the same implementation, but with the difference that we have a meta-posterior over two distributions. For the first 100 epoch we set them equal, mirroring the prior work. Afterwards, however, fix the meta-posterior over P_0, initialize $Q_1, ..., Q_n$ again by sampling from this distribution, and continue the optimization for another 100 epochs. Note that this results in the same amount of computation as the previous methods, but now split into first learn the network initialization, and then the regularization term conditioned on the learned initialization.

We compare our results with the prior-based bounds of Amit & Meir (2018); Rezazadeh (2022); Guan & Lu (2022), as well as independent learning. The experimental results are shown in Table 2. One can see that for the *permuted label* setting, having different parameters reduces the test error. This shows that the previous setups were indeed suboptimal, and it thereby confirms the benefits of out framework's added flexibility. In the *shuffled pixel* setting, the added flexibility did not yield any benefits, as the system learned almost identical parameters for the initialization and the regularization term. Consequently, the results of our framework are essentially the same as for previous ones. For more discussion on the results we refer the reader to Appendix B.

6. Conclusion

We presented a new framework for the theoretical analysis of meta-learning (or learning-to-learn) methods. Where previous approaches were limited to settings that can be

formulated as learning a prior distribution over models, our new approach takes a more direct approach and formulates the knowledge transfer as learning a preference for learning algorithms. Our main contributions are two PAC-Bayesian generalization bounds that are applicable to essentially all existing transfer mechanisms, including *model prototypes*, *regularization*, *representation learning*, *hypernetworks*, and the transfer of *optimization methods* or *hyperparameters*, or combinations thereof. We believe our approach will prove useful to put more practical meta-learning methods onto solid theoretical foundation, and ideally to inspire improvement, such as new forms of regularization, especially for the low-data regime.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Alquier, P. User-friendly introduction to PAC-Bayes bounds. *Foundations and Trends in Machine Learning*, 17(2):174–303, 2024.

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

Baxter, J. A model of inductive bias learning. *Journal of Artificial Intelligence Research (JAIR)*, 12:149–198, 2000.

Chen, Q., Shui, C., and Marchand, M. Generalization bounds for meta-learning: An information-theoretic analysis. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Denevi, G., Ciliberto, C., Grazzi, R., and Pontil, M. Learning-to-learn stochastic gradient descent with biased regularization. In *International Conference on Machine Learning (ICML)*, 2019.

Ding, N., Chen, X., Levinboim, T., Goodman, S., and Soricut, R. Bridging the gap between practice and PAC-Bayes theory in few-shot meta-learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Farid, A. and Majumdar, A. Generalization bounds for meta-learning via PAC-Bayes and uniform stability. *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

Friedman, L. and Meir, R. Adaptive meta-learning via data-dependent PAC-Bayes bounds. In *Conference on Lifelong Learning Agents (CoLLAs)*, 2023.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Guan, J. and Lu, Z. Fast-rate PAC-Bayesian generalization bounds for meta-learning. In *International Conference on Machine Learning (ICML)*, 2022.

Guedj, B. A primer on PAC-Bayesian learning. *arXiv preprint arXiv:1901.05353*, 2019.

Hellström, F. and Durisi, G. Evaluated CMI bounds for meta learning: Tightness and expressiveness. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Hellström, F., Durisi, G., Guedj, B., and Raginsky, M. Generalization bounds: Perspectives from information theory and PAC-Bayes. *arXiv preprint arXiv:2309.04381*, 2023.

Hochreiter, S., Younger, A. S., and Conwell, P. R. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks (ICANN)*, 2001.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2015.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 1998.

Lee, K., Maji, S., Ravichandran, A., and Soatto, S. Meta-learning with differentiable convex optimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Li, Z., Zhou, F., Chen, F., and Li, H. Meta-SGD: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

Liu, T., Lu, J., Yan, Z., and Zhang, G. PAC-Bayes bounds for meta-learning with data-dependent prior. *arXiv preprint arXiv:2102.03748*, 2021.

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

Maurer, A. Transfer bounds for linear feature learning. *Machine Learning*, 75(3):327–350, 2009.

Maurer, A., Pontil, M., and Romera-Paredes, B. The benefit of multitask representation learning. *Journal of Machine Learning Research (JMLR)*, 17(81):1–32, 2016.

McAllester, D. A. Some PAC-Bayesian theorems. In *Conference on Computational Learning Theory (COLT)*, 1998.

Nguyen, C., Do, T.-T., and Carneiro, G. PAC-Bayes meta-learning with implicit task-specific posteriors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 45(1):841–851, 2022.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Pentina, A. and Lampert, C. H. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning (ICML)*, 2014.

Pentina, A. and Lampert, C. H. Lifelong learning with non-iid tasks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2015.

Pérez-Ortiz, M., Rivasplata, O., Shawe-Taylor, J., and Szepesvári, C. Tighter risk certificates for neural networks. *The Journal of Machine Learning Research*, 22 (1):10326–10365, 2021.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Rezazadeh, A. A unified view on PAC-Bayes bounds for meta-learning. In *International Conference on Machine Learning (ICML)*, 2022.

Riou, C., Alquier, P., and Chérief-Abdellatif, B.-E. Bayes meets Bernstein at the meta level: an analysis of fast rates in meta-learning with PAC-Bayes. *arXiv preprint arXiv:2302.11709*, 2023.

Rothfuss, J., Fortuin, V., Josifoski, M., and Krause, A. PACOH: Bayes-optimal meta-learning with PAC-guarantees. In *International Conference on Machine Learning (ICML)*, 2021.

Rothfuss, J., Josifoski, M., Fortuin, V., and Krause, A. Scalable PAC-Bayesian Meta-Learning via the PAC-Optimal Hyper-Posterior: From Theory to Practice. *Journal of Machine Learning Research (JMLR)*, 2023.

Schmidhuber, J. Evolutionary principles in self-referential learning, 1987.

Scott, J., Zakerinia, H., and Lampert, C. H. PeFLL: Personalized Federated Learning by Learning to Learn. In *International Conference on Learning Representations (ICLR)*, 2024.

Seldin, Y., Laviolette, F., Cesa-Bianchi, N., Shawe-Taylor, J., and Auer, P. PAC-Bayesian inequalities for martingales. *IEEE Transactions on Information Theory*, 58(12): 7086–7093, 2012.

Thrun, S. and Pratt, L. (eds.). *Learning to Learn*. Kluwer Academic Press, 1998.

Tian, P. and Yu, H. Can we improve meta-learning model in few-shot learning by aligning data distributions? *Knowledge-Based Systems*, 277:110800, 2023.

Zhao, D., Kobayashi, S., Sacramento, J., and von Oswald, J. Meta-learning via hypernetworks. In *NeurIPS Workshop on Meta-Learning (MetaLearn)*, 2020.

A. Proofs

In this section, we provide the proofs of the results in the main body of the paper.

A.1. Proof of Theorem 3.1.

For the convenience of the reader we restate Theorem 3.1 here and then prove it.

Theorem A.1. *For any fixed meta-prior $\pi \in \mathcal{M}(\mathcal{A})$, fixed hyper-prior mapping $\mathcal{P} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ and any $\delta > 0$, it holds with probability at least $1 - \delta$ over the sampling of the training tasks, that for all meta-posterior distributions $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior mappings $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ it holds*

$$\mathcal{R}(\rho) \leq \widehat{\mathcal{R}}(\rho) + \sqrt{\frac{\mathbf{KL}(\rho\|\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}} + \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \mathbb{E}_{A\sim\rho}[C_1(A, \mathcal{Q}, \mathcal{P})] + \log(\frac{8mn}{\delta}) + 1}{2mn}}, \tag{22}$$

with

$$C_1(A, \mathcal{Q}, \mathcal{P}) = \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A)) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)} \sum_{i=1}^{n} \mathbf{KL}(A(S_i)||P). \tag{23}$$

The beginning of the proof coincides with the steps of the sketch in Section 4, while the later part provides additional details. As a reminder, we repeat the definitions of our main objects of interest: the risk of a meta-posterior, $\rho \in \mathcal{M}(\mathcal{A})$,

$$\mathcal{R}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \mathop{\mathbb{E}}_{(S,D)\sim\tau} \mathop{\mathbb{E}}_{z\sim D} \ell(z, A(S)), \tag{24}$$

its empirical analog,

$$\widehat{\mathcal{R}}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|S_i|} \sum_{z\in S_i} \ell(z, A(S_i)) \tag{25}$$

as well as the intermediate objective, which represents the true risk of the training tasks:

$$\widetilde{\mathcal{R}}(\rho) = \mathop{\mathbb{E}}_{A\sim\rho} \frac{1}{n} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{z\sim D_i} \ell(z, A(S_i)). \tag{26}$$

We divide the proof into two parts. First, we bound the difference of the true risks between training tasks and future tasks $\mathcal{R}(\rho) - \widetilde{\mathcal{R}}(\rho)$. For the second part, we bound the difference between the true risk and empirical risk of training tasks $\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho)$, and by combining the two bounds we obtain the final result.

Part I For the first part we can use classical PAC-Bayes arguments, because $\mathcal{R}(\rho)$ and $\widetilde{\mathcal{R}}(\rho)$ differ only in the fact that one is an empirical average with respect to the tasks while the other it is expectation. Consequently, one obtains:

Lemma A.2. *For all $\delta > 0$ it holds with probability at least $1 - \frac{\delta}{2}$ over the sampling of tasks that for all meta-posteriors $\rho \in \mathcal{M}(\mathcal{A})$:*

$$\mathcal{R}(\rho) - \widetilde{\mathcal{R}}(\rho) \leq \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}}. \tag{27}$$

Proof. For each algorithm A and task $T = (D, S)$ we define the loss as $l_{\text{task}}(A, T) = \mathbb{E}_{z\sim D}\, \ell(z, A(S))$. For this loss, $\widetilde{\mathcal{R}}(A)$ is the empirical risk, and $\mathcal{R}(A)$ is the true risk. Applying the standard PAC-Bayes bounds (Maurer, 2004; Pérez-Ortiz et al., 2021) to this setting results in this lemma. □

Part II We define the following two functions that produce distributions over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$, i.e. they assigns joint probabilities to tuples, $(A, P, f_1, ..., f_n)$, which contain a algorithm, a prior over models, and n models.

- *Posterior* $\mathfrak{Q}(\rho, \mathcal{Q})$*:* given as input a meta-posterior ρ over algorithms and a hyper-posterior mapping $\mathcal{Q}$ as input, it outputs the distribution over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample an algorithm $A \sim \rho$, *ii)* sample a prior $P \sim \mathcal{Q}(A)$, *iii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim A(S_i)$.
- *Prior* $\mathfrak{P}(\pi, \mathcal{P})$*:* given as input a meta-prior π over algorithms and a hyper-prior mapping $\mathcal{P}$ as input, it outputs the distribution over $\mathcal{A} \times \mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample an algorithm $A \sim \pi$, *ii)* sample a prior $P \sim \mathcal{P}(A)$, *iii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim P$.

Note that the inputs to $\mathfrak{Q}(\rho, \mathcal{Q})$ are data-dependent and will be learned using the data. In contrast, the input to $\mathfrak{P}(\pi, \mathcal{P})$ are data-independent and need to be fixed before seeing the data. With these definitions, we state the following key lemma:

Lemma A.3. *For any fixed meta-prior $\pi \in \mathcal{M}(\mathcal{A})$, fixed hyper-prior mapping $\mathcal{P} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$, any $\delta > 0$, and any $\lambda > 0$, it holds with probability at least $1 - \frac{\delta}{2}$ over the sampling of the training datasets that for all meta-posteriors $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior functions $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$:*

$$\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq \frac{1}{\lambda} \mathbf{KL}\left(\mathfrak{Q}(\rho, \mathcal{Q}) \| \mathfrak{P}(\pi, \mathcal{P})\right) + \frac{1}{\lambda} \log(\frac{2}{\delta}) + \frac{\lambda}{8nm}. \tag{28}$$

Proof. First for any task i and any model f_i we define:

$$\Delta_i(f_i) = \mathop{\mathbb{E}}_{z \sim D_i} \ell(z, f_i) - \frac{1}{|S_i|} \sum_{z \in S_i} \ell(z, f_i). \tag{29}$$

By this definition and the definitions of $\widetilde{\mathcal{R}}$ and $\widehat{\mathcal{R}}$ we have:

$$\mathop{\mathbb{E}}_{(A, P, f_1, \ldots, f_n) \sim \mathfrak{Q}(\rho, \mathcal{Q})} \Big[\frac{1}{n} \sum_{i=1}^{n} \Delta_i(f_i)\Big] = \widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \tag{30}$$

Using this equation and the *change of measure inequality* (Seldin et al., 2012) between the two distributions $\mathfrak{Q}(\rho, \mathcal{Q})$ and $\mathfrak{P}(\pi, \mathcal{P})$, for any $\lambda > 0$, any $\rho \in \mathcal{M}(\mathcal{A})$ and any $\mathcal{Q}$, we have:

$$\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) - \frac{1}{\lambda} \mathbf{KL}(\mathfrak{Q}(\rho, \mathcal{Q}) || \mathfrak{P}(\pi, \mathcal{P})) \leq \frac{1}{\lambda} \log \mathop{\mathbb{E}}_{(A, P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\pi, \mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} \tag{31}$$

where the second inequality is due to the *change of measure inequality* (Seldin et al., 2012).

By the construction of $\mathfrak{P}(\pi, \mathcal{P})$, we have

$$\mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{(A, P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\pi, \mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} = \mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{A \sim \pi} \mathop{\mathbb{E}}_{P \sim \mathcal{P}(A)} \mathop{\mathbb{E}}_{f_1 \sim P} \cdots \mathop{\mathbb{E}}_{f_n \sim P} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)}, \tag{32}$$

and, because it is independent of $S_1, \ldots, S_n$, we can rewrite this as

$$= \mathop{\mathbb{E}}_{A \sim \pi} \mathop{\mathbb{E}}_{P \sim \mathcal{P}(A)} \mathop{\mathbb{E}}_{S_1} \mathop{\mathbb{E}}_{f_1 \sim P} e^{\frac{\lambda}{n} \Delta_1(f_1)} \cdots \mathop{\mathbb{E}}_{S_n} \mathop{\mathbb{E}}_{f_n \sim P} e^{\frac{\lambda}{n} \Delta_n(f_n)}. \tag{33}$$

$$= \mathop{\mathbb{E}}_{A \sim \pi} \mathop{\mathbb{E}}_{P \sim \mathcal{P}(A)} \prod_{i=1}^{n} \mathop{\mathbb{E}}_{S_i} \mathop{\mathbb{E}}_{f_i \sim P} e^{\frac{\lambda}{n} \Delta_i(f_i)} \tag{34}$$

Each $\Delta_i(f_i)$ is a bounded random variable with support in an interval of size 1. By Hoeffding's lemma we have

$$\mathop{\mathbb{E}}_{S_i} \mathop{\mathbb{E}}_{f_i \sim P} e^{\frac{\lambda}{n} \Delta_i(f_i)} \leq e^{\frac{\lambda^2}{8n^2 m}}. \tag{35}$$

Therefore, by combining (34) and (35) we have:

$$\mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{(A, P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\pi, \mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} \leq e^{\frac{\lambda^2}{8nm}}. \tag{36}$$

By Markov's inequality, for any $\epsilon > 0$ we have

$$\mathbb{P}_{S_1,\ldots,S_n}\Big(\mathop{\mathbb{E}}_{(A,P,f_1,f_2,\ldots,f_n)\sim\mathfrak{P}(\pi,\mathcal{P})}\prod_{i=1}^{n} e^{\frac{\lambda}{n}\Delta_i(f_i)} \geq e^{\epsilon}\Big) \leq e^{\frac{\lambda^2}{8nm}-\epsilon} \tag{37}$$

Hence by combining (31) and (37) we get for any ϵ:

$$\mathbb{P}_{S_1,\ldots,S_n}\Big(\exists\rho,\mathcal{Q}:\widetilde{\mathcal{R}}(\rho)-\widehat{\mathcal{R}}(\rho)-\frac{1}{\lambda}\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P}))\geq\frac{1}{\lambda}\epsilon\Big)\leq e^{\frac{\lambda^2}{8nm}-\epsilon}, \tag{38}$$

or, equivalently, it holds for any $\delta > 0$ with probability at least $1-\frac{\delta}{2}$:

$$\forall\rho,\mathcal{Q}:\widetilde{\mathcal{R}}(\rho)-\widehat{\mathcal{R}}(\rho)\leq\frac{1}{\lambda}\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P}))+\frac{1}{\lambda}\log(\frac{2}{\delta})+\frac{\lambda}{8nm}. \tag{39}$$

□

The following lemma splits the $\mathbf{KL}$ term of (39) into more interpretable quantities.

Lemma A.4. *For the posterior, $\mathfrak{Q}(\rho,\mathcal{Q})$, and prior, $\mathfrak{P}(\pi,\mathcal{P})$, defined above it holds:*

$$\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P}))=\mathbf{KL}(\rho\|\pi)+\mathop{\mathbb{E}}_{A\sim\rho}\Big[\mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A))+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P)\Big]. \tag{40}$$

Proof.

$$\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P}))=\mathop{\mathbb{E}}_{A\sim\rho}\left[\mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\mathop{\mathbb{E}}_{f_i\sim A(S_i)}\ln\frac{\rho(A)\mathcal{Q}(A)(P)\prod_{i=1}^{n}A(S_i)(f_i)}{\pi(A)\mathcal{P}(A)(P)\prod_{i=1}^{n}P(f_i)}\Big]\right] \tag{41}$$

$$=\mathop{\mathbb{E}}_{A\sim\rho}\Big[\ln\frac{\rho(A)}{\pi(A)}\Big]+\mathop{\mathbb{E}}_{A\sim\rho}\left[\mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\ln\frac{\mathcal{Q}(A)(P)}{\mathcal{P}(A)(P)}\Big]+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\sum_{i=1}^{n}\mathop{\mathbb{E}}_{f_i\sim A(S_i)}\ln\frac{A(S_i)(f_i)}{P(f_i)}\Big]\right] \tag{42}$$

$$=\mathbf{KL}(\rho\|\pi)+\mathop{\mathbb{E}}_{A\sim\rho}\left[\mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}(A))+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P)\right]. \tag{43}$$

□

Part III We now combine the above parts to prove Theorem 3.1.

Proof. To get tight guarantees, we need to choose the value of λ in Lemma A.3 an optimal data-dependent way, but the statement of the Lemma holds only for individual values of λ. Therefore, we first create an version of inequality (28) by instantiating it for each $\lambda\in\Lambda$ with $\Lambda=\{1,\ldots,4mn\}$, and then applying a union-bound. It follows that

$$\mathbb{P}_{S_1,\ldots,S_n}\Big(\forall\rho,\mathcal{Q},\lambda\in\Lambda:\widetilde{\mathcal{R}}(\rho)-\widehat{\mathcal{R}}(\rho)\leq\frac{1}{\lambda}\Big[\mathbf{KL}\left(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})\right)+\frac{1}{\lambda}\log(\frac{8mn}{\delta})\Big]+\frac{\lambda}{8nm}\Big)\geq 1-\frac{\delta}{2} \tag{44}$$

Note that for real-valued $\lambda > 1$, it holds that $\lfloor\lambda\rfloor\leq\lambda$ and $\frac{1}{\lfloor\lambda\rfloor}\leq\frac{1}{\lambda-1}$. Thereby, we can allow real-valued λ and obtain

$$\mathbb{P}_{S_1,\ldots,S_n}\Big(\forall\rho,\mathcal{Q},\lambda\in(1,4mn]:\widetilde{\mathcal{R}}(\rho)-\widehat{\mathcal{R}}(\rho)\leq\underbrace{\frac{1}{\lambda-1}\Big[\mathbf{KL}\left(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})\right)+\log(\frac{8mn}{\delta})\Big]+\frac{\lambda}{8mn}}_{=:\Gamma(\lambda)}\Big)\geq 1-\frac{\delta}{2} \tag{45}$$

For any choice of $\rho, \mathcal{Q}$, let $\lambda^* = \sqrt{8mn(\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})) + \log(\frac{8mn}{\delta})) + 1}$. If $\lambda^* > 4mn$, that implies

$$\sqrt{\frac{\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})) + \log(\frac{8mn}{\delta}) + 1}{2mn}} > 1. \tag{46}$$

Otherwise, $\lambda^* \in (1, 4mn]$, so inequality (45) holds, and we have

$$\Gamma(\lambda^*) < \sqrt{\frac{\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})) + \log(\frac{8mn}{\delta}) + 1}{2mn}}. \tag{47}$$

Therefore,

$$\mathbb{P}_{S_1,\ldots,S_n}\left(\forall \rho, \mathcal{Q} : \widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq \sqrt{\frac{\mathbf{KL}(\mathfrak{Q}(\rho,\mathcal{Q})||\mathfrak{P}(\pi,\mathcal{P})) + \log(\frac{8mn}{\delta}) + 1}{2mn}}\right) \geq 1 - \frac{\delta}{2}. \tag{48}$$

In combination with Lemma A.4, with probability at least $1 - \frac{\delta}{2}$ it holds for all $\rho \in \mathcal{M}(\mathcal{A}), \mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{F})$:

$$\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \mathbb{E}_{A\sim\rho}[C_1(A,\mathcal{Q},\mathcal{P})] + \log(\frac{8mn}{\delta}) + 1}{2mn}}. \tag{49}$$

where $C_1(A, \mathcal{Q}, \mathcal{P})$ is defined as in (23). Combining (49) and Lemma A.2 by a union bound concludes the proof. □

A.2. Proof of Theorem 3.2.

We now restate and prove Theorem 3.2.

Theorem A.5. *For any fixed meta-prior $\pi \in \mathcal{M}(\mathcal{A})$, any fixed hyper-prior $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$ and any $\delta > 0$, it holds with probability at least $1 - \delta$ over the sampling of the datasets, that for all meta-posterior distributions $\rho \in \mathcal{M}(\mathcal{A})$ over algorithms, and for all hyper-posterior functions $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$ we have*

$$\mathcal{R}(\rho) \leq \widehat{\mathcal{R}}(\rho) + \sqrt{\frac{\mathbf{KL}(\rho||\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}} + \mathop{\mathbb{E}}_{A\sim\rho} \sqrt{\frac{C_2(A,\mathcal{Q},\mathcal{P}) + \log(\frac{8mn}{\delta}) + 1}{2mn}}, \tag{50}$$

with

$$C_2(A,\mathcal{Q},\mathcal{P}) = \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)} \sum_{i=1}^{n} \mathbf{KL}(A(S_i)||P). \tag{51}$$

The three-step proof largely follows that of Theorem 3.1, except for some differences that emerge because the constant hyper-prior allows some arguments to hold (with high probability over the datasets) uniformly for all algorithms at the same time.

Part I We bound the different $\mathcal{R}(\rho)$ and $\widetilde{\mathcal{R}}(\rho)$ the same way as in Lemma A.2.

Part II We define the following two functions that produce distributions over $\mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$, i.e. they assign joint probabilities to tuples, $(P, f_1, ..., f_n)$ that contain a prior over models and n models.

- *Posterior $\mathfrak{Q}(A, \mathcal{Q}(A))$:* given as input an algorithm $A \in \mathcal{A}$ and a hyper-posterior mapping $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{F})$ as input, it outputs the distribution over $\mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample a prior $P \sim \mathcal{Q}(A)$, *ii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim A(S_i)$.
- *Prior $\mathfrak{P}(\mathcal{P})$:* given a hyper-prior $\mathcal{P} \in \mathcal{M}(\mathcal{F})$ as input, it outputs the distribution over $\mathcal{M}(\mathcal{F}) \times \mathcal{F}^{\otimes n}$ with the following generating process: *i)* sample a prior $P \sim \mathcal{P}(A)$, *ii)* for each task, $i = 1, \ldots, n$, sample a model $f_i \sim P$.

Lemma A.6. *For any fixed meta-prior $\pi \in \mathcal{M}(\mathcal{A})$, fixed hyper-prior $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{F}))$, any $\delta > 0$, and any $\lambda > 0$, it holds with probability $1 - \frac{\delta}{2}$ over the sampling of datasets from training tasks that for all algorithms $A \in \mathcal{A}$ and for all hyper-posterior functions $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$:*

$$\widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) \leq \frac{1}{\lambda} \mathbf{KL}\left(\mathfrak{Q}(A, \mathcal{Q}(A)) \| \mathfrak{P}(\mathcal{P})\right) + \frac{1}{\lambda} \log(\frac{2}{\delta}) + \frac{\lambda}{8nm}, \tag{52}$$

where

$$\widetilde{\mathcal{R}}(A) = \frac{1}{n} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{z \sim D_i} \ell(z, A(S_i)), \qquad \textit{and} \qquad \widehat{\mathcal{R}}(A) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|S_i|} \sum_{z \in S_i} \ell(z, A(S_i)). \tag{53}$$

Proof. First for any task i and any model f_i we define:

$$\Delta_i(f_i) = \mathop{\mathbb{E}}_{z \sim D_i} \ell(z, f_i) - \frac{1}{|S_i|} \sum_{z \in S_i} \ell(z, f_i) \tag{54}$$

By this definition and the definitions of $\widetilde{\mathcal{R}}$ and $\widehat{\mathcal{R}}$ we have:

$$\mathop{\mathbb{E}}_{(P, f_1, f_2, \ldots, f_n) \sim \mathfrak{Q}(A, \mathcal{Q}(A))} \left[\frac{1}{n} \sum_{i=1}^{n} \Delta_i(f_i)\right] = \widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) \tag{55}$$

Using this equation and the *change of measure inequality* (Seldin et al., 2012) between the two distributions $\mathfrak{Q}(A, \mathcal{Q}(A))$ and $\mathfrak{P}(\mathcal{P})$, for any $\lambda > 0$, any $A \in \mathcal{A}$ and any $\mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$, we have:

$$\widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) - \frac{1}{\lambda} \mathbf{KL}\left(\mathfrak{Q}(A, \mathcal{Q}(A)) || \mathfrak{P}(\mathcal{P})\right) \leq \frac{1}{\lambda} \log \mathop{\mathbb{E}}_{(P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} \tag{56}$$

Because $\mathfrak{P}(\mathcal{P})$ is independent of $S_1, ..., S_n$, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{(P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\pi, \mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} &= \mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{P \sim \mathcal{P}} \mathop{\mathbb{E}}_{f_1 \sim P} \cdots \mathop{\mathbb{E}}_{f_n \sim P} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} && (57) \\
&= \mathop{\mathbb{E}}_{P \sim \mathcal{P}} \mathop{\mathbb{E}}_{S_1} \mathop{\mathbb{E}}_{f_1 \sim P} e^{\frac{\lambda}{n} \Delta_1(f_1)} \cdots \mathop{\mathbb{E}}_{S_n} \mathop{\mathbb{E}}_{f_n \sim P} e^{\frac{\lambda}{n} \Delta_n(f_n)} && (58) \\
&= \mathop{\mathbb{E}}_{P \sim \mathcal{P}} \prod_{i=1}^{n} \mathop{\mathbb{E}}_{S_i} \mathop{\mathbb{E}}_{f_i \sim P} e^{\frac{\lambda}{n} \Delta_i(f_i)} && (59)
\end{aligned}$$

Each $\Delta_i(f_i)$ is a bounded random variable with support in an interval of size 1. By Hoeffding's lemma we have

$$\mathop{\mathbb{E}}_{S_i} \mathop{\mathbb{E}}_{f_i \sim P} e^{\frac{\lambda}{n} \Delta_i(f_i)} \leq e^{\frac{\lambda^2}{8n^2 m}}. \tag{60}$$

Therefore by combining (59) and (60) we have:

$$\mathop{\mathbb{E}}_{S_1, \ldots, S_n} \mathop{\mathbb{E}}_{(P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} \leq e^{\frac{\lambda^2}{8nm}}. \tag{61}$$

By Markov's inequality, for any $\epsilon > 0$ we have

$$\mathbb{P}_{S_1, \ldots, S_n}\left(\mathop{\mathbb{E}}_{(P, f_1, f_2, \ldots, f_n) \sim \mathfrak{P}(\mathcal{P})} \prod_{i=1}^{n} e^{\frac{\lambda}{n} \Delta_i(f_i)} \geq e^{\epsilon} \right) \leq e^{\frac{\lambda^2}{8nm} - \epsilon} \tag{62}$$

Hence by combining (56) and (62) we get

$$\mathbb{P}_{S_1,\dots,S_n}\Big(\exists A, \mathcal{Q} : \widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) - \frac{1}{\lambda}\,\mathbf{KL}(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})) \geq \frac{1}{\lambda}\epsilon\Big) \leq e^{\frac{\lambda^2}{8nm}-\epsilon}, \tag{63}$$

or, equivalently, it holds for any $\delta > 0$ with probability at least $1 - \frac{\delta}{2}$:

$$\forall A, \mathcal{Q} : \quad \widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) \leq \frac{1}{\lambda}\,\mathbf{KL}(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})) + \frac{1}{\lambda}\log(\frac{2}{\delta}) + \frac{\lambda}{8nm} \tag{64}$$

□

The following lemma splits the $\mathbf{KL}$ term of (64) into more interpretable quantities.

Lemma A.7. *For the posterior, $\mathfrak{Q}(A,\mathcal{Q}(A))$, and prior, $\mathfrak{P}(\mathcal{P})$, defined above it holds:*

$$\mathbf{KL}\left(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})\right) = \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P) \tag{65}$$

Proof.

$$\begin{aligned}
\mathbf{KL}(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})) &= \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\mathop{\mathbb{E}}_{f_i\sim A(S_i)}\ln\frac{\mathcal{Q}(A)(P)\prod_{i=1}^n A(S_i)(f_i)}{\mathcal{P}(P)\prod_{i=1}^n P(f_i)}\Big] && (66)\\
&= \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\ln\frac{\mathcal{Q}(A)(P)}{\mathcal{P}(P)}\Big] + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\Big[\sum_{i=1}^{n}\mathop{\mathbb{E}}_{f_i\sim A(S_i)}\ln\frac{A(S_i)(f_i)}{P(f_i)}\Big]\Big] && (67)\\
&= \mathbf{KL}(\mathcal{Q}(A)\|\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}(A)}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P) && (68)
\end{aligned}$$

□

Part III We now finish the proof of Theorem 3.2 by combining the above results.

Proof. By applying a union bound for all the values of $\lambda \in \Lambda$ with $\Lambda = \{1,\dots,4mn\}$ we obtain that

$$\mathbb{P}_{S_1,\dots,S_n}\Big(\forall A, \mathcal{Q} : \widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A) \leq \sqrt{\frac{\mathbf{KL}\left(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})\right) + \log(\frac{8mn}{\delta}) + 1}{2mn}}\Big) \geq 1 - \frac{\delta}{2}. \tag{69}$$

Because this inequality holds (with high probability over the datasets) for all algorithms at the same time, it also holds in expectation over algorithms with respect to any distribution. Therefore we have:

$$\mathbb{P}_{S_1,\dots,S_n}\Big(\forall \rho, \mathcal{Q} : \mathop{\mathbb{E}}_{A\sim\rho}[\widetilde{\mathcal{R}}(A) - \widehat{\mathcal{R}}(A)] \leq \mathop{\mathbb{E}}_{A\sim\rho}\sqrt{\frac{\mathbf{KL}(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})) + \log(\frac{8mn}{\delta}) + 1}{2mn}}\Big) \geq 1 - \frac{\delta}{2}, \tag{70}$$

or, equivalently,

$$\mathbb{P}_{S_1,\dots,S_n}\Big(\forall \rho, \mathcal{Q} : \widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq \mathop{\mathbb{E}}_{A\sim\rho}\sqrt{\frac{\mathbf{KL}(\mathfrak{Q}(A,\mathcal{Q}(A))||\mathfrak{P}(\mathcal{P})) + \log(\frac{8mn}{\delta}) + 1}{2mn}}\Big) \geq 1 - \frac{\delta}{2}. \tag{71}$$

In combination with Lemma A.7, with probability at least $1 - \frac{\delta}{2}$ we have for all $\rho \in \mathcal{M}(\mathcal{A}), \mathcal{Q} : \mathcal{A} \to \mathcal{M}(\mathcal{M}(\mathcal{F}))$:

$$\widetilde{\mathcal{R}}(\rho) - \widehat{\mathcal{R}}(\rho) \leq \mathop{\mathbb{E}}_{A\sim\rho}\sqrt{\frac{C_2(A,\mathcal{Q},\mathcal{P}) + \log(\frac{8mn}{\delta}) + 1}{2mn}} \tag{72}$$

where $C_2(A,\mathcal{Q},\mathcal{P})$ is defined as in (51). Combining (72) and Lemma A.2 concludes the proof. □

B. Experimental Details

In this section, we provide the details of our experiments.

B.1. Setup

We follow the experimental setup proposed in Amit & Meir (2018) for benchmarking meta-learning methods. The experiment consists of two types of tasks based on the MNIST dataset (LeCun & Cortes, 1998). In the first one, each task is the MNIST classification task with a task-specific permutation in the labels. In the second experiment, each task has the MNIST images as samples, but a task-specific shuffle is applied to a subset of 200 pixels of the input. In both cases, there are 10 training tasks and 20 test tasks. We use 600 samples per training task and 100 samples per test task. This choice corresponds to a setup in which independent learning cannot be expected to provide good performance (the number of samples per task is small), and meta-learning is necessary.

B.2. Meta-learning Algorithm

As discussed in Section 2, the meta-learning mechanism introduced in Amit & Meir (2018) is to learn a hyper-posterior over priors in the training phase. For a future task, they minimize a PAC-Bayes bound based on this prior. As mentioned in Section 5 they train a neural network and use the same prior as the initialization point. This setup does not allow learning an initialization different from the prior used for regularization.

To show the benefits of the additional freedom provided by our framework, we use the same procedure except that we learn a separate initialization for the network which can differ from the prior used in the objective, as discussed in Sections 3.1 and 5. In the stochastic setting, we learn a meta-posterior ρ over the initialization prior and the regularization prior in the training phase. For future tasks, we sample the two distributions (P_0, P_1) from ρ, initialize our stochastic neural network by P_0, and optimize a PAC-Bayes bound with prior P_1. Note that the meta learner is free to make use of the added flexibility by learning P_0 different from P_1, or to recover the previous setup by learning P_0 identical to P_1. We do not have to fear overfitting from the larger set of parameters, because the objective is based on a generalization bound that enforces appropriate regularization.

We use Gaussians for all distributions, which allows us to compute the complexity terms in closed form. More precisely, let d be the number of weights in our neural network and we represent the prior and posteriors by their mean μ_i and the log variance value $\log \sigma_i$ for the weight w_i. Formally, we represent each ρ as $\rho_0 \times \rho_1$, which ρ_i is a distribution over P_i, and has the form of $\mathcal{N}(\theta_i, \kappa_\rho^2 I_{2d\times 2d})$. We use a fixed parameter for κ_ρ, and we learn the means θ_i. The meta-priors also have the same form, i.e. $\pi = \pi_0 \times \pi_1$ and $\pi_i = \mathcal{N}(0, \kappa_\pi^2 I_{2d\times 2d})$, with fixed κ_π.

To use our generalization bounds for this mechanism, we apply Theorem 3.1 (with $\delta = 0.1$) and we set we $\mathcal{Q}(P_0, P_1) = \mathcal{P}(P_0, P_1) = \delta_{P_1}$. The result is a bound

$$\mathcal{R}(\rho) \leq \widehat{\mathcal{R}}(\rho) + \sqrt{\frac{\mathbf{KL}(\rho\|\pi) + \log(\frac{4\sqrt{n}}{\delta})}{2n}} + \sqrt{\frac{\mathbf{KL}(\rho\|\pi) + \mathbb{E}_{(P_0,P_1)\sim\rho}\sum_{i=1}^{n}\mathbf{KL}(A(S_i)||P_1)] + \log(\frac{8mn}{\delta}) + 1}{2mn}} \tag{73}$$

in which $\mathbf{KL}(\rho\|\pi)$ has the following form:

$$\mathbf{KL}(\rho\|\pi) = \frac{4d\kappa_\rho + \|\theta_0\|^2 + \|\theta_1\|^2}{2\kappa_\pi} - 2d + 4d\log(\frac{\kappa_\pi}{\kappa_\rho}) \tag{74}$$

Training phase. In the training phase, we optimize the right-hand side of (73) to find the meta-posterior ρ. As in Amit & Meir (2018) we use the Monte Carlo method to approximate the values for calculating the expectation terms, and use the re-parametrization trick (Kingma et al., 2015) to optimize the expected value of the $\mathbf{KL}(A(S_i)||P_1)$ terms.

To find ρ we follow the optimization procedure defined in Amit & Meir (2018). For each task i, we assign a stochastic neural network Q_i, initiated in the following way: The mean of each weight is initiated randomly with the Glorot method (Glorot & Bengio, 2010), and the log-variance of each weight is initiated randomly from $\mathcal{N}(-10, 0.1^2)$.

In their original optimization procedure Amit & Meir (2018) optimize their objective for 200 epochs to find the hyper-posterior. Because our meta-distribution has two parts: one for initialization and one for regularization, we add the following change to this procedure: In the first 100 epochs, we assume ρ_0 and ρ_1 are equal and we minimize the bound to find $\rho_0 = \rho_1$

and posteriors Q_is. After 100 epochs, we fix ρ_0, initialize the Q_is by sampling from ρ_0 (Since ρ_0 is supposed to be the meta-distribution over initialization prior) and optimize the bound for ρ_1 and the Q_is for another 100 epochs.

Future tasks. For a future task T_{new} with its training dataset S_{new} we learn its posterior as follows: we sample the initialization prior from ρ_0 to initiate a stochastic neural network Q_{new}. Then, similar to previous works, we optimize the following PAC-Bayesian bound with the prior P_1 sampled from ρ_1 for 100 epochs.

$$\widehat{\mathcal{R}}(Q_{\text{new}}) + \sqrt{\frac{\mathbf{KL}(Q_{\text{new}}||P_1) + \log(\frac{8m}{\delta}) + 1}{2m}} \tag{75}$$

We use Monte Carlo sampling to approximate the expectations.

B.3. Implementation and Numeric Results

Our implementation is based on the code of Amit & Meir (2018), except that we fixed a bug in their computation of the $\mathbf{KL}$-divergences, which was also present in later works derived from it. Furthermore, we corrected an issue with how the gradients of the objective in Rezazadeh (2022) were computed. All experiments were done with the corrected implementation.[1]

We use the same network architectures as proposed Amit & Meir (2018): a small ConvNet with two convolutional layers and one fully-connected layer for the *permuted labels* task, and a three-layer fully-connected network for the *shuffled pixels* task. For further details on the architecture, see the original reference. We used the Adam optimizer with a learning rate of 10^{-3} and the number of Monte Carlo iterations was 1 in all experiments. For the fixed parameters of the minimization objective, we put $\delta = 0.1$, and for the variances of meta-prior π and meta-posterior ρ, we set $\kappa_\pi = 10^2$ and $\kappa_\rho = 10^{-3}$. Moreover, the batch size is 128 and the used loss function is Cross-Entropy loss.

The experimental results are shown in the Table 2. We compare our results with the following prior-based works: The (MLAP-M) bound of Amit & Meir (2018), The Classic bound of Rezazadeh (2022), and the kl-bound of Guan & Lu (2022). The results confirm that the extra flexibility of our framework can be beneficial. Specifically, in the *permuted labels* experiment, having a different initialization and regularization helps a lot. In the *shuffled pixel* setting, the flexibility does not help and we get the same performance as the previous methods. A noteworthy feature of Table 2 is the high error of Rezazadeh (2022) and Guan & Lu (2022) in the permuted labels task. These are a consequence of the fact that the complexity terms in their bounds are very big in low-data regime that we are interested in (where meta-learning is meant to help). As a result, the optimization mainly attempt as reducing the complexity terms, which leads to underfitting and classification performance as good as a random guess ($\approx 90\%$ error). The same problem occurs for the Catoni-type bound in Guan & Lu (2022) for both experimental settings, so we do not report its results.

Comparison of our bounds with and without different initialization and regularization: We present an ablation study in Table 3. To see if the added flexibility of our setting is indeed responsible for the improved results rather than the different objective compared to prior work, we compare our results with the case that the two distributions are equal when we use the distribution learned for the regularization for the initialization as well. As one can see, using different distributions for initialization and regularization reduce the error in the *permuted labels* task, but for *shuffled pixels* they stay the same.

Bound	Shuffled Pixel	Permuted Label
initialization identical to prior ($\rho_0 = \rho_1$)	10.1 ± 1.3	15.5 ± 4.3
initialization can differ from prior	9.9 ± 1.1	7.9 ± 1.7

Table 3. The ablation study to confirm that the gained performance is due to separate initialization and regularization. We compare our method with the case if we use the distributions in the end of our training procedure both as the initialization and regularization. As one can see, having different distributions leads to better performance in one of the tasks.

[1] https://github.com/hzakerinia/Flexible-PAC-Bayes-Meta-Learning/