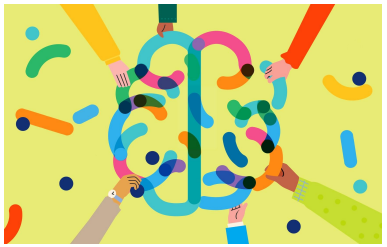
VECTOR INSTITUTE



SIMONS INSTITUTE for the Theory of Computing

The many faces of heterogeneity: Federated, Continual, & Modular Learning

Marco Ciccone

Distinguished Postdoctoral Fellow

Simons Workshop on Learning from Heterogeneous Sources

Works done in collaboration with

Debora

Eros

Riccardo

Leonardo




and Colin, Tatiana, Barbara, Massi and Carlo





Disclaimer

I am not an optimization researcher.

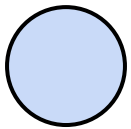
My background is in deep learning.

My research focuses on ***transfer learning*** *across data sources and tasks, as well as collaborative and modular learning.*

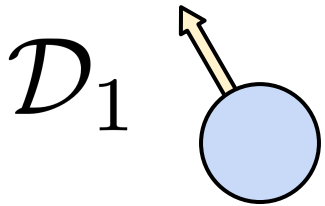
Goal for today

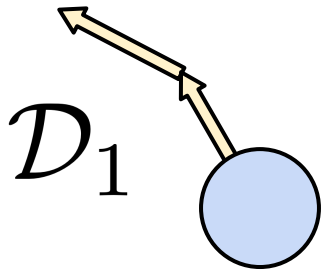
Show how different practical settings share the same underlying problem:

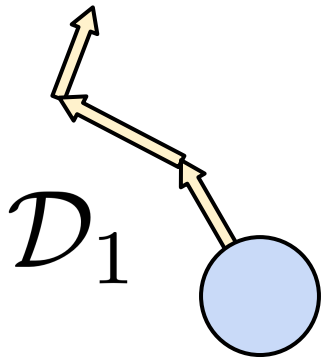
Learning from heterogeneous sources, under different constraints.

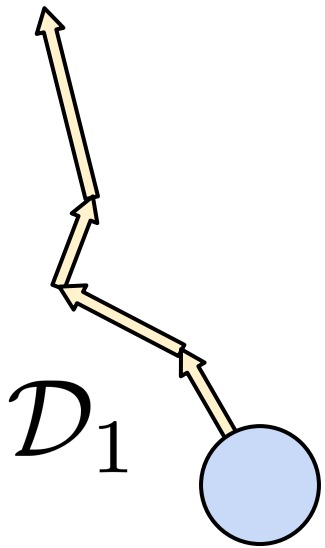


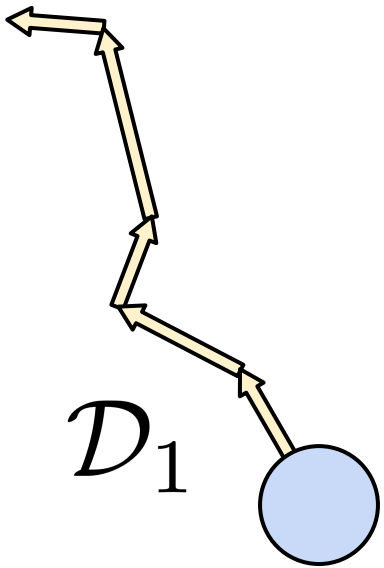
Pre-trained model / Initialization

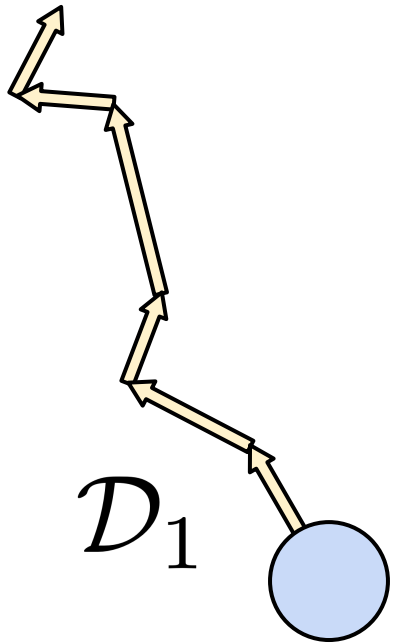

$\mathcal{D}_1$


$\mathcal{D}_1$

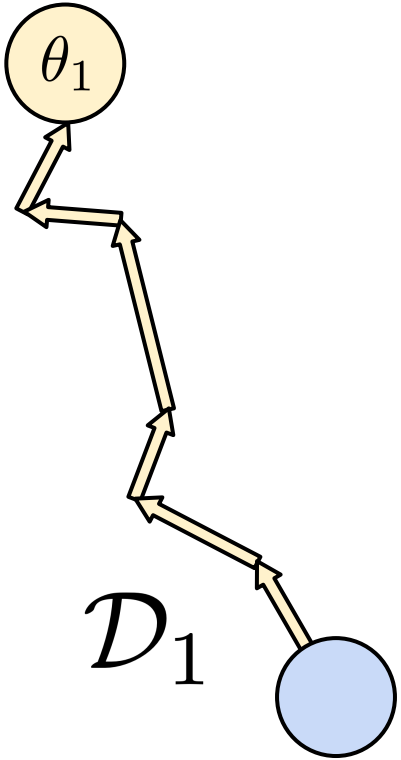

$\mathcal{D}_1$


$\mathcal{D}_1$

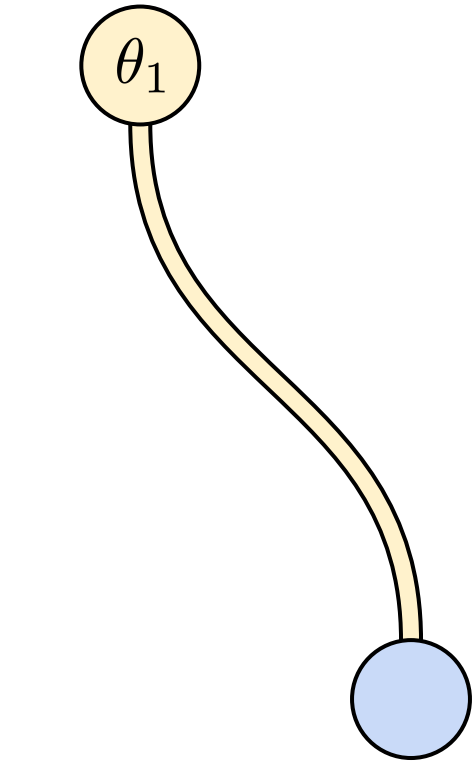

$\mathcal{D}_1$

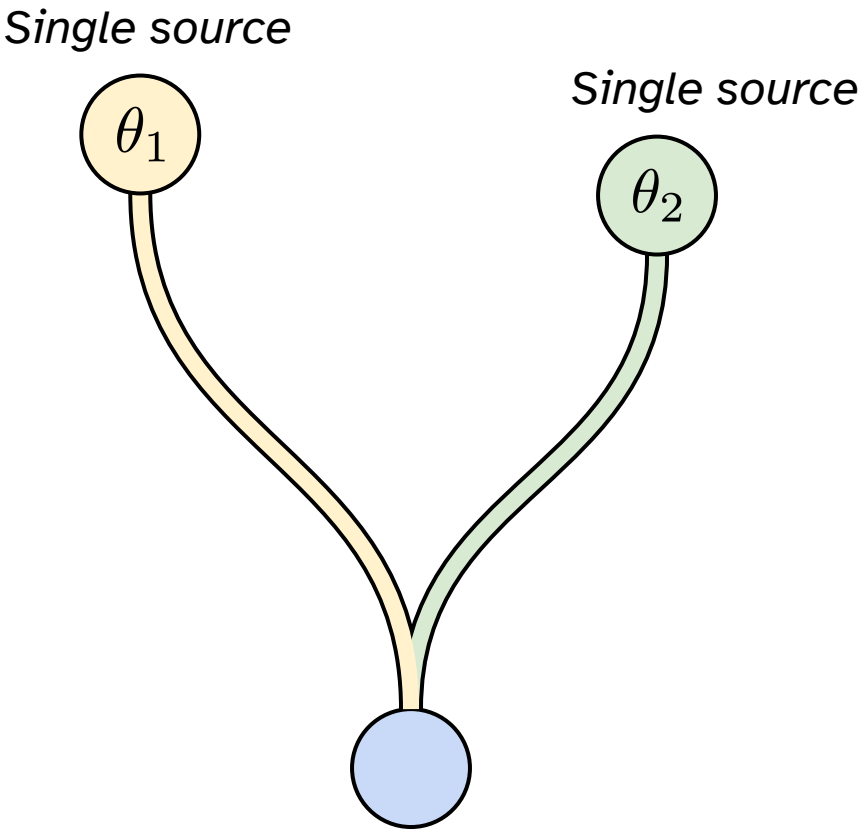

$\mathcal{D}_1$

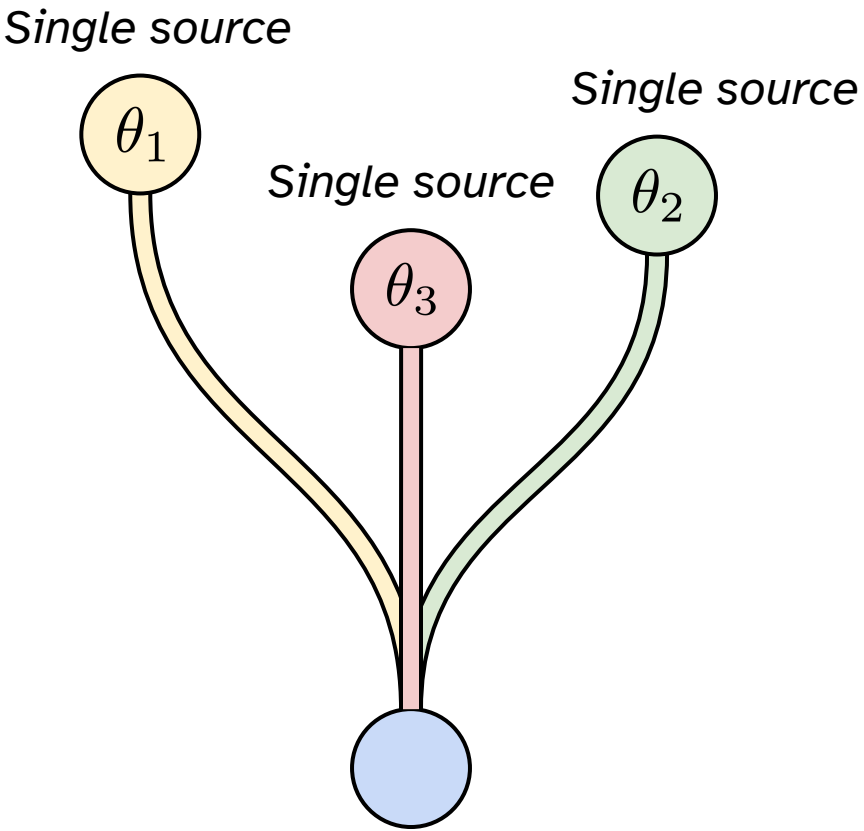
Single-source model



Single source




Single source
Single source
θ_1
θ_2

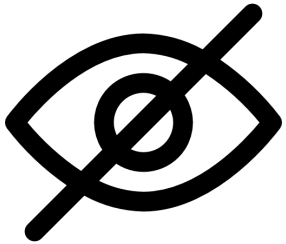

Single source
Single source
Single source
θ_1
θ_2
θ_3

Multi-task or multi-domain
(simultaneous access to all sources)

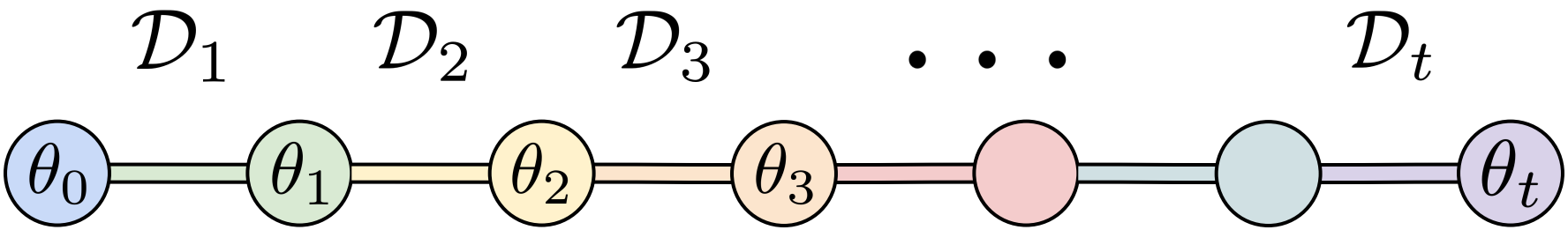
Challenge: *balance across sources*
find optimal data mixture, exploring pareto front is expensive

We cannot alway access all sources simultaneously.

Constraints on data access give different settings, but create similar challenges…



⏳ ***Continual Learning***: *Heterogeneity across time*



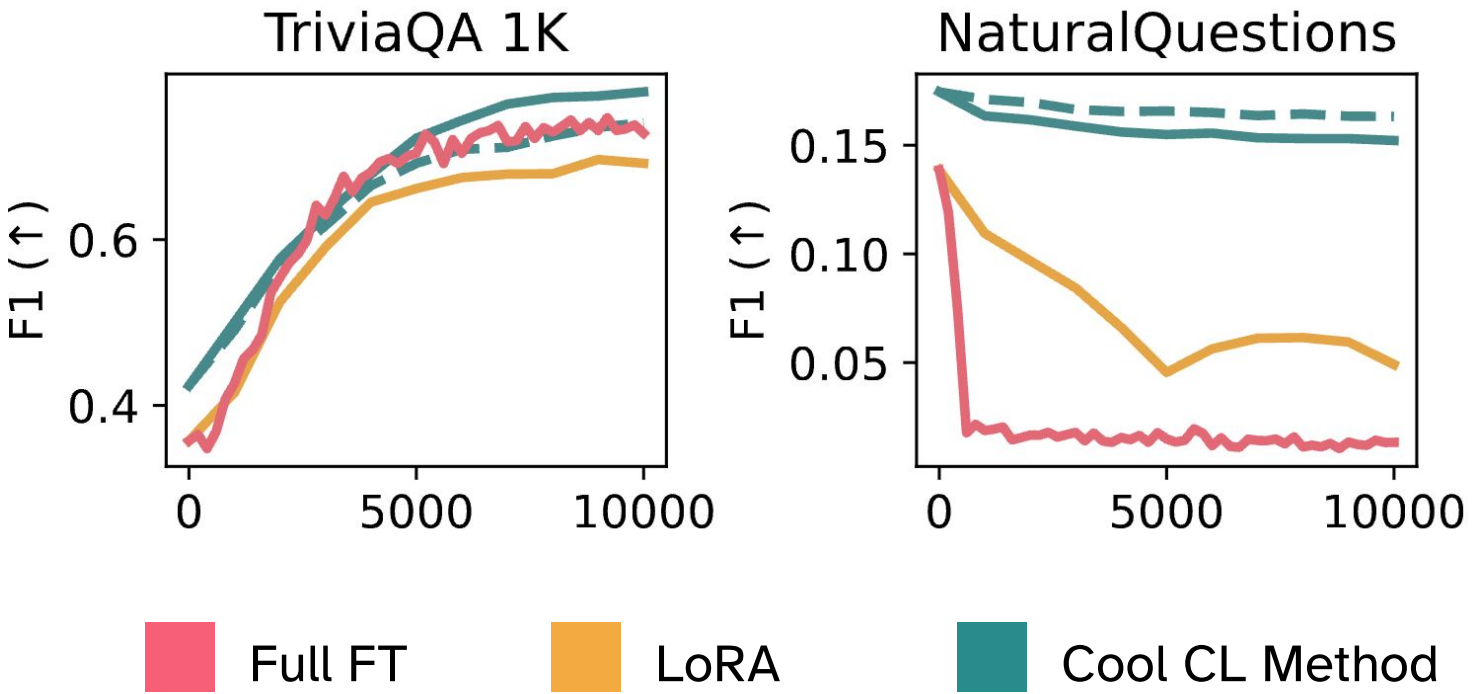
Constraint: *no access to past data!*

<u>Sequential</u> *training, only on* ***current data/task***

From "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem" by McCloskey and Cohen 1989
From "Catastrophic forgetting in connectionist networks" by French 1999

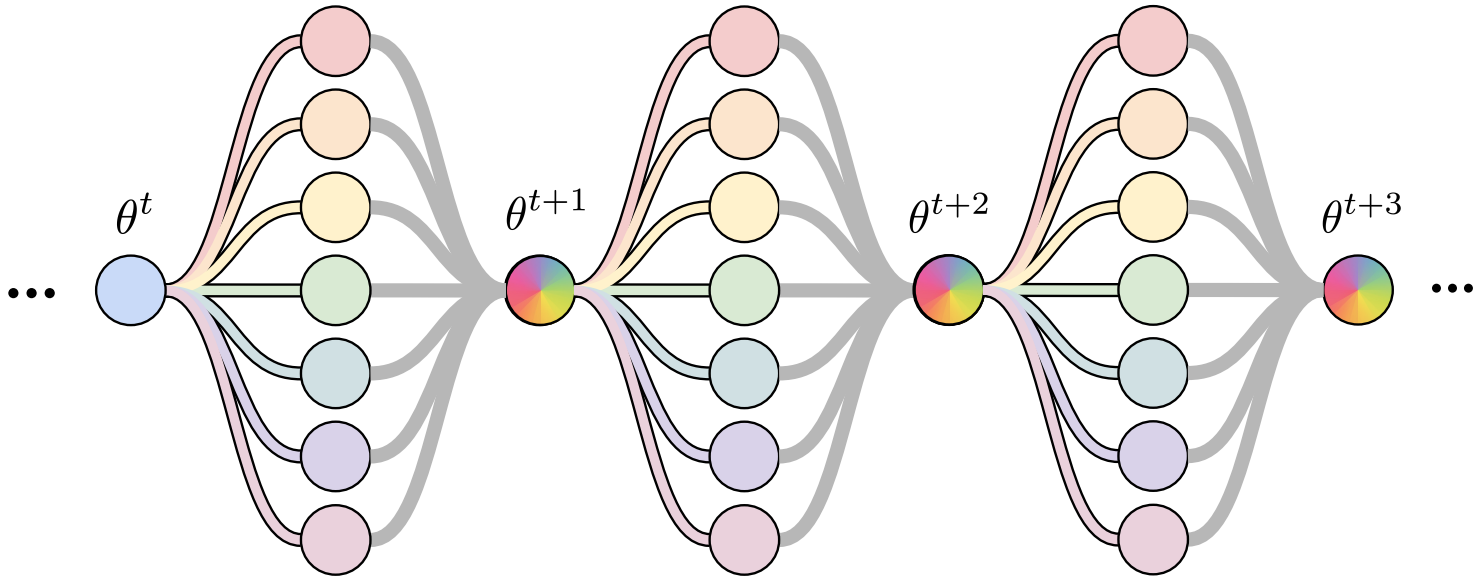
Key issue: ***Catastrophic Forgetting***

the model becomes ***biased*** *towards recent data*



From "Catastrophic forgetting in connectionist networks" by French 1999
Plot from "Continual Learning via Sparse Memory Finetuning" by Lin et al 2025

🌐 ***Federated Learning***: *Heterogeneity across space*

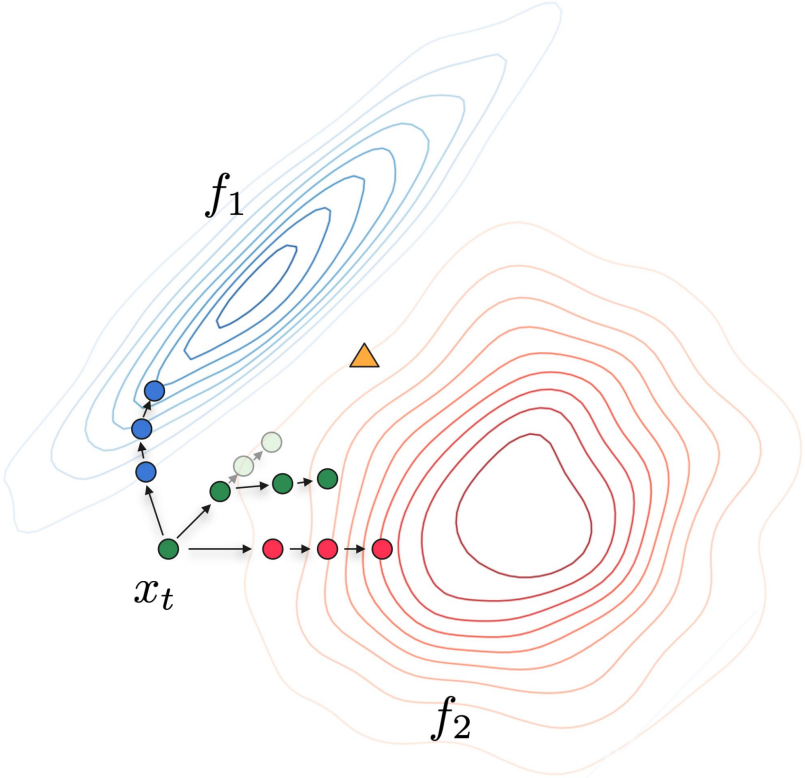


Constraint: *no access to full data, communication efficiency*

<u>Parallel</u> *training only on private clients' data and* ***model aggregation***

Client Drift Problem

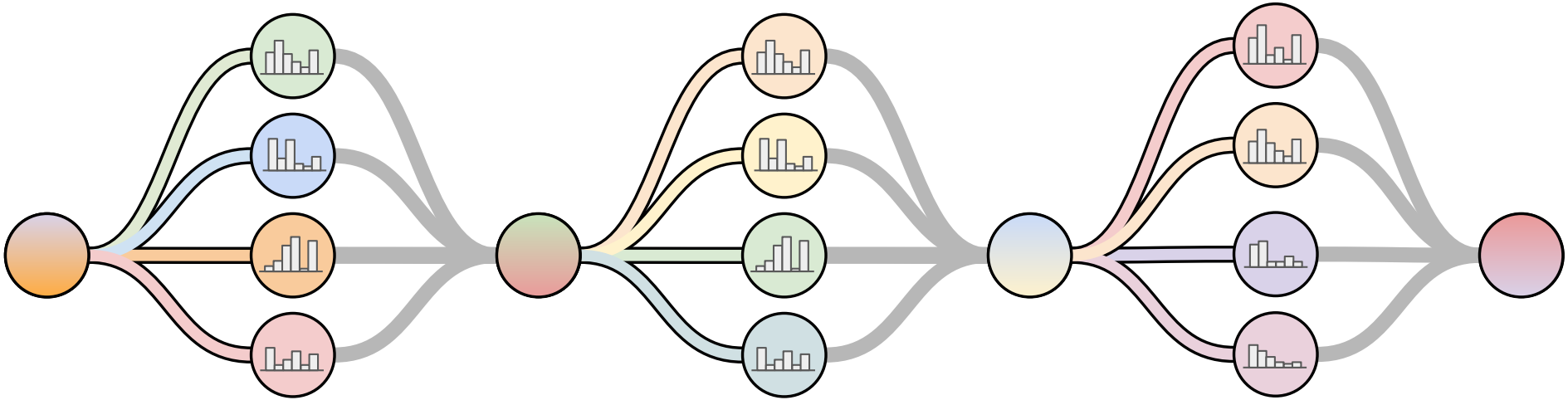
*Local models **overspecialize** on clients' data diverging from the global optimization goal*



Plot credit: SP Karimireddy, Scaffold presentation

FL with Partial Participation

(spatial and temporal heterogeneity)

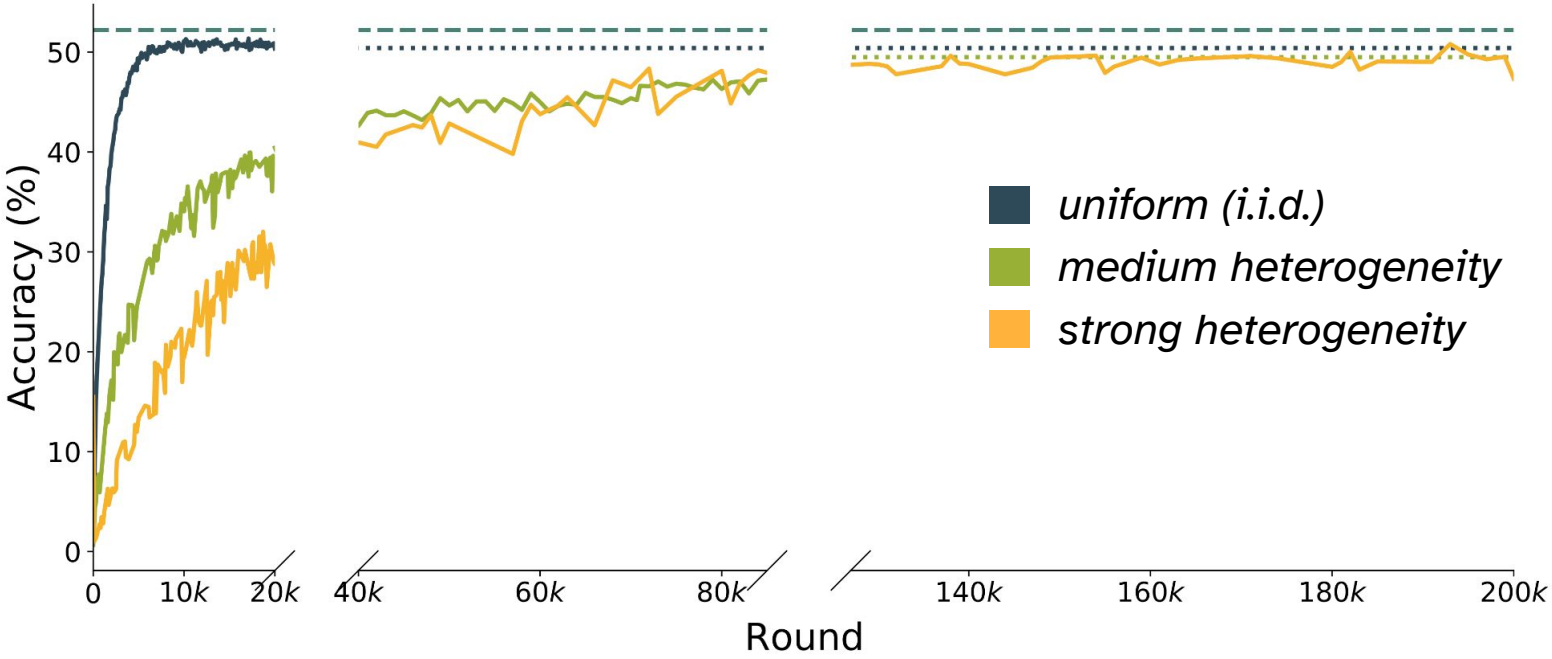


*This induces a **sequential learning** problem.*
At each round the global model is biased towards the most recent clients,
***forgetting** past knowledge.*

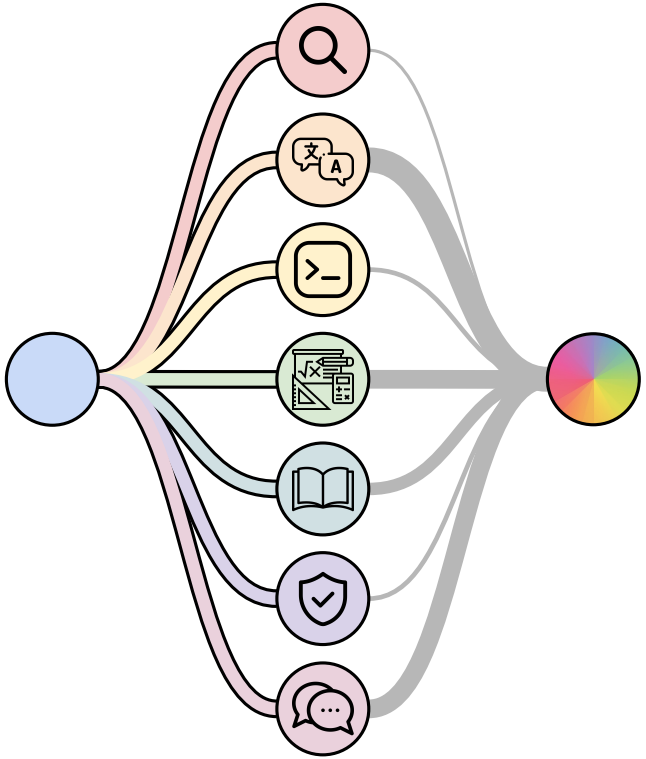
*FL with **one client per round** (and no client revisit)*
*reduces to **Continual Learning***

*This induces a **sequential learning** problem.*
At each round the global model is biased towards the most recent clients,
***forgetting** past knowledge.*

Key issue: ***slower and noisier convergence*** *(requires more communication)*



Model Merging *combines expert models*



Parallel *training and* ***single*** ***model aggregation***

Two core operations, two failure modes

“Local” Gradient Descent

optimizes on individual sources, minimizing partial objectives.

Model Aggregation

combines independently trained models in the parameter space.

Two core operations, two failure modes

"Local" Gradient Descent

optimizes on individual sources, minimizing partial objectives.

↓

***Forgetting** past knowledge,*
***Bias** towards local data,*
***Drift** from global solution.*

Model Aggregation

combines independently trained models in the parameter space.

↓

***Interference** from **conflicting parameters** destroys knowledge.*

Two core operations, two failure modes

"Local" Gradient Descent

optimizes on individual sources, minimizing partial objectives.

↓

***Forgetting** past knowledge,*
***Bias** towards local data,*
***Drift** from global solution.*

Model Aggregation

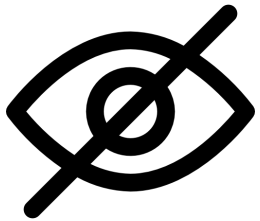
combines independently trained models in the parameter space.

↓

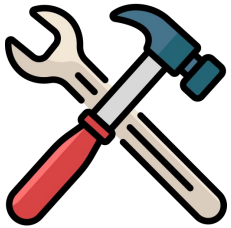
***Interference** from*
conflicting parameters
destroys knowledge.

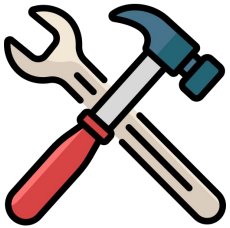
These compound, and different paradigms suffer from different combinations of problems

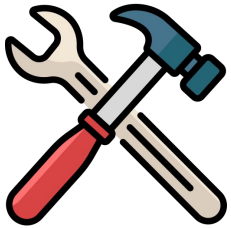
Similar problems...



...similar solutions.







Example: *local training with proximal terms*

Continual Learning: *Mitigate forgetting across tasks*

$$\mathcal{L}_{EWC}(\theta) = \mathcal{L}_{new}(\theta) + \frac{\lambda}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old})$$

Fisher Information Matrix

Federated Learning: *Mitigate client drift from global model*

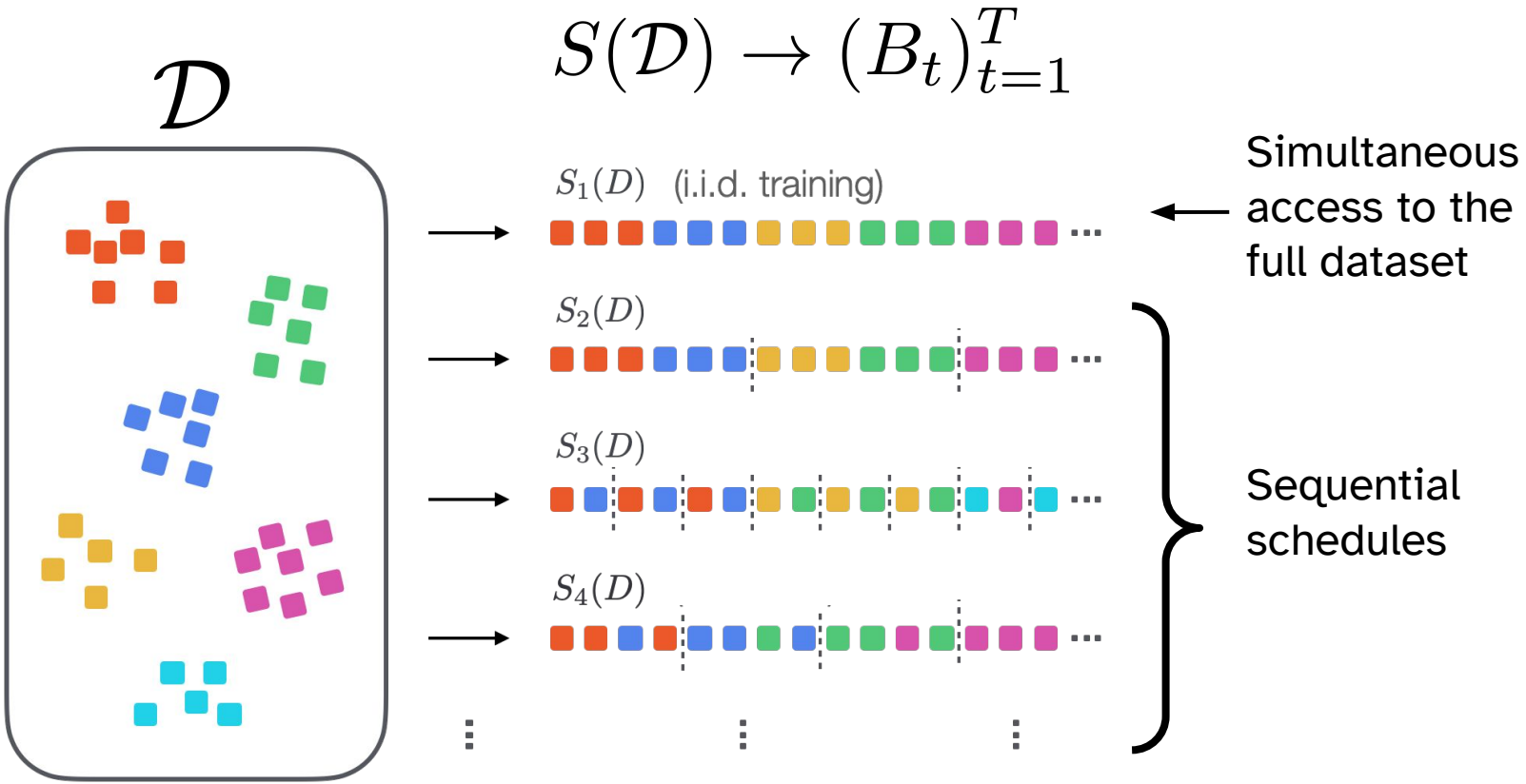
$$\mathcal{L}_{FedProx}(\theta) = \mathcal{L}_{local}(\theta) + \frac{\mu}{2}(\theta - \theta_{global})^T I(\theta - \theta_{global})$$

From "Overcoming catastrophic forgetting in neural networks" by Kirkpatrick et al

From "Federated Optimization in Heterogeneous Networks" by Li et al

Let's see a simple method
that can transfer across settings.

*In the CL setting, a dataset can be splitted into different "**schedules**" deciding how it is **streamed** into chunks of data*



GOAL: *Find an algorithm that is robust (or invariant) to different schedules.*
We can restrict ourselves to a ***specific class of "one vs all" predictors***

$$W^* = \arg\min_W \frac{1}{|D|} \sum_{(x,y)\in D} \left\| W^\top \psi(x) - \mathrm{OneHot}(y) \right\|^2 + \lambda \|W\|^2 .$$

pretrained model

GOAL: *Find an algorithm that is robust (or invariant) to different schedules.*
We can restrict ourselves to a ***specific class of "one vs all" predictors***

$$W^* = \arg\min_W \frac{1}{|D|} \sum_{(x,y)\in D} \left\| W^\top \psi(x) - \mathrm{OneHot}(y) \right\|^2 + \lambda \|W\|^2 .$$

ψ: pretrained model

Closed-form solution

$$W^* = (\underbrace{X^\top X}_{A} + \lambda I)^{-1} \underbrace{X^\top Y}_{b}$$

$$X = [\psi(x_1) \dots \psi(x_N)]^\top$$

$$Y = [\mathrm{OneHot}(y_1) \dots \mathrm{OneHot}(y_N)]^\top$$

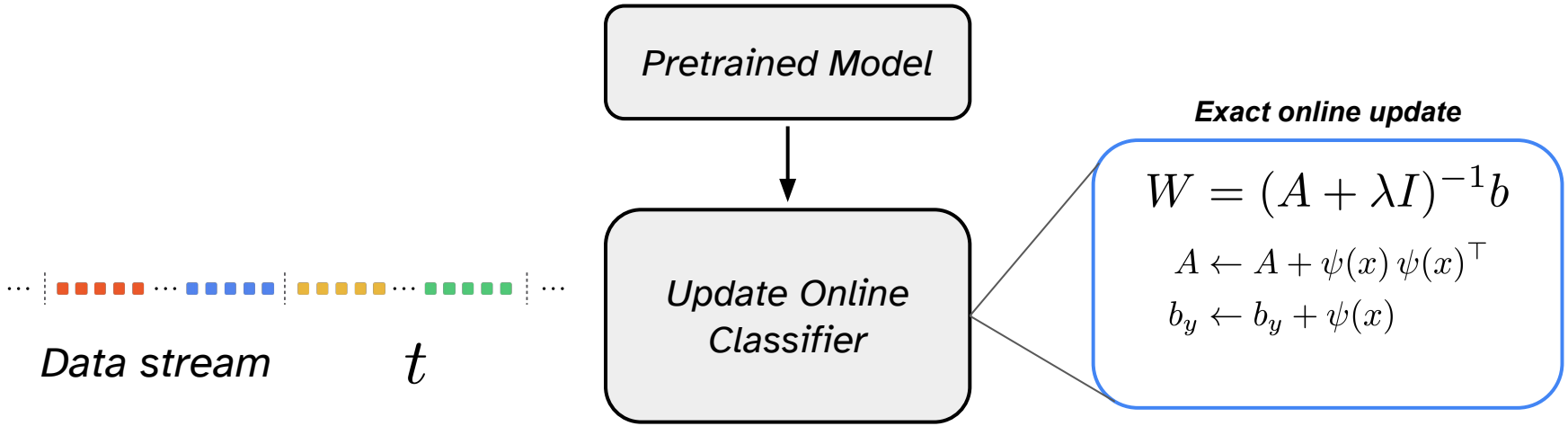
Exact online update

$$W = (A + \lambda I)^{-1} b$$

$$A \leftarrow A + \psi(x)\,\psi(x)^\top$$

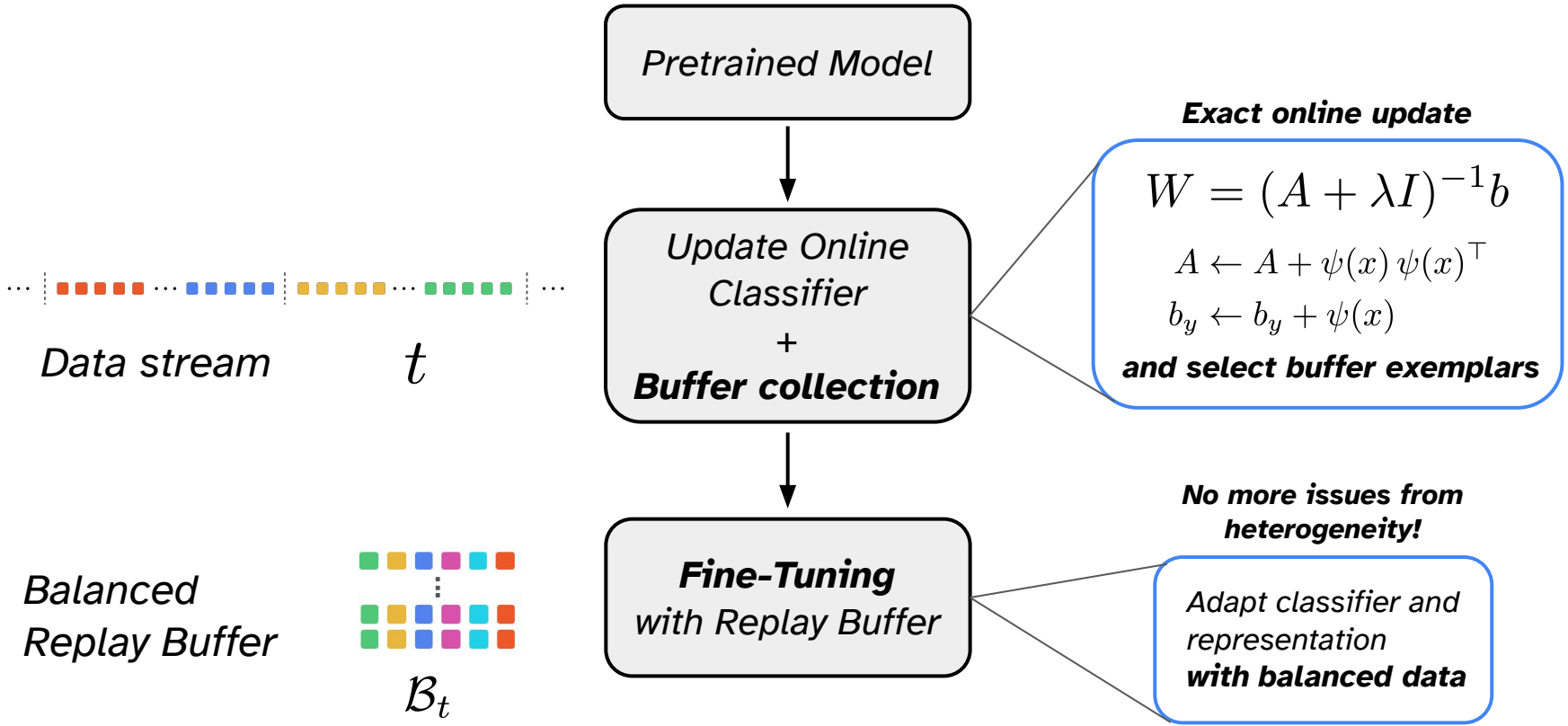
$$b_y \leftarrow b_y + \psi(x)$$

SCROLL: Schedule-Robust Continual Learning



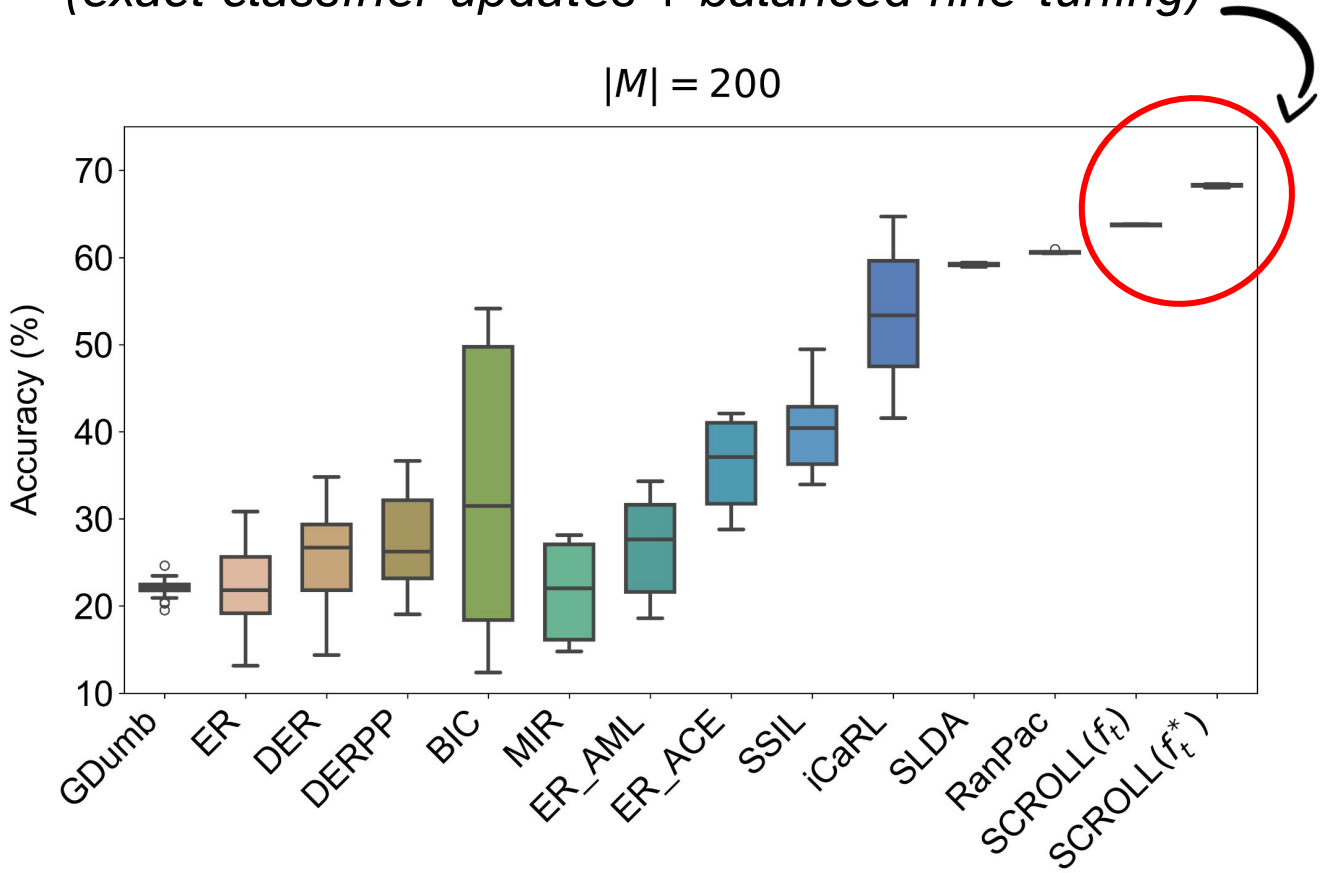
R. Wang, M. Ciccone, M. Pontil, C. Ciliberto "Schedule-Robust Continual Learning" IEEE TPAMI

SCROLL: Schedule-Robust Continual Learning



R. Wang, M. Ciccone, M. Pontil, C. Ciliberto "Schedule-Robust Continual Learning" IEEE TPAMI

SCROLL *is effectively robust across different schedules (exact classifier updates + balanced fine-tuning)*



R. Wang, M. Ciccone, M. Pontil, C. Ciliberto "Schedule-Robust Continual Learning" IEEE TPAMI

The same framework transfers to Federated Learning

Schedules as “client splits”

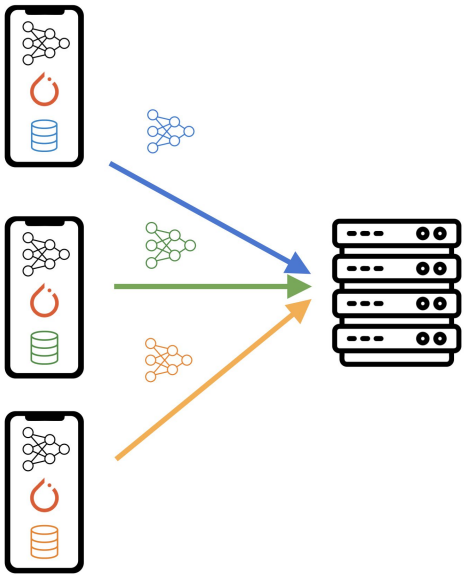
*Federated Recursive Ridge Regression (**Fed3R**)*
*builds a classifier **exactly** and **incrementally** across clients*

Compute Local Statistics

Client side

$$A_k^t = \sum_{(x,y)\in\mathcal{D}_k} \psi(x)^\top \psi(x)$$

$$b_k^t = \sum_{(x,y)\in\mathcal{D}_k} \psi(x)^\top e_y$$



Aggregate Statistics

Server side

$$A^t = A^{t-1} + \sum_{k\in\mathcal{K}} A_k^t$$

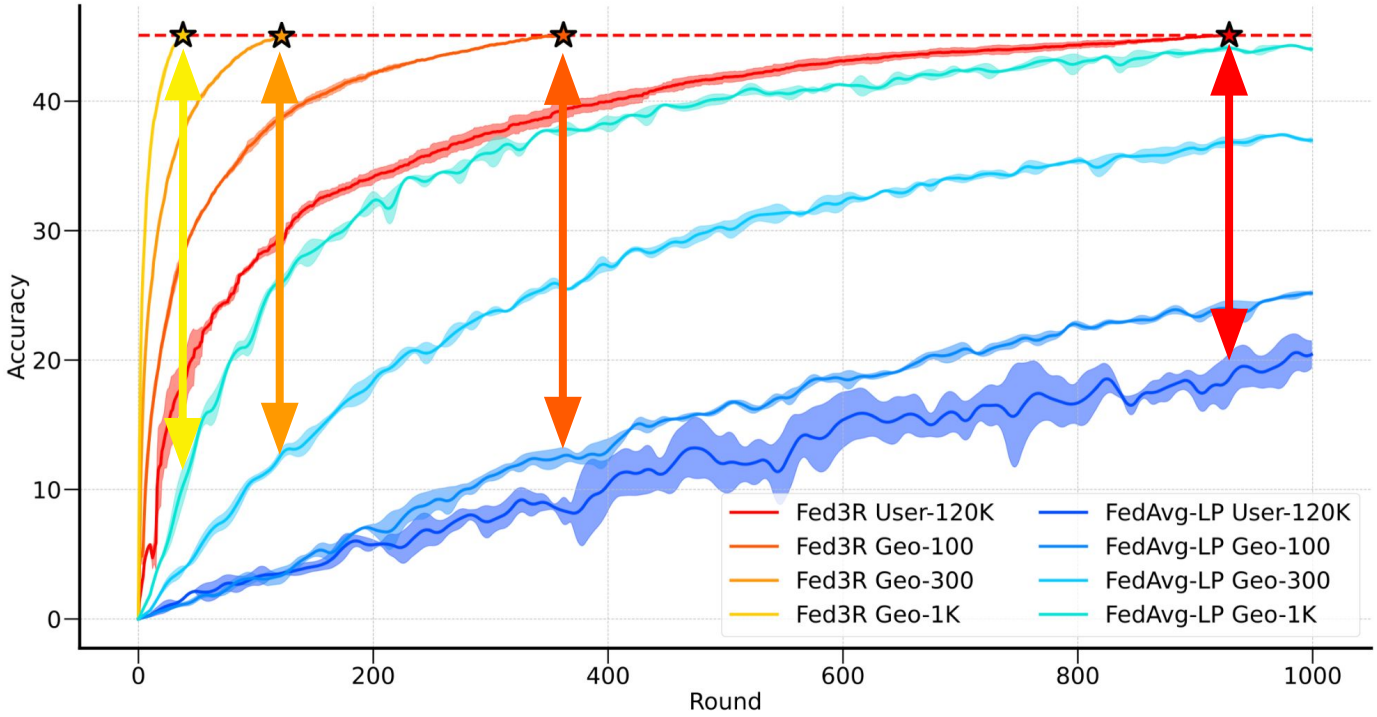
$$b^t = b^{t-1} + \sum_{k\in\mathcal{K}} b_k^t$$

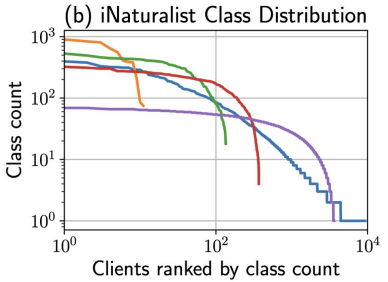
$$\downarrow$$

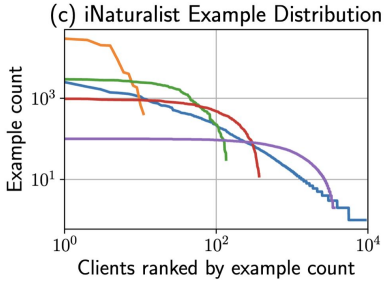
$$w = (A^t + \lambda I)^{-1} b^t$$

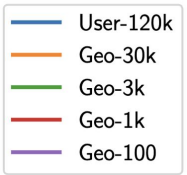
E. Fanì, R. Camoriano, B. Caputo, M. Ciccone "Accelerating Heterogeneous Federated Learning with Closed-form Classifiers" ICML 2024

*Fed3R solution is **equivalent to the centralized** solution (invariant to the **client split** and the **sampling order** by design)*



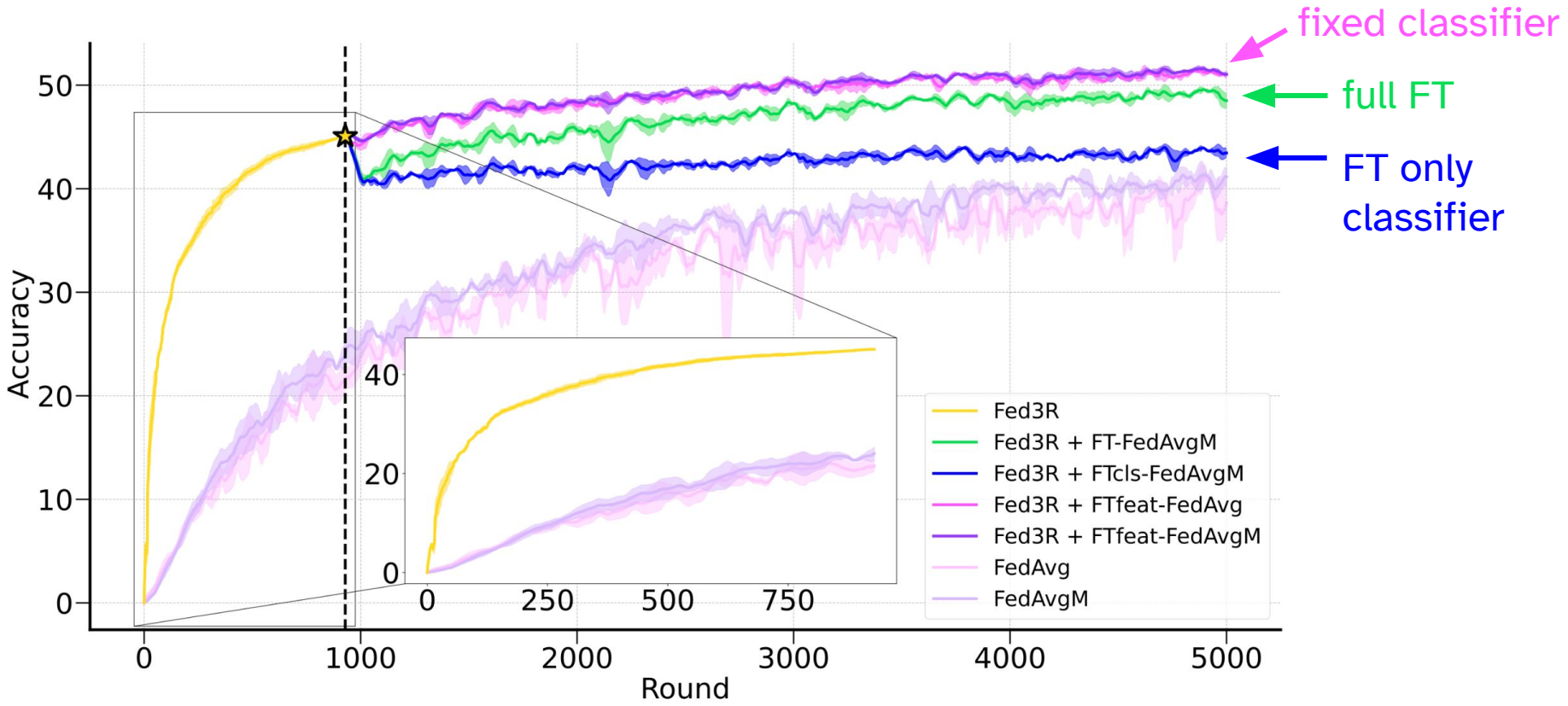






E. Fanì, R. Camoriano, B. Caputo, M. Ciccone "Accelerating Heterogeneous Federated Learning with Closed-form Classifiers" ICML 2024

Fed3R solution can be used as initialization for any federated algorithm



E. Fanì, R. Camoriano, B. Caputo, M. Ciccone "Accelerating Heterogeneous Federated Learning with Closed-form Classifiers" ICML 2024

We avoided the problem bypassing the issues of gradient-based learning with heterogeneous data…

(exact classifier aggregation + balanced buffer)

Because of constraints, often we can't use exactly the same approaches across setting.

Because of constraints, often we can't use exactly the same approaches across setting.

But we can transfer similar ideas.

Example:

We can't use replay buffers in FL, but…

many methods tried to ***correct the local update*** *by mimicking the centralized update*

Classical Heavy-Ball Momentum

$$\tilde{m}^t \leftarrow \beta \tilde{m}^{t-1} + \tilde{g}^t(\theta^{t-1}; \mathcal{D}^t)$$
$$\theta^t \leftarrow \theta^{t-1} - \eta \tilde{m}^t$$

Classical Heavy-Ball formulation (Polyak, 1964)

$$\theta^t \leftarrow \theta^{t-1} - \eta \tilde{g}^t(\theta^{t-1}; \mathcal{D}^t) + \beta(\theta^{t-1} - \theta^{t-2})$$

Problem: *in partial participation, with data heterogeneity the momentum is* ***biased*** *towards clients sampled during* ***last round***

Zaccone, Karimireddy, Masone, M. Ciccone "Communication-Efficient Heterogeneous Federated Learning with Generalized Heavy-Ball Momentum" TMLR 2025

Generalized Heavy-Ball Momentum (GHBM)

Main idea: *Recycle gradient from past $\tau > 1$ rounds to correct the bias*

$$\tilde{m}^t \leftarrow \beta \tilde{m}^{t-1} + \frac{1}{\tau} \sum_{k=t-\tau+1}^{t} \tilde{g}^k(\theta^{k-1}; \mathcal{D}^k)$$

$$\theta^t \leftarrow \theta^{t-1} - \eta \tilde{m}^t$$

Uniform Average across past gradients

Equivalent Heavy Ball formulation

$$\theta^t \leftarrow \theta^{t-1} - \eta \tilde{g}^t(\theta^{t-1}; \mathcal{D}^t) + \frac{\beta}{\tau}(\theta^{t-1} - \theta^{t-\tau-1})$$

recovers classical momentum for $\tau = 1$

Zaccone, Karimireddy, Masone, M. Ciccone "Communication-Efficient Heterogeneous Federated Learning with Generalized Heavy-Ball Momentum" TMLR 2025

Client update rule with GHBM

*Inject **global information** into the local optimization*
*to **guide it towards the global solution** and reduce the client drift.*

$$\theta_i^{t,j} \leftarrow \theta_i^{t,j-1} - \eta_l \tilde{g}_i^{t,j} + \frac{\beta}{\tau J} \underbrace{(\theta^{t-1} - \theta^{t-\tau-1})}_{\tau\text{-GHBM}}$$

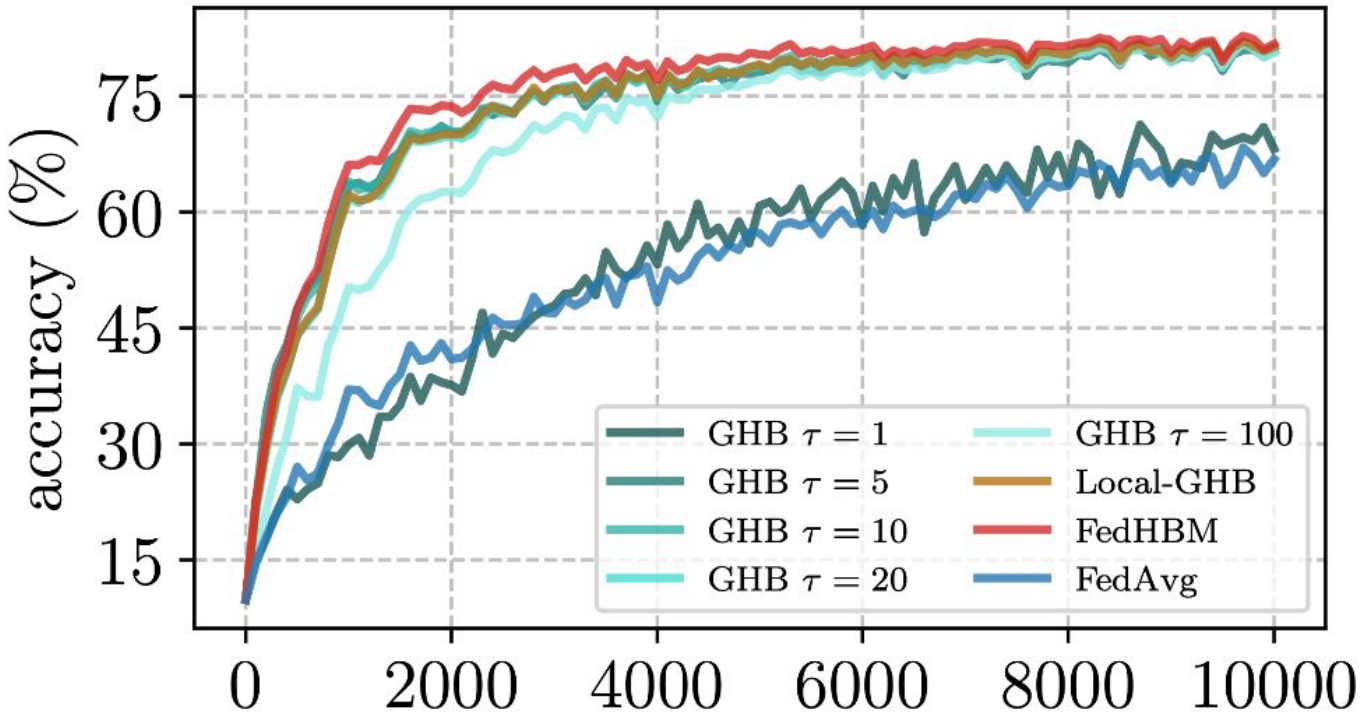
classic local SGD for i-th client

(j-th inner steps, t-th round)

GHBM correction computed with gradients from clients of past τ rounds

Zaccone, Karimireddy, Masone, M. Ciccone "Communication-Efficient Heterogeneous Federated Learning with Generalized Heavy-Ball Momentum" TMLR 2025

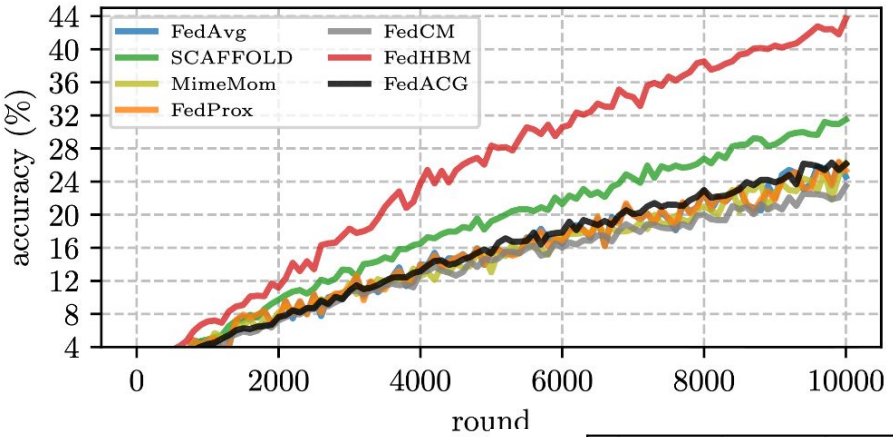
GHBM effectively speeds up convergence under heterogeneity and partial participation



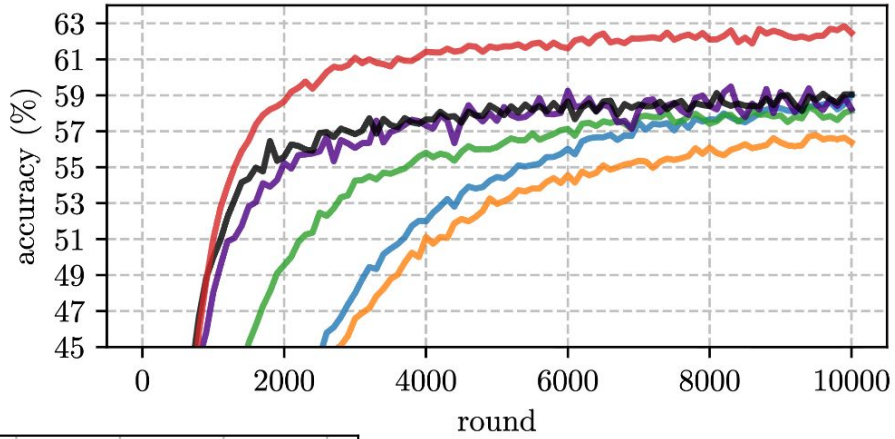
Zaccone, Karimireddy, Masone, M. Ciccone "Communication-Efficient Heterogeneous Federated Learning with Generalized Heavy-Ball Momentum" TMLR 2025

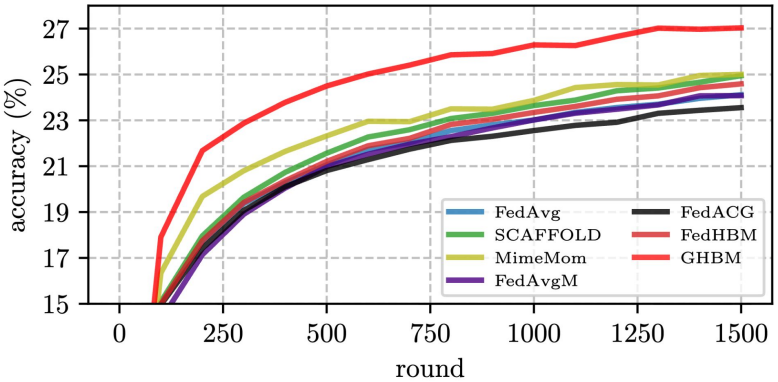
GHBM works well on several datasets and architectures

Cifar-100 Heterogeneous (ResNet)



Cifar-100 Homogeneous (ResNet)





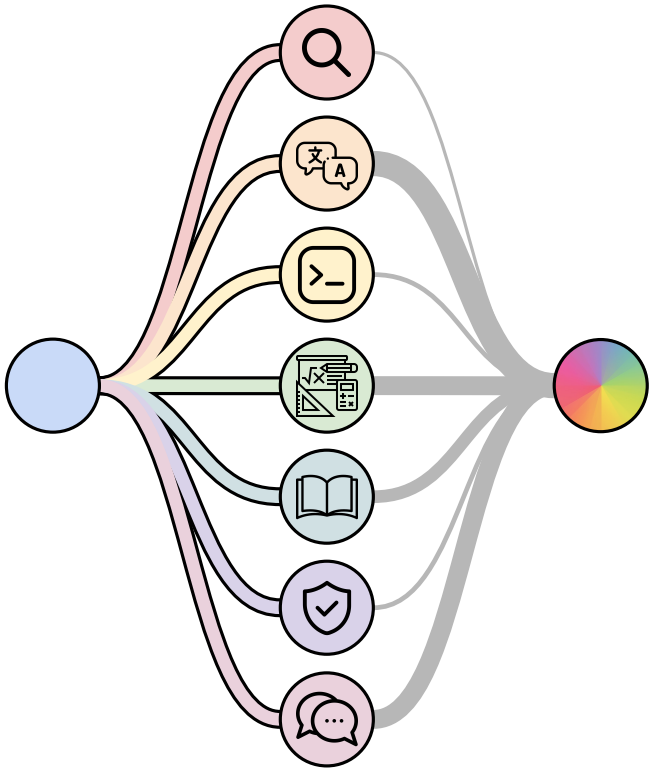
StackOverflow
(Heterogeneous
40K clients)

Zaccone, Karimireddy, Masone, M. Ciccone "Communication-Efficient Heterogeneous Federated Learning with Generalized Heavy-Ball Momentum" TMLR 2025

So far we mainly discussed about local training, what about model aggregation?

Model Merging *combines expert models post-hoc*

*(starting from the same **pre-trained model**)*



Core merging approach: ***Parameter Averaging***

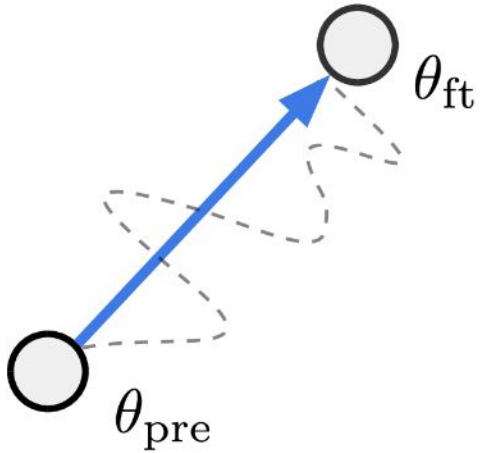
(linear interpolation in the weight space)

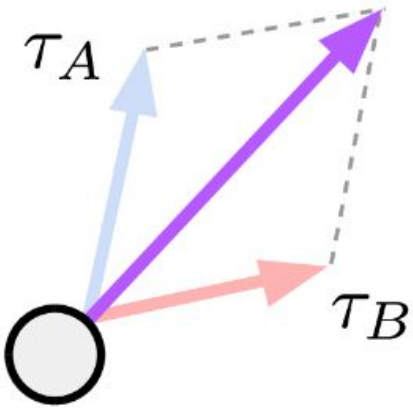
$$\theta_{\text{merged}} = \sum_{i=1}^{M} \lambda_i \theta_i$$

often, but not always, a convex combination

$$\sum_{i=1}^{M} \lambda_i = 1$$

Task vectors provide a general way of model editing (and merging)





$$\tau = \theta_{\text{ft}} - \theta_{\text{pre}}$$

$$\tau_{\text{merged}} = \tau_{\text{A}} + \tau_{\text{B}}$$

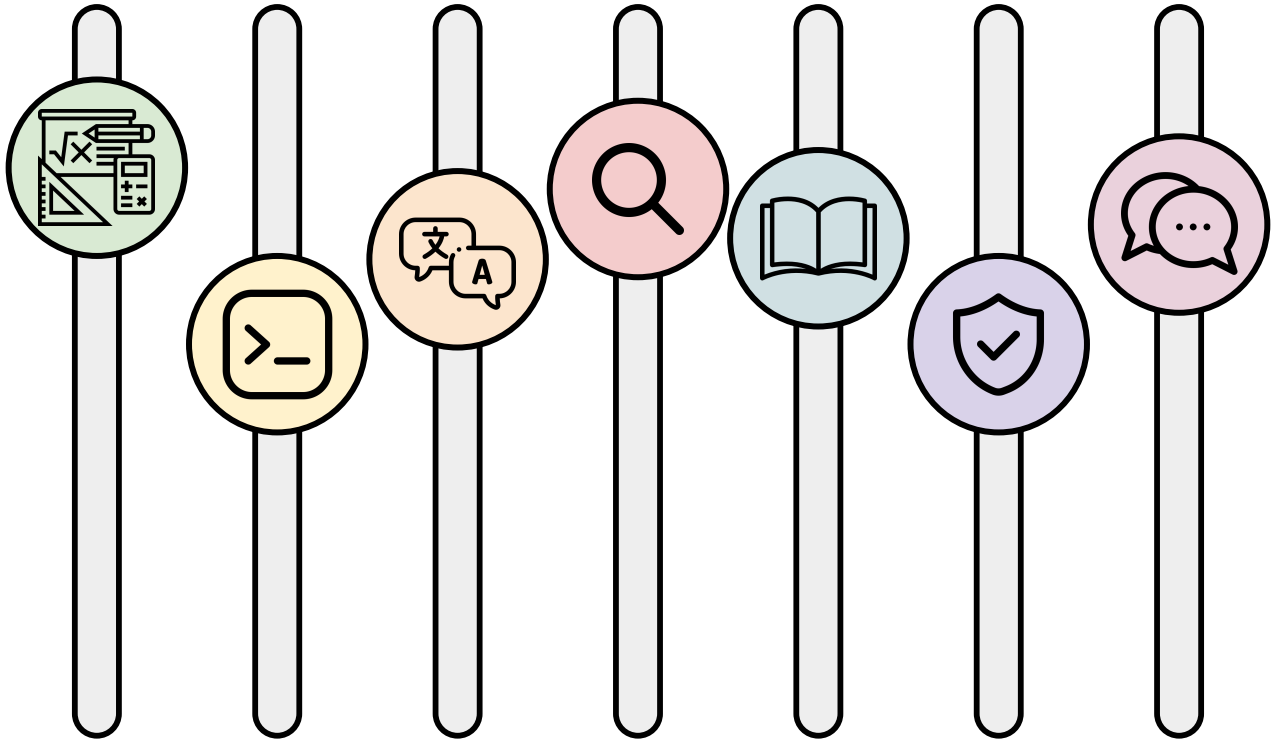
*Same concept of **"pseudo-gradient"** in FL (single round FL)*

From "Editing Models with Task Arithmetic" by Ilharco et al.

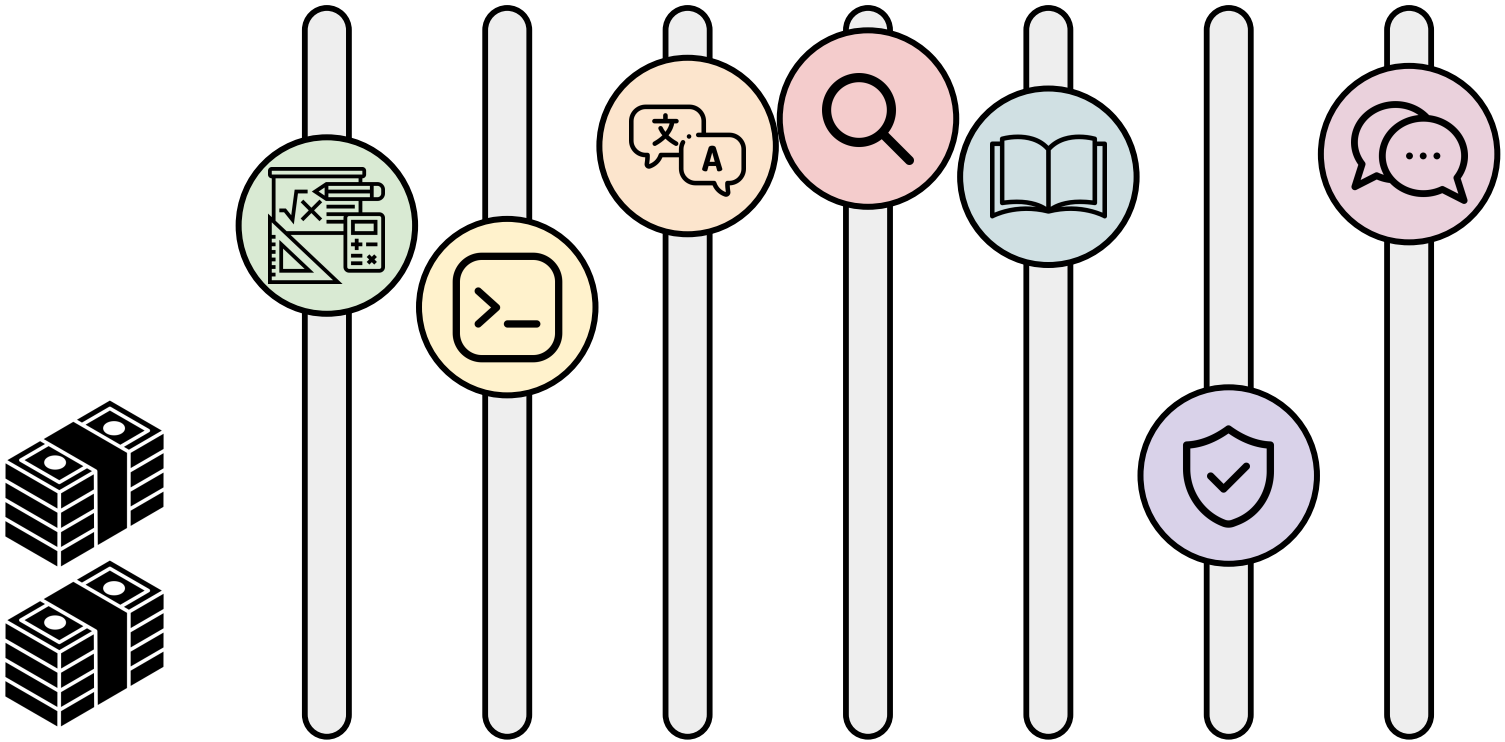
Multi-task or multi-domain
(simultaneous access to all sources)

Challenge: *balance across sources*
find optimal data mixture

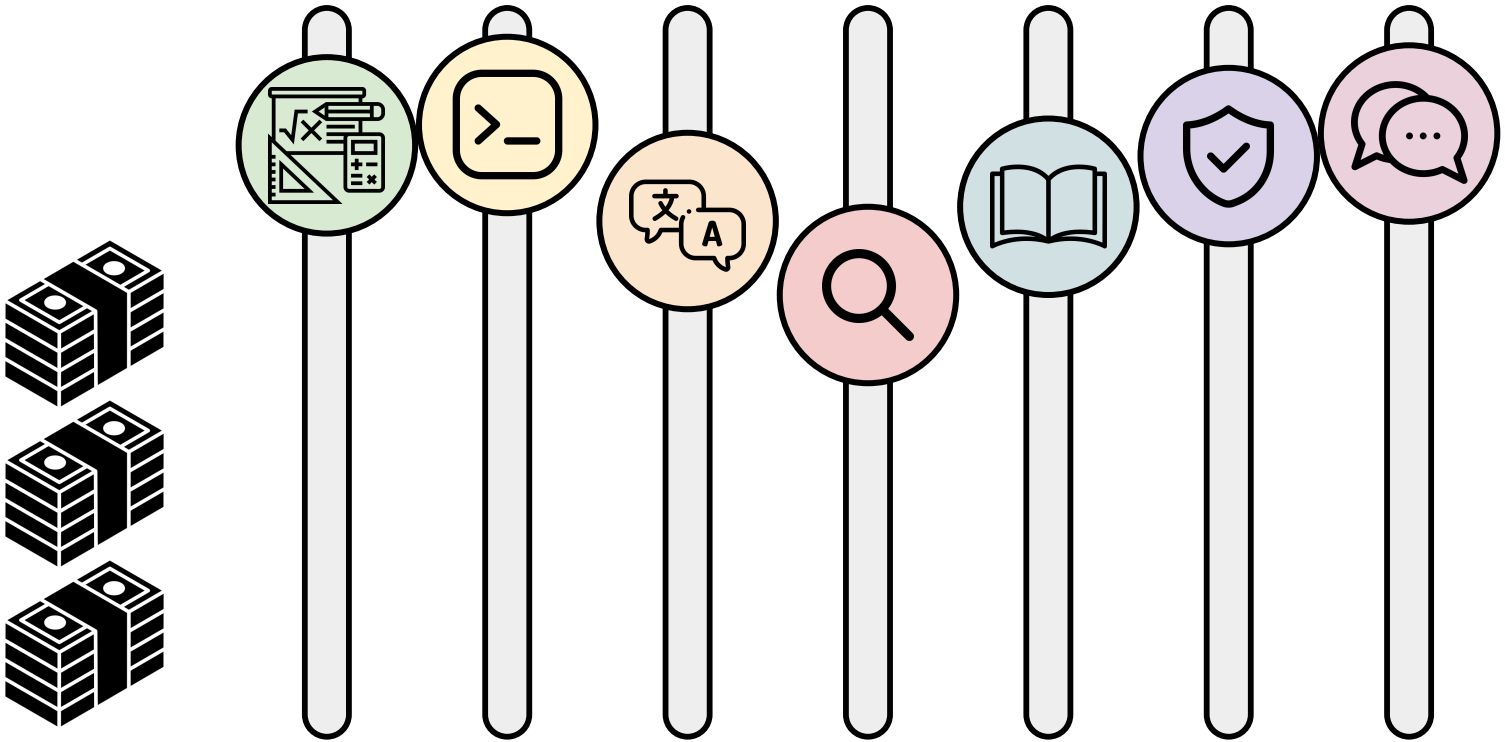
but exploring different mixtures is expensive!



but exploring different mixtures is expensive!

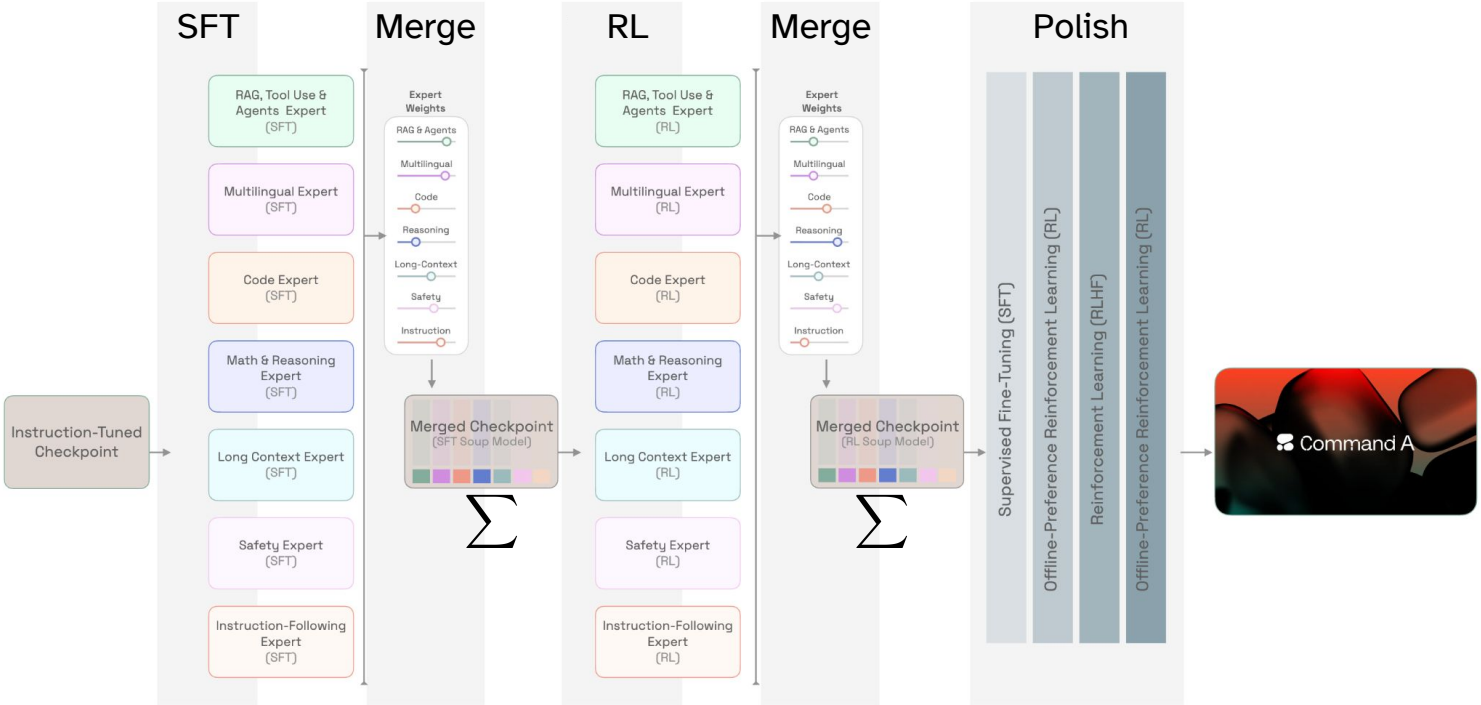


but exploring different mixtures is expensive!



*Merging enables teams to **work asynchronously** on improving capabilities and **cheaply explore different tradeoffs post-hoc***

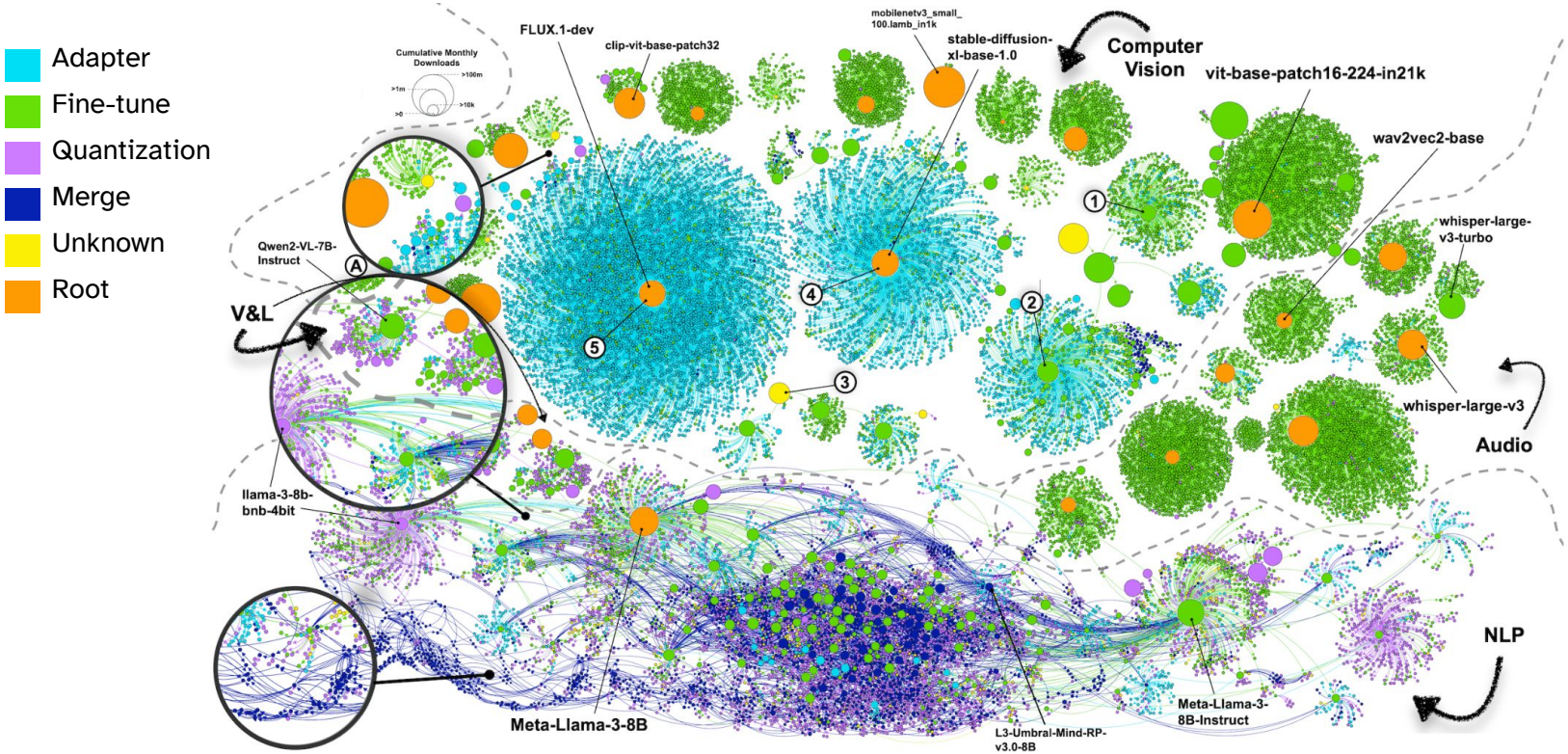
***Controlled setting:** in house experts, full access to data*



From "Command A: An Enterprise-Ready Large Language Model" by Ahmadian, Cohere, Team, et al.

Merging experts in the wild

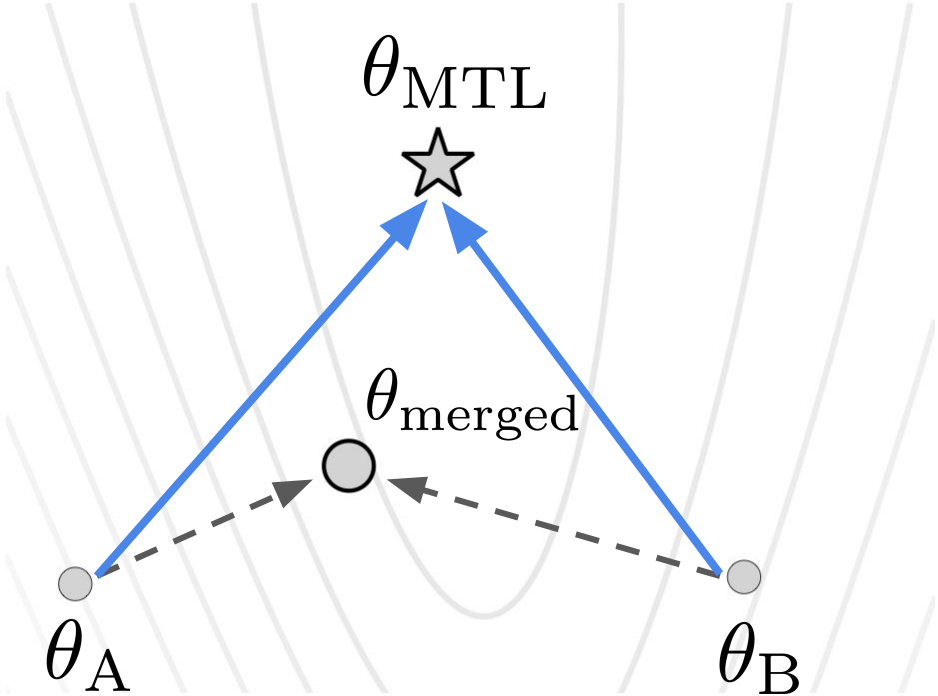
Combine experts from contributors: *no access to original data*



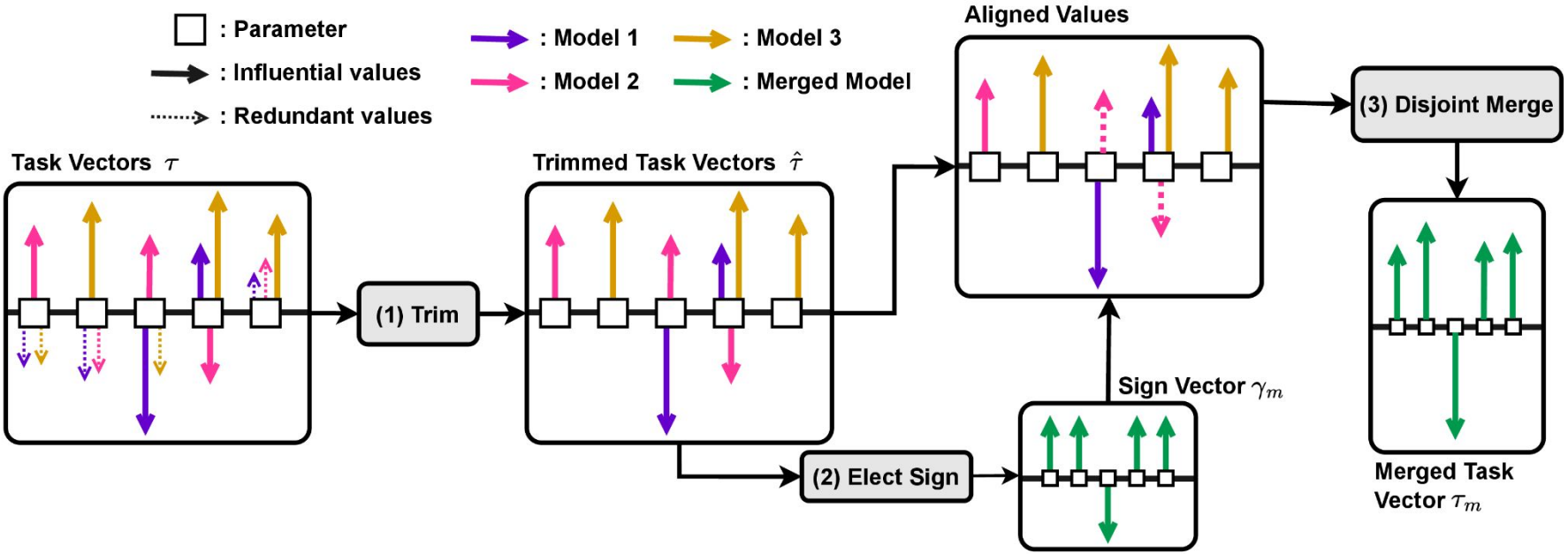
From "Charting and Navigating Hugging Face's Model Atlas" by Horwitz, Eliahu, et al.

Independently *fine-tuned models* ***"interfere"*** *when combined*

Several ***heuristics*** *exists to improve aggregation and* ***reduce interference***

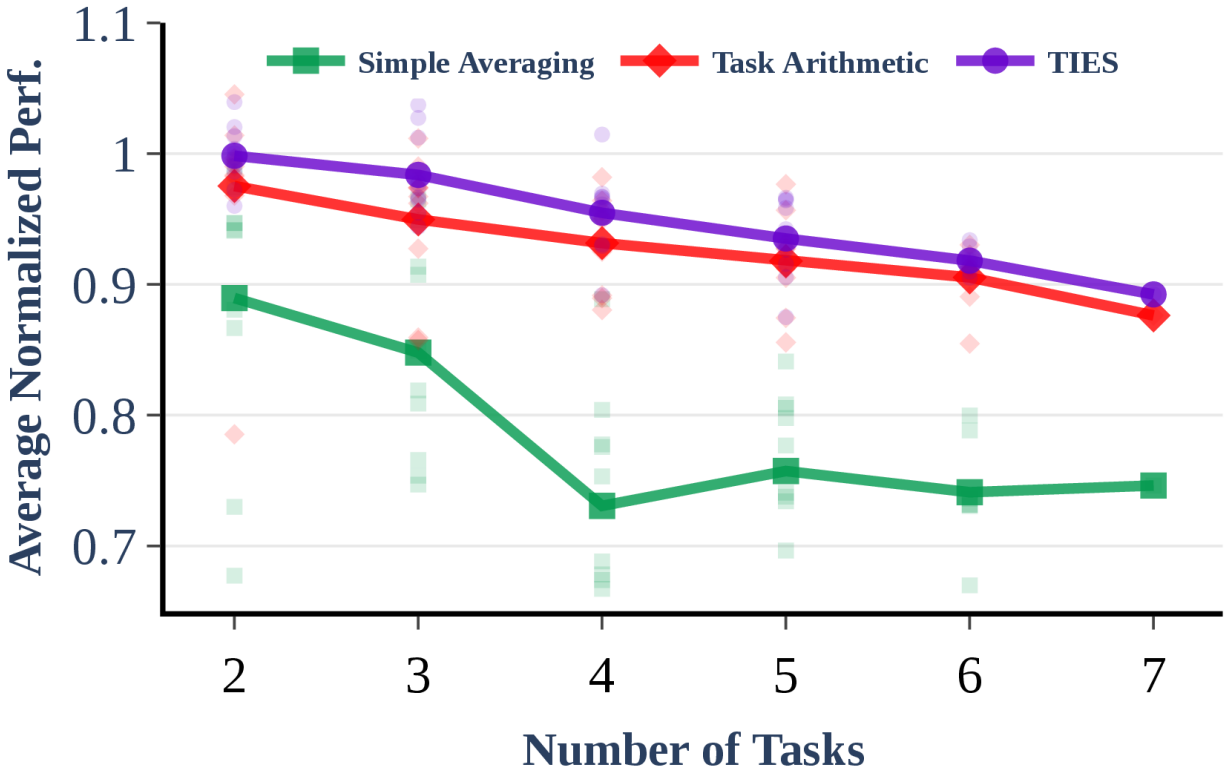


Example: *TIES Merging to resolve "interference" between task vectors via* ***pruning*** *and* ***sign agreement***



From "Resolving Interference When Merging Models" by Yadav et al.

TIES helps retain individual model performance



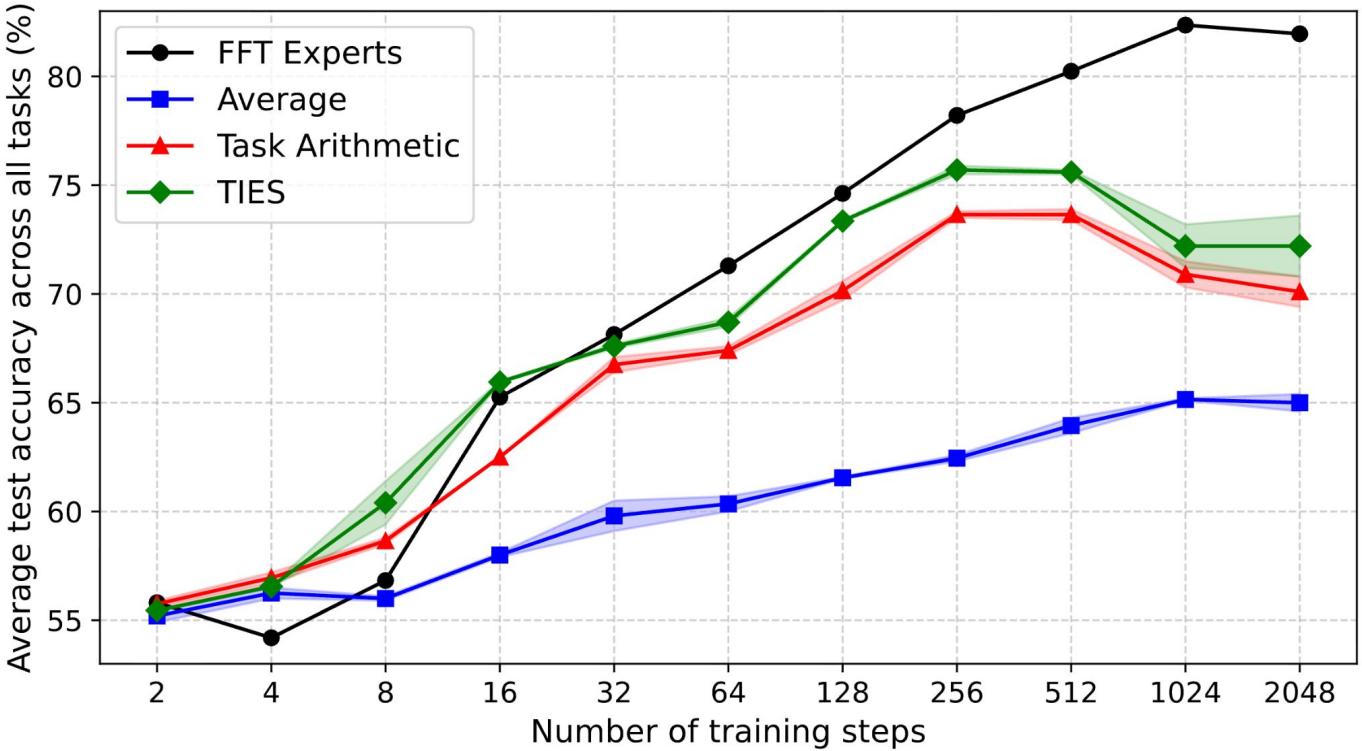
From "Resolving Interference When Merging Models" by Yadav et al.

Merging methods treat compatibility as an afterthought, relying on post-hoc fixes.

Merging methods treat compatibility as an afterthought, relying on post-hoc fixes.

How can we train models that are ***composable*** *by design?*

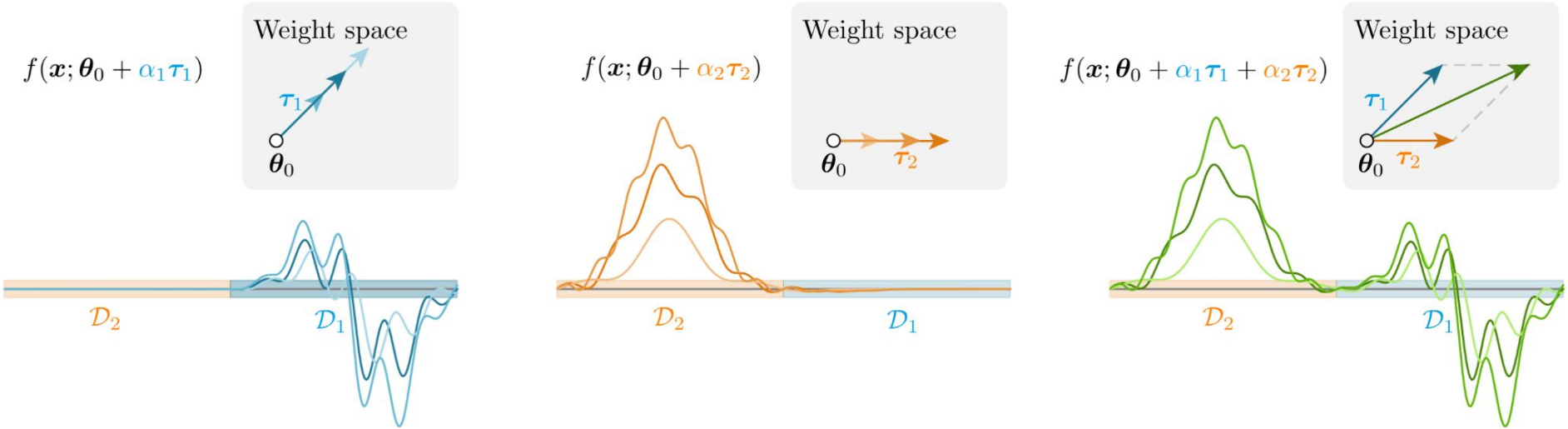
Local training has a clear effect on mergeability.
Overspecialization hurts aggregation - (client-drift?)



From "Less is More: Undertraining Experts Improves Model Upcycling", by Horoi et al

"Weight disentanglement" *property*

Merged model should behave as a composition of ***localized*** *functions*
where a task vector activates only for the support of its task



We can formalize this property and derive
a fine-tuning update that approximately satisfies it

From "Task Arithmetic in the Tangent Space: Improved Editing of Pre-Trained Models" by Ortiz-Jimenez et al.

Localized functions: *the task vector should activate only for the support of its task*

$$f\left(\boldsymbol{x}, \boldsymbol{\theta}_0 + \sum_{t=1}^{T} \alpha_t \boldsymbol{\tau}_t\right) = f(\boldsymbol{x}, \boldsymbol{\theta}_0)\mathbf{1}\left(\boldsymbol{x} \notin \bigcup_{t=1}^{T} \mathcal{D}_t\right) + \sum_{t=1}^{T} f(\boldsymbol{x}, \boldsymbol{\theta}_0 + \alpha_t \boldsymbol{\tau}_t)\mathbf{1}(\boldsymbol{x} \in \mathcal{D}_t)$$

Linearing the merged model we can derive constraints on the task vectors

$$f_{\text{lin}}\left(\boldsymbol{x}, \boldsymbol{\theta}_0 + \sum_{t=1}^{T} \alpha_t \boldsymbol{\tau}_t\right) = f(\boldsymbol{x}, \boldsymbol{\theta}_0) + \sum_{t=1}^{T} \alpha_t \boldsymbol{\tau}_t^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}, \boldsymbol{\theta}_0)$$

$$\forall \boldsymbol{x} \in \mathcal{D}_{t' \neq t}, \quad \boldsymbol{\tau}_t^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}, \boldsymbol{\theta}_0) = 0$$

We want $\boldsymbol{\tau}_t^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}, \boldsymbol{\theta}_0)$ *to be active only for* $\boldsymbol{x} \in \mathcal{D}_t$

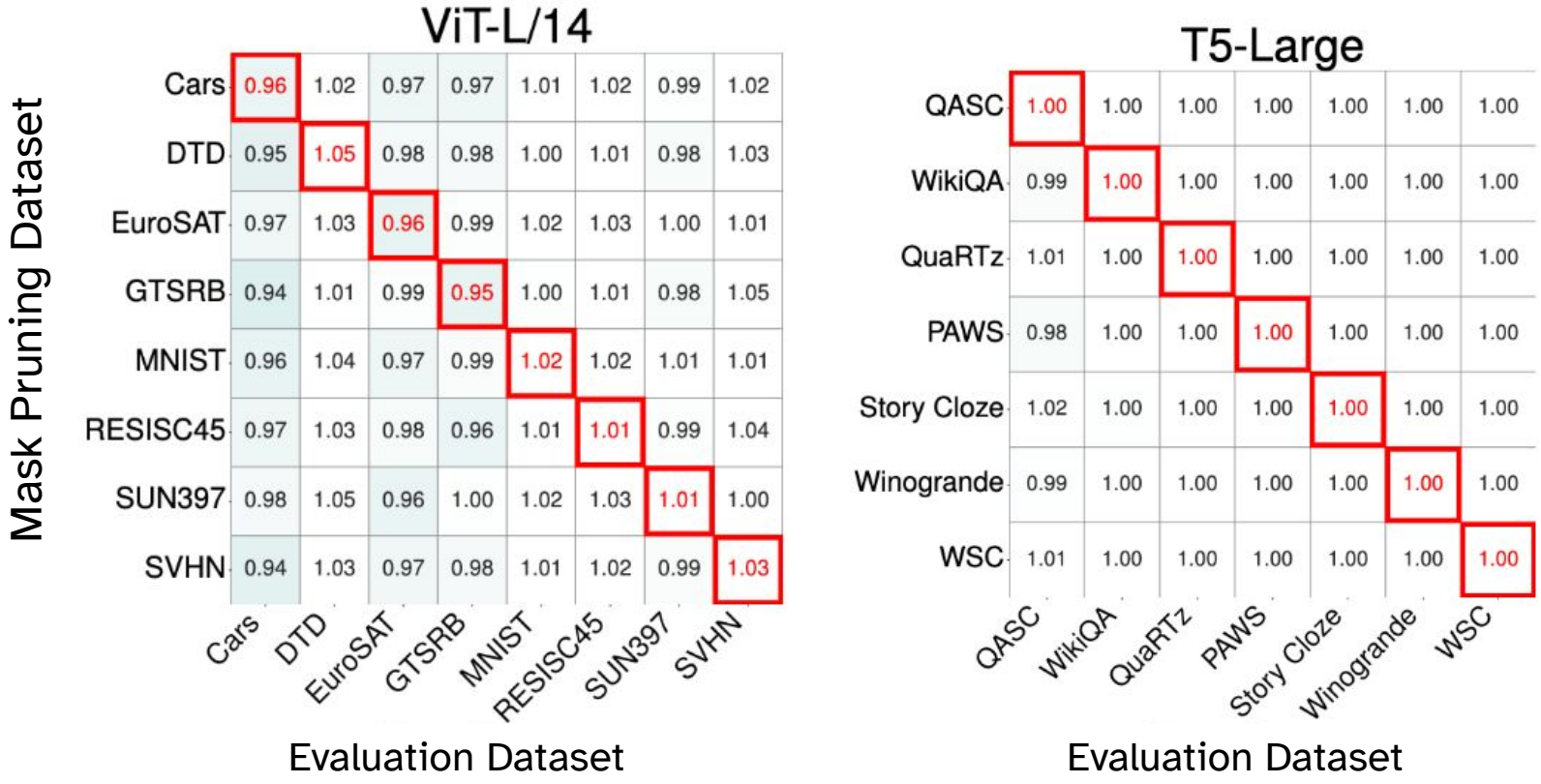
L. Iurada, ***M. Ciccone****, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025*

To satisfy these constraints during training we require data from other tasks, generally unknown a priori or not available :

$$\forall \boldsymbol{x} \in \mathcal{D}_{t' \neq t}, \quad \boldsymbol{\tau}_t^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}, \boldsymbol{\theta}_0) = 0$$

L. Iurada, M. Ciccone, T. Tommasi, “Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

We empirically found that ***low-sensitivity parameters*** $(\boldsymbol{\theta}_j \text{ s.t. } \nabla_{\boldsymbol{\theta}_j} f(\boldsymbol{x}, \boldsymbol{\theta}_0) \approx 0)$ ***are shared across tasks***



L. Iurada, M. Ciccone, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

We can derive a sparse fine-tuning rule
that approximately satisfy the constraints

Key idea: *update only parameters that have* ***low-sensitivity***

Sparse Fine-Tuning with mask c

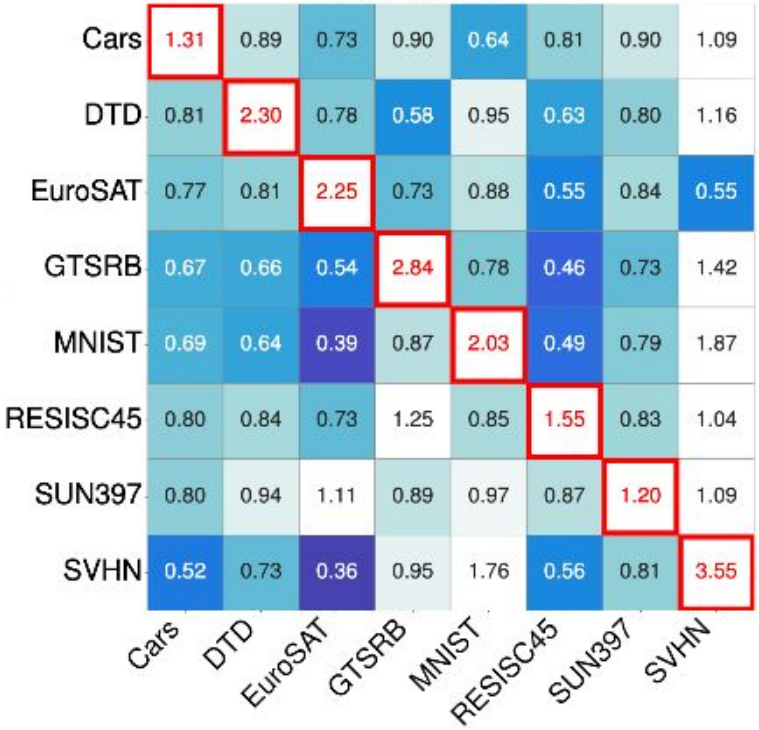
$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} - \gamma[\boldsymbol{c} \odot \nabla_{\boldsymbol{\theta}} \mathcal{L}(f(\boldsymbol{x}, \boldsymbol{\theta}^{(i-1)}), y)]$$

$$j = 1, \ldots, m \quad \boldsymbol{c}_j = \begin{cases} 1 & \text{if } \nabla_{\boldsymbol{\theta}_j} f(\boldsymbol{x}, \boldsymbol{\theta}_0) \approx 0 \\ 0 & \text{otherwise} \end{cases}$$

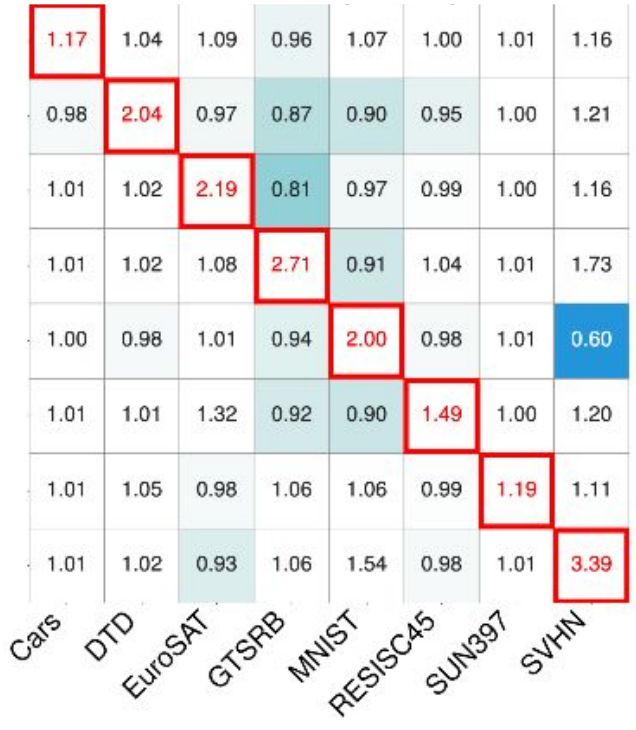
L. Iurada, M. Ciccone, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

Sparse Fine-Tuning ***preserves performance*** *of the pre-trained model*

Non-linear full FT

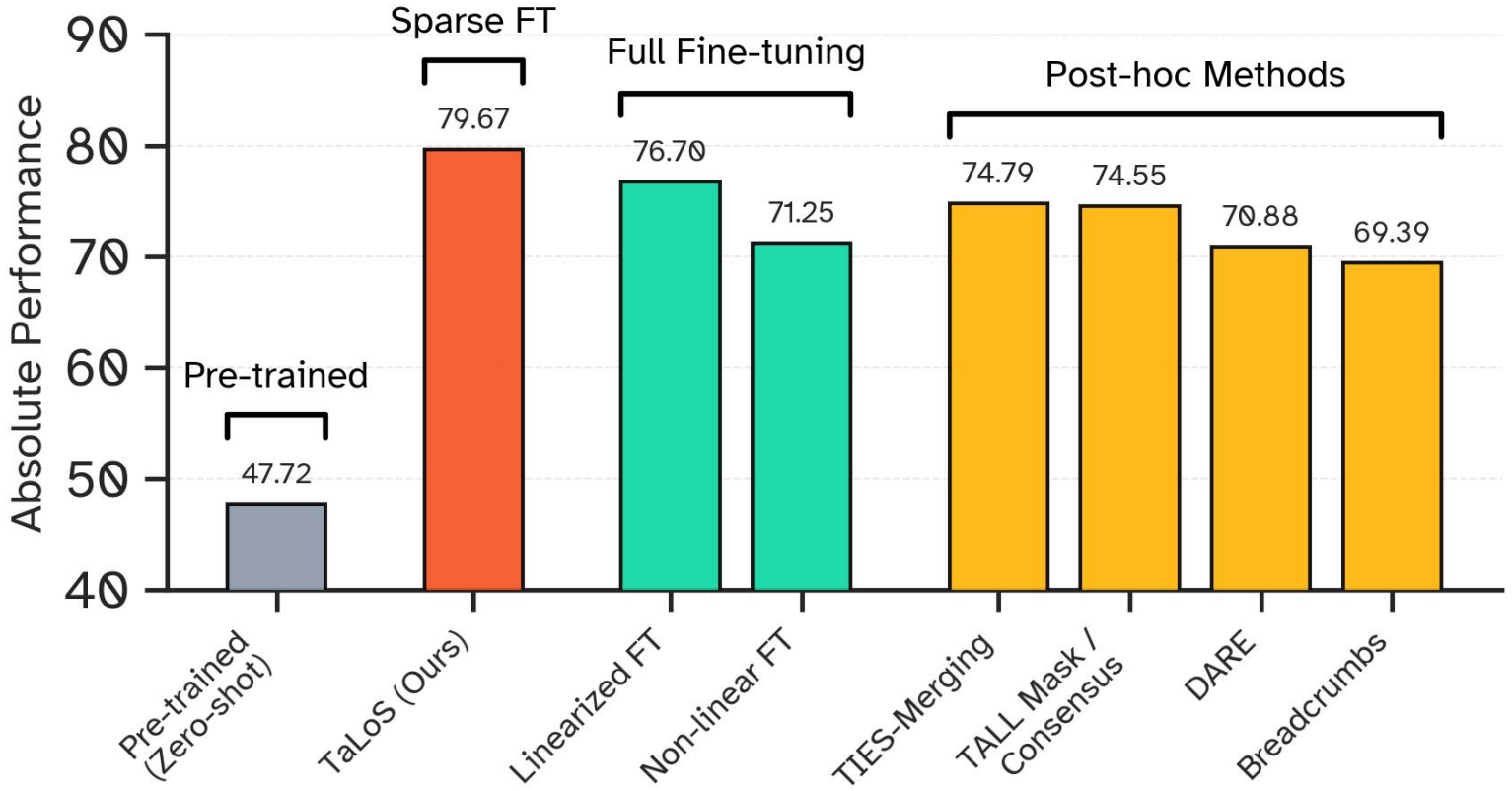


TaLoS (our sparse FT)



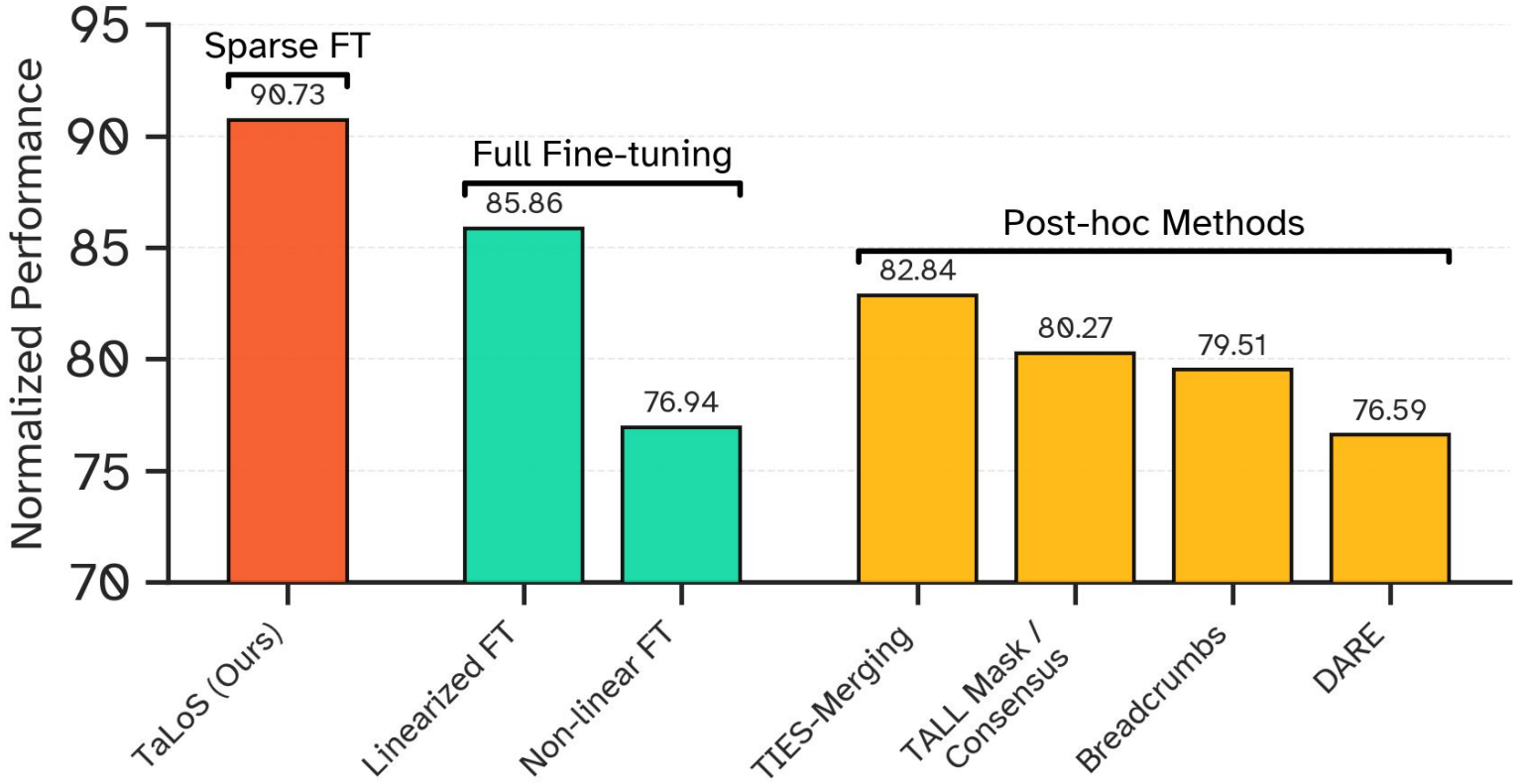
L. Iurada, M. Ciccone, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

*Sparse Fine-Tuning **reduces interference** when merging...*



L. Iurada, M. Ciccone, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

...and better retains individual performance



L. Iurada, M. Ciccone, T. Tommasi, "Efficient Model Editing with Task-Localized Sparse Fine-tuning", ICLR 2025

Takeaways

- *Learning from diverse sources presents* ***shared challenges across settings.***

- *Depending on constraints,*
  we can focus on local training, aggregation or both, with ***methods transferable across settings.***

- *Sub-communities should collaborate;*
  a ***unifying framework*** *could accelerate research and prevent redundant efforts.*

So, what's next?

Some meta-questions to spark discussions

- *How do large pre-trained models transform our approach to data heterogeneity and enable **building modular models?***

- *Can we automatically determine when **composition and transfer** are beneficial across sources, and when they are harmful?*

- *When should we prefer **merging** over other parameter isolation methods such as **routing over experts?***