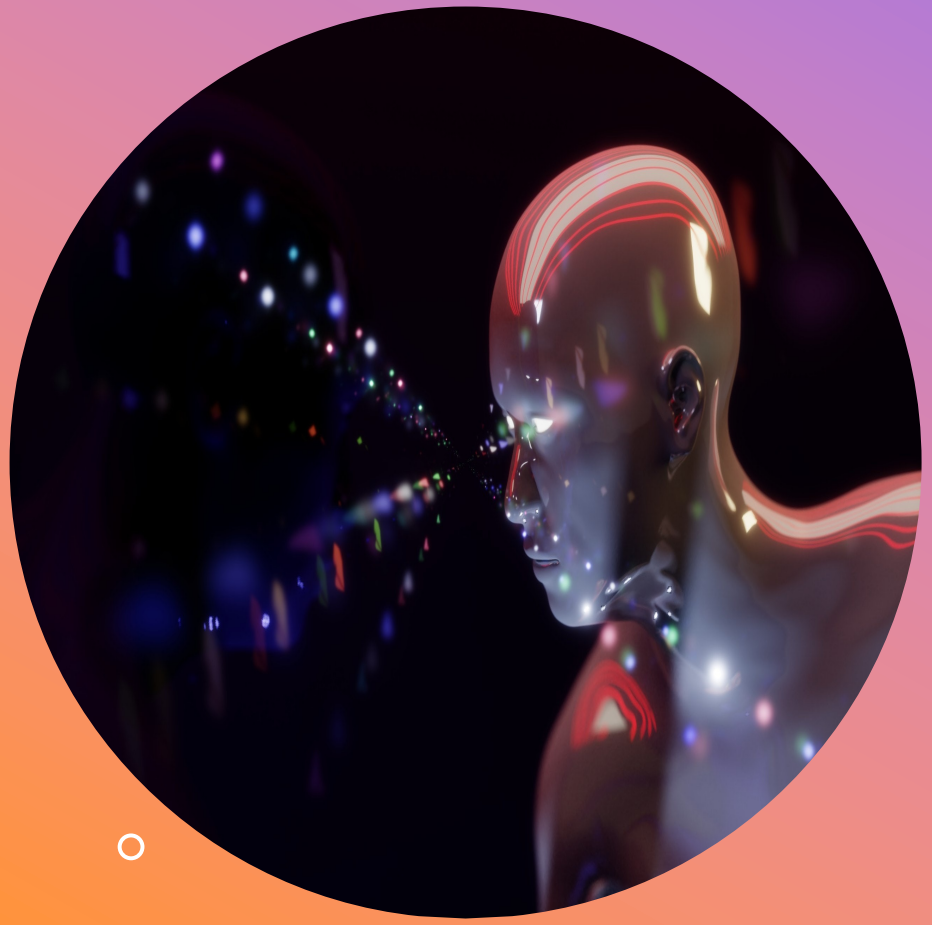
META-LEARNING WITH IMPLICIT GRADIENTS

Sowmiya Murugiah
sowmiyam@buffalo.edu
50485124



AGENDA

Introduction

Advanced ML Techniques

Meta-Learning

MAML vs iMAML

Formulation

Key Algorithm

Graph

Summary

Introduction

Supervised ML models are data-hungry. If you give large datasets to train large models, you learn a brilliant decision boundary. Traditional ML algorithms start to fall when situations tend to change fast and your model runs out of the training data. The motivation of meta-learning comes from being able to learn from small data. Meta parameters are learned in the outer loop while task-specific models are learned in the inner-loop. By drawing upon implicit differentiation, we develop the implicit MAML algorithm which depends only on the solution to the inner level optimization and not the path taken by the inner loop optimizer

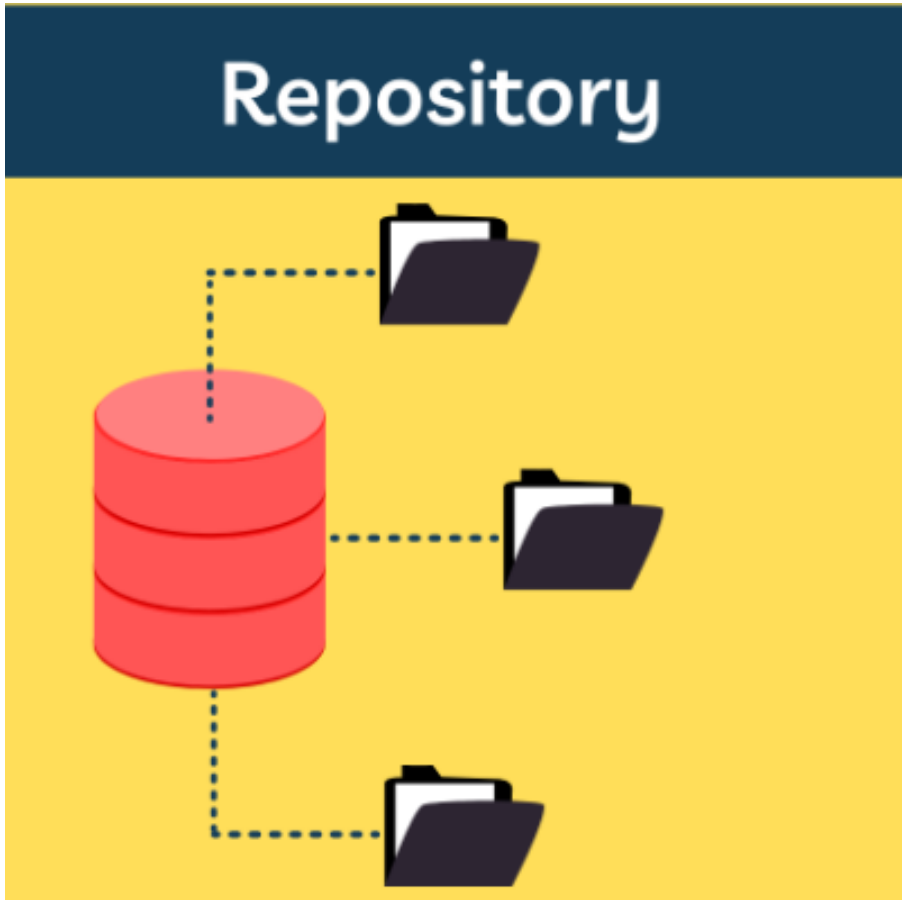
Advanced Machine Learning Techniques

- Transfer Learning – TL
- Multi-Transfer Learning – MTL
- Meta Learning

Transfer Learning

Transfer learning is a technique in machine learning and artificial intelligence where knowledge gained from one task or domain is applied to a different but related task or domain. In transfer learning, a pre-trained model is used as a starting point for a new learning task, rather than starting the learning process from scratch. The pre-trained model has learned features or representations that can be transferred to the new task, reducing the amount of training data required and improving the performance of the new model.

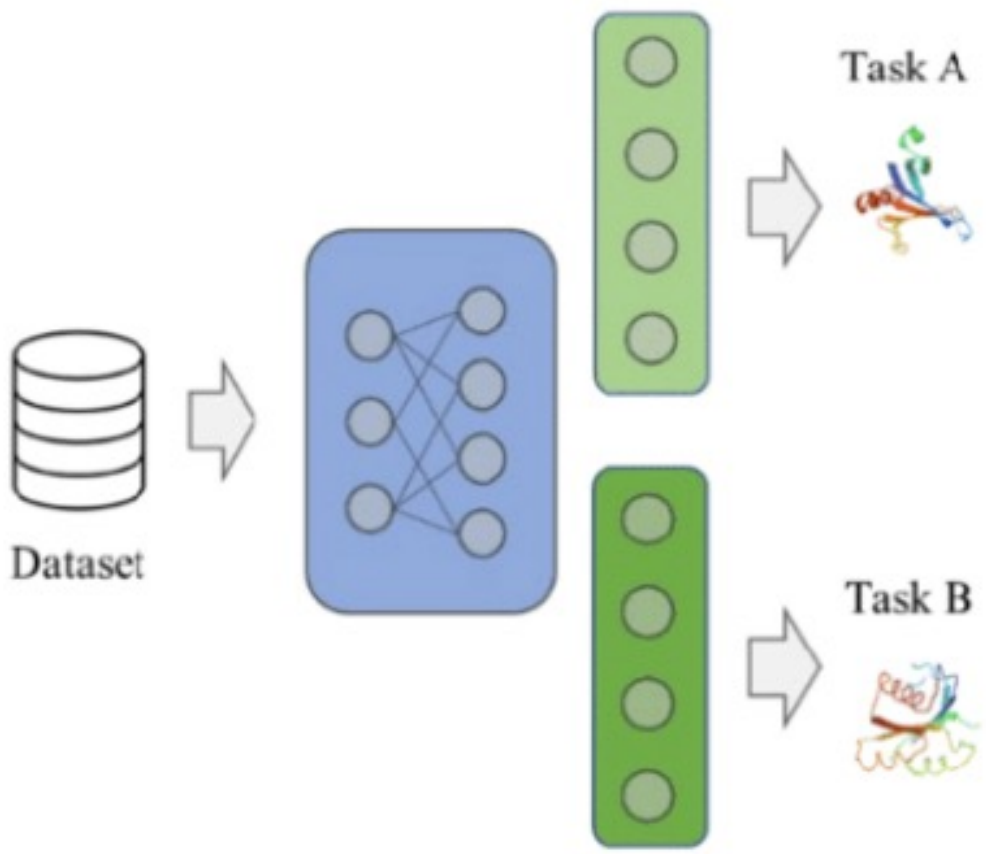
Disadvantage : It can be computationally expensive and time-consuming to train large models



Multi-Transfer Learning

Multi-Transfer Learning is a type of transfer learning that involves transferring knowledge from multiple source domains to a target domain. In traditional transfer learning, a pre-trained model is used to extract features from a single source domain, which are then used to train a new model for a target domain. However, in multi-transfer learning, the pre-trained model is trained on multiple source domains, each with different but related features.

Issue: limit top out accuracy because it is fully not dedicated to one task



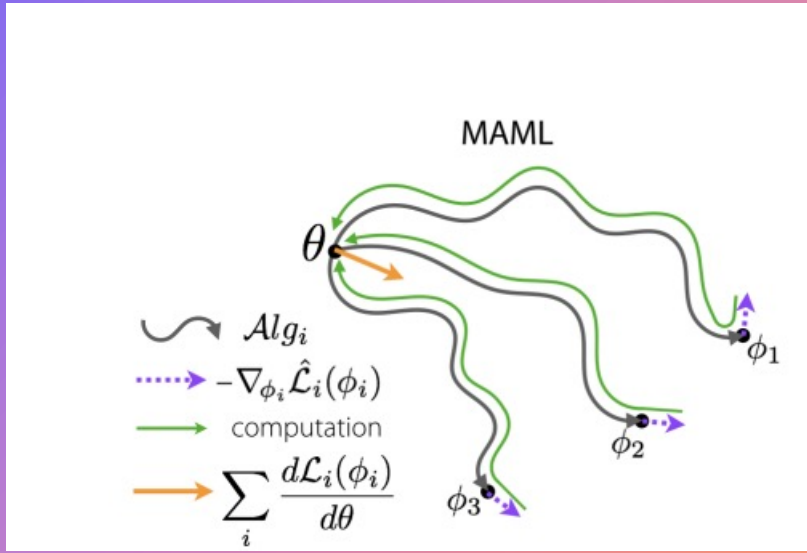
WHAT IS META-LEARNING

Meta-Learning

- It is referred ‘Learning to Learn’
- It is a subfield of machine learning and cognitive science that focuses on developing algorithms and models that can learn how to learn.
- The goal is to enable machines to quickly adapt to new tasks or environments by acquiring new skills and knowledge from past experiences.
- It uses bi-level optimization criteria. The meta-learning can be divided into 2 parts, inner loop, and outer loop. Inner loop may optimize some cost function however, outer loop optimizes environment.
- One popular method for meta-learning with implicit gradients is MAML

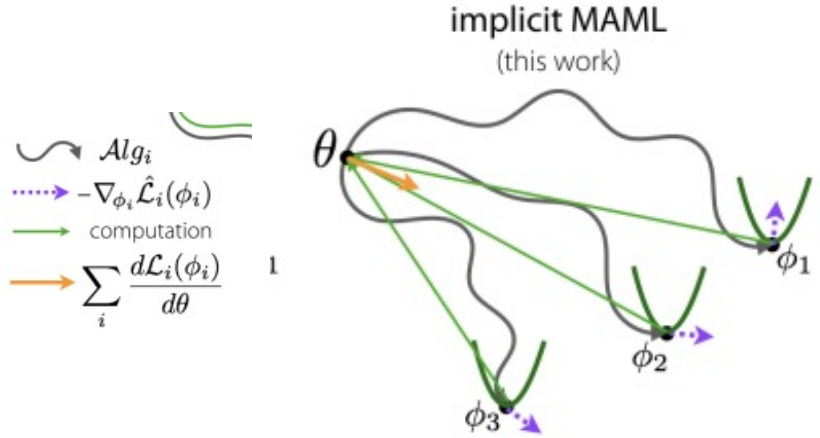
MAML - Model-Agnostic Meta-Learning

- MAML uses a first-order optimization method



iMAML

- iMAML uses a second-order optimization method. It depends only on the solution to the inner level optimization and not the path taken by the inner loop optimizer



Formulation

Outer and Inner Level

$$\overbrace{\boldsymbol{\theta}^*_{\mathrm{ML}} := \operatorname*{argmin}_{\boldsymbol{\theta}\in\Theta} F(\boldsymbol{\theta})}^{\mathrm{outer-level}},$$

$$\text{where } F(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} \mathcal{L}\left(\overbrace{\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^{\mathrm{tr}})}^{\mathrm{inner-level}}, \mathcal{D}_i^{\mathrm{test}}\right).$$

$$\boldsymbol{\phi}_i \equiv \mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^{\mathrm{tr}}) = \boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}_i^{\mathrm{tr}}). \quad \text{(inner-level of MAML)}$$

- θ_{ML} – Best Meta Learning parameter
- F - average validation loss function
- Alg - training algorithm performed on the training set of the particular task while starting from θ
- M – Number of tasks
- T_i – Task
- D_i - Dataset
- D_i^{tr} – Train set
- D_i^{test} – Test set
- Alpha – Learning Rate
- Φ_i – model parameters

Section 2.2 Proximal Regularization in the Inner Level

To have sufficient learning in the inner level while also avoiding over-fitting, Alg needs to incorporate some form of regularization. Since MAML uses a small number of gradient steps, this corresponds to early stopping and can be interpreted as a form of regularization and Bayesian prior [20]. In cases like ill-conditioned optimization landscapes and medium-shot learning, we may want to take many gradient steps, which poses two challenges for MAML. First, we need to store and differentiate through the long optimization path of Alg, which imposes a considerable computation and memory burden. Second, the dependence of the model-parameters $\{\phi_i\}$ on the meta-parameters $(\boldsymbol{\theta})$ shrinks and vanishes as the number of gradient steps in Alg grows, making meta-learning difficult

The old method has this constraint because of back propagation. New method don't have this issue and stays close to the actual parameter chosen by using λ which is a regularization parameter can be scalars, vectors or full matrices. But, in this paper they consider as scalar

$$Alg^{\star}(\boldsymbol{\theta}, \mathcal{D}_i^{\mathrm{tr}}) = \underset{\boldsymbol{\phi}' \in \Phi}{\operatorname{argmin}} \, \mathcal{L}(\boldsymbol{\phi}', \mathcal{D}_i^{\mathrm{tr}}) + \frac{\lambda}{2} ||\boldsymbol{\phi}' - \boldsymbol{\theta}||^2.$$

The proximal regularization term in Eq. 3 encourages ϕ_i to remain close to $\boldsymbol{\theta}$, thereby retaining a strong dependence throughout. The regularization strength (λ) plays a role similar to the learning rate (α) in MAML, controlling the strength of the prior $(\boldsymbol{\theta})$ relative to the data $(\mathcal{D}_{\mathcal{T}}^{\mathrm{tr}})$. Like α, the regularization strength λ may also be learned. Furthermore, both α and λ can be scalars, vectors, or full matrices. For simplicity, we treat λ as a scalar hyperparameter. In Eq. 3, we use $\star$ to denote that

Formulation

Bi-level meta learning problem

$$\overbrace{\boldsymbol{\theta}^*_{\mathrm{ML}} := \operatorname*{argmin}_{\boldsymbol{\theta}\in\Theta} F(\boldsymbol{\theta})}^{\text{outer-level}},$$

$$F(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} \mathcal{L}_i\left(\mathcal{A}lg_i^\star(\boldsymbol{\theta})\right),$$

$$\mathcal{A}lg_i^\star(\boldsymbol{\theta}) := \operatorname*{argmin}_{\boldsymbol{\phi}'\in\Phi} G_i(\boldsymbol{\phi}', \boldsymbol{\theta}), \text{ where } G_i(\boldsymbol{\phi}', \boldsymbol{\theta}) = \hat{\mathcal{L}}_i(\boldsymbol{\phi}') + \frac{\lambda}{2}\,||\boldsymbol{\phi}' - \boldsymbol{\theta}||^2.$$

- θ_{ML} – Best Meta Learning parameter
- F - average validation loss function
- Alg - training algorithm performed on the training set of the particular task while starting from θ
- M – Number of tasks
- T_i – Task
- D_i - Dataset
- D_i^{tr} – Train set
- D_i^{test} – Test set
- Alpha – Learning Rate
- λ - regularization parameter
- Φ_i – model parameters

How does it lead to gradient Descent

- We have to calculate $Df/D\theta$. The conventional approach involved blindly inverting the matrix, which is a computationally expensive process. So, the author suggests not to do that instead multiply the gradient with the invertible matrix.
- η outer step size

$$\frac{d\mathcal{A}lg_i^\star(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \left(\boldsymbol{I} + \frac{1}{\lambda}\,\nabla_{\boldsymbol{\phi}}^2 \hat{\mathcal{L}}_i(\boldsymbol{\phi}_i)\right)^{-1}.$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\,\frac{1}{M}\sum_{i=1}^{M} \frac{d\mathcal{A}lg_i^\star(\boldsymbol{\theta})}{d\boldsymbol{\theta}}\,\nabla_{\phi}\mathcal{L}_i(\mathcal{A}lg_i^\star(\boldsymbol{\theta})).$$

Algorithm

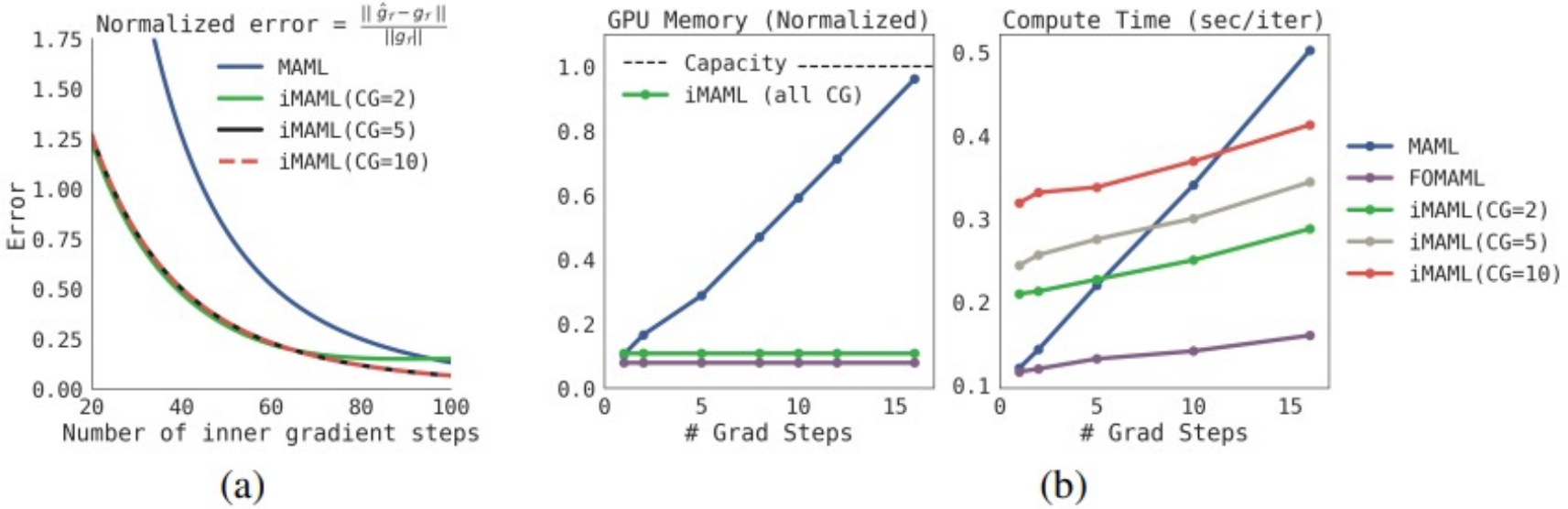
Algorithm 1 Implicit Model-Agnostic Meta-Learning (iMAML)

Require: Distribution over tasks $P(\mathcal{T})$, outer step size η, regularization strength λ,
while not converged **do**
  Sample mini-batch of tasks $\{\mathcal{T}_i\}_{i=1}^{B} \sim P(\mathcal{T})$
  for Each task $\mathcal{T}_i$ **do**
    Compute task meta-gradient $\boldsymbol{g}_i = \texttt{Implicit-Meta-Gradient}(\mathcal{T}_i, \boldsymbol{\theta}, \lambda)$
  end for
  Average above gradients to get $\hat{\nabla} F(\boldsymbol{\theta}) = (1/B) \sum_{i=1}^{B} \boldsymbol{g}_i$
  Update meta-parameters with gradient descent: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \hat{\nabla} F(\boldsymbol{\theta})$ *// (or Adam)*
end while

Algorithm 2 Implicit Meta-Gradient Computation

Input: Task $\mathcal{T}_i$, meta-parameters $\boldsymbol{\theta}$, regularization strength λ
Hyperparameters: Optimization accuracy thresholds δ and δ'
Obtain task parameters $\boldsymbol{\phi}_i$ using iterative optimization solver such that: $\|\boldsymbol{\phi}_i - \mathcal{A}lg_i^\star(\boldsymbol{\theta})\| \leq \delta$
Compute partial outer-level gradient $\boldsymbol{v}_i = \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathcal{T}}(\boldsymbol{\phi}_i)$
Use an iterative solver (e.g. CG) along with reverse mode differentiation (to compute Hessian vector products) to compute $\boldsymbol{g}_i$ such that: $\|\boldsymbol{g}_i - \left(\boldsymbol{I} + \frac{1}{\lambda} \nabla^2 \hat{\mathcal{L}}_i(\boldsymbol{\phi}_i)\right)^{-1} \boldsymbol{v}_i\| \leq \delta'$
Return: $\boldsymbol{g}_i$

Graph


Normalized error = $\frac{\|\hat{g}_r - g_r\|}{\|g_r\|}$
MAML
iMAML(CG=2)
iMAML(CG=5)
iMAML(CG=10)
Error
Number of inner gradient steps
(a)
GPU Memory (Normalized)
Capacity
iMAML (all CG)
Grad Steps
Compute Time (sec/iter)
Grad Steps
MAML
FOMAML
iMAML(CG=2)
iMAML(CG=5)
iMAML(CG=10)
(b)

Summary

In summary, meta-learning with implicit gradients is a promising approach for learning how to learn in scenarios where explicit gradients may not be available. These methods can enable models to quickly adapt to new tasks with minimal data, improving their efficiency and generalization. Also, this has the potential to improve the performance and versatility of autonomous systems, such as self-driving cars or robots

THANK YOU