Statistical Generalization Performance Guarantee for Meta-learning with Data Dependent Prior

Tianyu Liu, Jie Lu*, Zheng Yan, Guangquan Zhang

[a] *University of Technology Sydney, Broadway, Sydney, 2007, NSW, Australia*

Abstract

Meta-learning aims to leverage experience from previous tasks to achieve an effective and fast adaptation ability when encountering new tasks. However, it is unclear how the generalization property applies to new tasks. Probably approximately correct (PAC) Bayes bound theory provides a theoretical framework to analyze the generalization performance for meta-learning with an explicit numerical generalization error upper bound. A tighter upper bound may achieve better generalization performance. However, for the PAC-Bayes meta-learning bound, the prior distribution is selected randomly which results in poor generalization performance.

In this paper, we derive three novel generalization error upper bounds for meta-learning based on the PAC-Bayes relative entropy bound. Furthermore, in order to avoid randomly prior distribution, based on the empirical risk minimization (ERM) method, a data-dependent prior for the PAC-Bayes meta-learning bound algorithm is developed and the sample complexity and computational complexity are analyzed. The experiments illustrate that the proposed three PAC-Bayes bounds for meta-learning achieve a competitive generalization guarantee, and the extended PAC-Bayes bound with a data-dependent prior can achieve rapid convergence ability.

Keywords: Meta-learning, Bayesian network, statistical learning, PAC-Bayes bound

*Corresponding author

Email addresses: Tianyu.Liu-1@student.uts.edu.au (Tianyu Liu), jie.lu@uts.edu.au (Jie Lu), Yan.zheng@uts.edu.au (Zheng Yan), guangquan.zhangu@uts.edu.au (Guangquan Zhang)

1. Introduction

Machine learning models often require training with a large number of samples, for example, the image classification issue [1, 2, 3, 4]. Traditional machine learning algorithms mainly focus on a single task. But it is generally difficult to collect so much labelled data. So how to train a model when only a small amount of data is available? More to the point, since humans can learn new skills much faster and more effectively, how can we build such a model, which can reflect aspects of human learning? That is what meta-learning sets out to achieve. *Meta-learning* – or "learning to learn [5]" – is capable of accurately adapting or generalizing to new tasks and new environments that not encountered during training time. Using the experience acquired on previous tasks, meta-learning can adapt to new tasks quickly, even in the face of scant data.

Although meta-learning algorithms provide a powerful inductive bias based on various tasks, even with those which comprise only limited data, its generalization performance is poorly understood. How to evaluate the performance of a learnt meta-learning model when faced with new tasks is also an important issue. PAC-Bayes theory, known as generalization error bounds theory, provides a theoretical analysis framework for estimating the generalization performance of the machine learning model.

With high probability, PAC-Bayes bound provides the numerical generalization error upper bound for a learnt model. Different from the traditional neural networks which directly optimizes the empirical risk function, PAC-Bayes bound optimizes the neural network by minimizing both the empirical risk function and a regularization item, which is in proportion to the divergence between the prior distribution and posterior distribution of parameters. Therefore, this theory is more resistant to over-fitting. Compared to other generalization theory, PAC-Bayes bound can be directly used to train a neural network by selecting the generalization error upper bound as the training objective instead of the empirical loss. As shown in Figure. 1, based on the PAC-Bayes theory, the empirical test error is lower than the generalization error upper bound statistically.

PAC-Bayes theory is also extended to provide a theoretical framework for the generalization performance analysis of meta-learning, which yields a numerical generalization error upper bound for meta-learning that holds with arbitrarily high probability. Specifically, for meta-learning by minimizing an objective function derived from extended PAC-Bayes bounds, a gradient-

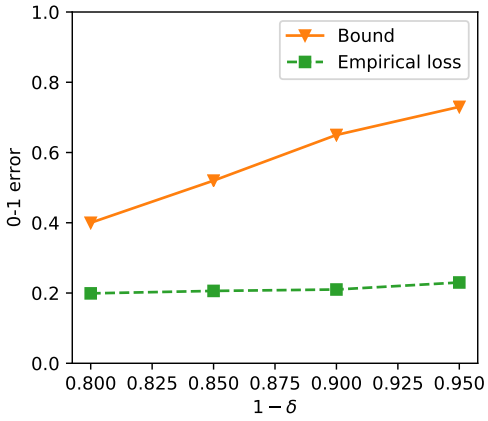


Figure 1: With a high probability $1-\delta$, the empirical test error is lower than the PAC-Bayes generalization error upper bound. The solid line indicates the generalization error upper bound, and the dashed line represents the empirical test loss. As shown in this figure, the higher the confidence probability, the looser the error upper bound.

based meta-learning algorithm [6] is developed to achieve better generalization performance and avoid overfitting.

For the PAC-Bayes theory, tighter bounds achieve a better generalization performance guarantee, therefore it is still an important issue to develop tighter generalization error upper bound theory. Generally, in PAC-Bayes theory, generalization error upper bound is primarily determined by the divergence between prior and posterior distributions. Obviously, the choice of prior distribution affects the performance of the PAC-Bayes bound significantly. However, in PAC-Bayes meta-learning theory, prior distribution is selected randomly before learning, which leads to a looser generalization error upper bound.

Motivated by the previous discussions, three novel generalization error bounds for meta-learning are presented. Furthermore, a data-based approach to adjust prior distribution is developed, and the specific implementations of these algorithms are given. The sample complexity and computational complexity of the proposed algorithm are also analyzed. The main contributions are concluded as follows:

- Tighter generalization error upper bound achieves better generalization performance. To improve generalization performance, three novel PAC-Bayes meta-learning bounds are proposed, including *meta-learning*

PAC-Bayes λ bound, *meta-learning PAC-Bayes quadratic bound* and *meta-learning PAC-Bayes variational bound*;

- In the original meta-learning PAC-Bayes theory, the prior distributions are selected randomly, which leads to a loose generalization error bound. Therefore, based on the ERM method, a PAC-Bayes bound for meta-learning with a data-dependent prior approach is developed; Instead of the random prior weight distribution, a data dependent prior attains fast convergence ability;

- The computational complexity of the proposed algorithms is also analyzed, and an empirical demonstration illustrates that the proposed algorithms achieve a competitive generalization guarantee and better convergence performance.

The rest of this paper is organized as follows. The literature review, which is related to meta-learning and PAC-Bayes bound, is investigated in Section 2. Then, the classical PAC-Bayes bounds for both single task and meta-learning are introduced in Section 3. Section 4 investigates three novel PAC-Bayes bounds for meta-learning, based on the PAC-Bayes relative entropy bound theory. A PAC-Bayes bound for meta-learning with a data-dependent prior is further developed in Section 5. The implementation details are described in Section 6. Section 7 provides numerical examples to verify the proposed algorithms. Finally, Section 8 draws some conclusions.

2. Related work

The literature review related to the generalization performance analysis of meta-learning approaches is addressed in this section. These can be divided into three categories: meta-learning algorithms, PAC-Bayes theory and its applications in meta-learning.

2.1. Meta-learning algorithms

Meta-learning algorithms can be divided into three major categories [7]: *black-box algorithms*, *non-parametric methods* and *optimization-based algorithms*.

Black-box adaptation meta-learning is to train a neural network to represent a meta-learner. With the aim of achieving a fast adaptation ability, [8] a meta-learning algorithm is presented with memory-augmented neural networks,

which can summarize and store important knowledge. When facing new learning tasks, the memory-based method can extract certain skills it has learned to assist in the current process. To access past experiences, a simple neural attentive-learner (SNAIL) is proposed in [9]. By using the attention architectures established in the meta-learner, SNAIL can determine which pieces of information it needs to select from the experience it has gathered. SNAIL architectures are easier to train than traditional RNNs, such as LSTM.

Non-parametric methods try to utilize a non-parametric learner as a meta-learner instead of parametric models. Non-parametric methods are simple and perform well in few-shot learning. [10] proposes a Siamese neural network, which contains two sub-networks with the same weights. During the training phase, the two sub-networks can extract features from two different input vectors, and then compute the distance between the two feature vectors. Matching networks are another non-parametric method, which is presented in [11]. In order to learn from a few examples, the matching network framework learns a net structure that maps a few labelled training datasets and an unlabelled instance to its label. Combined with recent advances in attention and memory, the matching networks enable rapid learning. Furthermore, [12] proposes a prototypical network, where the classification problem is regarded as finding the prototype center of each category in the semantic space and then predicting the category of the new sample using the nearest neighbor classifier. This method mainly combines the prototype network with the clustering algorithm.

Optimization-based meta-learning algorithms, different from the two aforementioned algorithms, learn to train the parameter vector to represent the meta-learner through optimization. In the traditional gradient descent approach, optimization update rules, for example the learning step, are still hard to design. [13] considers this issue as a learning problem, allowing the optimization algorithms to learn to exploit the update rules structure in an automatic way. Furthermore, [14] proposes another LSTM-based meta-learner model by combining gradient descent and the LSTM algorithm, which is applied to train neural networks. To extract common knowledge from previous tasks so as to achieve a fast adaptation ability, [15] proposes a model agnostic meta-learning (MAML) algorithm. The key idea of MAML is to learn a set of initialization parameters that allows the efficient learning of new tasks. However, MAML requires the computation of a second-order derivative which may exhibit instabilities. Therefore, [16] presents a scalable meta-learning algorithm, called Reptile, which does not calculate any second derivatives.

Furthermore, [17] addresses the training of MAML and proposes several tricks to improve the stability of MAML.

One of the majority challenges in few-shot learning is task ambiguity. [18] proposes a probabilistic MAML, which tries to incorporate a parameter distribution with a neural network that is trained via a variational lower bound. To improve the robustness of MAML, [19] propose a Bayesian MAML algorithm. Compared with a point estimate or a simple Gaussian approximation in the fast adaptation phase, this algorithm is capable of learning a very complex uncertainty structure. The Bayesian MAML outperforms vanilla MAML in terms of accuracy and robustness. Furthermore, based on the Bayesian inference framework and variational inference, [20] proposes a new Bayesian task-adaptive meta-learning (Bayesian TAML) algorithm for imbalanced and out-of-distribution tasks. In addition, several improved MAMLs are also introduced, such as Alpha MAML [21], meta-learning with latent embedding optimization [22] and Bayesian hierarchical modeling-based MAML [23].

2.2. PAC-Bayes theory

The first PAC-Bayes theory was established by McAllester [24], which provides generalization error upper bounds for the performance of randomized learning algorithms. Then, this method was subsequently used to analyze the generalization-error bound of the stochastic neural network [25]. PAC-Bayes bound theories were meant for a wide range of approximate Bayesian GP classification issues [26], [27]. Then work in [28] tries to explain the generalization in neural networks from the view of norm-based control, sharpness and robustness, and attempts to build a connection between sharpness and PAC-Bayes theory. The systemically undertaken study is addressed in [29] to train stochastic neural networks based on the PAC-Bayes bounds. Furthermore, PAC-Bayes bound also is applied in twin support vector machines at [30] and domain adaptation at [31].

The PAC-Bayes theory is only suitable for bounded loss function and i.i.d data. PAC-Bayesian bounds tailored for the sub-Gaussian or sub-Gamma loss family, such as negative log-likelihood function, is also developed by [32] and [33]. However, these algorithms require a distribution parameter, such as a variance factor and a scale parameter. Therefore [34] proposes an exponential bound under the assumption that the first three moments of the loss distribution are bounded. By introducing the special boundedness condition, [35] expands the PAC-Bayesian theory to learning problems with unbounded loss functions.

Recently, there has been a gradually increasing interest in research on overparameterized deep neural networks and SGD. [36] studies the generalization of randomized learning algorithms. trained with SGD. Inspired by [25], [37] obtains nonvacuous generalization numerical bounds for deep stochastic neural network classifiers with many more parameters than are present in the training data. The first non-vacuous generalization bound for compressed networks applied to the ImageNet classification problem is provided in [38]. Moreover, [39] further investigates the relationship between generalization performance and SGD.

As previously mentioned, the PAC-Bayesian bound is only valid for stochastic classifiers, although a growing body of literature attempts to construct PAC-Bayes bounds on deterministic classifiers. To address this gap, [40] develops a PAC-Bayesian transportation bound, by unifying the PAC-Bayesian analysis and the chaining method. This generalization error bound relates to the distance between any two predictors, both for stochastic classifiers and deterministic classifiers. A new perturbation bounds for feedforward neural networks is derived based on the sharpness of a model class in [41]. In addition, [42] presents a general PAC-Bayesian framework for the deterministic and uncompressed neural networks by leveraging the noise-resilience of deep neural networks on training data.

To achieve tighter generalization error bounds, [43] proposes two alternative prior distributions: one is to learn a prior distribution from a separate training data set which is not used to compute the bound, and another is to consider an expectation prior. [44] further investigates the PAC-Bayes bound with localized prior distribution defined in terms of the data generating distribution. Under the stability of the hypothesis, a Gaussian prior distribution, informed by the data-generating distribution and gathered at the expected output, is proposed for the SVM classifier [45]. More discussion can be seen in [46] and [47]. Furthermore, because data distribution is usually unknown, [48] develops a PAC-Bayes bound via ϵ-differentially private data-dependent prior.

2.3. PAC-Bayes bound for meta-learning

[49] provides a generalization error bound within the PAC-Bayes framework for lifelong learning. Furthermore, two principled algorithms are implemented, namely parameter and representation transfers. More recently, [6] developed a theoretical framework for meta-learning, allowing extended various PAC-Bayes bounds to meta-learning. To add to this, [50] considers the scenario

in which a common model set is used for model averaging via a model selection procedure that accounts for the model's uncertainty. Two data-based algorithms are proposed to obtain ideal priors for model averaging. Furthermore based on the algorithm stability and mutual information theory, the corresponding generalization error bounds are also proposed in [51] and [52]. These generalization theories are all follow the same task distribution assumption, [53] develops another two another PAC-Bayesian bound which can relax this assumption.

3. Preliminaries: PAC-Bayes theorem

In this section, the rigorous definitions for the standard PAC-Bayes bound and the PAC-Bayes meta-learning bound are introduced, which are given in Lemma 1 and Lemma 2, respectively.

3.1. Risk functions

Before introducing the PAC-Bayes theory, the following concepts and notations are introduced.

3.1.1. Single classifier case

In the classical supervised learning model setting, a set of dependent samples $S = \{z_i\}_{i=1}^m$ is randomly drawn from the unknown data distribution $\mathcal{D}$, each sample $Z_i = (X_i, Y_i)$ consisting of an input X_i and its corresponding label Y_i, where $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$. The learning objective is to find a classifier $h \in \mathcal{H}$ that predicts the label by minimizing the loss function $\{\ell(h, z)\}$, where $\mathcal{H}$ is considered as the hypothesis space and $\ell(h, z) : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}$ is the loss function which are used to measure the performance of classifier h. For the classification problems, the loss function is always bounded in $[0, 1]$. Furthermore, the *expected error* of the classifier h in the data distribution $\mathcal{D}$ can be written as $er(h, \mathcal{D})$

$$er(h, \mathcal{D}) = \mathbb{E}_{z\sim\mathcal{D}}\ell(h, z). \tag{1}$$

Since the real data distribution $\mathcal{D}$ is unknown, *generalization error* $er(h, \mathcal{D})$ cannot be calculated. Therefore, the *empirical error* $\widehat{er}(h, S)$ gives an observable estimation

$$\widehat{er}(h, S) = \frac{1}{n}\sum_{i=1}^n \ell(h, z_i). \tag{2}$$

Under the general neural network conditions, in order to minimize the empirical risk $\widehat{er}(h, S)$, a single classifier $h \in \mathcal{H}$ is selected. However, this may cause that the learned classifier $\widehat{h}$ over-fit the training dataset S . Various methods can be used to avoid over-fitting, including adding a complexity regularization.

3.1.2. Classifier distribution case

Similar to the Bayesian neural network, under the PAC-Bayes framework, learning algorithms will output a classifier distribution Q over the hypothesis space $\mathcal{H}$, instead of a single classifier h. The prior classifier can also be initialized as a prior distribution P. During the classifying phase, one can sample a classifier h from the classifier distribution Q.

Then *generalization error* $er(Q, \mathcal{D})$ over posterior distribution Q and unknown data distribution $\mathcal{D}$ is defined as

$$er(Q, \mathcal{D}) \triangleq \mathop{\mathbb{E}}_{h \sim Q} er(h, \mathcal{D}). \tag{3}$$

This *generalization error* $er(Q, \mathcal{D})$ is also cannot be calculated, therefore, *empirical error* $\widehat{er}(Q, S)$ is given as

$$\widehat{er}(Q, S) \triangleq \mathop{\mathbb{E}}_{h \sim Q} er(h, S). \tag{4}$$

3.2. PAC-Bayes bounds for a single task

PAC-Bayes theory provides a framework for the theoretical generalization performance analysis of a learnt model, which means the upper bound of the generalization error $er(Q, \mathcal{D})$. Based on the prior distribution P, PAC-Bayes theory tries to learn a posterior distribution $Q(S, P)$ from training data S.

Lemma 1 *(single-task PAC-Bayes bound [25]). Let $P \in \mathcal{M}$ be some prior distribution over $\mathcal{H}$. Then for any $\delta \in (0, 1]$, the following inequality holds uniformly for all posterior distributions $Q \in \mathcal{M}$ with a probability of at least $1 - \delta$*

$$\mathrm{kl}(\mathrm{er}(\mathrm{Q}, \mathcal{D}) \| \widehat{\mathrm{er}}(\mathrm{Q}, \mathrm{S})) \leq \frac{\mathrm{D}(\mathrm{Q} \| \mathrm{P}) + \log(\frac{2\sqrt{\mathrm{n}}}{\delta})}{\mathrm{n}}. \tag{5}$$

Here, $D(Q\|P)$ is a divergence measure to measure the difference between two distributions, kl is known as the binary KL divergence, which is the divergence of two Bernoulli distributions with parameters $q, q' \in [0, 1]$

$$\mathrm{kl}\left(q \| q'\right) = q \log\left(\tfrac{q}{q'}\right) + (1 - q) \log\left(\tfrac{1-q}{1-q'}\right). \tag{6}$$

As previously discussed, the generalization error $er(Q, \mathcal{D})$ cannot be calculated because of the unknown data distribution $\mathcal{D}$. With an arbitrarily high probability, PAC-Bayes theory shows that the generalization error can be bounded by the summation of empirical loss, and a regularization element involves the distance between prior distribution and posterior distributions. It also means that the empirical test error of the test samples sampled from the same data distribution $\mathcal{D}$ is lower than this error upper bound with high probability.

Generally, a PAC-Bayes generalization theory attempts to balance the empirical risk $\widehat{er}(Q, S)$ and a regularization term and the discrepancy between the prior distribution P and posterior distribution Q. Here, we should emphasize that prior distribution P is selected randomly before training, which must not be dependent on the training data. The prior distribution P is used chiefly to measure the distance of hypothesis space $\mathcal{H}$. Obviously, the choice of prior distribution P significantly affects the performance of the PAC-Bayes bound significantly.

Remark 1 *Compared with classical neural network algorithms which are trained by minimizing the empirical loss function $\widehat{er}(\cdot)$, in the PAC-Bayes theory, one can directly select the generalization error upper bound (the right side of this bound) as the training objective to achieve generalization performance guarantee. Furthermore, in the PAC-Bayes theory, the training objective includes empirical loss and the regularization item, which can avoid over-fitting.*

Remark 2 *From PAC-Bayes theory, if we use finite samples S to train the model using the PAC-Bayes rule, then the generalization error upper bound of the learnt model is decided by the empirical loss $\widehat{er}(Q, S)$ and the regularization term. This means that in the testing phase the test error is lower than this error upper bound with high probability.*

3.3. PAC-Bayes bounds for meta-learning

In this subsection, the PAC-Bayes bounds for meta-learning is introduced. Similar to the PAC-Bayes bound for a single task, the PAC-Bayes meta-learning bound also provides the generalization error upper bound for the meta-learning algorithm, as shown in Lemma 2.

In the meta-learning framework, following the same setting in [15, 6, 54], $\mathcal{T}$ represents the task distribution, loss function is defined as ℓ. In the *meta level*,

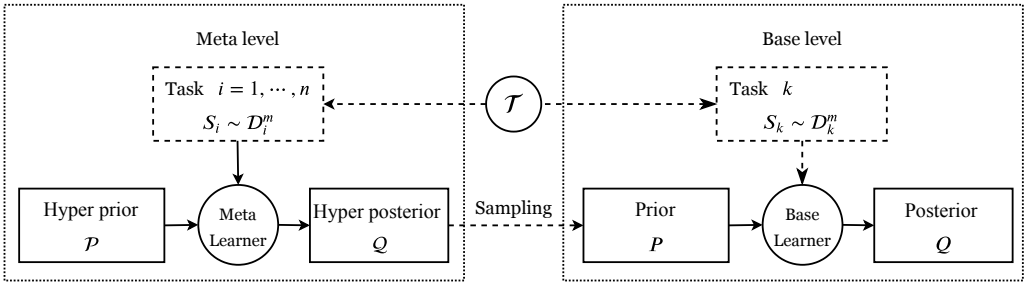


Figure 2: Simplified framework for PAC-Bayes meta-learning bound. Based on the *hyper prior* $\mathcal{P}$, *meta-learner* aims to learn a *hyper posterior* $\mathcal{Q}(P)$ by utilizing the observed tasks. When encountering new tasks, *base learner* samples a prior distribution P from the hyper-posterior $\mathcal{Q}(P)$. Then capitalizing on observed samples of the new task, *base learner* infers a posterior distribution $Q(S, P)$.

each observed task τ_i sampled from the task distribution $\mathcal{T}$, the corresponding sample set S_i are generated from the unknown data distribution $S_i \sim \mathcal{D}_i^{m_i}$, where m_i is the number of training samples for task i.

As previously mentioned, in the PAC-Bayes meta-learning theory the classifier is set as a distribution. In the *meta level*, based on the predefined *hyper prior* $\mathcal{P}$, the *meta-learner* aims to extract common knowledge from the observed tasks and learn a *hyper posterior* $\mathcal{Q}$. When encountering a new task, one can sample a prior distribution P from the hyper-posterior $\mathcal{Q}$ as the initialization, then based on this prior information P and the new task's training data S, the *base learner* learns the posterior information Q.

The performance of hyper-posterior $\mathcal{Q}$ can be measured by the expectation loss of prior P when learning new tasks, the so-called *generalization error*

$$er(\mathcal{Q}, \tau) \triangleq \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \mathop{\mathbb{E}}_{(\mathcal{D},m)\sim\mathcal{T}} \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} \mathop{\mathbb{E}}_{h\sim Q(S,P)} \mathop{\mathbb{E}}_{z\sim\mathcal{D}} \ell(h, z). \tag{7}$$

While $er(\mathcal{Q}, \tau)$ is still not commutable in practice because of the unknown task distribution and data distribution, nevertheless, we can estimate this by the *empiric error*

$$\widehat{er}\left(\mathcal{Q}, S_1, \ldots, S_n\right) \triangleq \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \frac{1}{n} \sum_{i=1}^{n} \widehat{er}\left(Q\left(S_i, P\right), S_i\right). \tag{8}$$

Different to the single task, the generalization error and empirical error in meta-learning relate to the expectation of P over hyper-posterior distribution

$\mathcal{Q}$. Then, the PAC-Bayes meta-learning bound proposed in [6] is introduced as follows

Lemma 2 *(Classical meta-learning PAC-Bayes bound [6]). Let $\mathcal{P} \in \mathcal{M}$ be some hyper-prior distribution over $\mathcal{H}$, and Q be a base learner. Then for any $\delta \in (0, 1]$, the following inequality holds uniformly for all hyper-posteriors distributions $\mathcal{Q} \in \mathcal{M}$ with a probability of at least $1 - \delta$*

$$\begin{aligned} & er(\mathcal{Q}, \tau) \leq \frac{1}{n} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{P \sim \mathcal{Q}} \hat{er} \left(Q_i \left(S_i, P \right), S_i \right) \\ & + \frac{1}{n} \sum_{i=1}^{n} \sqrt{\frac{\left(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P \sim \mathcal{Q}} D(Q||P) + \log \frac{2nm_i}{\delta} \right)}{2(m_i - 1)}} \\ & + \sqrt{\frac{1}{2(n-1)} \left(D(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \right)}. \end{aligned} \tag{9}$$

Meta-learning comprises two parts: the meta-learner extracts common knowledge (prior knowledge) from different observed tasks, and the base learner aims to adapt new tasks. Therefore the PAC-Bayes meta-learning bound also includes two regularization items: task complexity regularization item and sample complexity regularization term. The first task complexity term is the average of task-complexity terms of observed tasks, created by the finite number of samples in each observed tasks. This term converges to zero in the face of a large number of samples in each task. The second is an environment (or sample) complexity term, which is caused by a finite number of observed training tasks. Obviously, this term converges to zero if an infinite number of tasks is observed from the task environment.

4. PAC-Bayes meta-learning bounds

PAC-Bayes theory provides a theoretical framework for the generalization performance analysis in meta-learning. By selecting the PAC-Bayes bound as the training objective, one can develop a deep neural network with guaranteed generalization performance. Therefore, the tighter bound can achieve enhanced results. Motivated by this, in this section, based on the PAC-Bayes relative entropy theory, we propose three novel PAC-Bayes bounds for meta-learning, including *meta-learning PAC-Bayes λ bound* (Theorem 3 in Section 4.1), *meta-learning PAC-Bayes quadratic bound* (Theorem 4 in Section 4.2), and *meta-learning PAC-Bayes variational bound* (Theorem 5 in Section 4.3).

As shown in Lemma. 1, due to the binary KL divergence kl, this generalization error upper bound cannot be directly used as the training objective to guide the training of neural network. Therefore, utilizing different inequality, three novel PAC-Bayes meta-learning bounds are proposed.

4.1. Meta-learning PAC-Bayes λ bound

Theorem 3 *(meta-learning PAC-Bayes λ bound). Let $\mathcal{P}$ be some hyper-prior distribution, and Q be the posterior distribution, which is also called as the base learner. Then for any $\delta \in (0, 1]$ and $\lambda \in (0, 2)$, the following inequality holds uniformly for all hyper-posteriors distributions $\mathcal{Q} \in \mathcal{M}$ with probability at least $1 - \delta$*

$$\begin{aligned} er(\mathcal{Q}, \tau) \leq & \frac{1}{n} \sum_{i=1}^{n} \frac{1}{(1-\lambda_0/2)^2} \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q, S_i\right) \\ & + \frac{1}{n} \sum_{i=1}^{n} \frac{D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log \frac{4n\sqrt{m_i}}{\delta}}{m_i \lambda (1-\lambda/2)^2} \\ & + \sqrt{\frac{1}{2(n-1)} \left(D(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \right)}. \end{aligned} \tag{10}$$

With the reasonable selection of λ, this meta-learning PAC-Bayes bound can attain a tighter generalization error bound. The proof of this meta-learning bound is shown as follows.

Similar with the single task PAC Bayes theory, the Meta-learning PAC-Bayes λ bound also achieves fast convergence rate with $O(\frac{1}{m_i})$. Furthermore, based on the *PAC Bayes quadratic bound* in Eq. 32, the another PAC-Bayes meta-learning quadratic bound is proposed.

4.2. Meta-learning PAC-Bayes quadratic bound

Theorem 4 *(meta-learning PAC-Bayes quadratic bound). Let $\mathcal{P}$ be some hyper-prior distribution, and Q be the posterior distribution, which is also called as the base learner. Then for any $\delta \in (0, 1]$, the following inequality holds uniformly for all hyper-posteriors distributions $\mathcal{Q} \in \mathcal{M}$ with probability at least $1 - \delta$*

$$\begin{aligned} er(\mathcal{Q}, \tau) \leq & \frac{1}{n} \sum_{i=1}^{n} \Big(\sqrt{\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}(Q, S_i) + \epsilon_i} + \sqrt{\epsilon_i} \Big)^2 \\ & + \sqrt{\frac{1}{2(n-1)} \big(D(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \big)}. \end{aligned} \tag{11}$$

Here the $\epsilon_i = \frac{1}{2m_i} \big(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log \frac{4n\sqrt{m_i}}{\delta} \big)$.

As aforementioned, when the generalization error is smaller (especially $er(Q,\mathcal{D}) < 1/4$), this bound is tighter (see [29]) than the PAC-Bayes λ bound. When generalization error $er(Q,\mathcal{D})$ varies, different bounds can enact alternative different generalization performances. Motivated by this, [55] proposes a variational KL bound. with the fast convergence rate $O(\frac{1}{n})$.

With different situations, different bounds may lead to different generalization performances. One can combine the two above-mentioned meta-learning bounds by a function which is defined piecewise to improve performance. The variational KL bound can take the minimum value of Theorem 3 and Theorem 4, ensuring it is tight in both regimes. Prompted by PAC-Bayes variational bound, meta-learning PAC-Bayes variational bound is derived as follows:

4.3. Meta-learning PAC-Bayes variational bound

It has been proved that when the generalization error is smaller (especially $er(Q,\mathcal{D}) < 1/4$), this bound is tighter (see [29]) than the classical PAC-Bayes bound. However its convergence rate $O(\frac{1}{\sqrt{m}})$ is slower than the PAC-Bayes λ bound. When generalization error $er(Q,\mathcal{D})$ varies, different bounds can enact alternative different generalization performances.

Contrasting with the two previously described PAC-Bayes bounds, PAC-Bayes variational bound can take a minimum of two bounds, which might achieve a tighter generalization error for upper bound.

Theorem 5 *(meta-learning PAC-Bayes variational bound). Let P be some hyper-prior distribution, and Q be the posterior distribution, which is also known as base learner. Then for any $\delta \in (0,1]$, the following inequality holds uniformly for all hyper-posteriors distributions $\mathcal{Q} \in \mathcal{M}$ with probability at least $1-\delta$*

$$\begin{aligned} & er(\mathcal{Q},\tau) \leq \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q,S_i\right) \\ & +\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \min\Big(\varepsilon_i + \sqrt{\varepsilon_i\big(\varepsilon_i + 2\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q,S_i\right)\big)}, \sqrt{\tfrac{\varepsilon_i}{2}}\Big) \\ & +\sqrt{\tfrac{1}{2(n-1)}\left(D(\mathcal{Q}||\mathcal{P}) + \log\tfrac{2n}{\delta}\right)}. \end{aligned} \tag{12}$$

Here $\varepsilon_i = \frac{1}{m_i}\Big(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D\left(Q||P\right) + \log\frac{4n\sqrt{m_i}}{\delta}\Big)$.

The proof of those theorems have been attached in Appendix.

5. PAC-Bayes bounds with data-dependent prior

For the PAC-Bayes theory, the generalization error upper bound depends mainly on the regularization item involving the distance between prior distribution P and posterior distribution Q. However, the prior distribution is chosen randomly, with a view to measuring the parameter space. Especially in meta-learning, the generalization error bound involves both hyper-prior and hyper-posterior distributions, which are hard to converge. Seeking to solve this issue, as show in Figure 3, in this section we aim to learn a localized prior distribution through the *ERM approach* on a part of the training samples. Then the remaining data will be used to optimize generalization error bound.

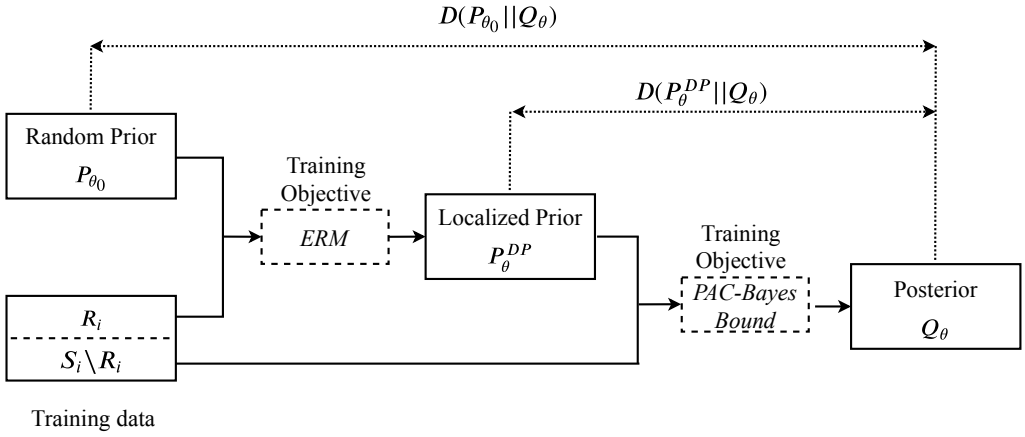


Figure 3: A simplified framework for PAC-Bayes bound with a data dependent prior. The generalization error upper bound is dominated by the distance between prior distribution and posterior distribution as well as the distance between the hyper-prior distribution and hyper-posterior distribution. The original prior distribution is selected as the random prior distribution, which leads to loose PAC-Bayes bound. Therefore we learn a localized prior distribution through the *ERM approach* on a part of the training samples, then the PAC-Bayes meta-learning bound is optimized based on this data dependent prior, which can achieve fast convergence ability.

Akin to the classical PAC-Bayes bound with data-dependent prior, for the meta-learning, we try to propose a novel extended PAC-Bayes bound with data-dependent prior. Specifically, during the *training phase* of meta-learning, the corresponding dataset of training task i is also divided into two separate datasets. Based on the *ERM approach*, one can learn a data-dependent prior

distribution over a section of the training samples

$$P_\theta = \arg\min \mathop{\mathbb{E}}_{h \in P_\theta} er_{\text{emp}}(h, S), \tag{13}$$

where $er_{\text{emp}}(h) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(h, R_i\right)$, n is the number of all training tasks, R_i is the sample subset of task i selected from the whole dataset S_i, providing information which can be used to calculate data-dependent prior. The remaining data $S_i \backslash R_i$ of task i is applied to evaluate the generalization error bound for meta-learning. In practice, expectations over distribution P are difficult to calculate. Therefore, the Monte Carlo method deemed most effective in obtaining the numerical results of (13). Furthermore, prior distribution P is selected directly from hyper posterior distribution $\mathcal{Q}_\theta$. So the learned parameters P_θ is designated as the initial mean parameter of $\mathcal{Q}_\theta$. The extended PAC-Bayes bound with data-dependent prior can then be shown as follows:

Theorem 6 *(Meta-learning PAC-Bayes bound with data-dependent prior). Let $Q : \mathcal{Z}^m \times \mathcal{M} \to \mathcal{M}$ be a base learner, and let $\mathcal{P}$ be some predefined hyper-prior distribution. Then for any $\delta \in (0, 1]$ the following inequality holds uniformly for all hyper-posterior distributions $\mathcal{Q}$ with probability at least $1 - \delta$,*

$$\begin{aligned} er(\mathcal{Q}, \tau) \leq \frac{1}{n}\sum_{i=1}^{n} \mathop{\mathbb{E}}_{P \sim \mathcal{Q}_D} \hat{er}\left(Q, S_i \backslash R_i\right) + \\ + \frac{1}{n}\sum_{i=1}^{n} \sqrt{\frac{D(\mathcal{Q}_D||\mathcal{P}) + \mathop{\mathbb{E}}_{P \sim \mathcal{Q}_D} D(Q(P, S_i \backslash R_i)||P) + \log\frac{2nm_i}{\delta}}{2(m_i - 1)}} \\ + \sqrt{\frac{D(\mathcal{Q}_D||\mathcal{P}) + \log\frac{2n}{\delta}}{2(n-1)}}. \end{aligned} \tag{14}$$

Similarly, in the testing phase, when encountering new tasks, one can also learn a data-dependent prior P through the *ERM approach*.

Remark 7 *Compared to random prior distribution, data dependent prior distribution can be considered as a weakly informative prior, which is more closer to the optimum parameters posterior distribution. Therefore, this algorithm can achieve fast convergence performance.*

6. Algorithms implementation and its performance analysis

Meta-learning PAC-Bayes bound tries to provide a generalization performance guarantee for the learned model with an arbitrarily high probability.

In this section, we mainly focus on how to utilize PAC-Bayes theory to design neural network with generalization performance guarantee, and analyzing the algorithm complexity of designed neural network.

6.1. Algorithms implementation

In this section, the specific algorithm implementation of the PAC-Bayes meta-learning bound is introduced . Different from the traditional neural network, which directly optimizes the empirical loss function, we can directly minimize the generalization error upper bound (or the right side of the bound) to achieve a generalization performance guarantee.

6.1.1. Training objectives

Based on the proposed three meta-learning PAC-Bayes bounds, the corresponding three training objectives are developed by selecting the generalization error upper bound as the training objective:

Theorem 3 and Theorem 4 lead to the meta-learning PAC-Bayes λ objective

$$\begin{aligned} J_\lambda(\theta) &= \tfrac{1}{n}\sum_{i=1}^{n} \tfrac{1}{(1-\lambda_0/2)^2} \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q, S_i\right) \\ &+ \tfrac{1}{n}\sum_{i=1}^{n} \tfrac{D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log\frac{4n\sqrt{m_i}}{\delta}}{m_i\lambda(1-\lambda/2)^2} \\ &+ \sqrt{\tfrac{1}{2(n-1)}\left(D(\mathcal{Q}||\mathcal{P}) + \log\tfrac{2n}{\delta}\right)}, \end{aligned} \tag{15}$$

and meta-learning quadratic PAC-Bayes objective

$$\begin{aligned} J_{\text{quad}}(\theta) &= \tfrac{1}{n}\sum_{i=1}^{n}\left(\sqrt{\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q, S_i\right) + \epsilon_i} + \sqrt{\epsilon_i}\right)^2 \\ &+ \sqrt{\tfrac{1}{2(n-1)}\left(D(\mathcal{Q}||\mathcal{P}) + \log\tfrac{2n}{\delta}\right)}. \end{aligned} \tag{16}$$

Here the $\epsilon_i = \frac{1}{2m_i}\left(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D\left(Q||P\right) + \log\frac{4n\sqrt{m_i}}{\delta}\right)$.

By comparison, the training objective from Theorem. 5 takes the following form

$$\begin{aligned} J_{\text{varia}}(\theta) &= \tfrac{1}{n}\sum_{i=1}^{n} \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q, S_i\right) \\ &+ \tfrac{1}{n}\sum_{i=1}^{n} \min\left(\varepsilon_i + \sqrt{\varepsilon_i\left(\varepsilon_i + 2\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q, S_i\right)\right)}, \sqrt{\tfrac{\varepsilon_i}{2}}\right) \\ &+ \sqrt{\tfrac{1}{2(n-1)}\left(D(\mathcal{Q}||\mathcal{P}) + \log\tfrac{2n}{\delta}\right)}, \end{aligned} \tag{17}$$

where the $\varepsilon_i = \frac{1}{m_i}\Big(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log\frac{4n\sqrt{m_i}}{\delta}\Big)$. Obviously this PAC-Bayes meta-learning bound includes three different items, empirical loss function and two regularization items.

6.1.2. Empirical loss function

In the training objective, the empirical loss function $\hat{er}(\cdot)$ is selected as bounded cross-entropy loss function.

As a rule, the standard loss function used on multi-class classification problems is the *cross-entropy loss function* $\ell : \mathbb{R}^k \times [k] \to \mathbb{R}$ defined by

$$\ell_{CE} = -\sum_{c=1}^{k} y_{o,c} \log(p_{o,c}), \tag{18}$$

where $y_{o,c}$ is the binary indicator (0 or 1) if class label c is the correct classification for observation o, k is the number of class and $p_{o,c}$ is the predicted probability observation o of class c. It is obviously that this loss function is unbounded loss function. However, the proposed meta-learning PAC-Bayes bound is only available for bounded loss function. Here, a "bounded cross-entropy" loss function is applied (See [29]) as the surrogate loss for training in all experiments with J_λ, J_{quad} and J_{varia}. Specifically, the loss function is clipped to $[0, \log(\frac{1}{p_{\min}})]$, where $p_{\min}$ is the lower bound of the network probabilities.

6.1.3. Stochastic neural network

In this paper we mainly focus on the stochastic neural network [56], where each weights are independent distributions. For simple, we initialize the weight distribution as the Gaussian distribution. In the meta-learning setting, as shown in Fig. 2, there consists of meta-level and base-level. Similarly the specific forms of hyper-prior distribution $\mathcal{P}$, hyper-posterior distribution $\mathcal{Q}$ and weights distribution of the stochastic neural network are selected.

For the meta-learning PAC-Bayes bound, the hyper-prior distribution $\mathcal{P}$ is set as a zero-mean Gaussian distribution

$$\mathcal{P} \triangleq \mathcal{N}\left(0, \kappa_{\mathcal{P}}^2 I_{N_P \times N_P}\right), \tag{19}$$

where $\kappa_{\mathcal{P}} > 0$ is constant and N_P is the number of neural network parameters w.

Correspondingly, the hyper-posterior distribution $\mathcal{Q}$, which consists of all distributions over $\mathbb{R}^{N_P}$, is defined as a family of isotropic Gaussian distributions as follows

$$\mathcal{Q}_\theta \triangleq \mathcal{N}\left(\theta, \kappa_{\mathcal{Q}}^2 I_{N_P \times N_P}\right), \tag{20}$$

Algorithm 1 Meta training phase, without data-dependent prior

Input: Datasets of n training tasks: $S_1, ..., S_n$.

Output: Learned meta-learner with parameter θ.

Initializing hyper-prior $\mathcal{P}$, hyper-posterior $\mathcal{Q}$, prior model θ, posterior model $\phi_i, i = 1, ..., n$;
while not done **do**
    for task $i, i = 1, ..., n$ **do**
        Sample mini-batch from datasets $S_i, i = 1, ..., n$
        Calculate $D(Q_{\phi_i} \| P_\theta)$(24)
        Calculate $\underset{P \sim \mathcal{Q}}{\mathbb{E}} \hat{er}\left(Q, S_i\right)$ by Monte-Carlo method
    end for
    Compute the training objective J (see 15, 16 or 17)
    Gradient step using $\begin{bmatrix} \nabla_\theta J \\ \nabla_{\phi_i} J \end{bmatrix}$
end while
return θ;

where $\kappa_{\mathcal{Q}} > 0$ is also a predefined constant. Therefore the KL divergence between the hyper-prior distribution $\mathcal{P}$ and hyper-posterior distribution $\mathcal{Q}$ equals

$$D\left(\mathcal{Q}_\theta \| \mathcal{P}\right) = \frac{\|\theta\|_2^2 + \kappa_{\mathcal{Q}}^2}{2\kappa_{\mathcal{P}}^2} + \log \frac{\kappa_{\mathcal{P}}}{\kappa_{\mathcal{Q}}} - \frac{1}{2}. \tag{21}$$

In the PAC-Bound theory, a probability neural network is applied, which means all weights w are stochastic variables drawing from prior or posterior distribution. In this paper, we define that each weight w_i in the neural network as it obeys Gaussian distribution. The prior P_θ and the posteriors $Q_{\phi_i}, i = 1, \ldots, n$, are defined as factorized Gaussian distributions

$$P_\theta(w) = \prod_{k=1}^{d} \mathcal{N}\left(w_k; \mu_{P,k}, \sigma_{P,k}^2\right), \tag{22}$$

$$Q_{\phi_i}(w) = \prod_{k=1}^{d} \mathcal{N}\left(w_k; \mu_{i,k}, \sigma_{i,k}^2\right), \tag{23}$$

where d is the number of neural network parameters and n is the number of tasks. The corresponding KL divergence between prior P_θ and the posteriors

Algorithm 2 Meta training phase, with data-dependent prior

Input: Datasets of n training tasks: $S_1, ..., S_n$.

Output: Learned meta-learner with parameter θ.

Initializing prior model θ;
Separate training datasets S_i into two parts $S_i/R_i, R_i, i = 1, ..., n$;
while not done **do**
    Sample mini-batch from datasets $R_i, i = 1, ..., n$
    Calculate $\underset{h \in P_\theta}{\mathbb{E}} er_{\text{emp}}(h, S)$ (13)
    Gradient step using $\nabla_\theta J$
end while
Initializing hyper-posterior $\mathcal{Q}$ with learned parameter θ, prior model P, posterior model $Q, i = 1, ..., n$;
while not done **do**
    for task $i, i = 1, ..., n$ **do**
        Sample mini-batch from $S_i/R_i, i = 1, ..., n$
        Calculate $D(Q_{\phi_i} \| P_\theta)$(24)
        Calculate $\underset{P \sim \mathcal{Q}}{\mathbb{E}} \hat{er}\,(Q, S_i)$ by Monte-Carlo method
    end for
    Compute the training objective f (see 15, 16 or 17)
    Gradient step using $\begin{bmatrix} \nabla_\theta J \\ \nabla_{\phi_i} J \end{bmatrix}$
end while
return θ;

$Q_{\phi_i}, i = 1, \ldots, n$, is

$$D\left(Q_{\phi_i} \| P_\theta\right) = \frac{1}{2} \sum_{k=1}^{d} \left(\log \frac{\sigma_{P,k}^2}{\sigma_{i,k}^2} + \frac{\sigma_{i,k}^2 + \left(\mu_{i,k} - \mu_{P,k}\right)^2}{\sigma_{P,k}^2} - 1 \right). \tag{24}$$

As we started earlier, the prior distribution $P_{\tilde{\theta}}$ is sampled from hyper-posterior distribution $\mathcal{Q}_\theta$. Practically, it follows that the prior distribution parameters $\tilde{\theta} = \theta + \varepsilon_P, \varepsilon_P \sim \mathcal{N}\left(0, \kappa_{\mathcal{Q}}^2 I_{N_P \times N_P}\right)$. In other words, prior distribution $P_{\tilde{\theta}}$ sampling from hyper-posterior distribution $\mathcal{Q}_\theta$ means adding Gaussian noise ε_P to the parameters θ during training. The specific pseudo code is shown in Algorithm 1 and Algorithm 2 for both random prior and data-dependent prior respectively.

Remark 8 *Without loss of generality, the initialization prior distributions of each network parameters are selected as Gaussian distribution. Utilizing the reparameterization trick, it is obviously that the posterior distributions of each parameters are also Gaussian distributions, which makes it convenience to calculate the KL divergence between prior and posterior distributions. For more details, please see [56, 57].*

6.2. Computational complexity analysis

In this section the computational complexity of this algorithm is analyzed, Similar to [56], the stochastic neural network (SNN) is applied in this paper, where each weight is assigned a distribution (Gaussian distribution here). By utilizing the reparameterization trick, each posterior weight distribution is still a Gaussian distribution. Therefore, the number of network parameters is twice that of other neural networks with the same structure. During the training phase there are two ways to decide the weight, one is that each weight can be sampled randomly from its distribution, or one can select the mean of the Gaussian distribution as the weight. Once the weight is decided, there is no difference between the SNN and classical deterministic neural network. The computational complexity of sampling is $O(N)$. Furthermore in our algorithm, the loss function is selected as the PAC-Bayes generalization error upper bound, thus its computational complexity is $O(N^2)$.

7. Experiments

In this section, the performance of our proposed meta-learning PAC-Bayes bound algorithms is illustrated using image classification tasks solved by stochastic neural networks. Specifically, we conduct our procedure in two different environments based on the MNIST dataset, these being *permuted pixels* and *permuted labels*. For the *permuted pixels* environment, each task is constructed by a shuffle of image pixels with 60000 training samples and 10000 testing samples. For the *permuted labels* environment, each task is generated by a permutation of image labels with the same number of training and testing samples as produced in the permuted pixels environment.

For the shuffled pixels experiment, the neural network structure selected is a fully connected neural network (FCN) with 4 layers (3 hidden layers and a linear output layer) and 400 units per layer. For the permuted labels experiment, the neural network structure is designated as a 4-layers convolutional neural network (CNN), comprising 2 fully convolution layers ,each with 5×5

kernels, and 2 full connected layers. For all experiments, ReLU activations are used. The optimizer is selected as Adam, with a learning rate of 10^{-3}.

For both of two experiments, each initialized log-var $\log \sigma_P^2$ of weights is drawn from $\mathcal{N}(-10, 0.01)$. The hyper-prior and hyper-posterior parameters are $\kappa_{\mathcal{P}} = 2000$ and $\kappa_{\mathcal{Q}} = 0.001$ respectively. In the meta-learning PAC-Bayes bound, the confidence parameter chosen is $\delta = 0.1$. The source code is available at GitHub[1].

In this section, we focus on the performance of the three proposed meta-learning PAC-Bayes bounds. For the meta-training phase, we run the total training of 50 epochs, maximal number of tasks in each meta-batch being 16, while 10 tasks are used for the meta-learner to learn. For the testing phase, we run a total testing of 20 epochs, using 20 tasks to confirm the meta-learner performance. We select 128 as the data batch size for training and testing. Specifically we mainly address on three issues using several experiments:

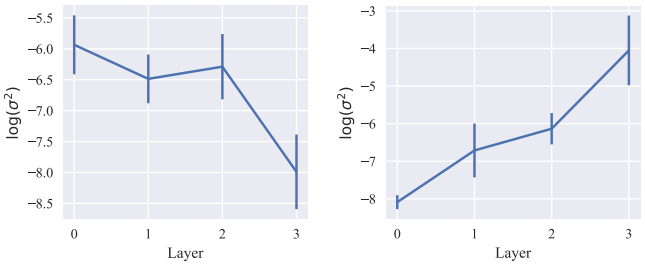
1. How does the number of tasks influence the model performance and neural network weight distributions?
2. Does the proposed three meta-learning PAC-Bayes bounds achieve a competitive generalization performance guarantee?
3. Does the meta-learning PAC-Bayes bound with a data dependent prior achieve fast convergence ability?

For a baseline, we measure the performance of learning from scratch using a stochastic network with no transfer from the meta-training tasks (See Table. 1). In the data-dependent prior experiment, the baseline is random prior.

Table 1: Comparing test error of various methods in both shuffled pixels and permuted labels environments ($\pm$ indicates the 95% confidence interval).

	Shuffled pixels	Permuted labels
Scratch	2.38 ± 0.13	1.48 ± 0.08
f_λ	2.48 ± 0.12	$0,78 \pm 0.11$
f	2.57 ± 0.09	0.93 ± 0.10
f	2.31 ± 0.15	0.88 ± 0.11

[1]Codes are available on `https://github.com/tyliu22/Meta-learning-PAC-Bayes-bound-with-data-depedent-prior.git`



(a) (b)

Figure 4: Model parameter analysis by layers, $\log(\sigma^2)$ represents the weight uncertainty of each layer. (a) Prior model parameter analysis for shuffled pixels environment; (b) Prior model parameter analysis for permuted labels environment.

7.1. How does the task number influence the model performance and network weight distributions?

First, we investigate the weights of the stochastic neural network. As shown in Figure 4(a), the average log-variance parameter of each layer's weights is analyzed. The higher the average $\log(\sigma^2)$ is, the more flexible are the weights are. In the shuffled pixel experiment, the lower layers perform with high variance which can extract the feature of shuffled-pixels image robustly, while the higher layers perform with a low variance which corresponds with fixed labels. In contrast, in the permuted label experiments, as shown in Figure 4(b), the higher layers perform with high variance which can adapt robustly to the permutation of the image label, and the lower layers' low variance performance corresponds to the fixed-image samples.

Next, the influence of a different number of training tasks on performance is analyzed, in relation to the generalization error bound, empiric loss and empiric error. In Figure 5, it is clear that, as the number of training tasks increases, the learned model achieves improved generalization performance and accuracy.

7.2. Does the proposed three meta-learning PAC-Bayes bounds achieve a competitive generalization performance guarantee?

We also compare five meta-learning PAC-Bayes bounds in two different MNIST environments, namely the consist of meta-learning McAllester PAC-Bayes bound (f_{classic}), meta-learning Seeger PAC-Bayes bound (f_{Seeger}), meta-learning PAC-Bayes λ bound (f_λ), meta-learning PAC-Bayes quadratic bound

Table 2: Comparison of various PAC-Bayes bounds in training phase in both shuffled pixels and permuted labels environment with different prior model.

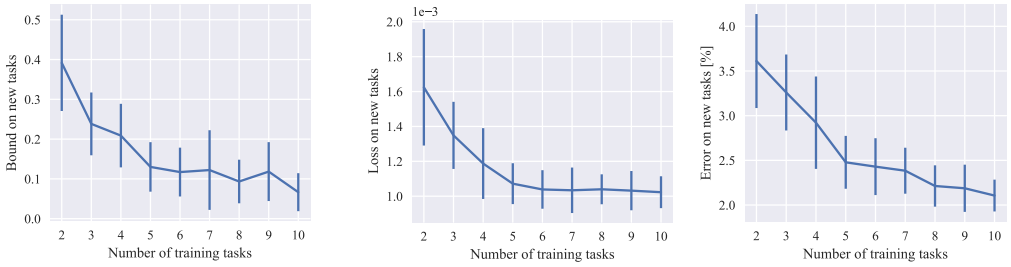
Environment	Prior model	Objective	Bound	Task complexity	Meta complexity	Empirical loss	Error (%)
Shuffled pixels	Random prior	f_{classic}	0.8117	0.1368	**0.6199**	0.05503	2.19
		f_{Seeger}	0.7924	0.1312	0.6312	0.02993	2.02
		f_{λ}	0.8094	**0.09615**	0.6327	0.04027	2.15
		f_{quad}	0.7761	0.1205	0.6286	0.02696	**2.01**
		f_{varia}	**0.7738**	0.1249	0.6262	**0.02272**	**2.01**
	data-dependent prior	f_{classic}	0.7213	0.01808	0.5519	0.1514	**4.02**
		f_{Seeger}	0.7030	0.01213	**0.5516**	0.1393	10.05
		f_{λ}	0.8095	**0.00223**	0.5547	**0.1263**	6.81
		f_{quad}	**0.7006**	0.01271	0.5523	0.1357	6.94
		f_{varia}	0.7681	0.01503	0.5518	0.2012	5.22
Permuted labels	Random prior	f_{classic}	0.6476	0.07143	**0.5504**	0.02574	0.813
		f_{Seeger}	0.6463	0.06144	0.5513	0.03357	0.880
		f_{λ}	0.6258	**0.04255**	0.5513	**0.01595**	**0.723**
		f_{quad}	0.6360	0.05332	0.5514	0.03136	0.878
		f_{varia}	**0.6208**	0.04597	0.5505	0.02419	0.811
	data-dependent prior	f_{classic}	0.6519	0.08259	**0.5505**	0.01887	0.781
		f_{Seeger}	0.6254	0.06053	0.5506	0.01422	0.771
		f_{λ}	0.6290	0.05074	0.5509	**0.01369**	0.748
		f_{quad}	0.6242	0.05539	0.5507	0.01816	**0.708**
		f_{varia}	**0.6143**	**0.04935**	0.5506	0.01440	0.781

The performance in the training phase of five meta-learning training objectives on the shuffled pixels and permuted labels MNIST environments with different prior models is analyzed in the above Table, in terms of PAC Bayes bound, task complexity, meta complexity, empirical loss and estimated error.

Table 3: Comparison of various PAC-Bayes bounds in testing phase in both shuffled pixels and permuted labels environment with different prior model ($\pm$ indicates the 95% confidence interval).

Environment	Prior model	Training objective	Test bound	Test loss (e − 04)	Test error (%)
Shuffled pixels	Random prior	f_{classic}	0.1580 ± 0.02075	$\mathbf{8.861 \pm 0.4593}$	2.541 ± 0.1341
		f_{Seeger}	0.2196 ± 0.04459	12.29 ± 0.9517	2.832 ± 0.169
		f_{λ}	$\mathbf{0.1271 \pm 0.01617}$	9.482 ± 0.5155	2.538 ± 0.1224
		f_{quad}	0.1957 ± 0.03627	10.61 ± 0.6422	2.753 ± 0.009047
		f_{varia}	0.1284 ± 0.02872	9.403 ± 0.3873	$\mathbf{2.432 \pm 0.07485}$
	data-dependent prior	f_{classic}	0.1660 ± 0.05372	$\mathbf{1.669 \pm 0.2091}$	$\mathbf{3.614 \pm 0.3152}$
		f_{Seeger}	0.1627 ± 0.04773	1.798 ± 0.2311	4.033 ± 0.3035
		f_{λ}	0.1893 ± 0.01472	1.816 ± 0.2451	3.692 ± 0.4079
		f_{quad}	0.1671 ± 0.08549	2.175 ± 0.4371	3.719 ± 0.3026
		f_{varia}	$\mathbf{0.1610 \pm 0.06316}$	1.790 ± 0.2783	3.809 ± 0.2905
Permuted labels	Random prior	f_{classic}	$\mathbf{2.905 \pm 0.1256}$	1.156 ± 0.09357	42.62 ± 4.999
		f_{Seeger}	3.196 ± 0.1018	1.374 ± 0.06323	54.83 ± 4.557
		f_{λ}	3.282 ± 0.1580	$\mathbf{1.074 \pm 0.09365}$	$\mathbf{40.73 \pm 5.466}$
		f_{quad}	3.284 ± 0.1081	1.393 ± 0.06071	55.60 ± 4.241
		f_{varia}	3.410 ± 0.1423	1.241 ± 0.09571	48.75 ± 5.178
	data-dependent prior	f_{classic}	0.1227 ± 0.02161	$\mathbf{3.354 \pm 0.1787}$	0.896 ± 0.04079
		f_{Seeger}	0.1220 ± 0.03072	3.426 ± 0.1126	0.950 ± 0.03477
		f_{λ}	$\mathbf{0.1071 \pm 0.03376}$	3.643 ± 0.1649	0.934 ± 0.03316
		f_{quad}	0.1358 ± 0.03387	3.374 ± 0.1499	0.913 ± 0.03068
		f_{varia}	0.1099 ± 0.02147	3.370 ± 0.1427	$\mathbf{0.882 \pm 0.02697}$

The performance in the testing phase of five meta-learning training objectives on the shuffled pixels and permuted labels MNIST environments with different prior models is analyzed in the above Table, in terms of test bound, test loss and test error.



(a) Generalization error bound (b) Empiric loss (c) Empiric error

Figure 5: The average performance of learning new tasks with different number of training tasks. (a) The average generalization error bound of learning new tasks; (b) The average empiric loss of learning new tasks; (c) The average empiric error of learning new tasks.

(f_{quad}) and meta-learning PAC-Bayes variational bound (f_{varia}). As shown in Table 2, the performance of various training objectives in the training phase with a random prior model is analyzed, in terms of bound, task complexity, meta complexity, empirical loss and estimated error. Figure 6(a) demonstrates that the proposed three meta-learning PAC-Bayes achieve a competitive bound, especially for the PAC-Bayes variational bound.

Furthermore, the performance of a learned meta-learner on new tasks with five training objectives is also established. Table 3 shows the specific result of generalization performance and accuracy in the testing phase. As shown in Figure 6(b), the proposed meta-learning PAC-Bayes λ bound and meta-learning PAC-Bayes variational bound perform a tighter generalization error bound. In addition, these two training objectives lead to improved accuracy.

7.3. Does the meta-learning PAC-Bayes bound with a data dependent prior achieve fast convergence ability?

Here, the meta-learning PAC-Bayes bounds with data-dependent prior algorithms are verified on the permuted MNIST dataset. We experiment both with priors gathered at randomly set weights and priors learnt using ERM on a part of the dataset. Specifically, training data is randomly divided into two separate datasets: 30% is used to learn a prior model using the ERM approach and the remaining data is applied to train the meta-learner. We run about 10 epochs in the training phase and 30 epochs in the testing phase to build the prior.

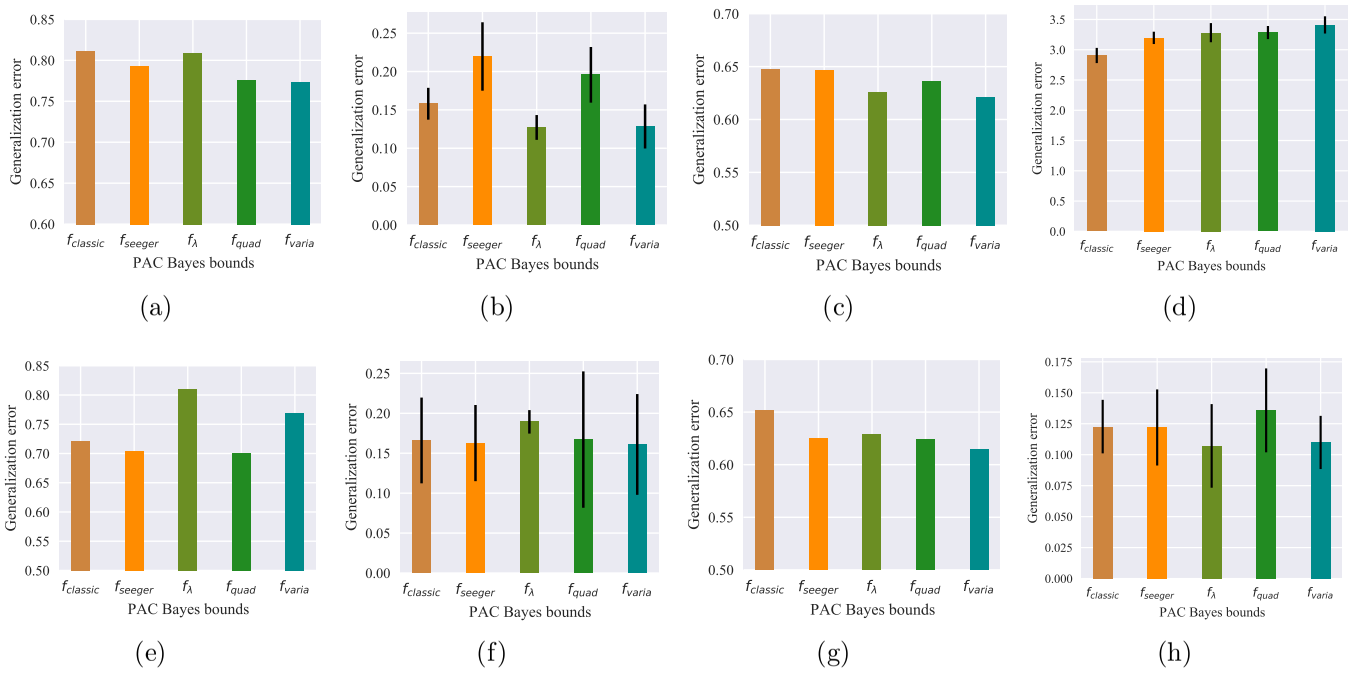
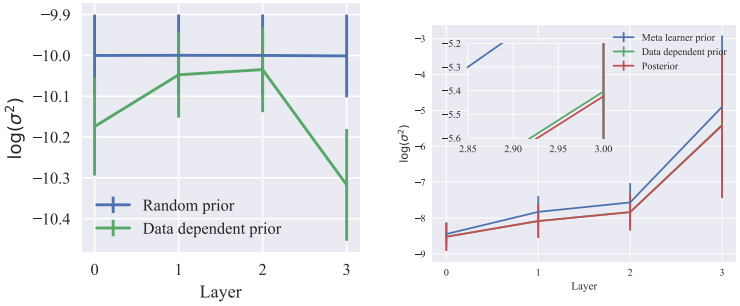


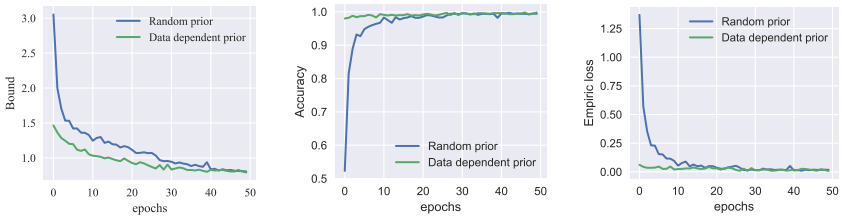
Figure 6: The average performance of learning new tasks with a different number of training tasks. (a) Comparison of different PAC-Bayes bounds in training phase in the permuted pixels environment; (b) The average test bound of learning new tasks for the permuted pixels with 95% confidence interval; (c) Comparison of different PAC-Bayes bounds in the training phase in the permuted labels environment; (d) The average test bound of learning new tasks for the permuted labels with 95% confidence interval; (e)Comparison of generalization error bounds for permuted pixels with data-dependent prior; (f) The average test bound with data-dependent prior for permuted pixels with 95% confidence interval; (g) Comparison of generalization error bounds for permuted labels with data-dependent prior; (h) The average test bound with data-dependent prior for permuted labels with 95% confidence interval.



(a) Weight prior distribution analysis

(b) Distance between weight prior and posterior distributions

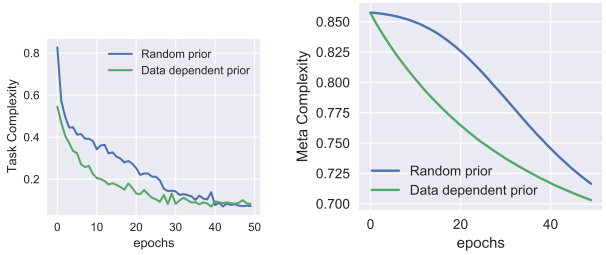
Figure 7: Parameter analysis of models with different prior. (a)Prior model parameter comparison between the random prior model and the learned data-dependent prior model; (b) Statistic analysis by layer for weight distributions, including random weight prior distribution, data-dependant prior distribution and learnt posterior distribution under a permuted labels setting.



(a) PAC-Bayes bound

(b) Accuracy

(c) Empiric loss



(d) Task complexity

(e) Meta complexity

Figure 8: Performance analysis of meta-learning with data-dependent prior (training objective is PAC-Bayes quadratic bound).

First, the discrepancy between the prior model with randomly initialized weights and the learned data-dependent prior model is compared in Figures 7(a) and Figure 7(b). It is obviously that, compared with the random prior model, the nature of the parameter learned in the data-dependent prior model is closer to the finally posterior model, which means it can achieve enhanced convergence performance. Besides, as shown in Figure 8, where the convergence performance between a random prior model and a data-dependent prior model during the training phase is analyzed. Obviously, the meta-learning PAC-Bayes bound with a data-dependent prior demonstrates a faster convergence ability with a series of epochs, in terms of the generalization error bound, accuracy, empiric loss, task complexity and meta-complexity.

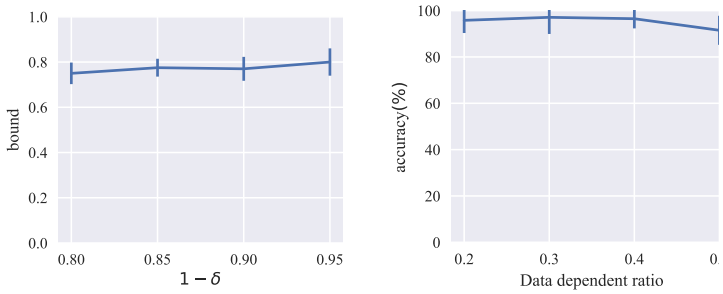
Furthermore, five meta-learning training objectives on two different MNIST environments are substantiated (See Table 2 and Table 3). As shown in Figure 6(e) and Figure 6(f), comparison between the two classical meta-learning PAC-Bayes bounds, the proposed meta-learning PAC-Bayes λ bound and meta-learning PAC-Bayes variational bound achieve a competitive generalization performance. The same conclusions can also can be drawn in the testing phase as shown in Figure 6(g) and Figure 6(h).

7.4. Ablation studies

In the Section, we focus on the study and evaluation of two most important hyperparameters: Confidence probability $1-\delta$and training data rate for data-dependent prior.

Results on the impact of confidence probability $1-\delta$ are depicted in Fig. 9(a). It is shown that PAC-Bayes bound varies with the change of $1-\delta$ influence the generalization error upper bound. Figure 4 shows the PAC-Bayes bound varies with the change of $1-\delta$. We find that the higher the confidence probability, the looser the error upper bound.

The training data rate for data-dependent prior measures how the training dataset is split into two separate datasets. As shown in Fig. 9(b), the split rate has a significant influence on the model performance. Large split rate may cause the learnt prior trapped in local minimum and over-fitting. The study shows that 20% data can be utilized to learn data-dependent prior, the remaining 80% data is then used to train model.



(a) Confidence probability (b) Data dependent ratio

Figure 9: Ablation analysis of meta-learning with data-dependent prior.

8. Conclusions and future work

In this paper, the meta-learning PAC-Bayes bounds with data-dependent prior algorithms are explored. The proposed theory can be applied to develop a practical algorithm, which can achieve a balance between model accuracy and generalization performance. First, based on the PAC-Bayes relative entropy bound, the meta-learning PAC-Bayes λ bound and meta-learning PAC-Bayes quadratic bound are derived. Furthermore, to achieving improved generalization performance, the meta-learning PAC-Bayes variational bound is also investigated. These bounds have been applied to develop a practical meta-learning model with a generalization performance guarantee and reduced overfitting. Next, in order to improve the convergence ability, it is also proposed that the ERM approach and the meta-learning PAC-Bayes bounds with data-dependent prior algorithms are combined.

The results of our experiments in two different MNIST environments, namely shuffled pixels and permuted labels, demonstrate that meta-learning PAC-Bayes λ bound and meta-learning PAC-Bayes variational bound can achieve competitive performances in terms of generalization error upper bound and estimation accuracy in both training and testing phases. Moreover, meta-learning PAC-Bayes bound with a data-dependent prior has a rapid convergence ability.

In future work, one could further investigate different prior distributions, such as a distribution-dependent prior, to achieve greater accuracy. Thus, how to learn a more efficient prior distribution is still an important issue. Besides, fast convergence rate algorithm can achieve better convergence performance with less training data or tasks. Thus meta learning fast rate PAC-Bayes

bound with convergence rate $O(\frac{1}{n})$ could be further investigated.

Acknowledgment

This work was supported by the Australian Research Council through the Discovery Project under Grant DP200100700.

Appendix

8.1. Proof of Theorem 3

In this section, a briefly proof of the extended PAC-Bayes λ bound is introduced.

Let n be the number of training tasks. The samples of task i are $z_{i,j}, j = 1, ..., K, K \triangleq m_i$, over the data distribution $\mathcal{D}_i$. The bounded loss function is defined as $\ell(h, z)$. We define the prior distribution P , which is sampled from hyper-prior distribution $\mathcal{P}$. The posterior distribution is defined as $Q = Q(S_i, P)$, which is sampled from hyper-posterior distribution $\mathcal{Q}$. Here as exemplified in [6], 'tuple hypothesis' is defined as $f = (P, h)$ where $P \in \mathcal{M}$ and $h \in \mathcal{H}$; 'prior over hypothesis' is defined as $\pi \triangleq (\mathcal{P}, P)$, where h is sampled from P. We note that the 'posterior over hypothesis' can be any distribution (even sample dependent). In particular, the PAC-Bayes bound will hold for the following family of distributions over $\mathcal{M} \times \mathcal{H}, \rho \triangleq (\mathcal{Q}, Q(S_i, P))$, where P is sampled from $\mathcal{Q}$ and h is sampled from $Q = Q(S_i, P)$ respectively.

The KL-divergence term is

$$\begin{aligned} D(\rho\|\pi) &= \mathop{\mathbb{E}}_{f\sim\rho} \log \frac{\rho(f)}{\pi(f)} \\ &= \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \mathop{\mathbb{E}}_{h\sim Q(S,P)} \log \frac{\mathcal{Q}(P)Q(S_i,P)(h)}{\mathcal{P}(P)P(h)} \\ &= \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \log \frac{\mathcal{Q}(P)}{\mathcal{P}(P)} + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \mathop{\mathbb{E}}_{h\sim Q(S,P)} \log \frac{Q(S_i,P)(h)}{P(h)} \\ &= D(\mathcal{Q}\|\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D\left(Q(S_i,P)\|P\right). \end{aligned} \tag{25}$$

Then by applying the *refined Pinsker inequality* $\mathrm{kl}(\hat{p}\|p) \geq \frac{(p-\hat{p})^2}{2p}$, $(\hat{p}, p \in (0, 1), \hat{p} < p)$ and the inequality $\sqrt{ab} \leq \frac{1}{2}\left(\lambda a + \frac{b}{\lambda}\right), \lambda > 0$ yields *PAC-Bayes λ bound* [58]

$$er(Q, \mathcal{D}) \leq \frac{\widehat{er}(Q,S)}{1-\lambda/2} + \frac{\mathrm{KL}(Q\|P)+\log(2\sqrt{n}/\delta)}{n\lambda(1-\lambda/2)}. \tag{26}$$

This generalization error upper bound can be used to guide the training of neural network by selecting the right side as the training objective with the fast convergence rate $O(\frac{1}{n})$.

Based on the *PAC Bayes* λ *bound* in Eq. 26, the Meta-learning PAC-Bayes λ bound is proved as follow. Just as with classical extended PAC-Bayes theory, our proof also involves two steps:

Step 1: For the task i, we use PAC-Bayes relative entropy bound to evaluate the generalization error for each observed task i

$$\begin{aligned}&\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} er\left(Q,\mathcal{D}_i\right) \leq \frac{1}{1-\lambda_i/2}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\widehat{er}\left(Q,S_i\right)\\&+\frac{D(\mathcal{Q}\|\mathcal{P})+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q\|P)+\log\frac{2\sqrt{m_i}}{\delta_i}}{m_i\lambda(1-\lambda_i/2)}.\end{aligned} \tag{27}$$

Step 2: We try to bound the environment-level generalization. Due to observing only a finite number of tasks from the environment, re-using the classical PAC-Bayes bound yields

$$\begin{aligned}&\mathop{\mathbb{E}}_{(\mathcal{D},m)\sim\tau}\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\mathop{\mathbb{E}}_{h\sim Q(S,P)}\mathop{\mathbb{E}}_{z\sim\mathcal{D}}\ell(h,z)\\&\leq\frac{1}{n}\sum_{i=1}^{n}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\mathop{\mathbb{E}}_{h\sim Q(S_i,P)}\mathop{\mathbb{E}}_{z\sim\mathcal{D}_i}\ell(h,z)\\&+\sqrt{\frac{1}{2(n-1)}\big(D(\mathcal{Q}\|\mathcal{P})+\log\frac{n}{\delta_0}\big)}.\end{aligned} \tag{28}$$

For simplicity, we can rewrite the above formula as

$$\begin{aligned}&er(\mathcal{Q},\tau)\leq\frac{1}{n}\sum_{i=1}^{n}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} er(Q(S_i,P),\mathcal{D}_i)\\&+\sqrt{\frac{1}{2(n-1)}\big(D(\mathcal{Q}\|\mathcal{P})+\log\frac{n}{\delta_0}\big)}.\end{aligned} \tag{29}$$

Combining Eq. 27 and Eq. 29 by union bound yields

$$\begin{aligned}&er(\mathcal{Q},\tau)\leq\frac{1}{n}\sum_{i=1}^{n}\frac{1}{1-\lambda_i/2}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\hat{er}\left(Q,S_i\right)\\&+\frac{1}{n}\sum_{i=1}^{n}\frac{D(\mathcal{Q}||\mathcal{P})+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P)+\log\frac{2\sqrt{m_i}}{\delta_i}}{m_i\lambda_i(1-\lambda_i/2)}\\&+\sqrt{\frac{1}{2(n-1)}\big(D(\mathcal{Q}||\mathcal{P})+\log\frac{n}{\delta_0}\big)}.\end{aligned} \tag{30}$$

Assuming that $\delta_0 = \frac{\delta}{2}$, $\delta_i = \frac{\delta}{2n}$ and $\lambda_0 = \lambda_i = \lambda$, this then yields

$$\begin{aligned} er(\mathcal{Q}, \tau) \leq & \frac{1}{n} \sum_{i=1}^{n} \frac{1}{(1-\lambda/2)} \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}\left(Q_i, S_i\right) \\ & + \frac{1}{n} \sum_{i=1}^{n} \frac{D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log \frac{4n\sqrt{m_i}}{\delta}}{m_i \lambda (1-\lambda/2)} \\ & + \sqrt{\frac{1}{2(n-1)} \left(D(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \right)}. \end{aligned} \tag{31}$$

8.2. Proof of Theorem 4

Different with *PAC-Bayes λ bound* in Eq. 26, one may view *refined Pinsker inequality* as a quadratic inequality on $\sqrt{er(Q, \mathcal{D})}$. Solving this inequality yields the following *PAC-Bayes quadratic bound* [29]

$$er(Q, \mathcal{D}) \leq \left(\sqrt{\widehat{er}\left(Q, S\right) + \varepsilon} + \sqrt{\varepsilon} \right)^2, \tag{32}$$

where $\varepsilon = \frac{1}{2m} \left(D\left(Q \| P\right) + \log \frac{2\sqrt{m}}{\delta} \right)$.

For the task i, applying PAC-Bayes quadratic bound to evaluate the generalization error in each of the observed tasks i yields

$$\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} er(Q, \mathcal{D}_i) \leq \left(\sqrt{\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \widehat{er}(Q, S_i) + \varepsilon_i} + \sqrt{\varepsilon_i} \right)^2, \tag{33}$$

where $\varepsilon_i = \frac{1}{2m_i} \left(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log \frac{2\sqrt{m_i}}{\delta_i} \right)$.

By utilizing the first step of the proof in Theorem 3, and assuming that $\delta_0 = \frac{\delta}{2}$, $\delta_i = \frac{\delta}{2n}$, we can get the following meta-learning PAC-Bayes bound

$$\begin{aligned} er(\mathcal{Q}, \tau) \leq & \frac{1}{n} \sum_{i=1}^{n} \left(\sqrt{\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} \hat{er}(Q, S_i) + \epsilon_i} + \sqrt{\epsilon_i} \right)^2 \\ & + \sqrt{\frac{1}{2(n-1)} \left(D(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \right)}. \end{aligned} \tag{34}$$

Here $\varepsilon_i = \frac{1}{m_i} \left(D(\mathcal{Q}||\mathcal{P}) + \mathop{\mathbb{E}}_{P\sim\mathcal{Q}} D(Q||P) + \log \frac{4n\sqrt{m_i}}{\delta} \right)$.

8.3. Proof of Theorem 5

First, a variational KL bound is introduced [55]

$$er(h, \mathcal{D}) \leq \min \begin{cases} \widehat{er}(Q, S) + \varepsilon + \sqrt{\varepsilon(\varepsilon + 2\widehat{er}(Q, S))}, \\ \widehat{er}(Q, S) + \sqrt{\varepsilon/2}, \end{cases} \tag{35}$$

where $\varepsilon = \frac{1}{m} \left(D(Q\|P) + \log \frac{2\sqrt{m}}{\delta} \right)$.

For the task i, we use PAC-Bayes variational bound to estimate generalization error in each of the observed task i

$$\begin{aligned}&\mathop{\mathbb{E}}_{P\sim\mathcal{Q}} er\left(Q_i,\mathcal{D}_i\right) \leq \tfrac{1}{n}\sum_{i=1}^{n}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\widehat{er}\left(Q,S_i\right)\\&+\tfrac{1}{n}\sum_{i=1}^{n}\min\Big(\varepsilon_i+\sqrt{\varepsilon_i\big(\varepsilon_i+2\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\hat{er}\left(Q,S_i\right)\big)},\sqrt{\tfrac{\varepsilon_i}{2}}\Big),\end{aligned} \tag{36}$$

where $\varepsilon_i=\frac{1}{2m_i}\Big(D(\mathcal{Q}||\mathcal{P})+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}D\left(Q||P\right)+\log\frac{2\sqrt{m_i}}{\delta_i}\Big)$.

By applying the first step of the proof in Theorem 3, and assuming that $\delta_0=\frac{\delta}{2}$, $\delta_i=\frac{\delta}{2n}$ yields

$$\begin{aligned}&er(\mathcal{Q},\tau)\leq\tfrac{1}{n}\sum_{i=1}^{n}\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\hat{er}\left(Q_i\left(S_i,P\right),S_i\right)\\&+\tfrac{1}{n}\sum_{i=1}^{n}\min(\varepsilon_i+\sqrt{\varepsilon_i(\varepsilon_i+2\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}\hat{er}(Q_i,S_i))},\sqrt{\tfrac{\varepsilon_i}{2}})\\&+\sqrt{\tfrac{1}{2(n-1)}\left(D(\mathcal{Q}||\mathcal{P})+\log\tfrac{2n}{\delta}\right)}.\end{aligned} \tag{37}$$

Here $\varepsilon_i=\frac{1}{m_i}\Big(D(\mathcal{Q}||\mathcal{P})+\mathop{\mathbb{E}}_{P\sim\mathcal{Q}}D\left(Q||P\right)+\log\frac{4n\sqrt{m_i}}{\delta}\Big)$.

References

[1] F. Xing, Y. Xie, H. Su, F. Liu, L. Yang, Deep learning in microscopy image analysis: A survey, IEEE Transactions on Neural Networks and Learning Systems 29 (10) (2017) 4550–4568.

[2] P. Tang, X. Wang, B. Shi, X. Bai, W. Liu, Z. Tu, Deep fishernet for image classification, IEEE Transactions on Neural Networks and Learning Systems 30 (7) (2018) 2244–2250.

[3] J. Luo, C.-M. Vong, P.-K. Wong, Sparse bayesian extreme learning machine for multi-classification, IEEE Transactions on Neural Networks and Learning Systems 25 (4) (2013) 836–843.

[4] Y. Liu, S. Liu, Y. Wang, F. Lombardi, J. Han, A survey of stochastic computing neural networks for machine learning applications, IEEE Transactions on Neural Networks and Learning Systems (2020).

[5] J. Schmidhuber, Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook, Ph.D. thesis, Technische Universität München (1987).

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

[7] C. Finn, S. Levine, Meta-learning: from few-shot learning to rapid reinforcement learning, ICML meta-learning 2019 tutorial (2019).

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

[9] N. Mishra, M. Rohaninejad, X. Chen, P. Abbeel, A simple neural attentive meta-learner, arXiv preprint arXiv:1707.03141 (2017).

[10] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, R. Shah, Signature verification using a "siamese" time delay neural network, in: Advances in Neural Information Processing Systems, 1994, pp. 737–744.

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

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

[13] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, N. De Freitas, Learning to learn by gradient descent by gradient descent, in: Advances in Neural Information Processing Systems, 2016, pp. 3981–3989.

[14] S. Ravi, H. Larochelle, Optimization as a model for few-shot learning, in: International Conference on Learning Representations, 2016.

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

[16] A. Nichol, J. Schulman, Reptile: a scalable metalearning algorithm, arXiv preprint arXiv:1803.02999 2 (2018) 2.

[17] A. Antoniou, H. Edwards, A. Storkey, How to train your MAML, arXiv preprint arXiv:1810.09502 (2018).

[18] C. Finn, K. Xu, S. Levine, Probabilistic model-agnostic meta-learning, in: Advances in Neural Information Processing Systems, 2018, pp. 9516–9527.

[19] T. Kim, J. Yoon, O. Dia, S. Kim, Y. Bengio, S. Ahn, Bayesian model-agnostic meta-learning, arXiv preprint arXiv:1806.03836 (2018).

[20] H. B. Lee, H. Lee, D. Na, S. Kim, M. Park, E. Yang, S. J. Hwang, Learning to balance: Bayesian meta-learning for imbalanced and out-of-distribution tasks, arXiv preprint arXiv:1905.12917 (2019).

[21] H. S. Behl, A. G. Baydin, P. H. Torr, Alpha maml: Adaptive model-agnostic meta-learning, arXiv preprint arXiv:1905.07435 (2019).

[22] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, R. Hadsell, Meta-learning with latent embedding optimization, arXiv preprint arXiv:1807.05960 (2018).

[23] E. Grant, C. Finn, S. Levine, T. Darrell, T. Griffiths, Recasting gradient-based meta-learning as hierarchical bayes, arXiv preprint arXiv:1801.08930 (2018).

[24] D. A. McAllester, Some PAC-Bayesian theorems, Machine Learning 37 (3) (1999) 355–363.

[25] J. Langford, R. Caruana, (not) bounding the true error, Advances in Neural Information Processing Systems 14 (2001) 809–816.

[26] M. Seeger, PAC-Bayesian generalisation error bounds for Gaussian process classification, Journal of Machine Learning Research 3 (Oct) (2002) 233–269.

[27] O. Catoni, PAC-Bayesian supervised classification: the thermodynamics of statistical learning, arXiv preprint arXiv:0712.0248 (2007).

[28] B. Neyshabur, S. Bhojanapalli, D. McAllester, N. Srebro, Exploring generalization in deep learning, Advances in Neural Information Processing Systems 30 (2017) 5947–5956.

[29] M. Pérez-Ortiz, O. Rivasplata, J. Shawe-Taylor, C. Szepesvári, Tighter risk certificates for neural networks, arXiv preprint arXiv:2007.12911 (2020).

[30] X. Xie, S. Sun, Pac-bayes bounds for twin support vector machines, Neurocomputing 234 (2017) 137–143.

[31] P. Germain, A. Habrard, F. Laviolette, E. Morvant, Pac-bayes and domain adaptation, Neurocomputing 379 (2020) 379–397.

[32] P. Germain, F. Bach, A. Lacoste, S. Lacoste-Julien, PAC-Bayesian theory meets Bayesian inference, Advances in Neural Information Processing Systems 29 (2016) 1884–1892.

[33] P. Alquier, J. Ridgway, N. Chopin, On the properties of variational approximations of Gibbs posteriors, The Journal of Machine Learning Research 17 (1) (2016) 8374–8414.

[34] M. Holland, PAC-Bayes under potentially heavy tails, in: Advances in Neural Information Processing Systems, 2019, pp. 2715–2724.

[35] M. Haddouche, B. Guedj, O. Rivasplata, J. Shawe-Taylor, PAC-Bayes unleashed: generalisation bounds with unbounded losses, arXiv preprint arXiv:2006.07279 (2020).

[36] B. London, A PAC-Bayesian analysis of randomized learning with application to stochastic gradient descent, Advances in Neural Information Processing Systems 30 (2017) 2931–2940.

[37] G. K. Dziugaite, D. M. Roy, Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data, arXiv preprint arXiv:1703.11008 (2017).

[38] W. Zhou, V. Veitch, M. Austern, R. P. Adams, P. Orbanz, Non-vacuous generalization bounds at the imagenet scale: a PAC-Bayesian compression approach, arXiv preprint arXiv:1804.05862 (2018).

[39] G. K. Dziugaite, D. Roy, Entropy-SGD optimizes the prior of a PAC-Bayes bound: Generalization properties of Entropy-SGD and data-dependent priors, in: International Conference on Machine Learning, PMLR, 2018, pp. 1377–1386.

[40] K. Miyaguchi, PAC-Bayesian transportation bound, arXiv preprint arXiv:1905.13435 (2019).

[41] B. Neyshabur, S. Bhojanapalli, N. Srebro, A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks, arXiv preprint arXiv:1707.09564 (2017).

[42] V. Nagarajan, J. Z. Kolter, Deterministic PAC-bayesian generalization bounds for deep networks via generalizing noise-resilience, arXiv preprint arXiv:1905.13344 (2019).

[43] E. Parrado-Hernández, A. Ambroladze, J. Shawe-Taylor, S. Sun, PAC-Bayes bounds with data dependent priors, The Journal of Machine Learning Research 13 (1) (2012) 3507–3531.

[44] G. Lever, F. Laviolette, J. Shawe-Taylor, Tighter PAC-Bayes bounds through distribution-dependent priors, Theoretical Computer Science 473 (2013) 4–28.

[45] O. Rivasplata, E. Parrado-Hernández, J. S. Shawe-Taylor, S. Sun, C. Szepesvári, PAC-Bayes bounds for stable algorithms with instance-dependent priors, Advances in Neural Information Processing Systems 31 (2018) 9214–9224.

[46] G. Lever, F. Laviolette, J. Shawe-Taylor, Distribution-dependent PAC-Bayes priors, in: International Conference on Algorithmic Learning Theory, Springer, 2010, pp. 119–133.

[47] L. Oneto, D. Anguita, S. Ridella, PAC-Bayesian analysis of distribution dependent priors: Tighter risk bounds and stability analysis, Pattern Recognition Letters 80 (2016) 200–207.

[48] G. K. Dziugaite, D. M. Roy, Data-dependent PAC-Bayes priors via differential privacy, Advances in Neural Information Processing Systems 31 (2018) 8430–8441.

[49] A. Pentina, C. Lampert, A PAC-Bayesian bound for lifelong learning, in: International Conference on Machine Learning, 2014, pp. 991–999.

[50] Y. Huang, W. Huang, L. Li, Z. Li, Meta-Learning PAC-Bayes Priors in Model Averaging, Proceedings of the AAAI Conference on Artificial Intelligence 34 (04) (2020) 4198–4205.

[51] A. Farid, A. Majumdar, Pac-bus: Meta-learning bounds via pac-bayes and uniform stability, arXiv preprint arXiv:2102.06589 (2021).

[52] S. T. Jose, O. Simeone, Information-theoretic generalization bounds for meta-learning and applications, Entropy 23 (1) (2021) 126.

[53] N. Ding, X. Chen, T. Levinboim, S. Goodman, R. Soricut, Bridging the gap between practice and pac-bayes theory in few-shot meta-learning, arXiv preprint arXiv:2105.14099 (2021).

[54] J. Rothfuss, V. Fortuin, M. Josifoski, A. Krause, Pacoh: Bayes-optimal meta-learning with pac-guarantees, arXiv preprint arXiv:2002.05551 (2020).

[55] G. K. Dziugaite, K. Hsu, W. Gharbieh, D. M. Roy, On the role of data in PAC-Bayes bounds, arXiv preprint arXiv:2006.10929 (2020).

[56] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural network, in: International Conference on Machine Learning, PMLR, 2015, pp. 1613–1622.

[57] D. P. Kingma, T. Salimans, M. Welling, Variational dropout and the local reparameterization trick, arXiv preprint arXiv:1506.02557 (2015).

[58] N. Thiemann, C. Igel, O. Wintenberger, Y. Seldin, A strongly quasiconvex PAC-Bayesian bound, in: International Conference on Algorithmic Learning Theory, PMLR, 2017, pp. 466–492.