More on Meta-Learning & Its Connections to Multi-Task Learning

Haoxiang Wang
PhD Candidate
ECE, UIUC
Overview

1. Simplifications of MAML — reduce the cost of 2nd-order optimization
   • First-order MAML & Reptile: 1st-order approximation
   • ANIL: only apply 2nd-order optimization on head
   • iMAML & MetaOptNet — implicit differentiation on 2nd-order gradients

2. A Unified View of the MAML Family
   • Two key variables: inner-loop regularization & optimized layers

3. Connection between Meta-Learning & Multi-Task Learning [ICML’21]
   • Multi-Task Learning is equivalent to a subclass of MAML family
     - Optimization perspective & Neural Tangent Kernel perspective
   • Empirically, Multi-Task Learning can match meta-learning w/ 10x speedup
MAML vs. 1st-order MAML

MAML (2nd order optimization) on \( N \) tasks

**Loss:** \( L(\theta) = \sum_{i=1}^{N} L_i(\theta - \lambda \nabla_\theta L_i(\theta)) = \sum_{i=1}^{N} L_i(\theta'_i) \)

where \( \theta'_i = \theta - \lambda \nabla_\theta L_i(\theta) \)

**Gradient:** \( \nabla_\theta \theta'_i = \nabla_\theta \theta - \lambda \nabla_\theta L_i(\theta) = I - \lambda \nabla^2_\theta L_i(\theta) \)

**Outer-Loop:**

\[
\theta \leftarrow \theta - \eta \sum_{i=1}^{N} \nabla_\theta L_i(\theta'_i) = \theta - \eta \sum_{i=1}^{N} \nabla_\theta \theta'_i \nabla_\theta L_i(\theta'_i) = \theta - \eta \sum_{i=1}^{N} (I - \lambda \nabla^2_\theta L_i(\theta)) \nabla L_i(\theta'_i) \]

1st-order MAML (1st-order optimization) [1]

**Loss:** Same as MAML

**Assumption:** \( \lambda \to 0 \) and/or \( \| \nabla^2_\theta L_i(\theta) \|_F \to 0 \)

**Outer-Loop:** \( \theta \leftarrow \theta - \eta \sum_{i=1}^{N} (I - \lambda \nabla^2_\theta L_i(\theta)) \nabla L_i(\theta'_i) \approx 0 \)

Reptile (1st-order optimization) [2]

**Loss:** the objective is same as 1st-order MAML

**Algorithm:** Update on task-specific losses sequentially.

---

MAML (1-step inner-loop step) vs. iMAML (infinite inner-loop steps)

**MAML** (2nd order optimization) on $N$ tasks

**Loss:**

$$L(\theta) = \sum_{i=1}^{N} L_i(\theta - \lambda \nabla_{\theta} L_i(\theta)) = \sum_{i=1}^{N} L_i(\theta_i)$$

where $\theta_i' = \theta - \lambda \nabla_{\theta} L_i(\theta)$

**Gradient:**

$$\nabla_{\theta} L_i \theta_i' = \nabla_{\theta} L_i - \lambda \nabla_{\theta} L_i(\theta) = I - \lambda \nabla_{\theta}^2 L_i(\theta)$$

**Outer-Loop:**

$$\theta \leftarrow \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} L_i(\theta_i') = \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} \theta_i' \nabla_{\theta_i'} L_i(\theta_i')$$

$$= \theta - \eta \sum_{i=1}^{N} \left( I - \lambda \nabla_{\theta}^2 L_i(\theta) \right) \nabla L_i(\theta_i')$$

**i-MAML** (Implicit Differentiation on MAML) [1]

**Loss:**

$$L(\theta) = \sum_{i=1}^{N} L_i(\theta_i^*)$$

**Inner-Loop Objective (w/ $L_2$ regularization)**

where $\theta_i^* = \arg\min_{\theta'} L_i(\theta') + \frac{\gamma}{2} \| \theta' - \theta \|^2_2$

**Assumption:** $\gamma$ is large s.t. the inner-loop objective is strongly convex

**Gradient:** $\theta_i^*$ is the global minimum of the inner-loop objective, thus

$$\nabla_{\theta} L_i(\theta_i^*) + \gamma (\theta^* - \theta) = 0 \quad \nabla_{\theta}(\cdot)$$

$$\nabla_{\theta} \theta_i^* = I - \frac{1}{\gamma} \nabla_{\theta'}^2 L_i(\theta_i^*) \nabla_{\theta'} \theta_i^* \implies \nabla_{\theta} \theta_i^* = \left( I + \frac{1}{\gamma} \nabla_{\theta'}^2 L_i(\theta_i^*) \right)^{-1}$$

**Outer-Loop:**

$$\theta \leftarrow \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} L_i(\theta_i^*) = \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} \theta_i^* L_i(\theta_i^*)$$

$$= \theta - \eta \sum_{i=1}^{N} \left( 1 + \frac{1}{\gamma} \nabla_{\theta_i^*}^2 L_i(\theta_i^*) \right)^{-1} \nabla_{\theta_i^*} L_i(\theta_i^*)$$

Empirical Benchmark: Few-Shot Image Classification

**Meta-Training**: Given multiple supervised learning tasks (i.e., multi-class classification)

**Meta-Testing**: Given a few labelled samples from several unseen classes, learn a model that can classify these classes correctly

**Empirical Comparison on Few-Shot Image Classification Datasets**

### Omniglot

Neural Net Backbone: Conv-4 (4 Conv Layers of 32 Channels)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>5-way 1-shot</th>
<th>5-way 5-shot</th>
<th>20-way 1-shot</th>
<th>20-way 5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML [15]</td>
<td>98.7 ± 0.4%</td>
<td><strong>99.9 ± 0.1%</strong></td>
<td>95.8 ± 0.3%</td>
<td>98.9 ± 0.2%</td>
</tr>
<tr>
<td>first-order MAML [15]</td>
<td>98.3 ± 0.5%</td>
<td>99.2 ± 0.2%</td>
<td>89.4 ± 0.5%</td>
<td>97.9 ± 0.1%</td>
</tr>
<tr>
<td>Reptile [43]</td>
<td>97.68 ± 0.04%</td>
<td>99.48 ± 0.06%</td>
<td>89.43 ± 0.14%</td>
<td>97.12 ± 0.32%</td>
</tr>
<tr>
<td>iMAML, GD (ours)</td>
<td>99.16 ± 0.35%</td>
<td>99.67 ± 0.12%</td>
<td>94.46 ± 0.42%</td>
<td>98.69 ± 0.1%</td>
</tr>
<tr>
<td>iMAML, Hessian-Free (ours)</td>
<td><strong>99.50 ± 0.26%</strong></td>
<td>99.74 ± 0.11%</td>
<td><strong>96.18 ± 0.36%</strong></td>
<td><strong>99.14 ± 0.1%</strong></td>
</tr>
</tbody>
</table>

### Mini-ImageNet

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>5-way 1-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML</td>
<td>48.70 ± 1.84 %</td>
</tr>
<tr>
<td>first-order MAML</td>
<td>48.07 ± 1.75 %</td>
</tr>
<tr>
<td>Reptile</td>
<td>49.97 ± 0.32 %</td>
</tr>
<tr>
<td>iMAML GD (ours)</td>
<td>48.96 ± 1.84 %</td>
</tr>
<tr>
<td>iMAML HF (ours)</td>
<td>49.30 ± 1.88 %</td>
</tr>
</tbody>
</table>

MAML (inner-loop optimize all layers) vs. ANIL (inner-loop optimize last layer only)

MAML (2\textsuperscript{nd} order optimization) on \( N \) tasks

**Loss:**  
\[
L(\theta) = \sum_{i=1}^{N} L_i \left( \theta - \lambda \nabla_{\theta} L_i(\theta) \right) = \sum_{i=1}^{N} L_i(\theta_i)
\]

where \( \theta_i' = \theta - \lambda \nabla_{\theta} L_i(\theta) \)

**Gradient:**  
\[
\nabla_{\theta} \theta_i' = \nabla_{\theta} \theta - \lambda \nabla_{\theta} L_i(\theta) = I - \lambda \nabla_{\theta}^{2} L_i(\theta)
\]

**Outer-Loop:**
\[
\theta \leftarrow \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} L_i(\theta_i') = \theta - \eta \sum_{i=1}^{N} \nabla_{\theta} \theta_i' \nabla_{\theta_i'} L_i(\theta_i')
\]
\[
= \theta - \eta \sum_{i=1}^{N} \left( I - \lambda \nabla_{\theta}^{2} L_i(\theta) \right) \nabla L_i(\theta_i')
\]

Almost-No-Inner-Loop (ANIL) [1]

**Model:** \( L \)-layer neural net \( f_\theta(x) = w^T \phi_{\theta^{<L}}(x) \), where \( \theta = \{ \theta^{<L}, w \} \) is all parameters, \( \theta^{<L} \) is parameters of first \( L - 1 \) layers, and \( w \) is output layer.

**Loss:**  
\[
L(\theta^{<L}, w) = \sum_{i=1}^{N} L_i \left( \theta^{<L}, w - \lambda \nabla_{w} L_i(\theta^{<L}, w) \right) = \sum_{i=1}^{N} L_i \left( \theta^{<L}, w'_i \right)
\]

where \( w'_i = w - \lambda \nabla_{w} L_i(\theta^{<L}, w) \)

**Gradient:**  
\[
\nabla_w w'_i = \nabla_w w - \lambda \nabla_{w} L_i(\theta^{<L}, w) = I - \lambda \nabla_{w}^{2} L_i(\theta^{<L}, w)
\]

**Outer-Loop:**
\[
\theta^{<L} \leftarrow \theta^{<L} - \eta \sum_{i=1}^{N} \nabla_{\theta^{<L}} L_i(\theta^{<L}, w_i')
\]
\[
w \leftarrow w - \eta \sum_{i=1}^{N} \left( I - \lambda \nabla_{w}^{2} L_i(\theta^{<L}, w) \right) \nabla_{w_i'} L_i(\theta, w_i')
\]

Not updating \( w \) in the outer loop does not affect the performance [2]

---


Empirical Studies on on MAML

The inner loop of MAML does not change the learned representation of hidden layers much — it only significantly change the head (i.e., output layer).

MAML’s inner loop updates have little effect on learned representations from early stages of training.

Empirical Comparison: ANIL vs. MAML

A Unified View of MAML Family (i.e, Gradient-Based Meta-Learning / GBML)

\[
\min_{\theta^{<L}} \left[ \min_{\{w_i^r\}_{i=1}^N} \sum_{i=1}^N L_i(\theta^{<L}, w_i^r) + R(w_i^r) \right]
\]

Implicit Differentiation on ANIL

\[
\min_{\theta^{<L}} \left[ \min_{\{\theta_i^r\}_{i=1}^N} \sum_{i=1}^N L_i(\theta_i^r) + R(\theta_i^r) \right]
\]

Regularization (early stopping or \(\ell_2\))

<table>
<thead>
<tr>
<th>Inner-Loop Optimized Layers</th>
<th>Early Stopping</th>
<th>(\ell_2 ) Regularizer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Layer</td>
<td>ANIL (Raghu et al., 2020)</td>
<td>MetaOptNet (Lee et al., 2019b) R2D2 (Bertinetto et al., 2019)</td>
</tr>
<tr>
<td>All Layers</td>
<td>MAML (Finn et al., 2017)</td>
<td>iMAML (Rajeswaran et al., 2019) Meta-MinibatchProx (Zhou et al., 2019b)</td>
</tr>
</tbody>
</table>

An Optimization Perspective

1st order optimization  1st+2nd order optimization  2nd order optimization

Single Task
- Supervised Learning

Multiple Tasks
- Multi-Task Learning
- ANIL
- MAML
- iMAML
- MetaOptNet

References:
[ICML 2021] Bridging Multi-Task Learning and Meta-Learning: Towards Efficient Training and Effective Adaptation

Haoxiang Wang
PhD Candidate
ECE, UIUC

Han Zhao
Assistant Professor
Computer Science, UIUC

Bo Li
Assistant Professor
Computer Science, UIUC
**Intro to Multi-Task Learning and Meta-Learning**

**Multi-Task Learning (MTL)**

- **Setting:** Test task = Training tasks
- **Goal:** Be a master on a set of tasks
- **Assumption:** The training tasks might be helpful to each other (e.g., riding bicycles → riding horses), thus training a model jointly on all of them might be better than training individual models on each task.

**Meta-Learning**

- **Setting:** Test task \( \notin \) Training tasks
- **Goal:** Adapt to an unseen task quickly.
- **Assumption:** The test task has some shared knowledge (i.e., meta-knowledge) with the training tasks.
Multi-Task Learning
Solve multiple tasks $\mathcal{T}_1, \ldots, \mathcal{T}_T$ at once.

$$\min_{\theta} \sum_{i=1}^{T} \mathcal{L}(\theta, \mathcal{D}_i)$$

Transfer Learning
Solve target task $\mathcal{T}_b$ after solving source task $\mathcal{T}_a$ by transferring knowledge learned from $\mathcal{T}_a$

The Meta-Learning Problem
Given data from $\mathcal{T}_1, \ldots, \mathcal{T}_n$, quickly solve new task $\mathcal{T}_{\text{test}}$

In all settings: tasks must have some shared structure.
Goal: Given a few labelled samples from several unseen classes, learn a model that can classify these classes correctly.

Given 1 example of 5 classes:

- Training data $D_{\text{train}}$
- Test set $X_{\text{test}}$

Classify new examples.
**Meta-Training**: Given multiple supervised learning tasks (i.e., multi-class classification)

**Meta-Testing**: Given a few labelled samples from several unseen classes, learn a model that can classify these classes correctly.
Real-World Applications of Meta-Learning

Adapting to new objects provided demo resulting policy
Yu*, Finn*, Xie, Dasari, Zhang, Abbeel, Levine. One-Shot Imitation from Observing Humans. RSS 2018

Adapting to new terrains & conditions
Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. Learning to Adapt in Dynamic Real-World Environments. ICLR 2019

Adapting to new molecules

<table>
<thead>
<tr>
<th>CHEMBL ID</th>
<th>e-NN</th>
<th>TH sut Ht ALL</th>
<th>TH sut Ht T op</th>
<th>FO-MAML</th>
<th>ANEL</th>
<th>MAML</th>
</tr>
</thead>
<tbody>
<tr>
<td>16421106</td>
<td>0.699</td>
<td>0.675</td>
<td>0.673</td>
<td>0.700</td>
<td>0.707</td>
<td>0.706</td>
</tr>
<tr>
<td>16434909</td>
<td>0.650</td>
<td>0.615</td>
<td>0.608</td>
<td>0.629</td>
<td>0.639</td>
<td>0.641</td>
</tr>
<tr>
<td>16435166</td>
<td>0.533</td>
<td>0.520</td>
<td>0.520</td>
<td>0.553</td>
<td>0.553</td>
<td>0.553</td>
</tr>
<tr>
<td>16435066</td>
<td>0.611</td>
<td>0.607</td>
<td>0.607</td>
<td>0.648</td>
<td>0.648</td>
<td>0.648</td>
</tr>
<tr>
<td>16416513</td>
<td>0.708</td>
<td>0.703</td>
<td>0.703</td>
<td>0.748</td>
<td>0.748</td>
<td>0.748</td>
</tr>
<tr>
<td>16430138</td>
<td>0.607</td>
<td>0.591</td>
<td>0.590</td>
<td>0.714</td>
<td>0.714</td>
<td>0.714</td>
</tr>
<tr>
<td>16439349</td>
<td>0.895</td>
<td>0.881</td>
<td>0.880</td>
<td>0.952</td>
<td>0.952</td>
<td>0.952</td>
</tr>
<tr>
<td>16443423</td>
<td>0.605</td>
<td>0.604</td>
<td>0.604</td>
<td>0.642</td>
<td>0.642</td>
<td>0.642</td>
</tr>
<tr>
<td>16442823</td>
<td>0.679</td>
<td>0.674</td>
<td>0.674</td>
<td>0.756</td>
<td>0.756</td>
<td>0.756</td>
</tr>
<tr>
<td>16467517</td>
<td>0.789</td>
<td>0.785</td>
<td>0.785</td>
<td>0.856</td>
<td>0.856</td>
<td>0.856</td>
</tr>
<tr>
<td>16461156</td>
<td>0.705</td>
<td>0.692</td>
<td>0.691</td>
<td>0.765</td>
<td>0.765</td>
<td>0.765</td>
</tr>
</tbody>
</table>

Nguyen et al. Meta-Learning GNN Initializations for Low-Resource Molecular Property Prediction. 2020

Adapting to new regions of the world

Multi-Task Learning (MTL) vs. Gradient-Based Meta-Learning (GBML)

**Multi-Task Learning (MTL)**

*Multi-Head Structure*

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$w_2$</td>
<td>$w_3$</td>
</tr>
</tbody>
</table>

- Task-Specific Last Layers (Heads)
- Share Hidden Layers (Body)

**Training Objective:**

1. **1st-order optimization** (a form of Empirical Risk Minimization) → **Efficient Training**

Loss: $L(\theta) = \sum_{i=1}^{n} L_i(\theta)$

Optimization: $\theta = \theta - \sum_i \nabla L_i(\theta)$

**Gradient-Based Meta-Learning (GBML)**

*Single-Head Structure*

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Unseen Test Task**

**Training Objective:**

2. **2nd order optimization** (e.g., MAML, MetaOptNet, ANIL, iMAML) → **Expensive Training**

Loss: $L(\theta) = \sum_{i=1}^{n} L_i(\theta - \lambda \nabla L_i(\theta))$

Optimization:

\[
\theta = \theta - \sum_i \left( I - \lambda \nabla^2 L_i(\theta) \right) \nabla L_i(\theta')
\]

where $\theta'_i = \theta - \lambda \nabla L_i(\theta)$

- **Adapt to unseen tasks**
- **No adaptation to unseen tasks**
Motivation: 
Can we combine the best of both worlds from multi-task learning and meta-learning, i.e., effective adaptation to unseen tasks with efficient training? 

Our answer: Yes!

Contribution: 
Our paper bridges Multi-Task Learning (MTL) and Gradient-Based Meta-Learning (GBML) by theoretical and empirical studies.
Multi-Task Learning for Unseen Tasks by **Fine-Tuning Last Layer**

**Fine-tuning**: For a trained MTL model, we can adapt it to an unseen test task by

1. Randomly initialize a new head
2. Fine-tune the head on a few labelled data of the test task
3. Use the fine-tuned head for predictions on the new task
Gradient-Based Meta-Learning: Similarity to Multi-Task Learning

**Step I:** Obtain task-specific transient heads by gradient descent on each task

\[ w_i = w - \nabla_w L_i(\phi, w) \]

**Step II:** Use the transient heads to give predictions on each task.

<table>
<thead>
<tr>
<th>Inner-Loop Optimized Layers</th>
<th>Early Stopping</th>
<th>$\ell_2$ Regularizer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Last Layer</strong></td>
<td>ANIL (Raghu et al., 2020)</td>
<td>MetaOptNet (Lee et al., 2019b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R2D2 (Bertinetto et al., 2019)</td>
</tr>
<tr>
<td><strong>All Layers</strong></td>
<td>MAML (Finn et al., 2017)</td>
<td>iMAML (Rajeswaran et al., 2019)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Meta-MinibatchProx (Zhou et al., 2019b)</td>
</tr>
</tbody>
</table>
Equivalence in Optimization Objective: MTL $\approx$ a Subclass of MAML Family

**Multi-Task learning:**
$$\min_{\theta^{<L}, \{w'_i\}_{i=1}^N} \sum_{i=1}^N L_i(\theta^{<L}, w'_i) + R(w'_i)$$

**ANIL/MetaOptNet/R2D2:**
$$\min_{\{w'_i\}_{i=1}^N} \sum_{i=1}^N L_i(\theta^{<L}, w'_i) + R(w'_i)$$

- Usually $= 0$ (no regularization)
- Inner-Loop
- Regularization (early stopping or $\ell_2$)

<table>
<thead>
<tr>
<th>Inner-Loop Optimized Layers</th>
<th>Early Stopping</th>
<th>$\ell_2$ Regularizer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Last Layer</strong></td>
<td>ANIL (Raghu et al., 2020)</td>
<td>MetaOptNet (Lee et al., 2019b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R2D2 (Bertinetto et al., 2019)</td>
</tr>
<tr>
<td><strong>All Layers</strong></td>
<td>MAML (Finn et al., 2017)</td>
<td>iMAML (Rajeswaran et al., 2019)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Meta-MinibatchProx (Zhou et al., 2019b)</td>
</tr>
</tbody>
</table>

### Theoretical Results

<table>
<thead>
<tr>
<th>Inner-Loop Optimized Layers</th>
<th>Early Stopping</th>
<th>$\ell_2$ Regularizer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Layer</td>
<td>ANIL (Raghu et al., 2020)</td>
<td>MetaOptNet (Lee et al., 2019b) R2D2 (Bertinetto et al., 2019)</td>
</tr>
<tr>
<td>All Layers</td>
<td>MAML (Finn et al., 2017)</td>
<td>iMAML (Rajeswaran et al., 2019) Meta-MinibatchProx (Zhou et al., 2019b)</td>
</tr>
</tbody>
</table>

**Equivalence:** MTL and a class of GBML shares the **same optimization objective**

**Difference:** MTL uses *joint training*, while GBML adopts *bi-level optimization with regularization*

**Closeness in the function space:**

- We compare neural nets trained by ANIL (a MAML simplification) and MTL in an NTK-based meta-learning framework [1]
- We prove that, on any test task, the difference between predictions is upper bounded as

$$
\|\text{ANIL prediction} - \text{MTL prediction}\|_2 \leq O(\lambda \tau + \frac{1}{L})
$$

where

- $\lambda, \tau$: Learning rates
- $L$: Network depth

---

Empirical Validation of Theoretical Results

**Setting:** Synthetic dataset of Gaussian data points for few-shot learning.

\[ \| \text{ANIL prediction} - \text{MTL prediction} \|_2 \leq O(\lambda \tau + \frac{1}{L}) \]

\[ L: \text{Network depth} \]

\[ \lambda, \tau: \text{Learning rates} \]

Experiments on **Few-Shot Learning**

**Benchmarks**: 4 popular few-shot learning datasets, extracted from ImageNet and CIFAR-100.

- mini-ImageNet
- tiered-ImageNet
- CIFAR-FS
- FC-100

**Remarks**: The number of unique training tasks is quite large (due to combinatorial explosion), e.g., it’s 4.3 billions for tiered-ImageNet. Thus, we cannot afford an individual head for each training task.
**Few-Shot Learning Setup**

**Training Set:** a bunch of N-way classification tasks

**Test Set:** an unseen N-way classification tasks with only a few labelled samples (i.e., K-shot)
Example: 5-way few-shot classification; Each task has 5 task-specific classes drawn from 10 base classes.

Objective: $L(θ) = \sum_{i=1}^{n} L_i(θ)$

Shared Hidden Layers
Experimental Results on **Few-Shot Learning**


**MTL-ours**: Our memory-efficient implementation of multi-task learning.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Architecture</th>
<th>mini-ImageNet</th>
<th></th>
<th>tiered-ImageNet</th>
<th>CIFAR-FS</th>
<th>FC100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1-shot (%)</td>
<td>5-shot (%)</td>
<td>1-shot (%)</td>
<td>5-shot (%)</td>
<td>1-shot (%)</td>
</tr>
<tr>
<td>MAML [Finn et al., 2017a]</td>
<td>CNN-4</td>
<td>48.70 ± 1.84</td>
<td>63.11 ± 0.92</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MetaOptNet [Lee et al., 2019]</td>
<td>ResNet-12</td>
<td><strong>62.64 ± 0.61</strong></td>
<td><strong>78.63 ± 0.46</strong></td>
<td>65.99 ± 0.72</td>
<td>81.56 ± 0.53</td>
<td><strong>72.0 ± 0.7</strong></td>
</tr>
<tr>
<td>MTL-ours [Wang et al., 2021]</td>
<td>ResNet-12</td>
<td>59.84 ± 0.22</td>
<td><strong>77.72 ± 0.09</strong></td>
<td>67.11 ± 0.12</td>
<td><strong>83.69 ± 0.02</strong></td>
<td>69.5 ± 0.3</td>
</tr>
</tbody>
</table>

**Multi-task learning can match the SOTA of gradient-based meta-learning on few-shot learning benchmarks!**
**Training Efficiency** of Multi-Task Learning vs. Gradient-Based Meta-Learning

<table>
<thead>
<tr>
<th></th>
<th>Test Accuracy</th>
<th>GPU Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>MetaOptNet</td>
<td>78.63%</td>
<td>85.6 hrs</td>
</tr>
<tr>
<td>MTL</td>
<td>77.72%</td>
<td>3.7 hrs</td>
</tr>
</tbody>
</table>

**Mini-ImageNet (5-way 5 shot)**

*Multi-task learning* can be more than 10x faster, since it does not use any 2\(^{nd}\) order optimization.
An Implicit Indication: as $L \to \infty$, $\lambda \tau \to 0$, MTL/Meta-Learning $\approx$ Supervised

$$\|\text{ANIL prediction} - \text{MTL prediction}\|_2 \leq O(\lambda \tau + \frac{1}{L})$$

$\lambda$, $\tau$: Learning rates
$L$: Network depth

1st order optimization 1st+2nd order optimization 2nd order optimization

Single Task

Multiple Tasks

### Empirical Evidence (for Deep NN, supervised learning is optimal)

<table>
<thead>
<tr>
<th>model</th>
<th>backbone</th>
<th>minImageNet 5-way</th>
<th>tieredImageNet 5-way</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>MAML [12]</td>
<td>32-32-32-32</td>
<td>48.70 ± 1.84</td>
<td>63.11 ± 0.92</td>
</tr>
<tr>
<td>Matching Networks [54]</td>
<td>64-64-64-64</td>
<td>43.56 ± 0.84</td>
<td>55.31 ± 0.73</td>
</tr>
<tr>
<td>IMP [2]</td>
<td>64-64-64-64</td>
<td>49.2 ± 0.7</td>
<td>64.7 ± 0.7</td>
</tr>
<tr>
<td>Prototypical Networks^ [46]</td>
<td>64-64-64-64</td>
<td>49.42 ± 0.78</td>
<td>68.20 ± 0.66</td>
</tr>
<tr>
<td>TAML [21]</td>
<td>64-64-64-64</td>
<td>51.77 ± 1.86</td>
<td>66.05 ± 0.85</td>
</tr>
<tr>
<td>SAML [15]</td>
<td>64-64-64-64</td>
<td>52.22 ± n/a</td>
<td>66.49 ± n/a</td>
</tr>
<tr>
<td>GCR [27]</td>
<td>64-64-64-64</td>
<td>53.21 ± 0.80</td>
<td>72.34 ± 0.64</td>
</tr>
<tr>
<td>KTN(Visual) [35]</td>
<td>64-64-64-64</td>
<td>54.61 ± 0.80</td>
<td>71.21 ± 0.66</td>
</tr>
<tr>
<td>PARN [59]</td>
<td>64-64-64-64</td>
<td>55.22 ± 0.84</td>
<td>71.55 ± 0.66</td>
</tr>
<tr>
<td>Dynamic Few-shot [14]</td>
<td>64-64-128-128</td>
<td>56.20 ± 0.86</td>
<td>73.00 ± 0.64</td>
</tr>
<tr>
<td>Relation Networks [48]</td>
<td>64-96-128-256</td>
<td>50.44 ± 0.82</td>
<td>65.32 ± 0.70</td>
</tr>
<tr>
<td>R2D2 [3]</td>
<td>96-192-384-512</td>
<td>51.2 ± 0.6</td>
<td>68.8 ± 0.1</td>
</tr>
<tr>
<td>SNAIL [29]</td>
<td>ResNet-12</td>
<td>55.71 ± 0.99</td>
<td>68.88 ± 0.92</td>
</tr>
<tr>
<td>AdaResNet [32]</td>
<td>ResNet-12</td>
<td>56.88 ± 0.62</td>
<td>71.94 ± 0.57</td>
</tr>
<tr>
<td>TADAM [34]</td>
<td>ResNet-12</td>
<td>58.50 ± 0.30</td>
<td>76.70 ± 0.30</td>
</tr>
<tr>
<td>Shot-Free [41]</td>
<td>ResNet-12</td>
<td>59.04 ± n/a</td>
<td>77.64 ± n/a</td>
</tr>
<tr>
<td>TEWAM [37]</td>
<td>ResNet-12</td>
<td>60.07 ± n/a</td>
<td>75.90 ± n/a</td>
</tr>
<tr>
<td>MTIL [47]</td>
<td>ResNet-12</td>
<td>61.20 ± 1.80</td>
<td>75.30 ± 0.80</td>
</tr>
<tr>
<td>Variational FSL [62]</td>
<td>ResNet-12</td>
<td>61.23 ± 0.26</td>
<td>77.69 ± 0.17</td>
</tr>
<tr>
<td>MetaOptNet [26]</td>
<td>ResNet-12</td>
<td>62.64 ± 0.61</td>
<td>78.63 ± 0.46</td>
</tr>
<tr>
<td>Diversity w/ Cooperation [11]</td>
<td>ResNet-18</td>
<td>59.48 ± 0.65</td>
<td>75.62 ± 0.48</td>
</tr>
<tr>
<td>Fine-tuning [9]</td>
<td>WRN-28-10</td>
<td>57.73 ± 0.62</td>
<td>78.17 ± 0.49</td>
</tr>
<tr>
<td>LEO-trainval [44]</td>
<td>WRN-28-10</td>
<td>61.76 ± 0.08</td>
<td>77.59 ± 0.12</td>
</tr>
<tr>
<td>Ours-simple</td>
<td>ResNet-12</td>
<td>62.02 ± 0.63</td>
<td>79.64 ± 0.44</td>
</tr>
<tr>
<td>Ours-distill</td>
<td>ResNet-12</td>
<td>64.82 ± 0.60</td>
<td>82.14 ± 0.43</td>
</tr>
</tbody>
</table>

Table 1. Comparison to prior work on minImageNet and tieredImageNet. Average few-shot classification accuracies (%) with 95% confidence intervals on minImageNet and tieredImageNet meta-test splits. Results reported with input image size of 84x84. a-b-c-d denotes a 4-layer convolutional network with a, b, c, and d filters in each layer. \^ results obtained by training on the union of training and validation sets.

Takeaway: We can combine the benefits of multi-task learning and meta-learning, i.e., effective adaptation to unseen tasks with efficient training.

Code: https://github.com/AI-secure/multi-task-learning

Contact Information:
- Haoxiang Wang: hwang264@illinois.edu
- Han Zhao: hanzhao@illinois.edu
- Bo Li: lbo@illinois.edu