Modular Meta-Learning with Shrinkage

Motivation

For meta learning, updating only these task-specific modules then allows the model to be adapted to low-data tasks for as many steps as necessary without risking overfitting. Unfortunately, **existing meta-learning methods either do not scale to long adaptation or else rely on handcrafted task-specific architectures**. Here, we propose a meta-learning approach that obviates the need for this often sub-optimal hand-selection. I

Contribution

- Introduce a hierarchical Bayesian model for modular meta-learning along with two parameter-estimation methods, which we show **generalize existing meta-learning algorithms**.
- Demonstrate that our approach enables identification of a small set of meaningful task-specific modules.
- our method prevents overfitting and improves predictive performance on problems that require many adaptation steps given only small amounts of data.

Background

Gradient-based Meta-Learning

Algorithm 1: Meta-learning pseudocode.

Input: Batch size B, steps L and learning rate α
Initialize ϕ
while *not done* **do**
  $\{t_1, \ldots, t_B\} \leftarrow$ sample mini-batch of tasks
  for *each task t in* $\{t_1, \ldots, t_B\}$ **do**
    Initialize $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\phi}$
    for *step* $l = 1 \ldots L$ **do**
      $\boldsymbol{\theta}_t \leftarrow \text{TASKADAPT}(\mathcal{D}_t, \boldsymbol{\phi}, \boldsymbol{\theta}_t)$
    end
  end
  // Meta update
  $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \alpha \cdot \frac{1}{B} \sum_t \Delta_t(\mathcal{D}_t, \boldsymbol{\phi}, \boldsymbol{\theta}_t)$
end

Algorithm 1 is a structure of a typical meta-learning algorithm, which could be: MAML, iMAML, Reptile. And

- TASKADAPT: task adaptation (inner loop)

- The meta-update specifies the contribution of task t to the meta parameters. (outer loop)

Modular Bayesian Meta-Learning

- Standard meta-learning, the meta parameters φ provide an initialization for the task parameters θ at test time. That is, all the neural network parameters are treated equally, and hence they must all be updated at test time. This strategy is inefficient and prone to overfittin.
- Split the network parameters into two groups: a group varying across tasks and a group that is shared across tasks.

To address the above issue, this paper proposed Modular Bayesian Meta-Learning. Instead of adapt entire parameters (as in MAML), the proposed approach adapt each module separately

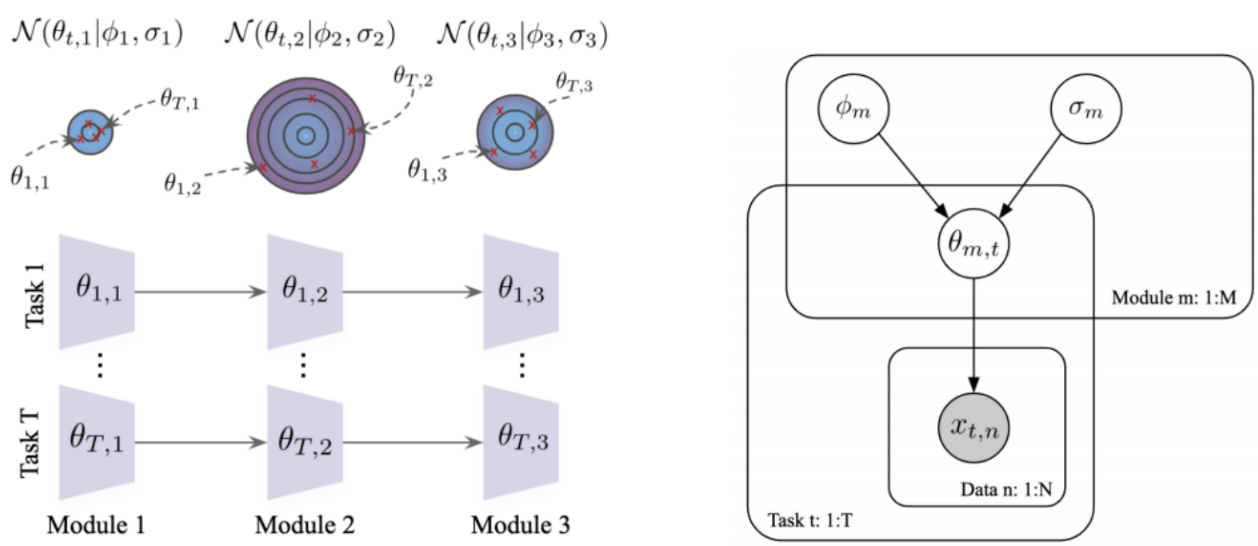

Fig.1 Modular meta-learning

The density of the hierarchical Bayesian model is

$$p(\boldsymbol{\phi}, \boldsymbol{\theta}, \mathcal{D}|\boldsymbol{\sigma}) = p(\boldsymbol{\phi}) \prod_{m=1}^{M} \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{\theta}_{m,t}|\boldsymbol{\phi}_m, \sigma_m^2 \mathbf{I}) \prod_{t=1}^{T} \prod_{n=1}^{N_t} p(\mathbf{x}_{t,n}|\boldsymbol{\theta}_t).$$

However, jointly maximizing the objective may collapse, Hence, the authors propose to alternatively optimize θ and Ø via MAP (with train set).

$$\hat{\boldsymbol{\theta}}_{1:T}(\boldsymbol{\sigma}), \hat{\boldsymbol{\phi}}(\boldsymbol{\sigma}) = \underset{\boldsymbol{\theta}_{1:T}, \boldsymbol{\phi}}{\operatorname{argmax}}\, \ell_{\mathrm{MAP}}, \quad \text{where} \quad \ell_{\mathrm{MAP}} := \log p\left(\boldsymbol{\theta}_{1:T}, \boldsymbol{\phi}|\mathcal{D}_{1:T}^{\text{train}}, \boldsymbol{\sigma}\right).$$

and sigma via predictive log-likelihood (with validation set)

$$\hat{\boldsymbol{\sigma}} = \underset{\boldsymbol{\sigma}}{\operatorname{argmax}} \log p\left(\mathcal{D}_{1:T}^{\text{val}} | \mathcal{D}_{1:T}^{\text{train}}, \boldsymbol{\sigma}\right) \approx \underset{\boldsymbol{\sigma}}{\operatorname{argmax}} \sum_{t=1}^{T} \log p(\mathcal{D}_t^{\text{val}} | \hat{\boldsymbol{\theta}}_t(\boldsymbol{\sigma})) := \underset{\boldsymbol{\sigma}}{\operatorname{argmax}} \, \ell_{\text{PLL}} \, ,$$

Discussion

- **If optimize Ø (also meta parameter) via log-likelihood, Then it becomes identical to first-order MAML**
- **If sigma → ∞, it becomes identical to Reptile**
- **Similar to iMAML, meta-gradient of sigma requires expensive 2nd-order gradients over θ and Ø**, Instead, one can approximate gradient using similar Jacobian technique of stationary points