

Inexact-ADMM Based Federated Meta-Learning for Fast and Continual Edge Learning

MobiHoc '21: Proceedings of the Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing July (ACM 2021)

Yue, Sheng and Ren, Ju and Xin, Jiang and Lin, Sen and Zhang, Junshan
Reporter: Fengjiao Gong

November 2, 2023

Outline

1. Problem Formulation
2. Inexact-ADMM Based Algorithm
3. Performance Analysis
4. Experiments

Problem Formulation

- ▶ **Continual Learning** (also known as *Incremental Learning*, *Life-long Learning*) is a concept to learn a model for a large number of tasks sequentially *without forgetting knowledge obtained from the preceding tasks*, where the data in the old tasks are not available anymore during training new ones.
- ▶ **Edge Learning** Edge learning is a subset of artificial intelligence (AI) in which processing takes place on-device, or “at the edge” of where the data originates, using a pre-trained set of algorithms. In order to meet the requirements for performance, safety, and latency in many IoT applications, intelligent decisions must be made right here right now at the network edge.

Problem Formulation

- ▶ **Meta-Learning** Meta-learning refers to machine learning algorithms that learn from the output of other machine learning algorithms. It trains the model's initial parameters over many tasks, such that the pre-trained model, and can achieve maximal performance on a new task after quick adaptation using only a small amount of data .
- ▶ **Federated Meta-Learning** All source edge nodes collaboratively learn a global model initialization such that maximal performance can be obtained with the model parameters updated with only a few data samples at the target edge node, thereby achieving real-time edge intelligence.

Continual Edge learning via federated meta-learning

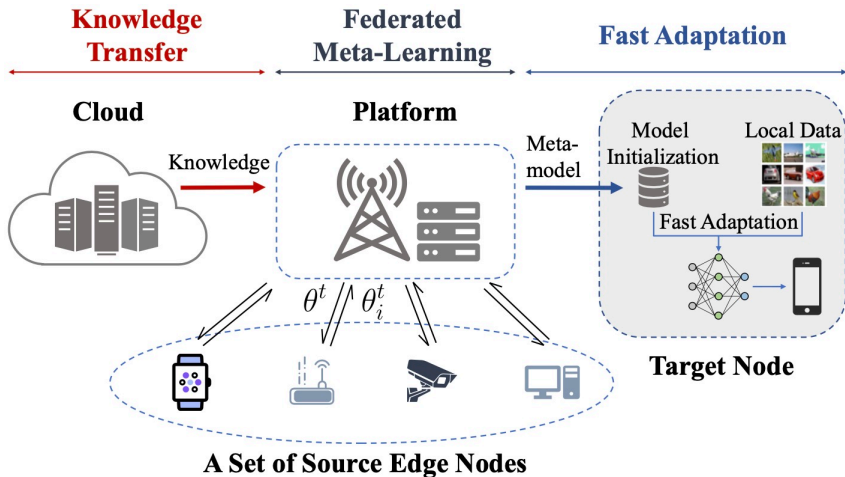


Figure 1: The platform-aided federated meta-learning with knowledge transfer

Notation

Consider supervised learning, for node $i \in \mathcal{I} \cup \{m\}$

- ▶ Data Sample $(\mathbf{x}_i^j, \mathbf{y}_i^j) \in \mathcal{X}_i \times \mathcal{Y}_i \sim \text{distribution } P_i$
- ▶ Dataset $\mathcal{D}_i = \left\{ (\mathbf{x}_i^j, \mathbf{y}_i^j) \right\}_{j=1}^{D_i}$
- ▶ Model Parameter $\phi_i \in \mathbb{R}^n$
- ▶ Empirical Loss function

$$L_i(\phi_i, \mathcal{D}_i) \triangleq (1/D_i) \sum_{j=1}^{D_i} l_i(\phi_i, (\mathbf{x}_i^j, \mathbf{y}_i^j))$$

where $l_i(\cdot, \cdot)$ is a general differentiable non-convex loss function.

Problem Formulation — Federated Meta-Learning

$$\begin{aligned} \min_{\theta} \quad & \sum_{i \in \mathcal{I}} w_i L_i(\phi_i(\theta), \mathcal{D}_i^q) + \lambda D_h(\theta, \theta_p) \\ \text{s.t.} \quad & \phi_i(\theta) = \theta - \alpha \nabla L_i(\theta, \mathcal{D}_i^s), i \in \mathcal{I} \end{aligned} \tag{1}$$

where

- ▶ dataset \mathcal{D}_i is divided as

support set $\mathcal{D}_i^s \cup$ query set $\mathcal{D}_i^q, \forall i \in \mathcal{I}$

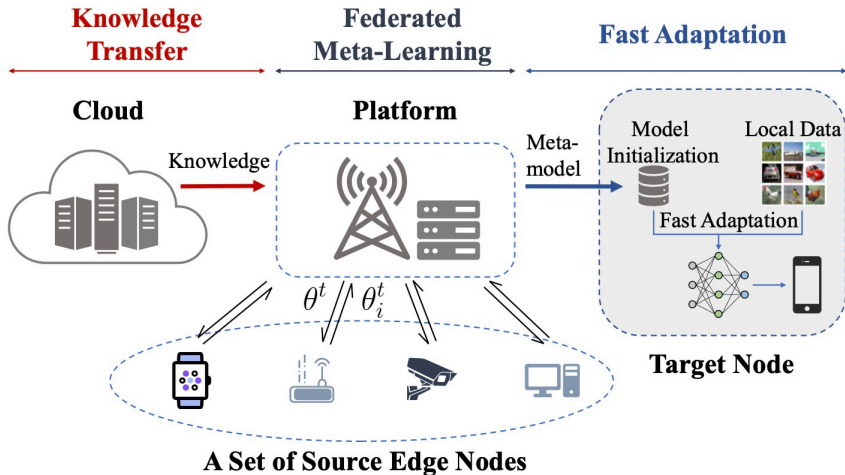
- ▶ Bregman divergence about prior model parameter $\theta_p \in \mathbb{R}^n$

$$D_h(\theta, \theta_p) \triangleq h(\theta) - h(\theta_p) - \langle \nabla h(\theta_p), \theta - \theta_p \rangle \tag{2}$$

- ▶ α is learning rate, λ is a penalty parameter, and $w_i \triangleq D_i / \sum_{i \in \mathcal{I}} D_i$.

We aim to find **a good meta-model** such that *slight updating, i.e., **one-step gradient descent**, results in substantial performance improvement* for any task across the edge nodes.

Continual Edge learning via federated meta-learning



Problem Formulation — Fast Adaptation

the platform $\xrightarrow[\text{meta-model } \theta]{\text{transfer}}$ target node $\xrightarrow[\text{one-step SGD}]{\text{local dataset } \mathcal{D}_m^s}$ new model ϕ_m

$$\phi_m = \theta - \alpha \nabla L_m(\theta, \mathcal{D}_m^s) \quad (3)$$

Algorithm for Federated Meta-Learning

- ▶ Conventional ADMM
- ▶ Inexact-ADMM: **ADMM-FedMeta**

Conventional ADMM

For the constrained optimization problem

$$\begin{aligned} \min_{\theta} \quad & \sum_{i \in \mathcal{I}} w_i L_i(\phi_i(\theta), \mathcal{D}_i^q) + \lambda D_h(\theta, \theta_p) \\ \text{s.t.} \quad & \phi_i(\theta) = \theta - \alpha \nabla L_i(\theta, \mathcal{D}_i^s), i \in \mathcal{I} \end{aligned} \tag{1}$$

Introduce another variable θ_i , (1) is **equivalent to**

$$\begin{aligned} \min_{\{\theta_i\}, \theta} \quad & \sum_{i \in \mathcal{I}} w_i L_i(\phi_i(\theta_i), \mathcal{D}_i^q) + \lambda D_h(\theta, \theta_p) \\ \text{s.t.} \quad & \theta_i - \theta = 0, i \in \mathcal{I} \end{aligned} \tag{4}$$

Conventional ADMM

For the constrained optimization problem

$$\begin{aligned} \min_{\{\theta_i\}, \theta} \quad & \sum_{i \in \mathcal{I}} w_i L_i(\phi_i(\theta_i), \mathcal{D}_i^q) + \lambda D_h(\theta, \theta_p) \\ \text{s.t.} \quad & \theta_i - \theta = 0, i \in \mathcal{I} \end{aligned} \tag{4}$$

Construct the **augmented Lagrangian function** as follows:

$$\begin{aligned} \mathcal{L}(\{\theta_i, y_i\}, \theta) \triangleq \quad & \sum_{i \in \mathcal{I}} \left(w_i L_i(\phi_i(\theta_i), \mathcal{D}_i^q) + \langle y_i, \theta_i - \theta \rangle \right. \\ & \left. + \frac{\rho_i}{2} \|\theta_i - \theta\|^2 \right) + \lambda D_h(\theta, \theta_p) \end{aligned} \tag{5}$$

where $y_i \in \mathbb{R}^n$ is a dual variable and $\rho_i > 0$ is a penalty parameter for each $i \in \mathcal{I}$.

Conventional ADMM

When the classical ADMM method is applied, the variables θ_i, θ and y_i are **updated alternatively** in solving (5) as follows:

$$\begin{cases} \theta^{t+1} = \arg \min_{\theta} \mathcal{L}(\{\theta_i^t, y_i^t\}, \theta) \\ \theta_i^{t+1} = \arg \min_{\theta_i} \mathcal{L}_i(\theta_i, y_i^t, \theta^{t+1}) \\ y_i^{t+1} = y_i^t + \rho_i (\theta_i^{t+1} - \theta^{t+1}) \end{cases} \quad (6)$$

where

$$\mathcal{L}_i(\theta_i, y_i, \theta) \triangleq w_i L_i(\phi_i(\theta_i), \mathcal{D}_i^q) + \langle y_i, \theta_i - \theta \rangle + \frac{\rho_i}{2} \|\theta_i - \theta\|^2 \quad (7)$$

To **decouple** the regularizer from the edge nodes,

1. *updating θ at the platform* — **Global**
2. *updating $\{\theta_i, y_i\}$ at the source edge nodes in a distributed manner* — **Local**

Inexact-ADMM: ADMM-FedMeta

(1) Local update of $\{\theta_i, y_i\}$

- Update node-specific model ϕ_i

$$\phi_i^t = \theta^t - \alpha \nabla L_i(\theta^t, \mathcal{D}_i^s) \quad (8)$$

- Update local parameter θ_i

$$\begin{aligned} \theta_i^t = \arg \min_{\theta_i} \{ & w_i L_i(\phi_i(\theta_i), \mathcal{D}_i^q) + \langle y_i^{t-1}, \theta_i - \theta^t \rangle \\ & + \frac{\rho_i}{2} \|\theta_i - \theta^t\|^2 \}. \end{aligned} \quad (9)$$

- Update local dual variable y_i

$$y_i^t = y_i^{t-1} + \rho_i(\theta_i^t - \theta^t) \quad (10)$$

Inexact-ADMM: ADMM-FedMeta

(2) Global Aggregation towards Meta-Model θ

- Global update of θ

$$\theta^{t+1} = \frac{\sum_{i \in \mathcal{I}} (y_i^t + \rho_i \theta_i^t) - \lambda \nabla_{\theta^t} D_h(\theta^t, \theta_p)}{\sum_{i \in \mathcal{I}} \rho_i} \quad (11)$$

To simplify the computation reduce the *computational cost*, this paper relaxed the subproblem (9).

Inexact-ADMM: ADMM-FedMeta

- ▶ Linear approximation (i.e., first-order Taylor expansion) around θ^t

$$\begin{aligned}\theta_i^t = & \arg \min_{\theta_i} \left\{ w_i L_i(\phi_i^t, \mathcal{D}_i^q) \right. \\ & + \langle w_i (I - \alpha \nabla^2 L_i(\theta^t, \mathcal{D}_i^s)) \nabla L_i(\phi_i^t, \mathcal{D}_i^q) \\ & \left. + y_i^{t-1}, \theta_i - \theta^t \rangle + \frac{\rho_i}{2} \|\theta_i - \theta^t\|^2 \right\},\end{aligned}\tag{12}$$

- ▶ First-order estimator of the Hessian-gradient product

$$g_i^t \triangleq \frac{\nabla L_i(\theta^t + \delta_{i,t} r_i^t, \mathcal{D}_i^s) - \nabla L_i(\theta^t - \delta_{i,t} r_i^t, \mathcal{D}_i^s)}{2\delta_{i,t}}\tag{13}$$

where $r_i^t \triangleq \nabla L_i(\phi_i^t, \mathcal{D}_i^q)$ and $\delta_{i,t} > 0$ is the degree of freedom capturing the estimation accuracy.

[reference] On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In International Conference on Artificial Intelligence and Statistics. 1082–1092. 2020

Inexact-ADMM: ADMM-FedMeta

Algorithm 1: Inexact-ADMM Based Meta-Learning Algorithm (ADMM-FedMeta)

Input: $\theta_p, \alpha, \lambda, \rho_i, \mathcal{D}_i^s, \mathcal{D}_i^q$ for $i \in \mathcal{I}$

Output: Final meta-model θ

- 1 Each edge node $i \in \mathcal{I}$ initializes y_i^{-1} ;
 - 2 Platform initializes θ^0 and sends it to all edge nodes;
 - 3 **for** $t = 0$ **to** T **do**
 - 4 **for** $i = 1$ **to** I **do**
 - 5 Compute $\phi_i^t \leftarrow \theta^t - \alpha \nabla L_i(\theta^t, \mathcal{D}_i^s)$;
 - 6 Compute θ_i^t by (14);
 - 7 Compute $y_i^t \leftarrow y_i^{t-1} + \rho_i(\theta_i^t - \theta^t)$;
 - 8 Send θ_i^t and y_i^t back to the platform;
 - 9 **end**
 - 10 Platform updates θ^{t+1} by (11) and sends it to all edge nodes $i \in \mathcal{I}$;
 - 11 **end**
 - 12 Platform transfers θ^T to target node for fast adaptation;
-

Update θ_i :

$$\theta_i^t = \theta^t - \frac{y_i^{t-1} + w_i (\nabla L_i(\phi_i^t, \mathcal{D}_i^q) - \alpha g_i^t)}{\rho_i} \quad (14)$$

Computation complexity $O(n)$

Performance Analysis

1. Convergence Analysis
2. Performance of Rapid Adaptation at Target Node
3. Forgetting effect to Prior Knowledge

Convergence Analysis

Theorem 1 (*Convergence and Communication Complexity*).

1. $\{\theta^t\}$ has at least one limit point and each limit point θ^* is a stationary solution of (1), i.e., $\|\nabla F(\theta^*)\| = 0$.
2. Algorithm 1 finds an ϵ -FOSP of (1) after at most $O(1/\epsilon^2)$ communication rounds.

Performance of Rapid Adaptation at Target Node

Theorem 2 (Fast Adaptation Performance).

$$\begin{aligned} \mathbb{E} \{ \|\nabla F_m(\theta_\epsilon)\| \} \leq & \epsilon + \alpha\beta_m \sum_{i \in \mathcal{I}} w_i \psi_i^h + (\alpha\mu + 1)^2 \sum_{i \in \mathcal{I}} w_i \psi_i^g \\ & + \alpha\mu(\alpha\mu + 1) \sum_{i \in \mathcal{I}} w_i \sigma_i^g \left(\frac{1}{\sqrt{D_m^q}} + \frac{1}{\sqrt{D_i^q}} \right) \\ & + (\alpha\mu + 1) \sum_{i \in \mathcal{I}} w_i \sigma_i^g \left(\frac{1}{\sqrt{D_m^s}} + \frac{1}{\sqrt{D_i^s}} \right) \\ & + \alpha\beta_m \sum_{i \in \mathcal{I}} w_i \sigma_i^h \left(\frac{1}{\sqrt{D_m^s}} + \frac{1}{\sqrt{D_i^s}} \right) \end{aligned} \tag{15}$$

where

$$\begin{aligned} F_m(\theta) &\triangleq L_m(\theta - \alpha \nabla L_m(\theta, \mathcal{D}_m^s), \mathcal{D}_m^q) + \lambda D_h(\theta, \theta_p), \forall \mathcal{D}_m^s, \mathcal{D}_m^q \sim P_m \\ \mu &= \max_{i \in \mathcal{I}} \{\mu_i\} \end{aligned}$$

Forgetting effect to Prior Knowledge



Theorem 3. Suppose that $L_p(\cdot)$ is μ_p -smooth, $\|\nabla L_p(\theta_p)\| \leq \epsilon_p$ for some $\epsilon_p > 0$, and $D_h(\theta, \theta_p) = \|\theta - \theta_p\|^2$. For the ϵ -FOSP solution θ_ϵ , we have the following result:

$$\|\nabla L_p(\theta_\epsilon)\| \leq \epsilon_p + \frac{v_p}{2\lambda} \left(\epsilon + \sum_{i \in \mathcal{I}} w_i \left(\beta_i + \frac{\alpha \mu_i \sigma_i^g}{\sqrt{D_i^s}} + \frac{\sigma_i^g}{\sqrt{D_i^q}} \right) \right). \quad (16)$$

Experiments

- ▶ **Algorithm** — ADMM-FedMeta
 1. FedAvg
 2. Per-FedAvg
- ▶ **Regularization** — Knowledge transfer
 1. Forgetting effect
 2. Prior knowledge impact

Baselines

-  Fallah, A., Mokhtari, A., and Ozdaglar, A. E.
Personalized federated learning: A meta-learning approach.
[ArXiv abs/2002.07948 \(2020\)](#).
-  McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y.
Communication-Efficient Learning of Deep Networks from Decentralized Data.

In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (20–22 Apr 2017), A. Singh and J. Zhu, Eds., vol. 54 of Proceedings of Machine Learning Research, PMLR, pp. 1273–1282.

► FedAvg[1]

$$\min_{w \in \mathbb{R}^d} f(w) := \frac{1}{n} \sum_{i=1}^n f_i(w)$$

► Per-FedAvg [2]

$$\min_{w \in \mathbb{R}^d} F(w) := \frac{1}{n} \sum_{i=1}^n f_i(w - \alpha \nabla f_i(w))$$

Fashion-MNIST, CIFAR-10, and CIFAR-100

- ▶ Each node has samples from only two random classes.
- ▶ The number of samples per node follows a discrete uniform distribution, i.e., $D_i \sim U(a, b)$ for $i \in \mathcal{I}$.
- ▶ Randomly select 80% and 20% nodes as the source nodes and the target nodes respectively.
- ▶ For each node, divide the local dataset into a support set and a query set (i.e., \mathcal{D}_i^s and \mathcal{D}_i^q), each with 50% of the local data.

Results

Dataset	# local updates	FedAvg	Per-FedAvg	ADMM-FedMeta
Fashion-MNIST	1	83.99% \pm 3.90%	87.55% \pm 2.42%	95.69%\pm0.37%
	5	88.86% \pm 1.57%	89.65% \pm 3.26%	N/A
	10	85.29% \pm 1.93%	90.95% \pm 2.71%	N/A
CIFAR-10	1	41.97% \pm 1.33%	60.53% \pm 1.12%	74.61%\pm2.19%
	5	56.58% \pm 2.27%	65.93% \pm 9.97%	N/A
	10	56.58% \pm 1.15%	67.43% \pm 0.99%	N/A
CIFAR-100	1	42.35% \pm 1.55%	48.19% \pm 2.18%	63.56%\pm0.87%
	5	50.00% \pm 1.09%	49.56% \pm 1.09%	N/A
	10	49.97% \pm 1.04%	48.73% \pm 1.23%	N/A

Figure 2: Comparison of the accuracy on target nodes of different algorithms.

Results

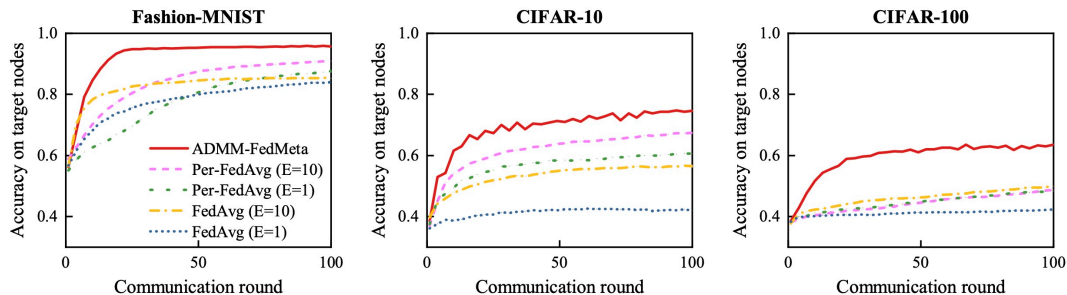


Figure 3: Comparison of the convergence speed of different algorithms.

Results

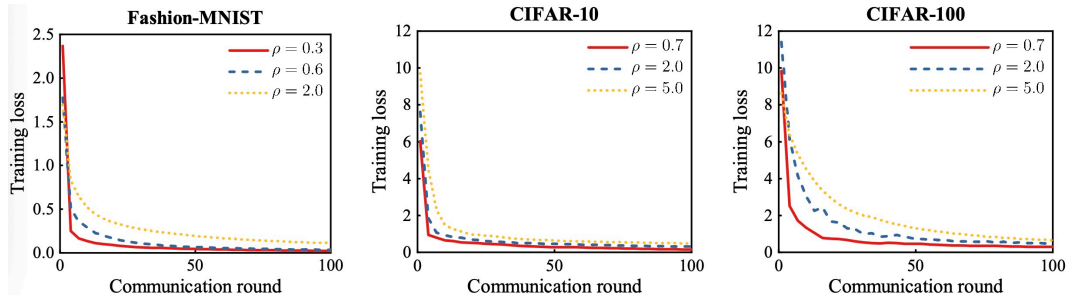


Figure 4: Impact of penalty parameter ρ .

Results

Dataset	Task	Prior Model	FedAvg	Per-FedAvg	ADMM-FedMeta
Fashion-MNIST	Prior	95.63%	41.27%	49.60%	92.86%
	New	49.21%	94.05%	94.84%	94.04%
CIFAR-10	Prior	75.74%	41.08%	42.08%	62.38%
	New	17.33%	55.45%	47.03%	71.29%
CIFAR-100	Prior	66.27%	35.32%	37.62%	59.52%
	New	45.63%	40.48%	57.92%	63.10%

Figure 5: Comparison of the accuracy on prior and current tasks.

- Pre-train a model θ_p with satisfactory performance on the data of the first 50% number of classes as a prior task
- Use θ_p as the initialization to train the meta-models on data of the last 50% number of classes as a new task

Results

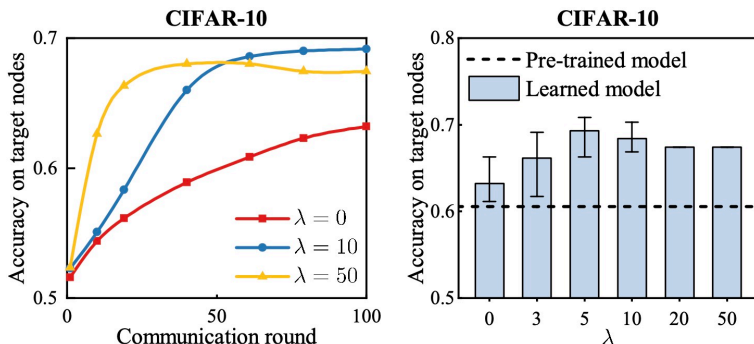


Figure 6: Impact of λ and θ_p .

- Pre-train a prior model as θ_p using images of 3-10 classes on CIFAR-10
- Train the meta-model θ on source nodes with images of 1-8 classes
- Test the accuracy on the target nodes with all 1-10 classes

Conclusion

This paper presents an inexact-ADMM based federated meta-learning approach for fast and continual edge learning.

- ▶ proposed a platform-aided federated meta-learning architecture enabling edge nodes to collaboratively learn a meta-model with the knowledge transfer of previous tasks.
- ▶ cast the federated meta-learning problem as a regularized optimization problem, with the previous knowledge as regularization.
- ▶ developed a variant of inexact-ADMM method that reduces the computational cost per round to $O(n)$.

Thanks!