UNIVERSITY OF TORONTO

VECTOR INSTITUTE

Model Merging: Foundations, Practice and Applications

Marco Ciccone

Distinguished Postdoctoral Fellow, Vector Institute

Malikeh Ehghaghi

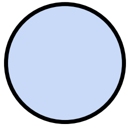
Ph.D student, University of Toronto & Vector Institute

Colin Raffel

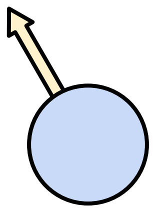
Associate Professor, University of Toronto
Associate Research Director, Vector Institute
Faculty Researcher, Hugging Face

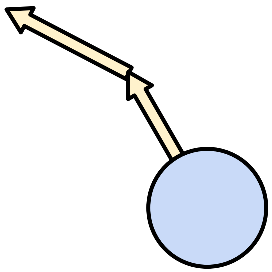
NEURAL INFORMATION PROCESSING SYSTEMS 2025

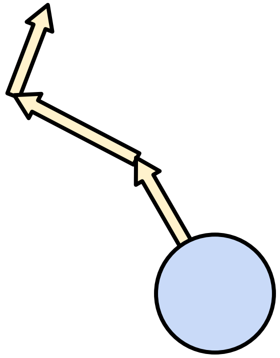
Hugging Face

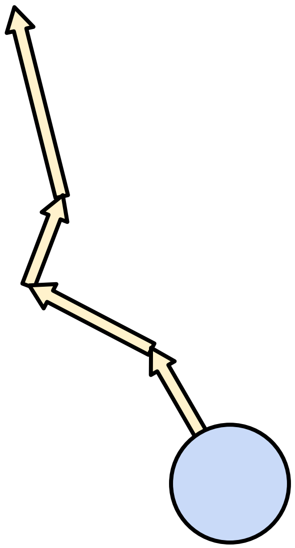


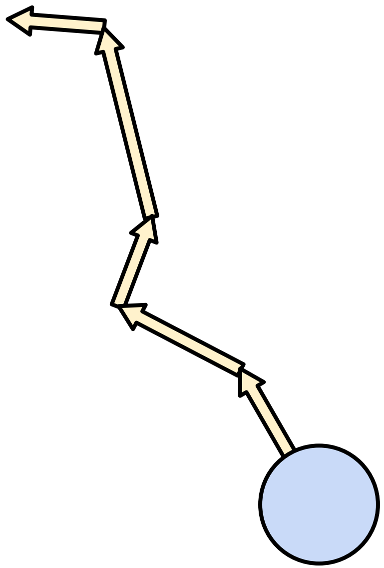
Pre-trained model

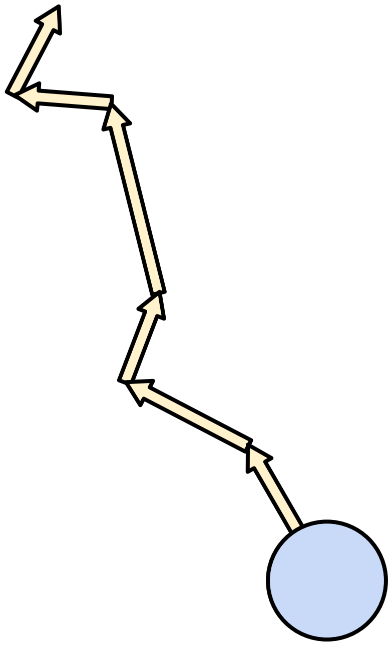


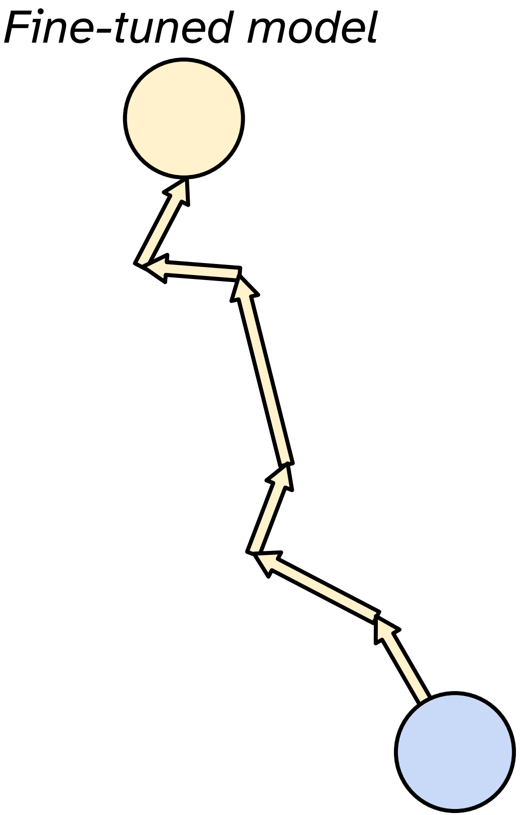


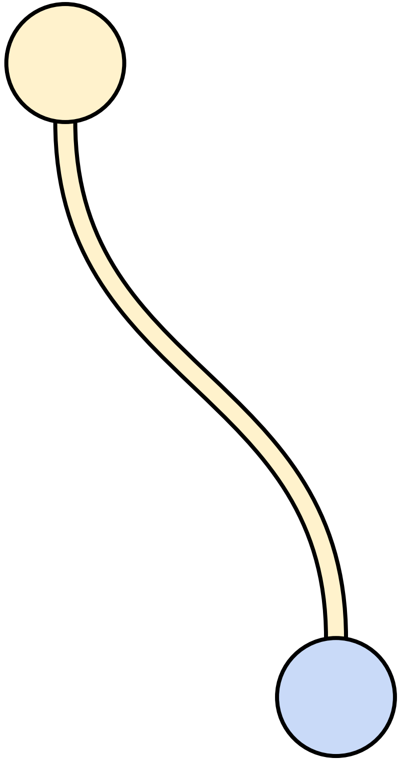


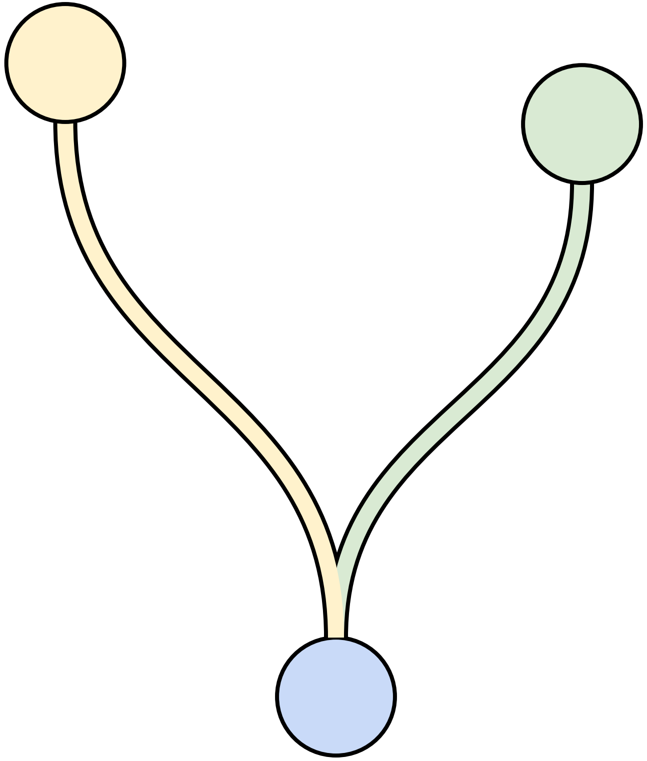




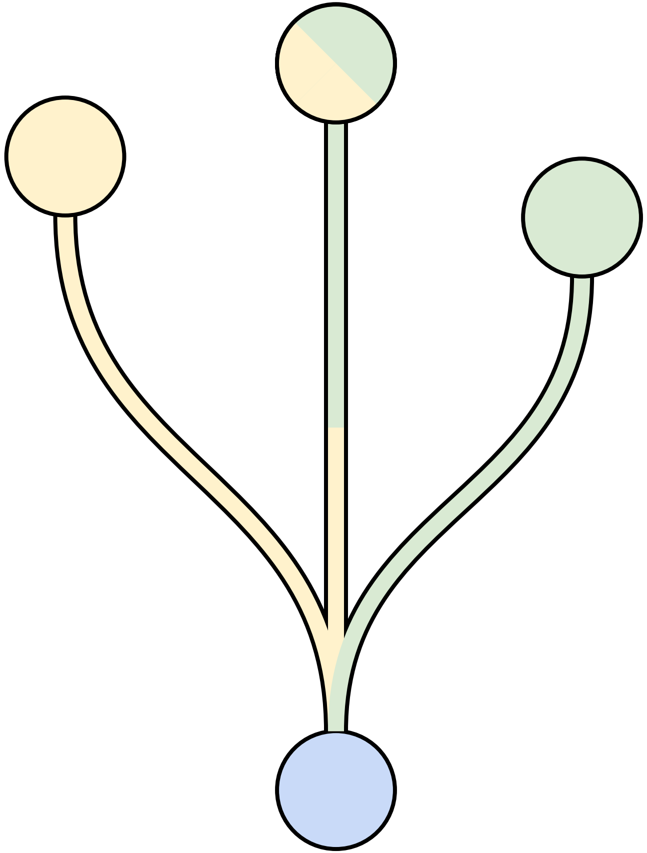


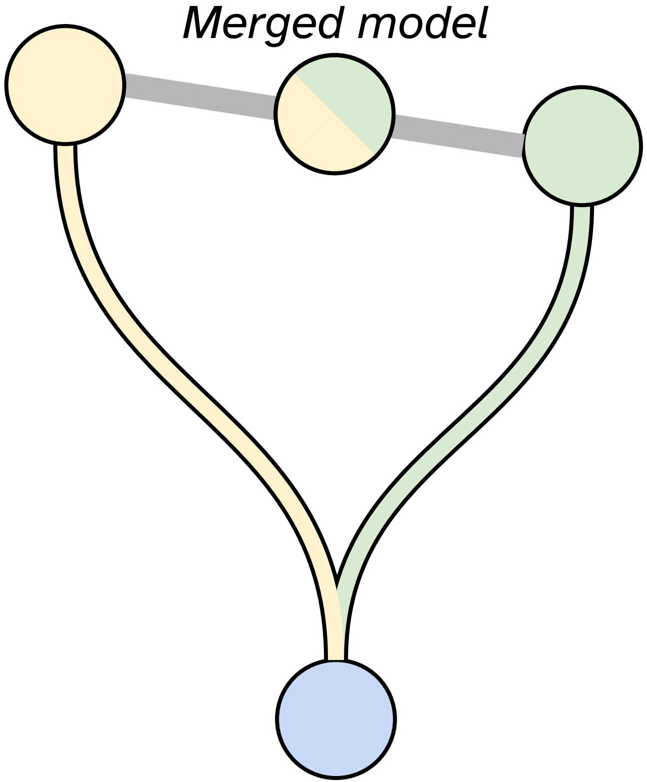

Fine-tuned model

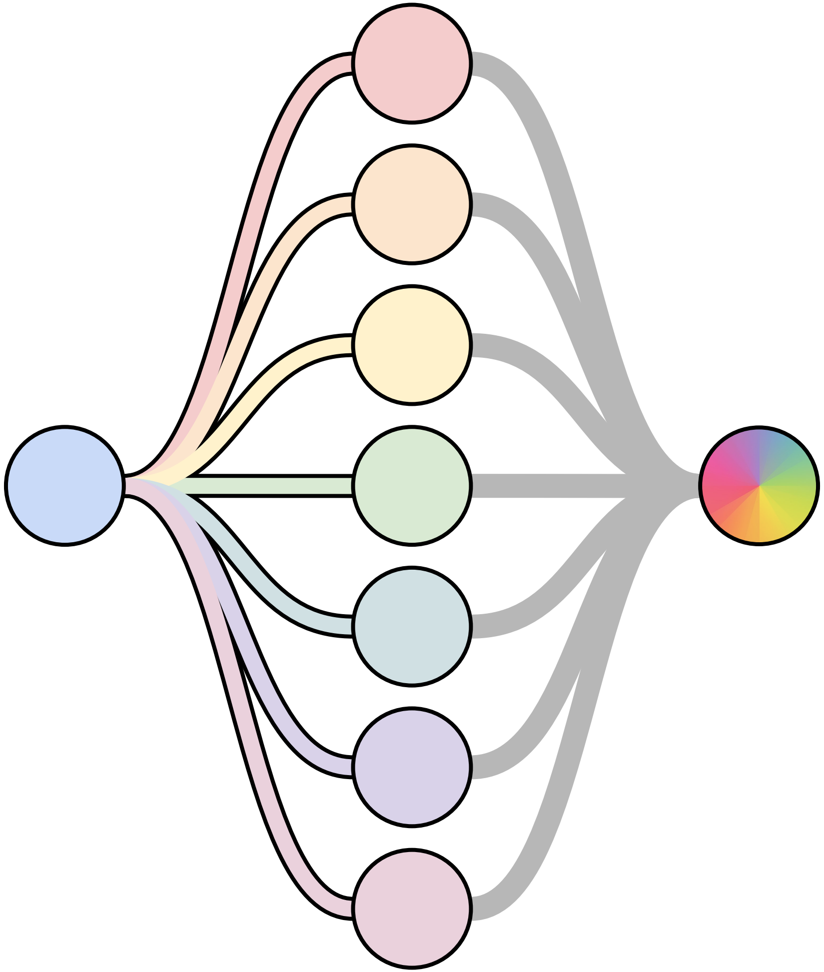


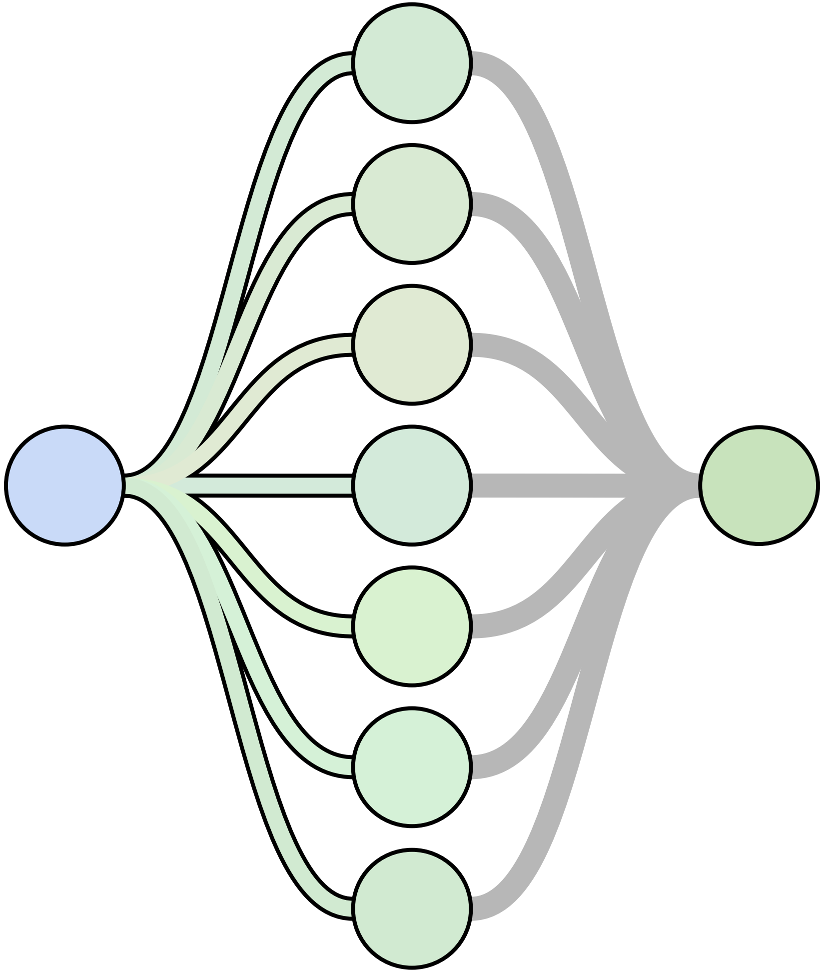


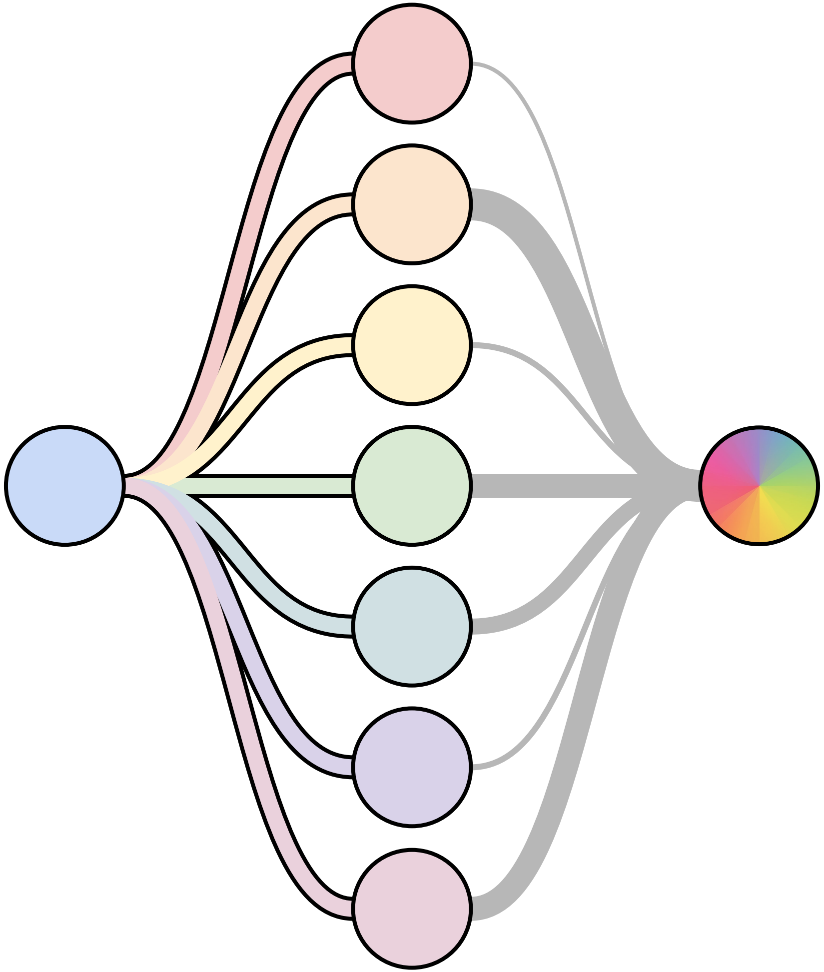
Multitask model

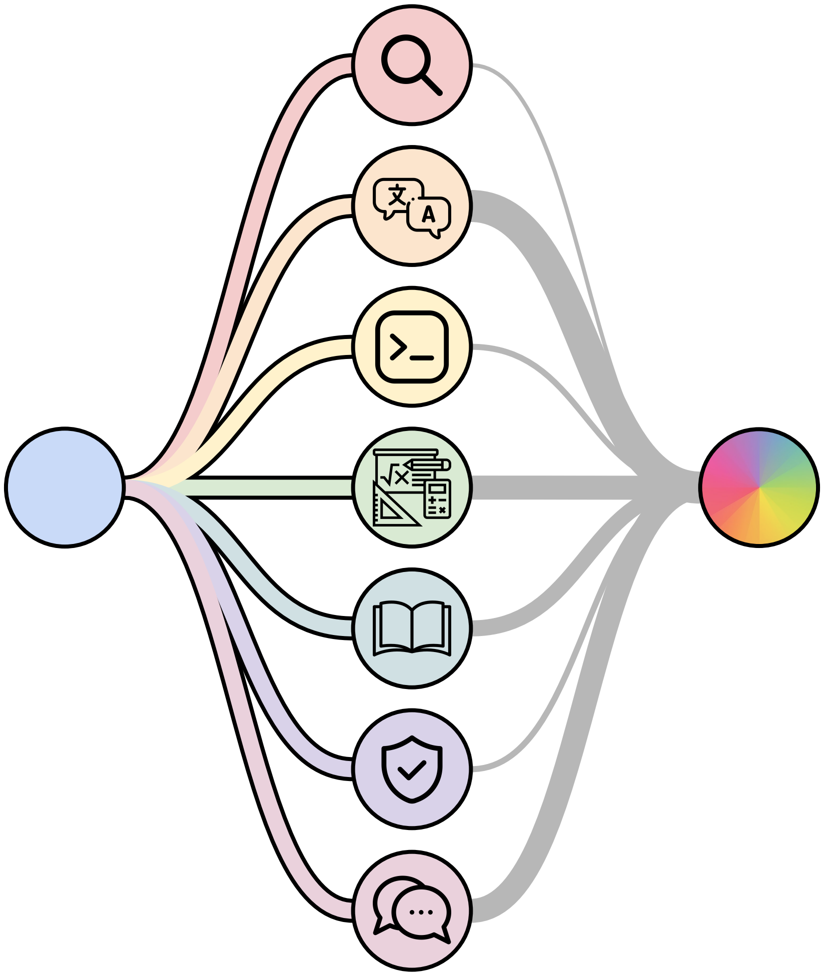


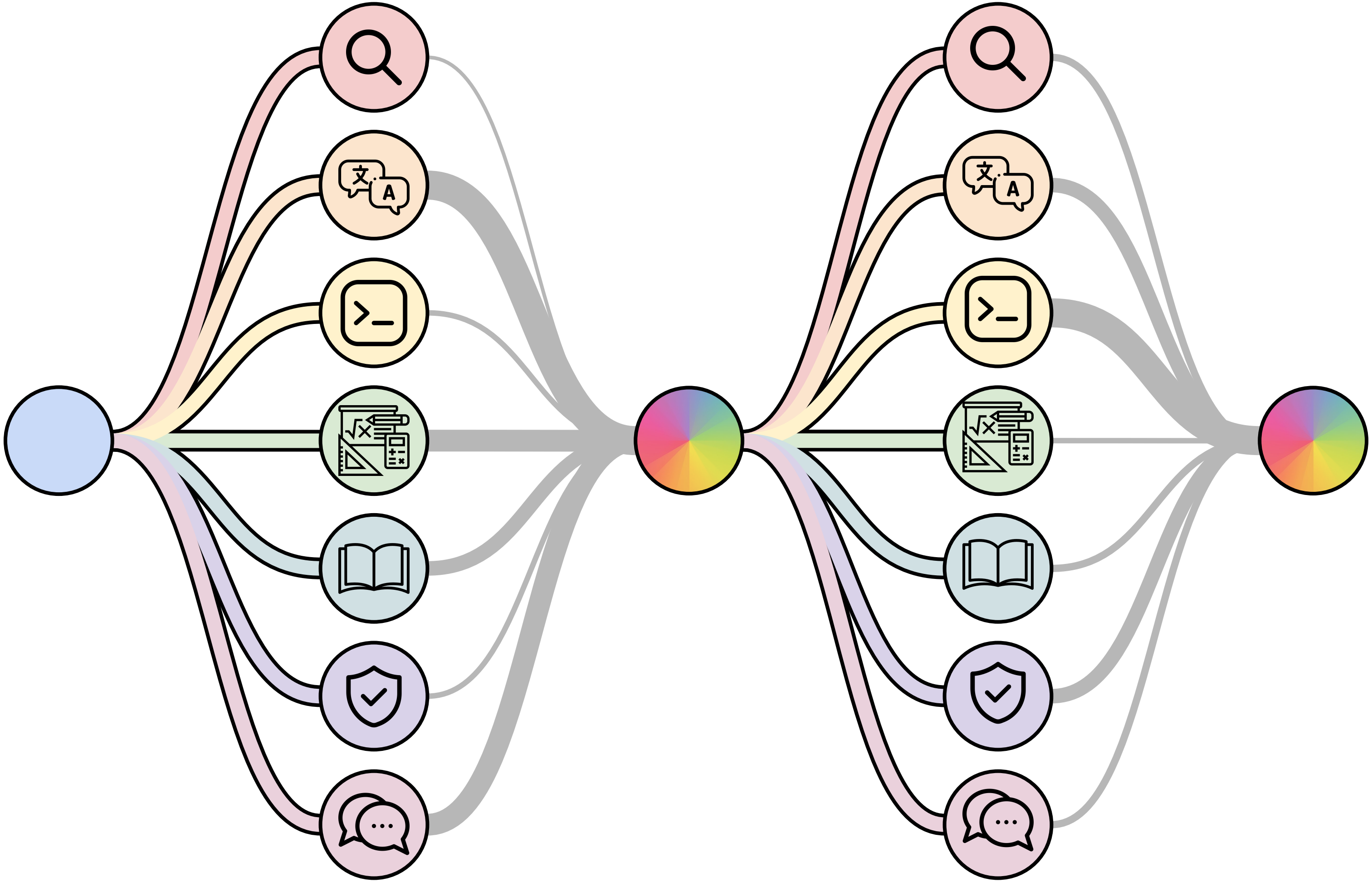

Merged model











Tutorial outline

📚 **Foundations** (Marco)

- History and foundations of parameter averaging
- Linear mode connectivity and beyond

Tutorial outline

📚 **Foundations** (Marco)

- History and foundations of parameter averaging
- Linear mode connectivity and beyond

🛠️ **Practice** (Colin)

- Overview of merging methods
- Evaluating model merging

Tutorial outline

📚 **Foundations** (Marco)

- History and foundations of parameter averaging
- Linear mode connectivity and beyond

🛠️ **Practice** (Colin)

- Overview of merging methods
- Evaluating model merging

🚀 **Applications** (Malikeh)

- Fruitful applications of merging
- Merging libraries and toolkits

Tutorial outline

📚 **Foundations** (Marco)

- History and foundations of parameter averaging
- Linear mode connectivity and beyond

🛠️ **Practice** (Colin)

- Overview of merging methods
- Evaluating model merging

🚀 **Applications** (Malikeh)

- Fruitful applications of merging
- Merging libraries and toolkits

💬 **Panel** with Charles Goddard, Sara Hooker, Edoardo Ponti, Alexandra Chronopoulou, and Margaret Li

📚 *Foundations*

OK, but what does it mean to combine models?

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

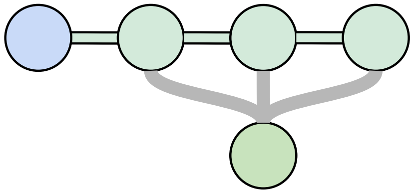
$$\sum_{i=1}^{M} \lambda_i = 1$$

Averaging weights?
That doesn't sound right.

Our roadmap

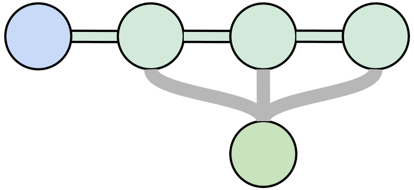
Our roadmap

single training trajectory

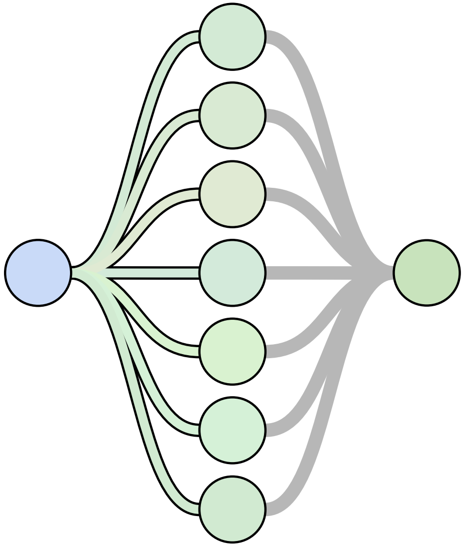


Our roadmap

single training trajectory

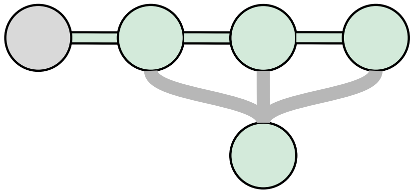


multiple training trajectory (same data)

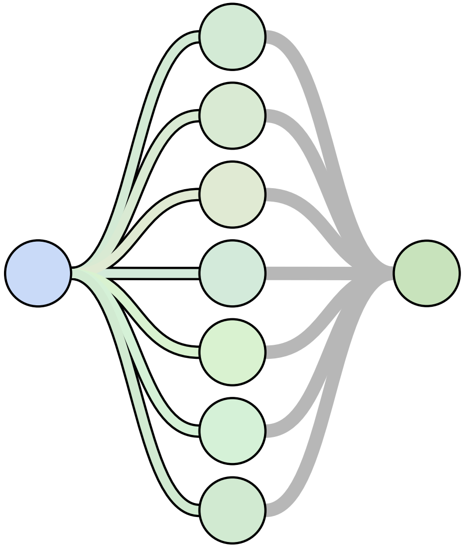


Our roadmap

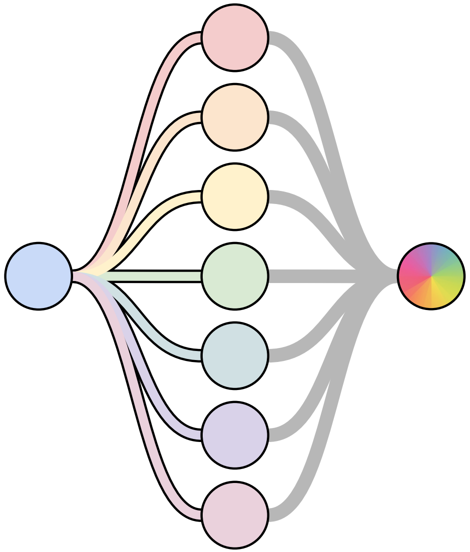
single training trajectory



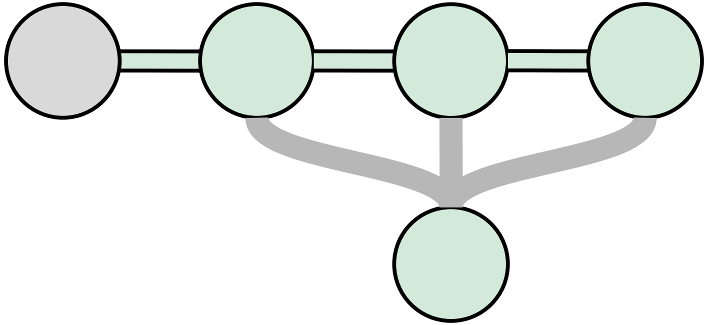
multiple training trajectory (same data)



multiple training trajectory (different data)



Single training trajectory



Polyak-Ruppert averaging

uniform average of SGD iterates accelerates stochastic optimization

SGD update

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$

Running Average

$$\bar{\theta}_{t+1} = \bar{\theta}_t + \frac{1}{t+1}(\theta_{t+1} - \bar{\theta}_t)$$

From “Efficient estimations from a slowly convergent Robbins-Monro process" by Ruppert (1988)
From “Acceleration of stochastic approximation by averaging" by Polyak & Juditsky (1992)

"**Tail averaging**" is a common "folk-law" for improving performance in deep learning

found to be useful to stabilize the training. Model evaluations are performed using a running average of the parameters computed over time.

Szegedy et al (2015)

produce long term coherent waveforms. The learning rate was held constant at 2×10^{-4}, and Polyak averaging (Polyak & Juditsky, 1992) was applied over the parameters. The

van den Oord et al (2017)

Motivated by this observation, we investigate averaged SGD (ASGD) to further improve the training process.

Merity et al (2017)

For the base models, we used a single model obtained by averaging the last 5 checkpoints, which were written at 10-minute intervals. For the big models, we averaged the last 20 checkpoints. We

Vaswani et al (2017)

rates and take two **D** steps per **G** step. For evaluation, we employ moving averages of **G**'s weights following Karras et al. (2018); Mescheder et al. (2018); Yazc et al. (2018), with a decay of 0.9999.

Brock et al (2018)

We examine two different techniques for parameter averaging in GAN training. Moving Average (MA) computes the time-average of parameters, whereas Exponential Moving Average (EMA) computes an exponentially discounted sum.

Yazıcı et al (2018)

***U**niform **A**verage applied only to the last T checkpoints*

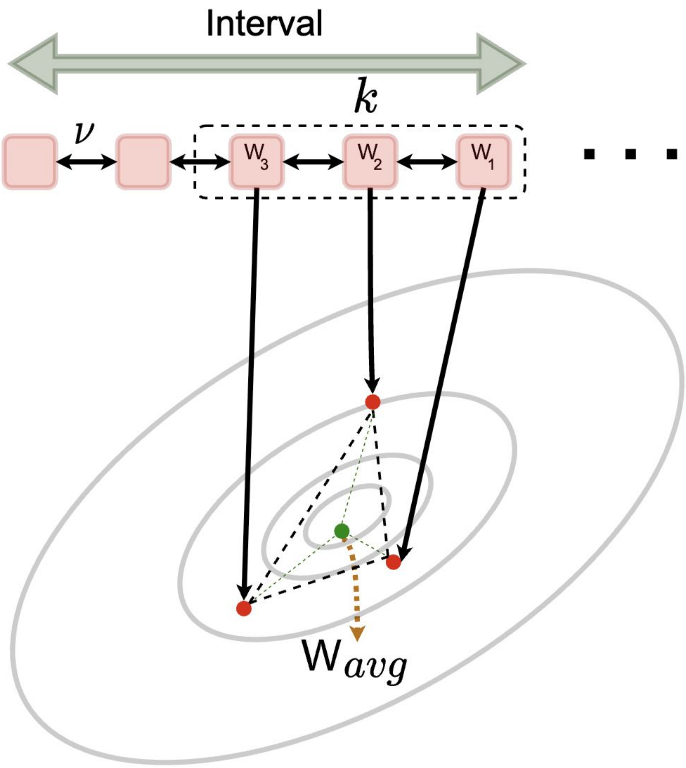
$$\bar{\theta}_T = \frac{1}{T}\sum_{t=1}^{T}\theta_t$$

***E**xponential **M**oving **A**verage, more importance to the latest checkpoints*

$$\theta_t^{\mathrm{EMA}} = \alpha\theta_{t-1}^{\mathrm{EMA}} + (1-\alpha)\theta_t$$

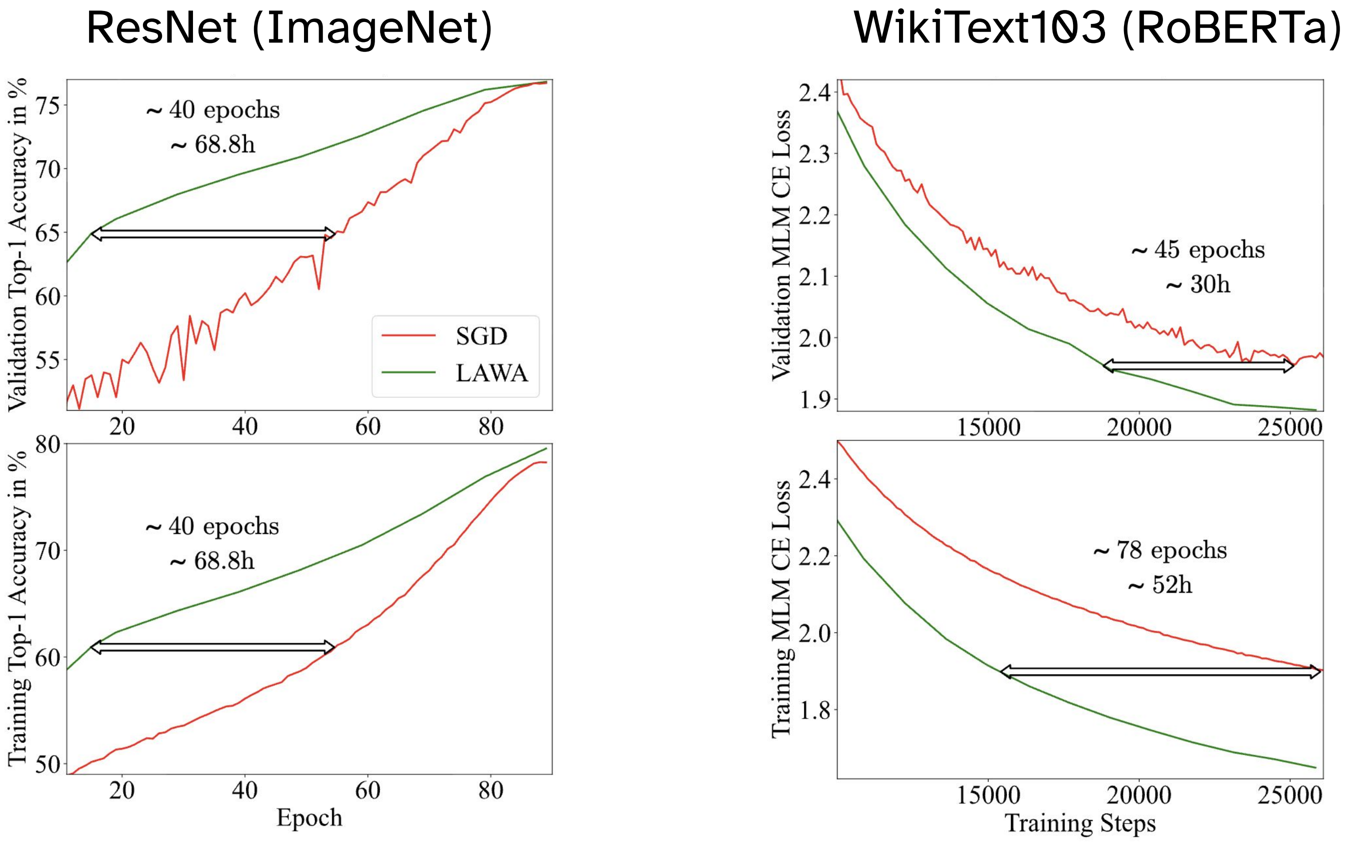
Moving Window Average: *LAtest Weight Averaging (LAWA)*

average K periodically sampled checkpoints stored in a FIFO queue



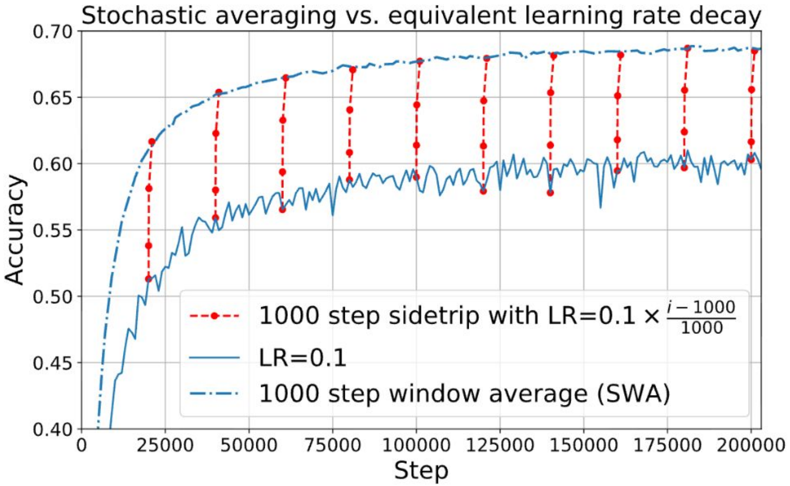
From "Stop Wasting My Time! Saving Days of ImageNet and BERT Training with Latest Weight Averaging" by Kaddour (2022)

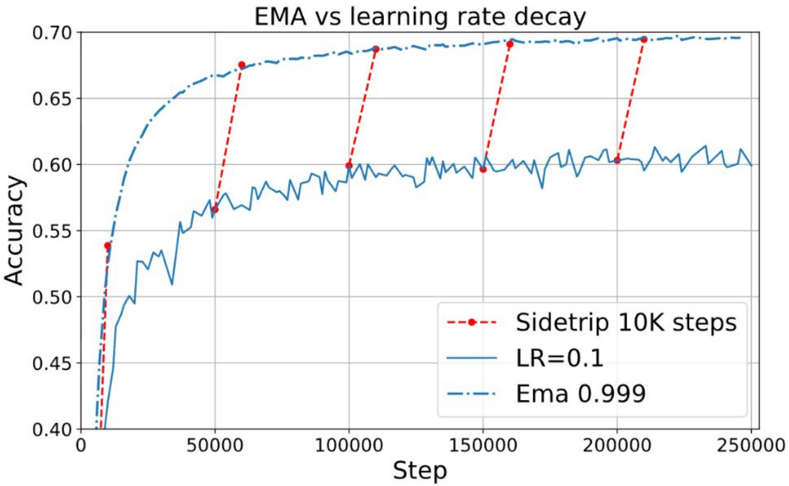
Moving Window Average: *LAtest Weight Averaging (LAWA)*



From "Stop Wasting My Time! Saving Days of ImageNet and BERT Training with Latest Weight Averaging" by Kaddour (2022)

Parameter Averaging can be equivalent to LR decay schedule

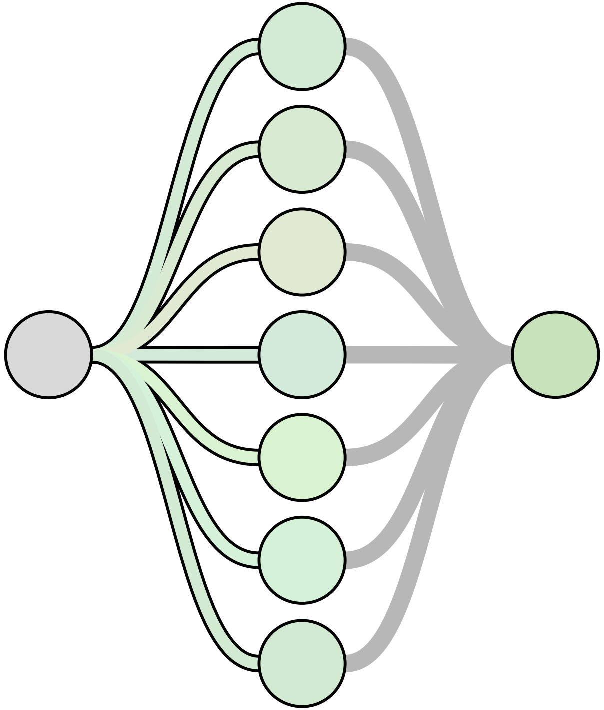




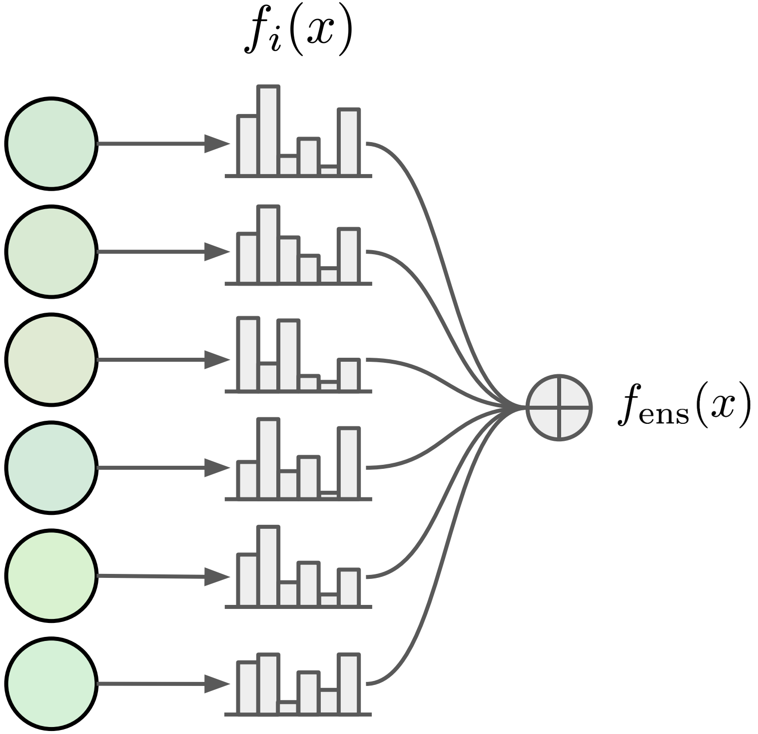
Averaging method	Equivalent learning rate for last k steps
Stochastic Weight Averaging (Izmailov et al., 2018), window size k, cycle $c = 1$	$\lambda(i)\frac{k+1-i}{k}$
Two point average k step apart	$\lambda(i)/2$
Exponential Moving Average (decay δ)	$\lambda(i)(1-(1-\delta)^{k-i})$, where $(k \gg 1/\delta)$

From "Training trajectories, mini-batch losses and the curious role of the learning rate" by Sandler et al. (2023)

Multiple training trajectory
(same data)



***Ensembling**: Combining predictions from independent models to improve model robustness and performance*



$$f_{\text{ens}}(x) = g(f_1(x), f_2(x), \ldots, f_M(x))$$

Ensembling problems

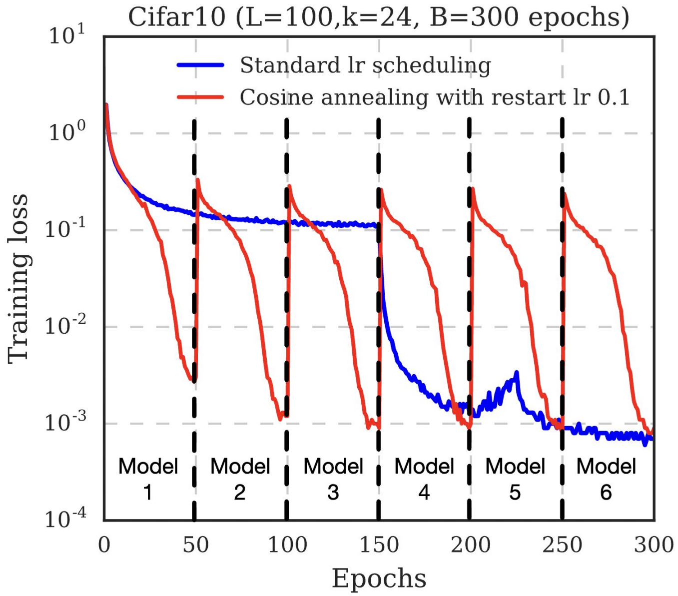
Training cost:
train N models

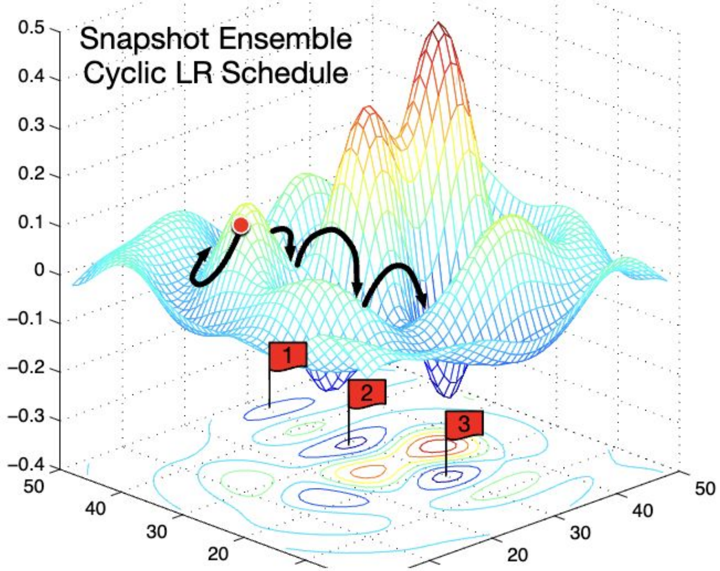
expensive for Neural Networks
need to balance diversity

Inference cost
produce N predictions

time O(N)
memory O(N)

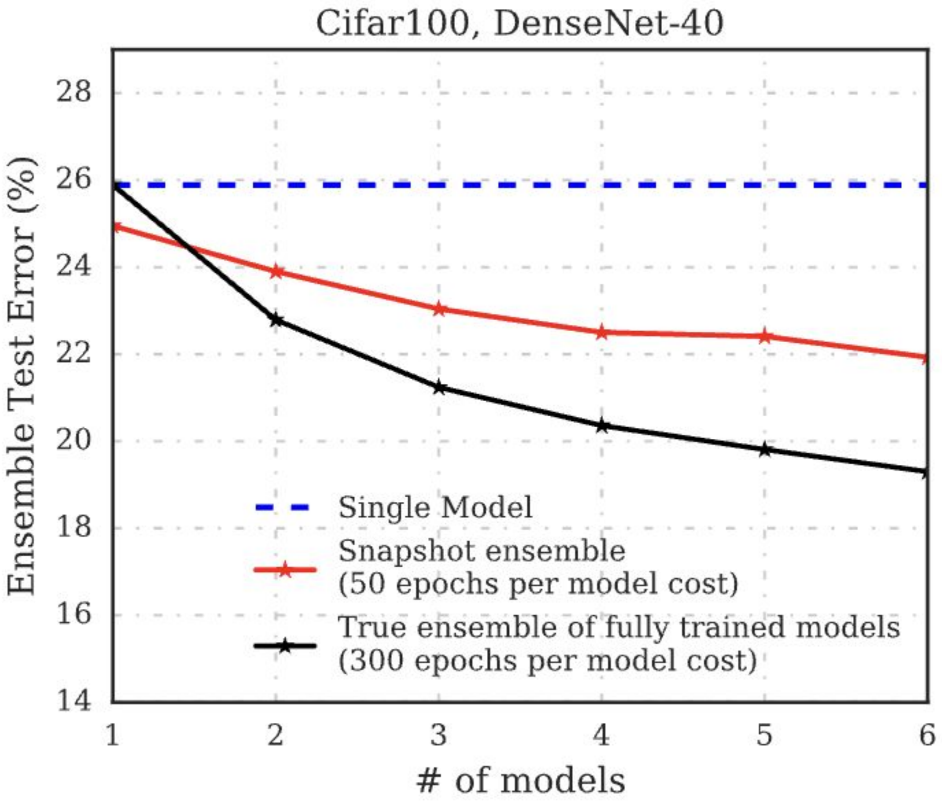
Snapshot ensembles *train 1, get M for free, by exploring different solutions*





From "Snapshot Ensembles: Train 1, get M for free" by Huang et al. (2017)

Snapshot ensembles



From "Snapshot Ensembles: Train 1, get M for free" by Huang et al. (2017)

Ensembling problems

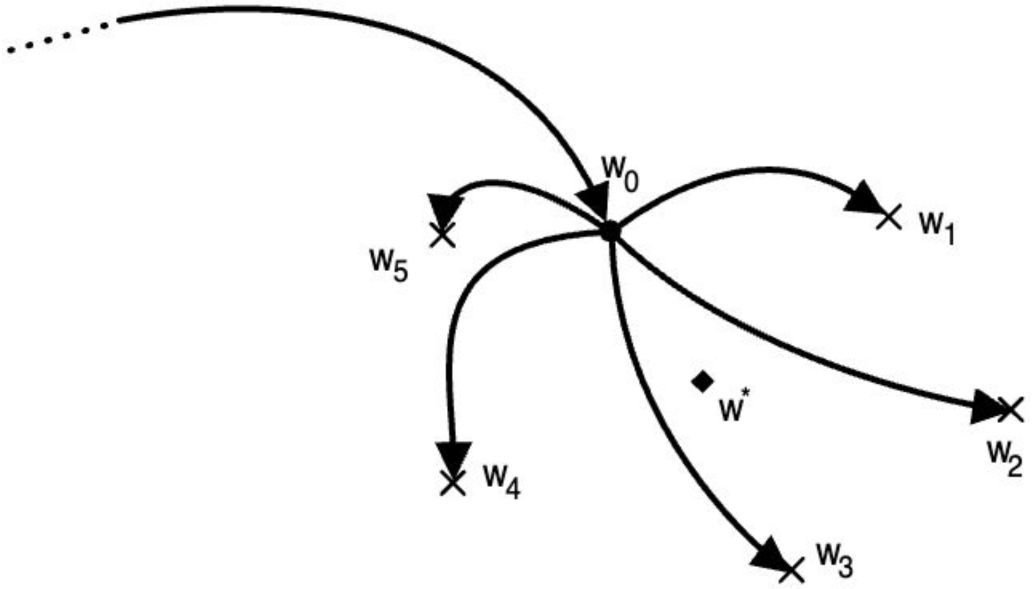
Training cost

explore solutions
while training

Inference cost

time O(N)
memory O(N)

*Simulating ensembling in weight space (Utans, **1996**)*



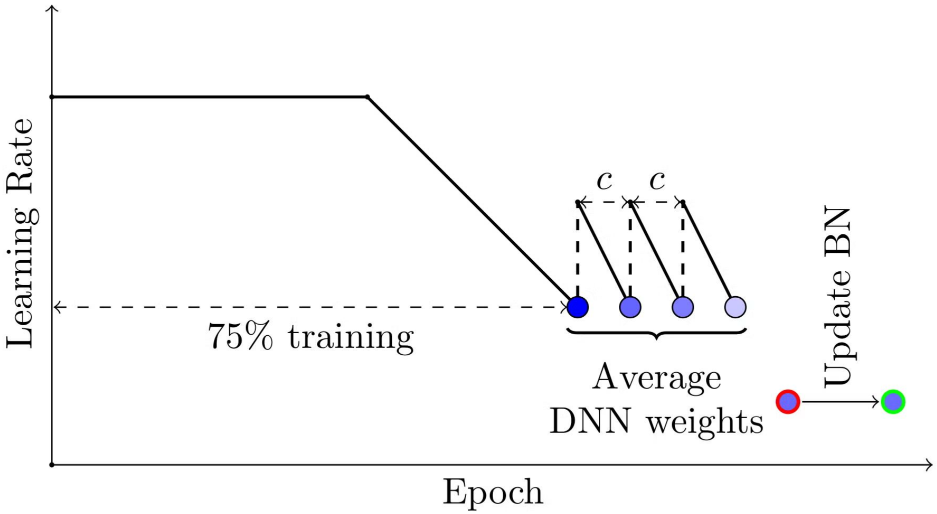
Efficient training, less overhead than from scratch

Constant prediction cost

From “Weight Averaging for Neural Networks and Local Resampling" by Utans (1996)

Stochastic Weight Averaging (SWA)

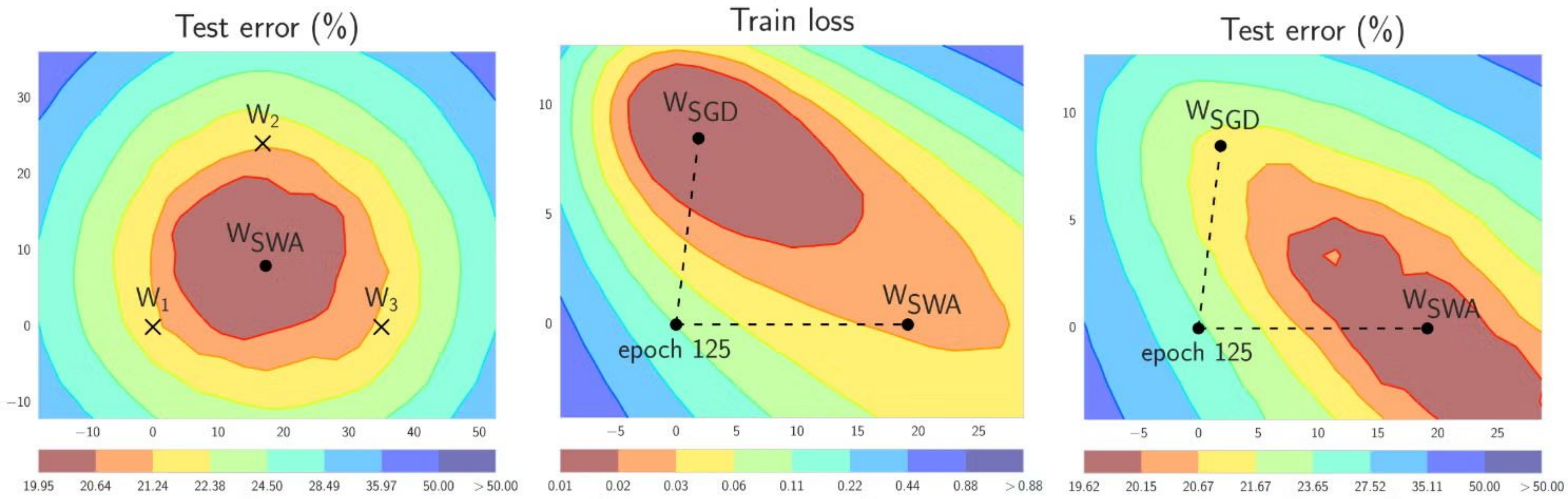
combines loss basin exploration and checkpoints averaging



$$\theta_{\text{SWA}} \leftarrow \frac{\theta_{\text{SWA}} \cdot n + \theta}{n+1}$$

From "Averaging Weights Leads to Wider Optima and Better Generalization" by Izamilov et al. (2018)

SWA improves generalization via averaging SGD solutions

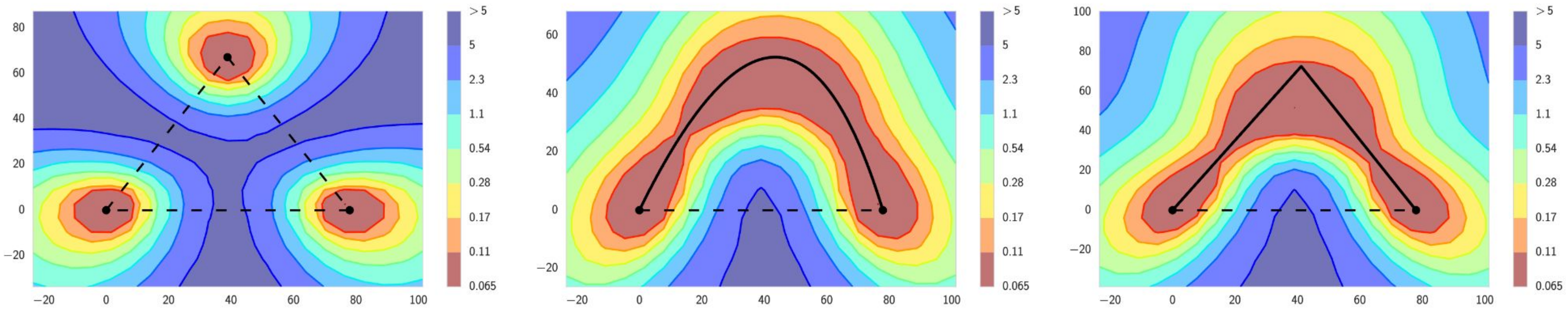


From "Averaging Weights Leads to Wider Optima and Better Generalization" by Izamilov et al. (2018)

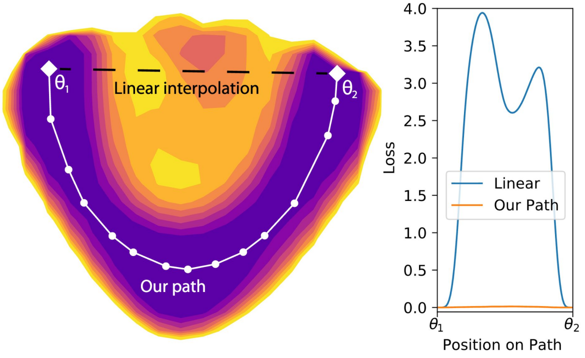
*But **when** does Weight Averaging work?*

Mode Connectivity

“minima are connected by paths with constantly low loss”



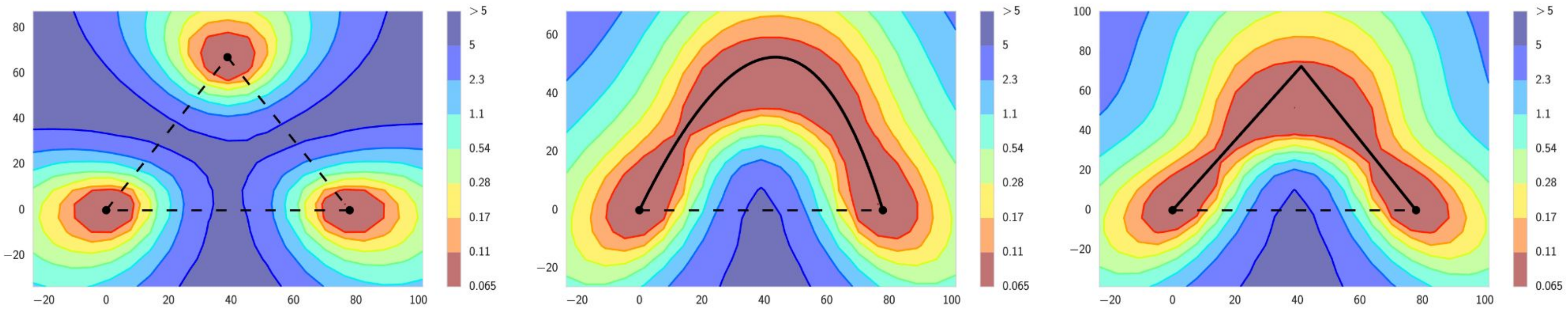
From “Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs" by Garipov et al. (2018)



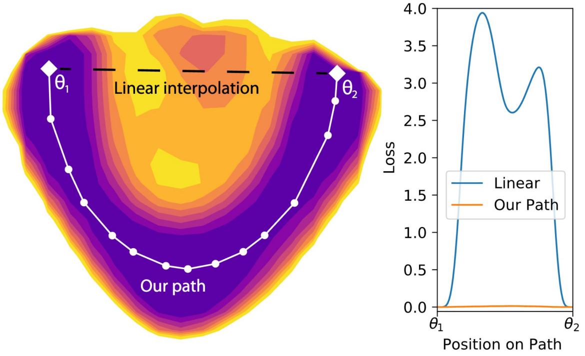
From “Essentially No Barriers in Neural Network Energy Landscape" by Draxler et al. (2018)

(Linear) *Mode Connectivity*

“minima are connected by ***(linear)*** *paths with constantly low loss”*

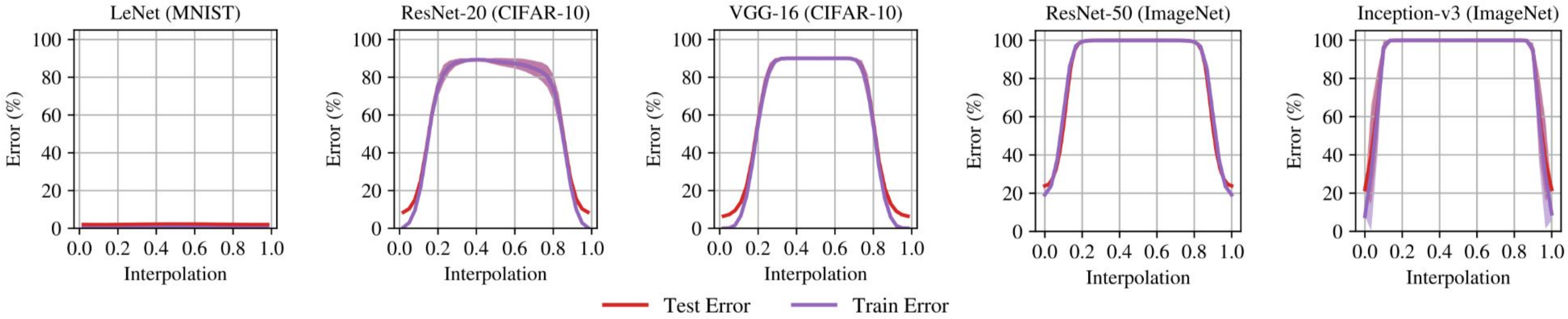


From “Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs" by Garipov et al. (2018)



From “Essentially No Barriers in Neural Network Energy Landscape" by Draxler et al. (2018)

Independently trained models are ***not linearly connected***



From "Linear Mode Connectivity and the Lottery Ticket Hypothesis" by Frankle et al. (2019)

Neural Networks have ***permutation symmetries***

The same function can be represented by different parameterizations

$$f(x) = \mathbf{W}_2\sigma(\mathbf{W}_1 x + \mathbf{b}_1) + \mathbf{b}_2$$

$$\mathbf{W}_1^{'} = \mathbf{P}\mathbf{W}_1$$ permute rows

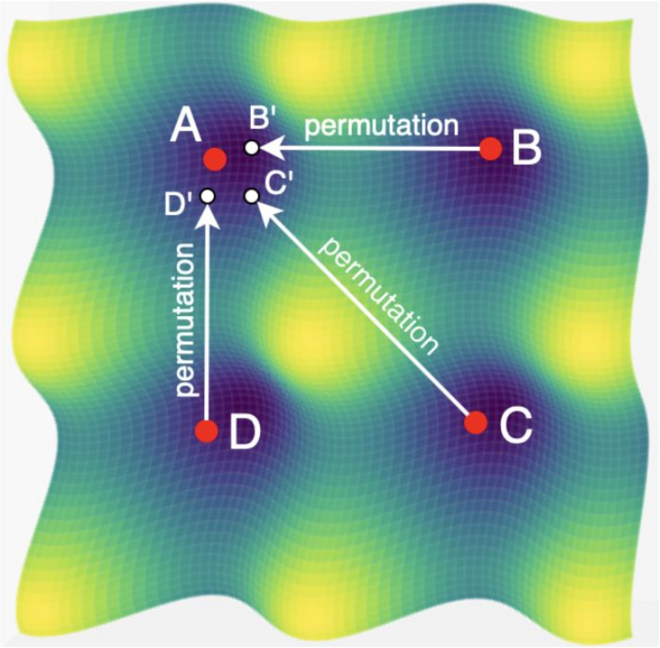
$$\mathbf{b}_1^{'} = \mathbf{P}\mathbf{b}_1,$$ permute rows

$$\mathbf{W}_2^{'} = \mathbf{W}_2\mathbf{P}^\top$$ permute columns

$$f(x) = f^{'}(x)$$

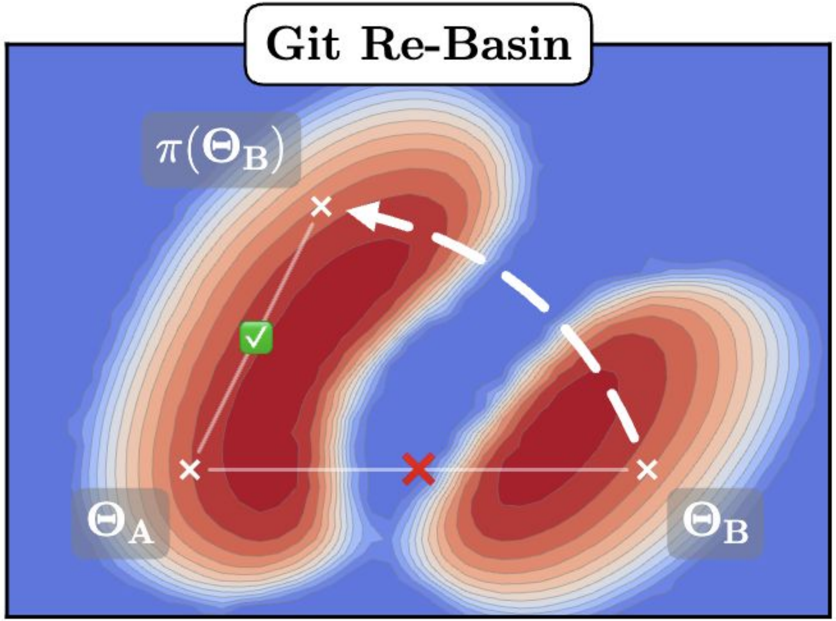
"On the algebraic structure of feedforward network weight spaces" by Hecht-Nielsen (1990)

Conjecture: *SGD solutions could be linearly connected modulo permutation symmetries*



From "The Role of Permutation Invariance in Linear Mode Connectivity of Neural Networks" by Entezari et al. (2021)

Finding explicit permutations makes model linearly connected



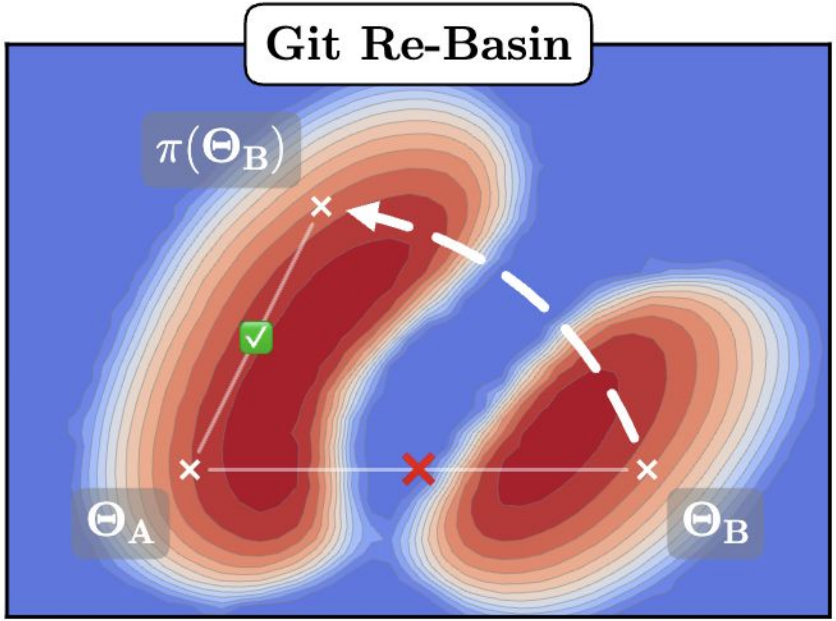
Activation Matching
(Linear Assignment Problem – Hungarian Algorithm)

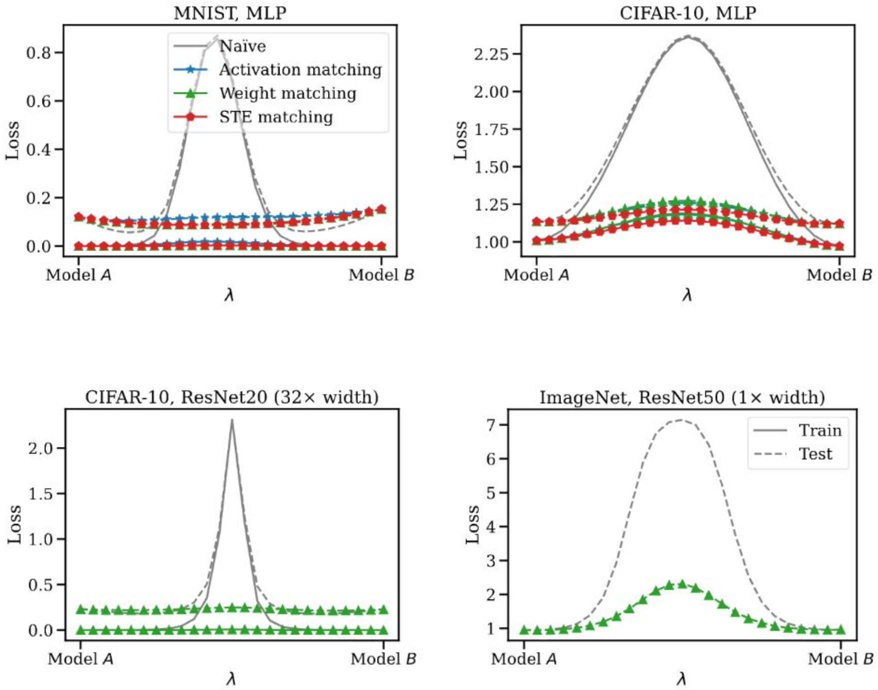
Weight Matching
Quadratic Assignment Problem- Coordinate Descent)

Learning Permutations
(directly minimize loss barrier)

From "Git Re-Basin: Merging Models modulo Permutation Symmetries" by AInsworth et al. (2022)

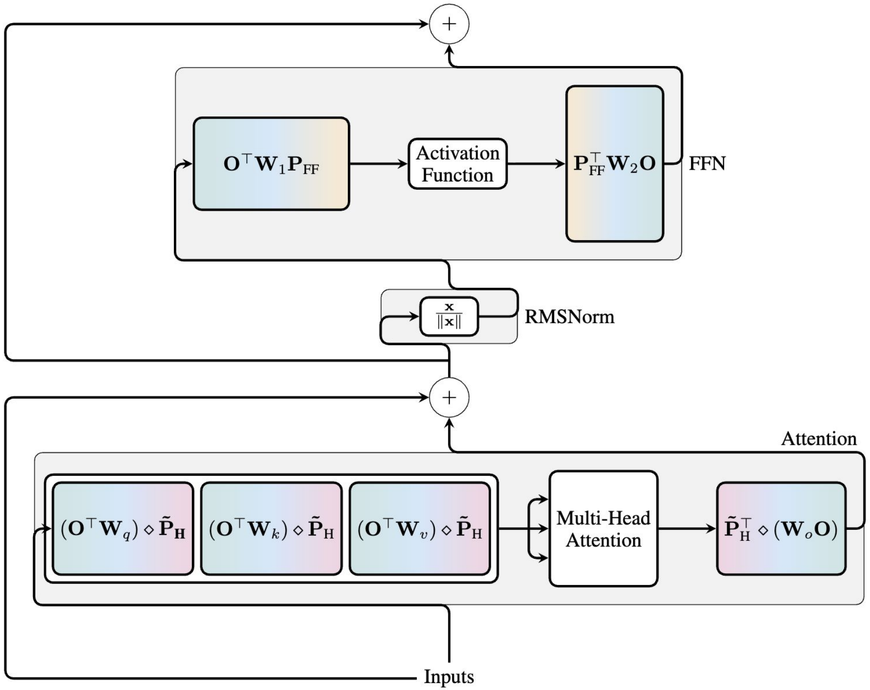
Finding explicit permutations makes model linearly connected

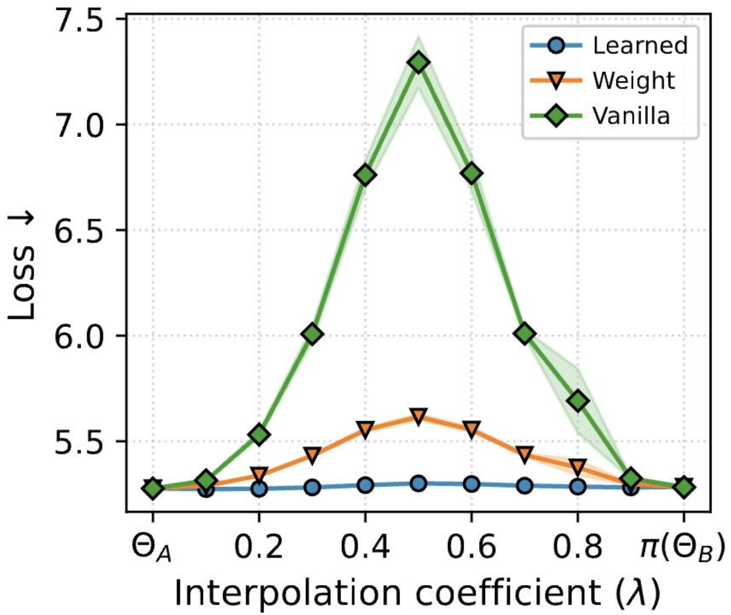




From "Git Re-Basin: Merging Models modulo Permutation Symmetries" by AInsworth et al. (2022)

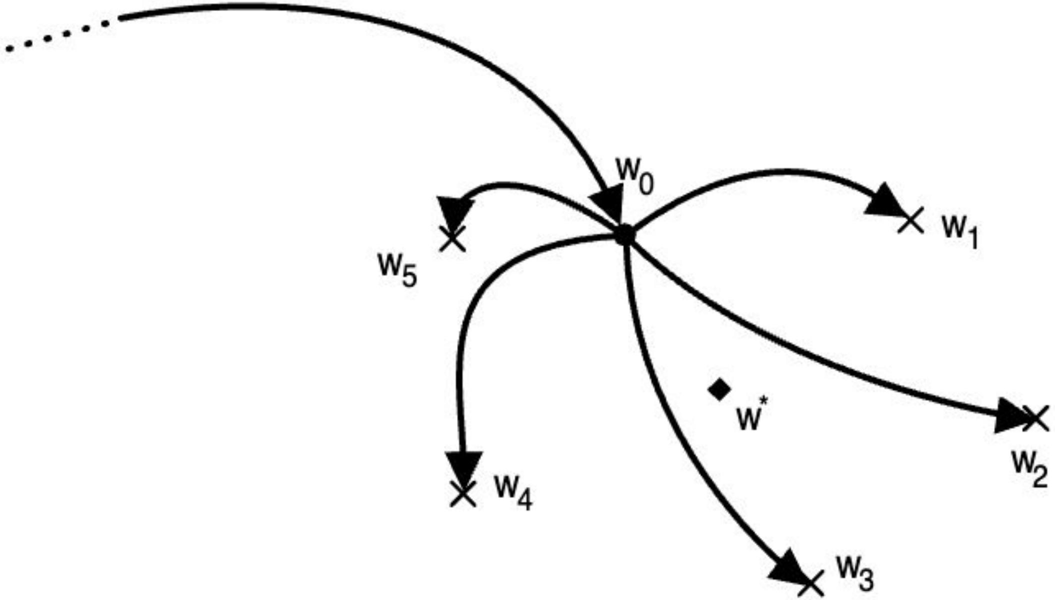
In the Transformer we have even richer continuous symmetries





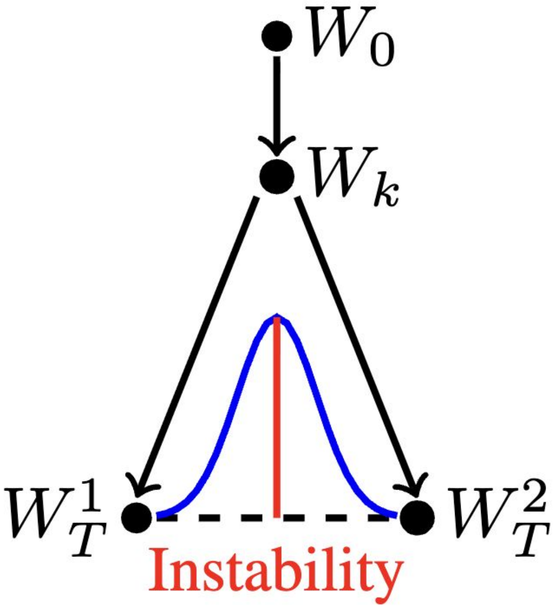
From "Generalized Linear Mode Connectivity for Transformers" by Theus et al. (2025)

Sharing part of the same trajectory enables LMC



From "Weight Averaging for Neural Networks and Local Resampling" by Utans (1996)

***Spawning Experiment**: after k iterations, add different SGD noise*

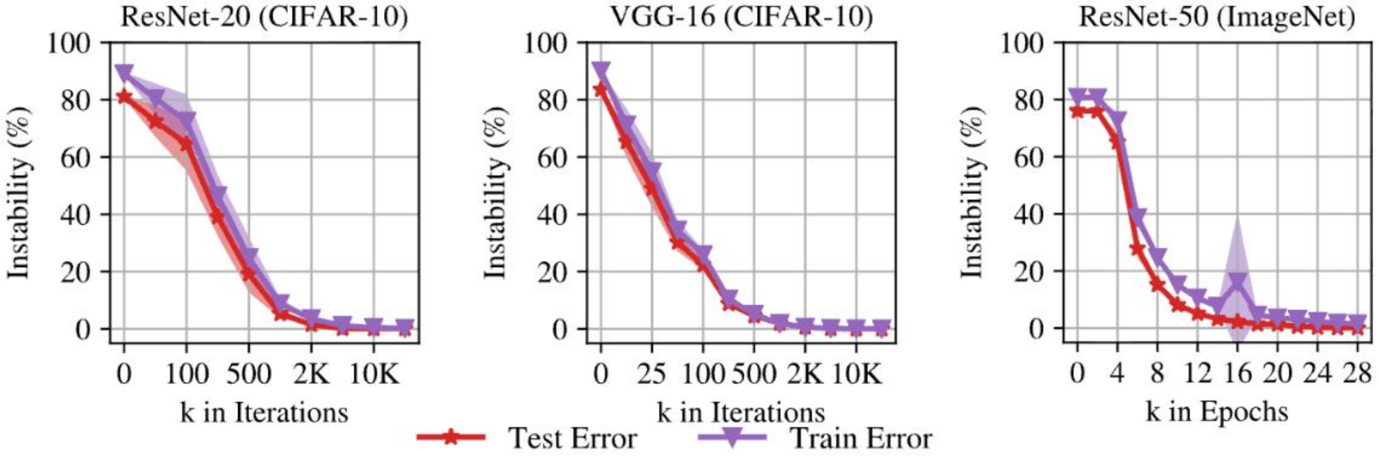


$$\mathcal{E}_{\text{sup}}(W_1, W_2) - \text{mean}(\mathcal{E}(W_1), \mathcal{E}(W_2))$$

From "Linear Mode Connectivity and the Lottery Ticket Hypothesis" by Frankle et al. (2019)

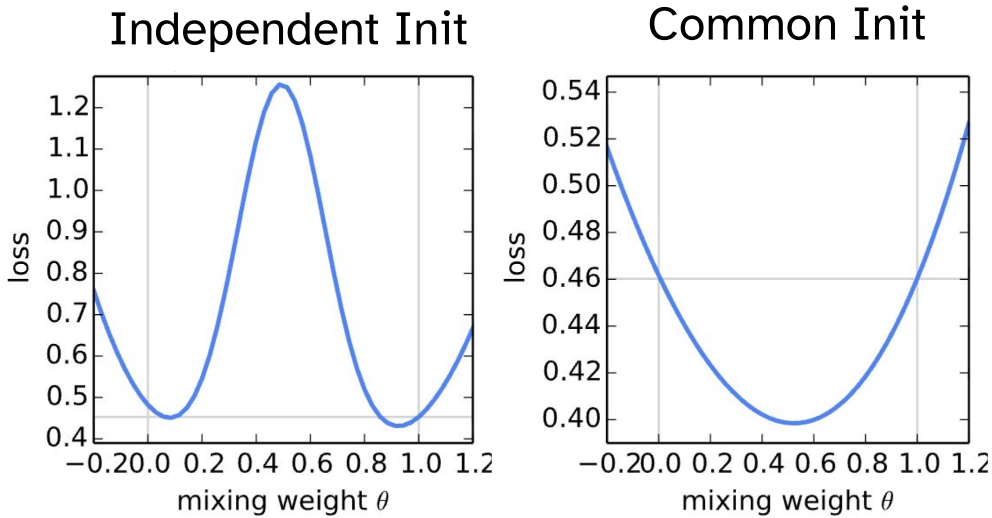
Linear *Mode Connectivity (LMC) emerges through training*

after shared training, ***independent*** *optimization trajectories*
settle into the same linearly connected basin



From "Linear Mode Connectivity and the Lottery Ticket Hypothesis" by Frankle et al. (2019)

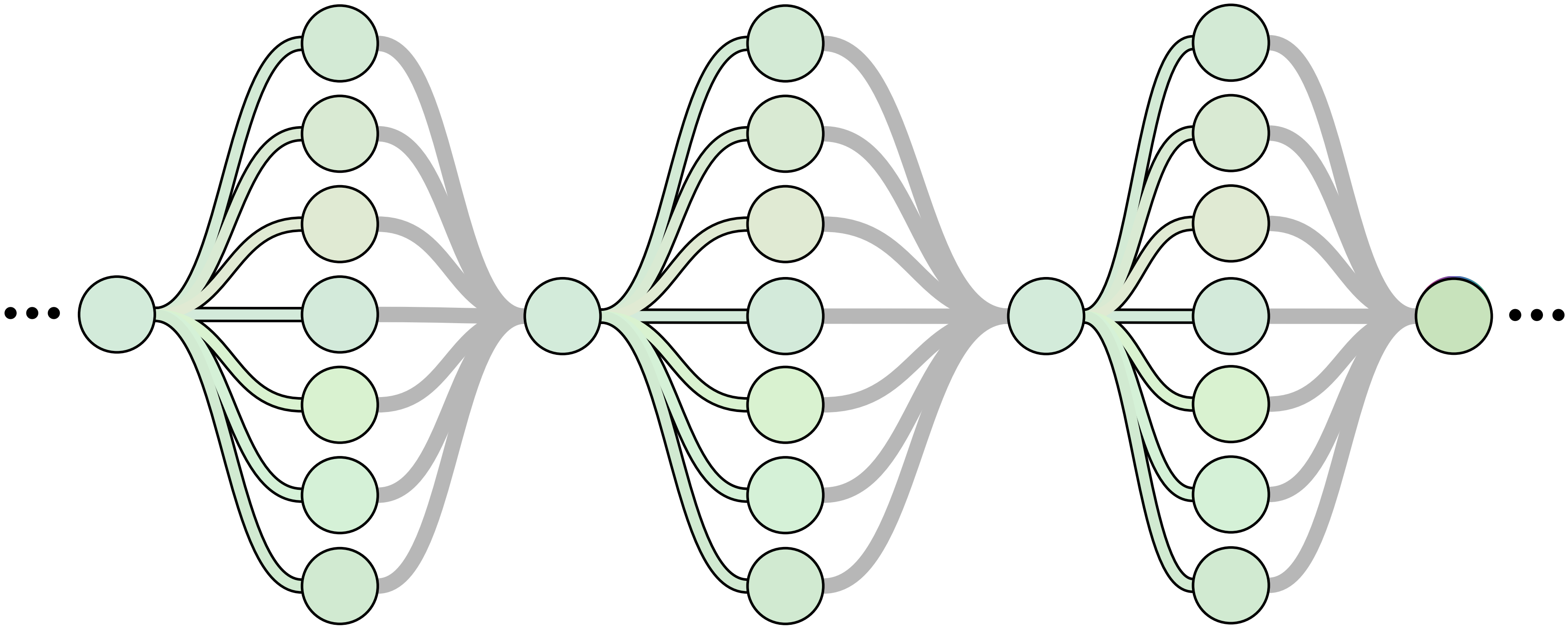
Parameter Averaging *as core operation for* ***Federated Learning***



$$\theta^{t+1} = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \theta_i^t$$

*From "Federated Learning of Deep Networks using **Model Averaging**" by McMahan et al. (2016)*
renamed later as "Communication-Efficient Learning of Deep Networks from Decentralized Data"

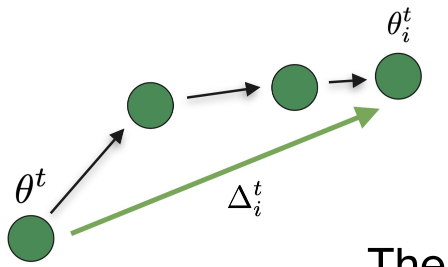
Federated Learning as multistage training and merging



From "Communication-Efficient Learning of Deep Networks from Decentralized Data" by McMahan et al. (2016)

Parameter Averaging *as core operation for* ***distributed learning***

$$\theta^{t+1} = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \theta_i^t = \theta^t - \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \underbrace{\theta^t - \theta_i^t}_{\Delta_i^t}$$



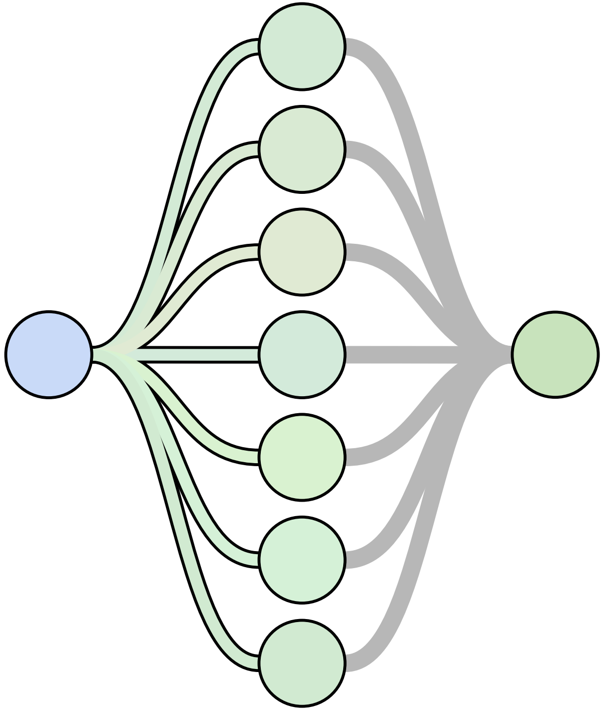
$$\theta^{t+1} = \theta^t - \eta \Delta^t$$

The displacement from initialization can be used as
pseudo-gradient for an outer/server optimizer

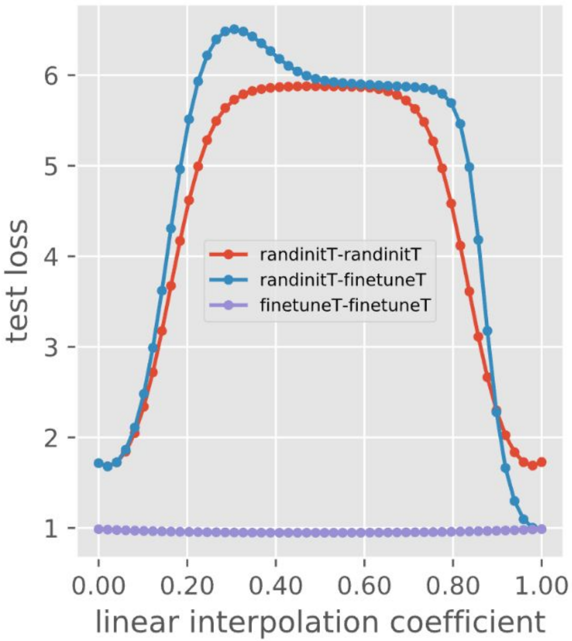
$$\theta^{t+1} = \textsc{ServerOpt}(\theta^t, -\Delta^t, \eta)$$

From "Adaptive Federated Optimization by Reddi et al. (2020)

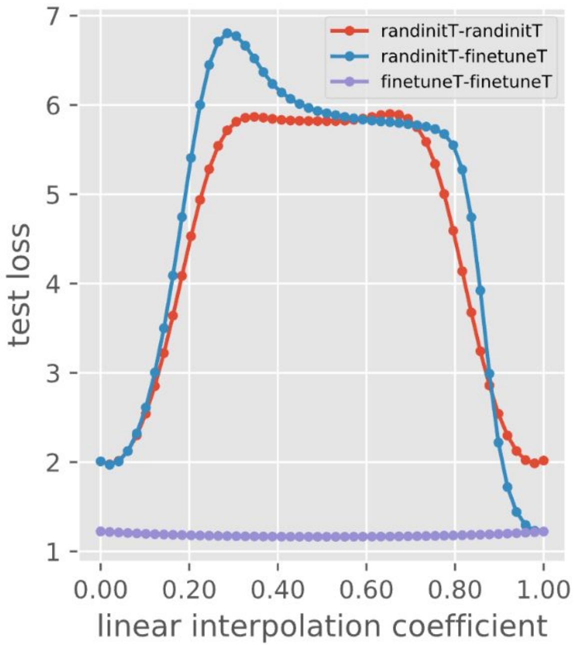
Multiple training trajectories
*(**fine-tuning** with new data)*



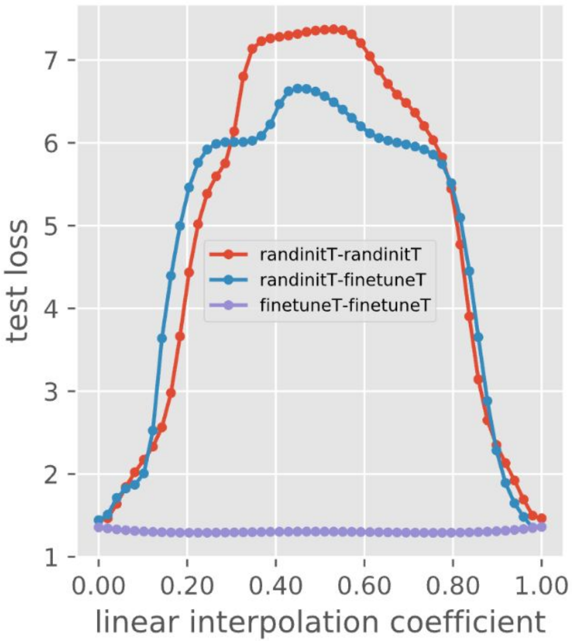
Pretraining *(ImageNet) consistently removes barriers during Fine-tuning (DomainNet)*



real



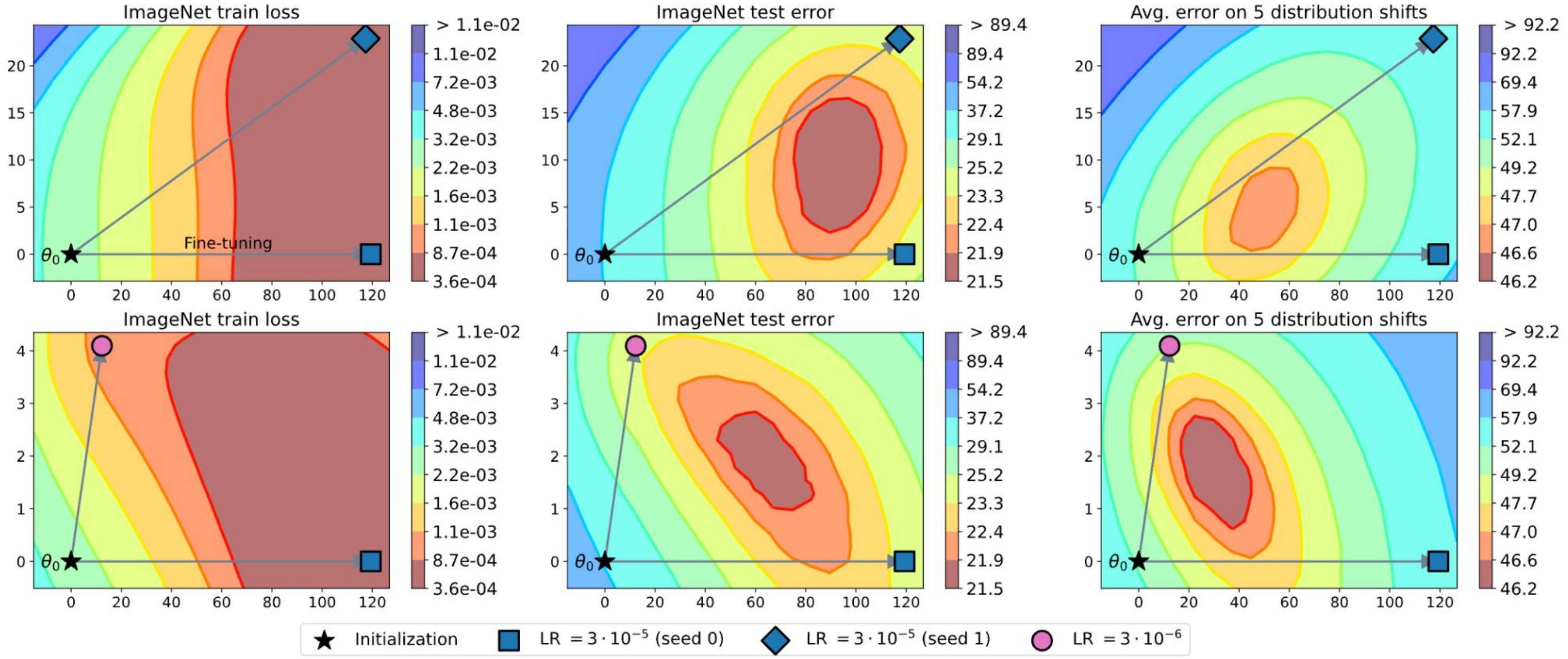
clipart



quickdraw

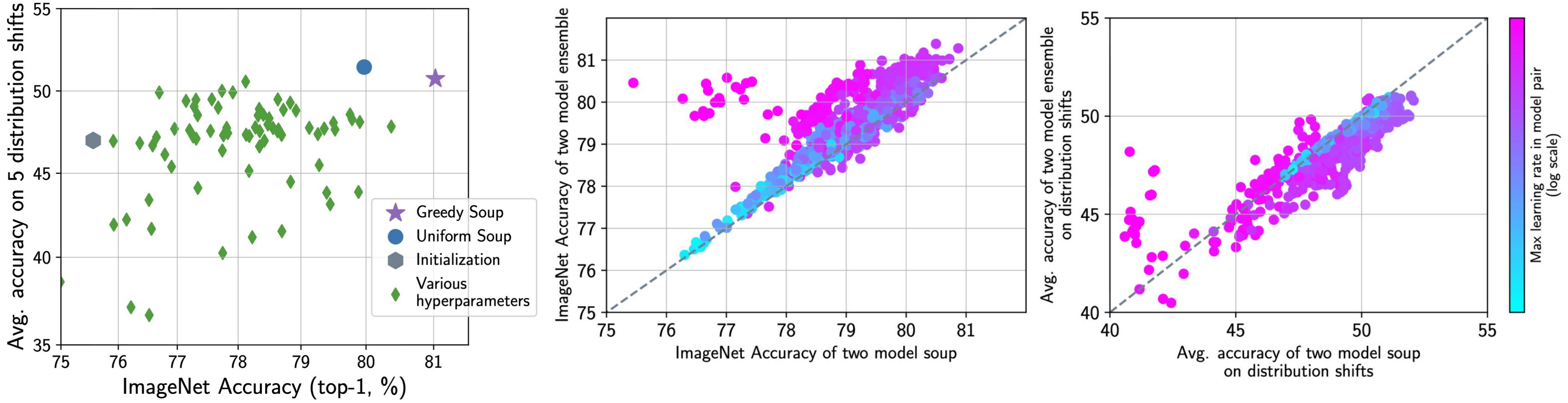
From "What is being transferred in transfer learning?" by Neyshabur et al. (2020)

Model Soups**: The solution with the **highest accuracy** is often not a fine-tuned model but **lies between fine-tuned models



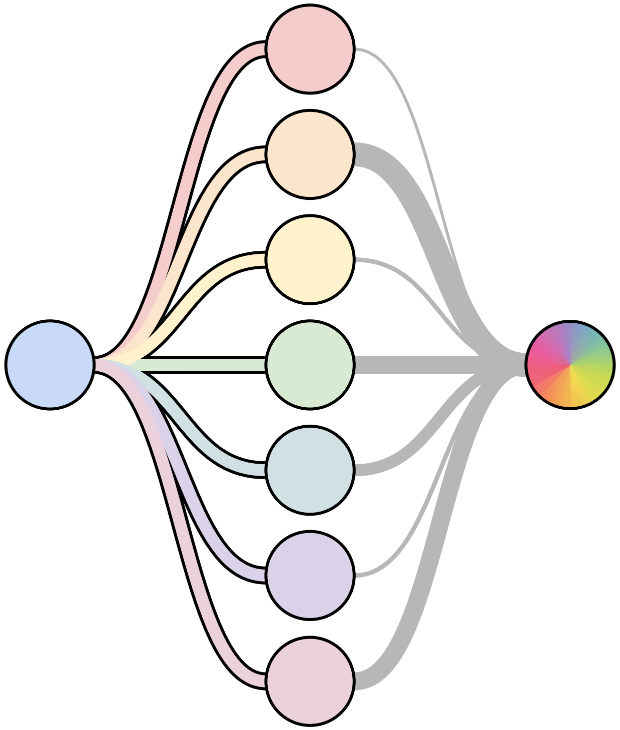
From "Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time" by Wortsman et al. (2022)

Ensemble *performance* ***correlates*** *with* ***model soup*** *performance*

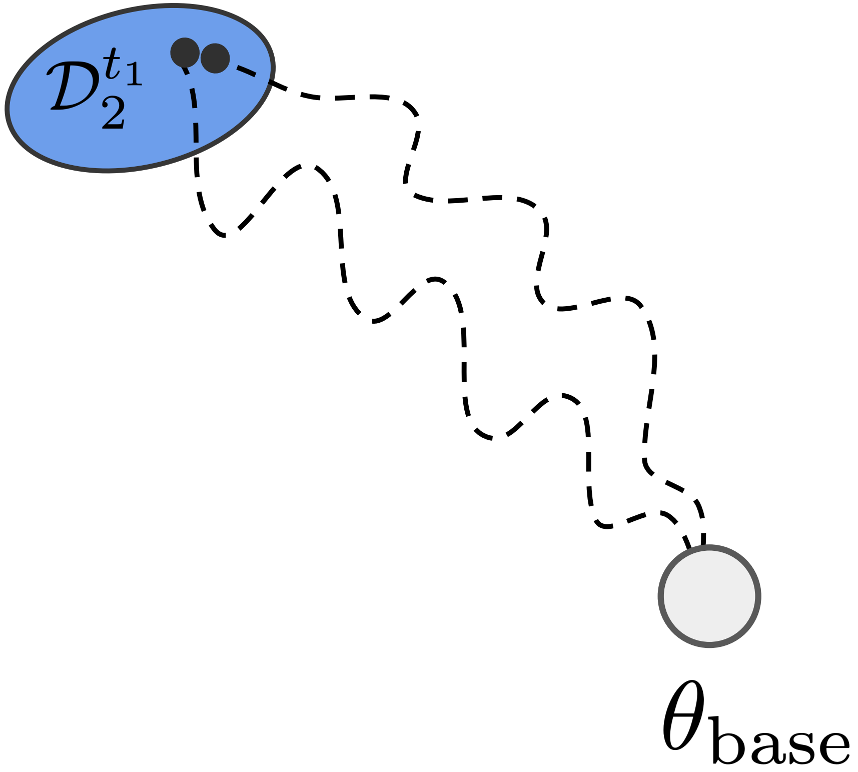


From "Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time" by Wortsman et al. (2022)

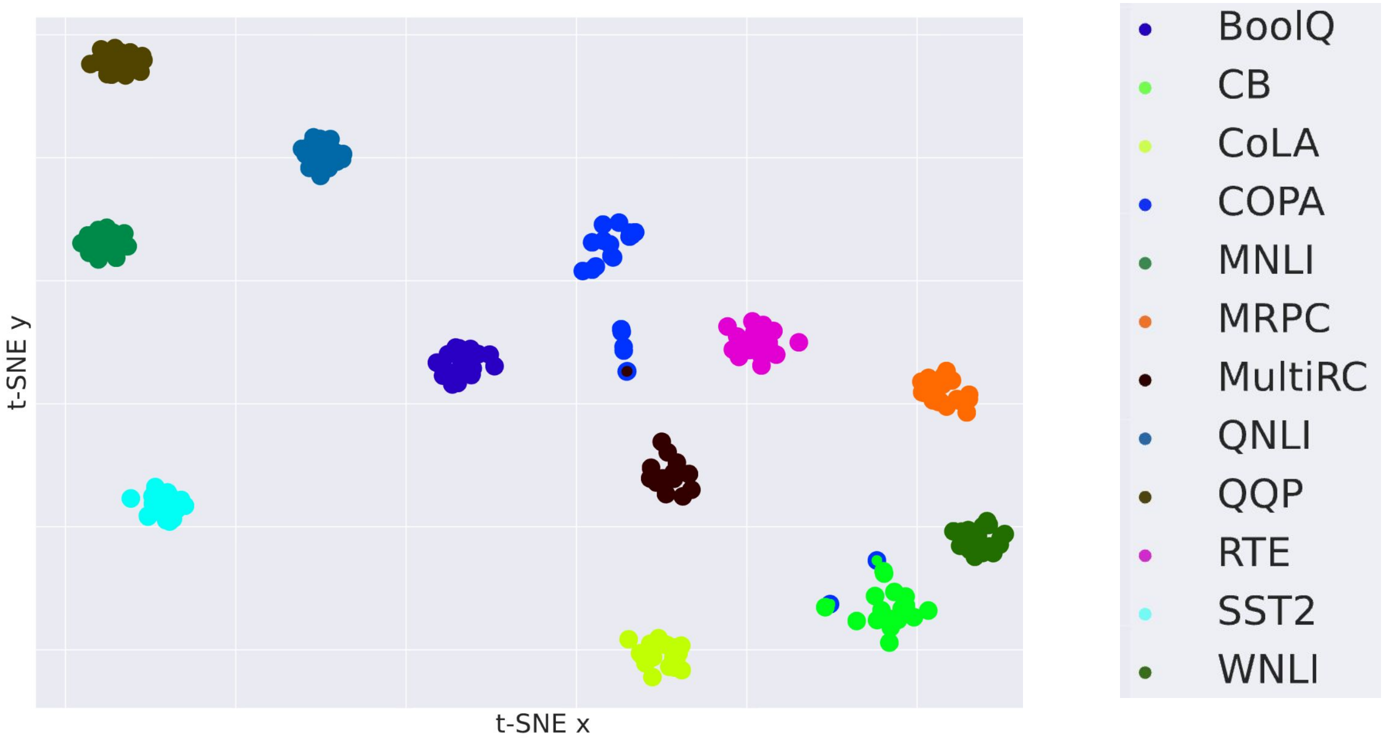
Multiple training trajectories
*(**fine-tuning** with **different** data)*



Models fine-tuned on similar data fall into the same region
*(Same **dataset**)*

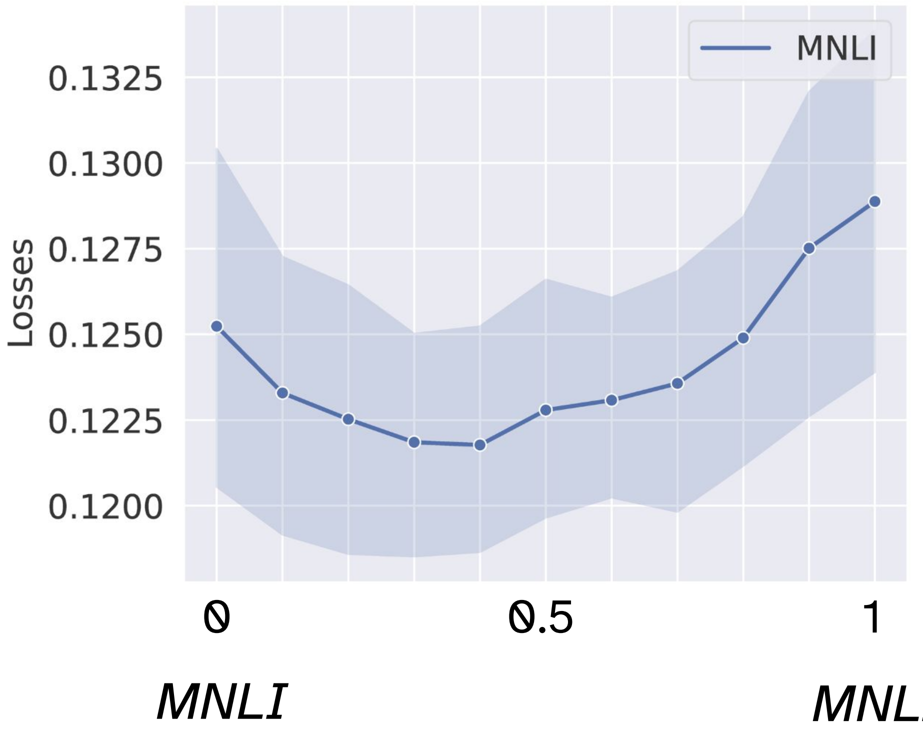


Models fine-tuned on similar data fall into the same region
*(Same **dataset**)*



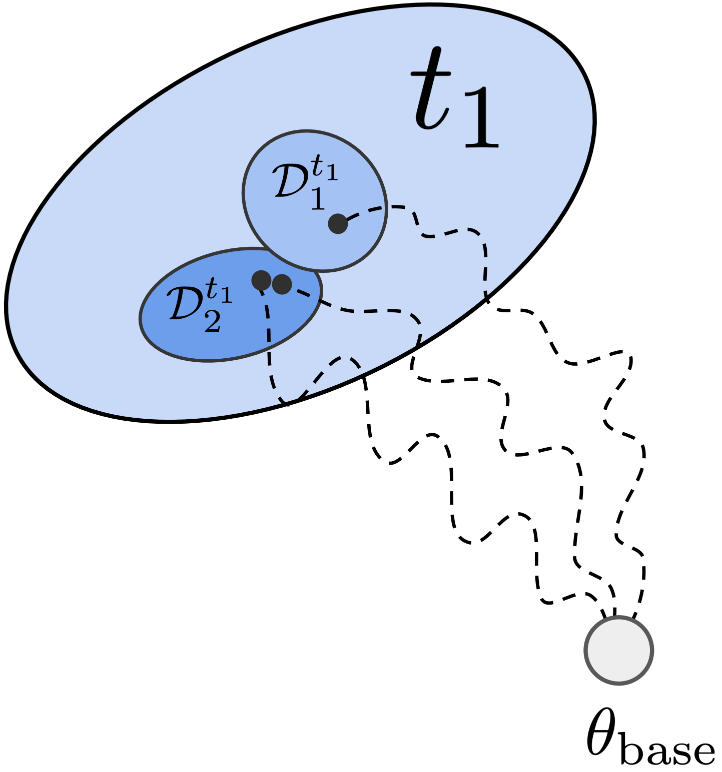
From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

*Interpolating models fine-tuned on the **same dataset***
(interpolation MNLI models)

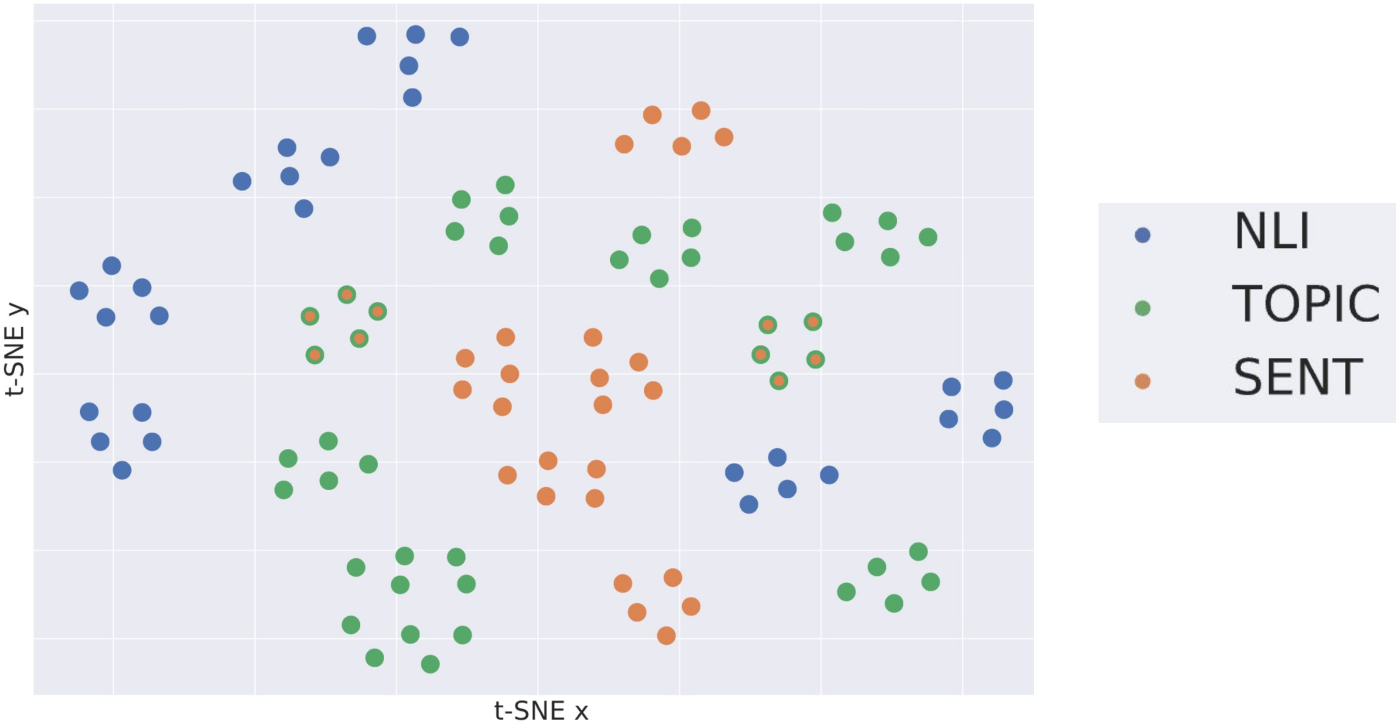


From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

Hypothesis: *Models fine-tuned on* ***different datasets*** *from the* ***same task*** *are also close in the weight space, forming a continuous region*

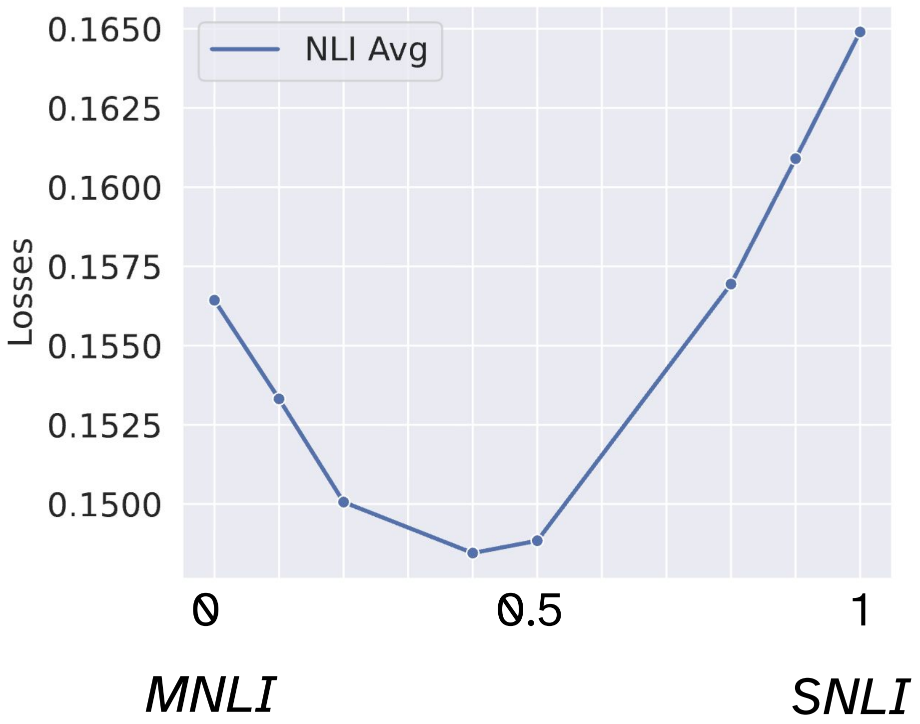


Models fine-tuned on ***different datasets*** *from the* ***same task*** *are also close in the weight space*



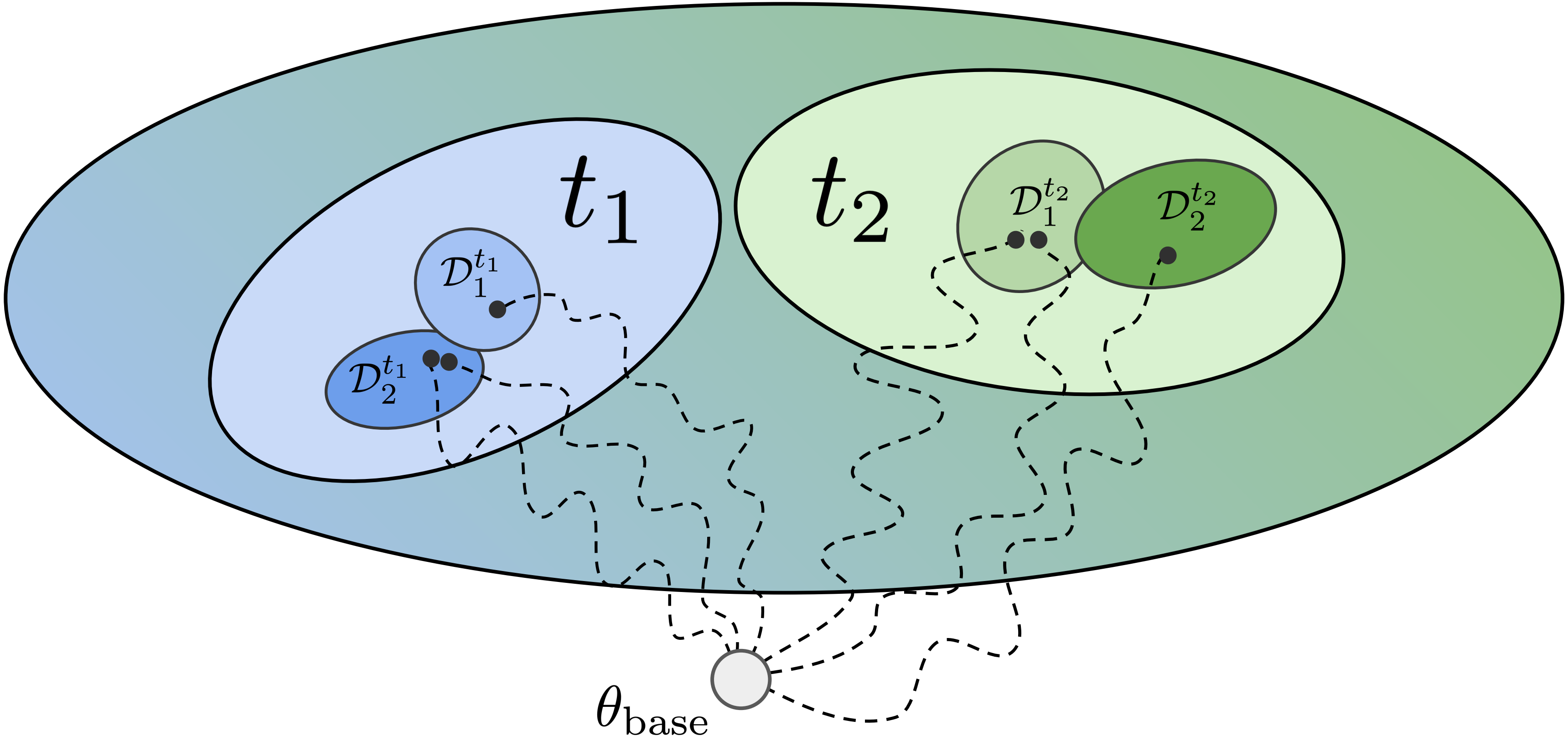
From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

*Interpolating models fine-tuned on **same task (NLI), but different datasets***
avg perf across NLI datasets (MNLI, SNLI, QNLI, ESNL, advNLI)



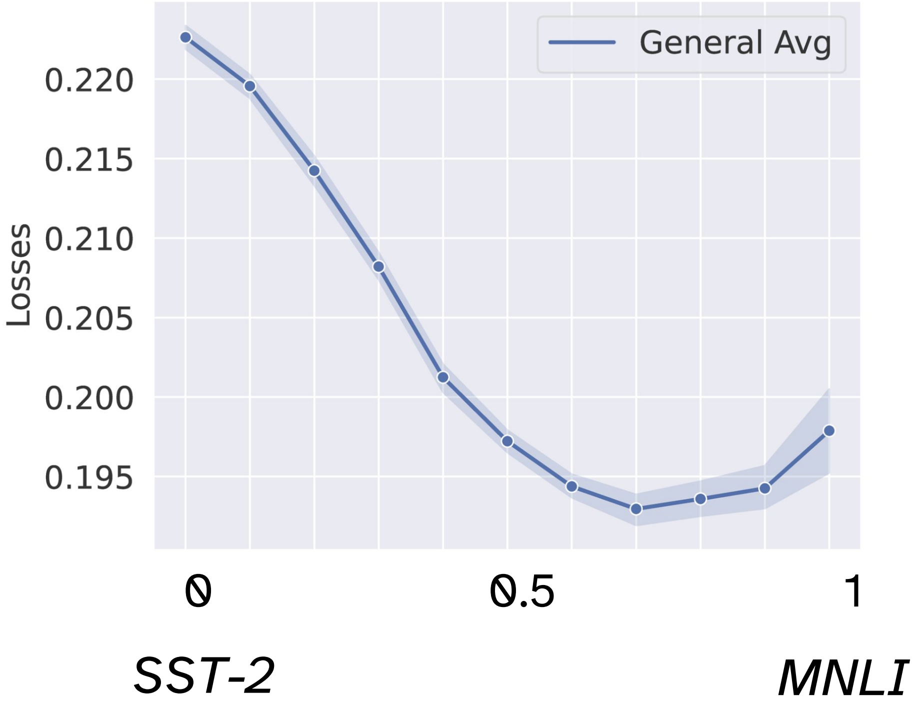
From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

Knowledge is a Region in Weight Space



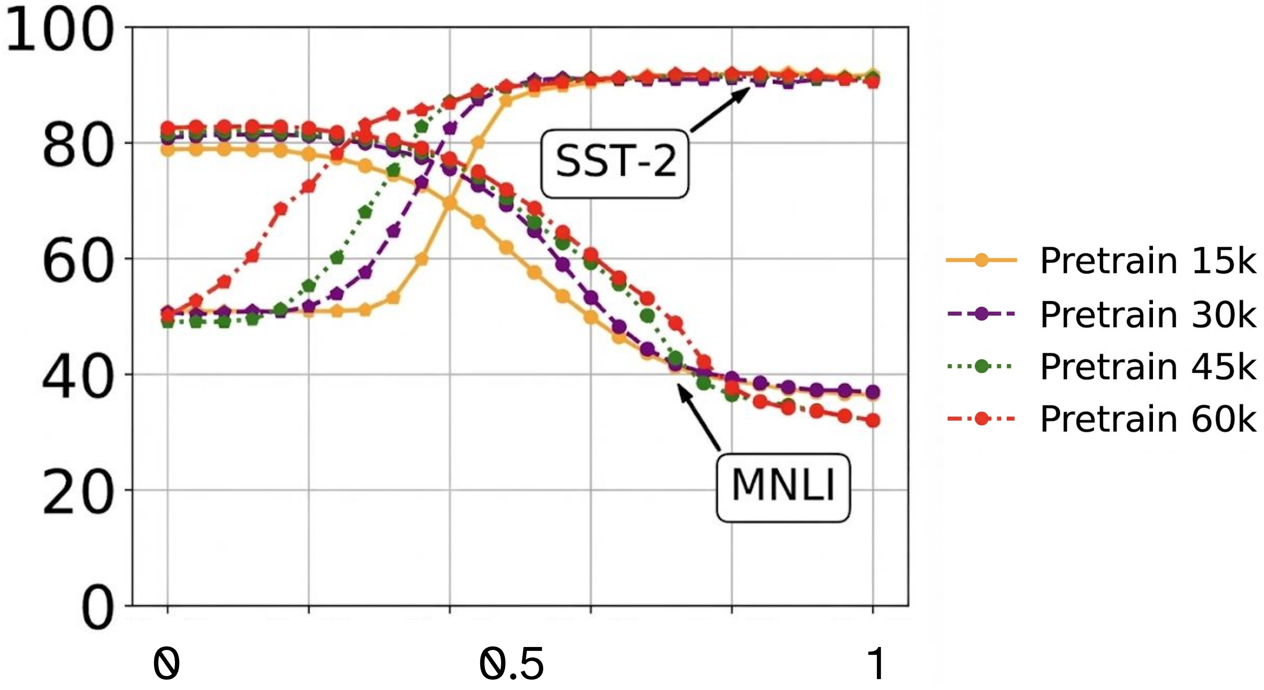
From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

*Interpolating across different tasks/datasets improve **general** performance (avg across tasks)*



From "Knowledge is a Region in Weight Space for Finetuned Language Models" by Gueta et al. (2023)

Pre-training pulls task boundaries closer

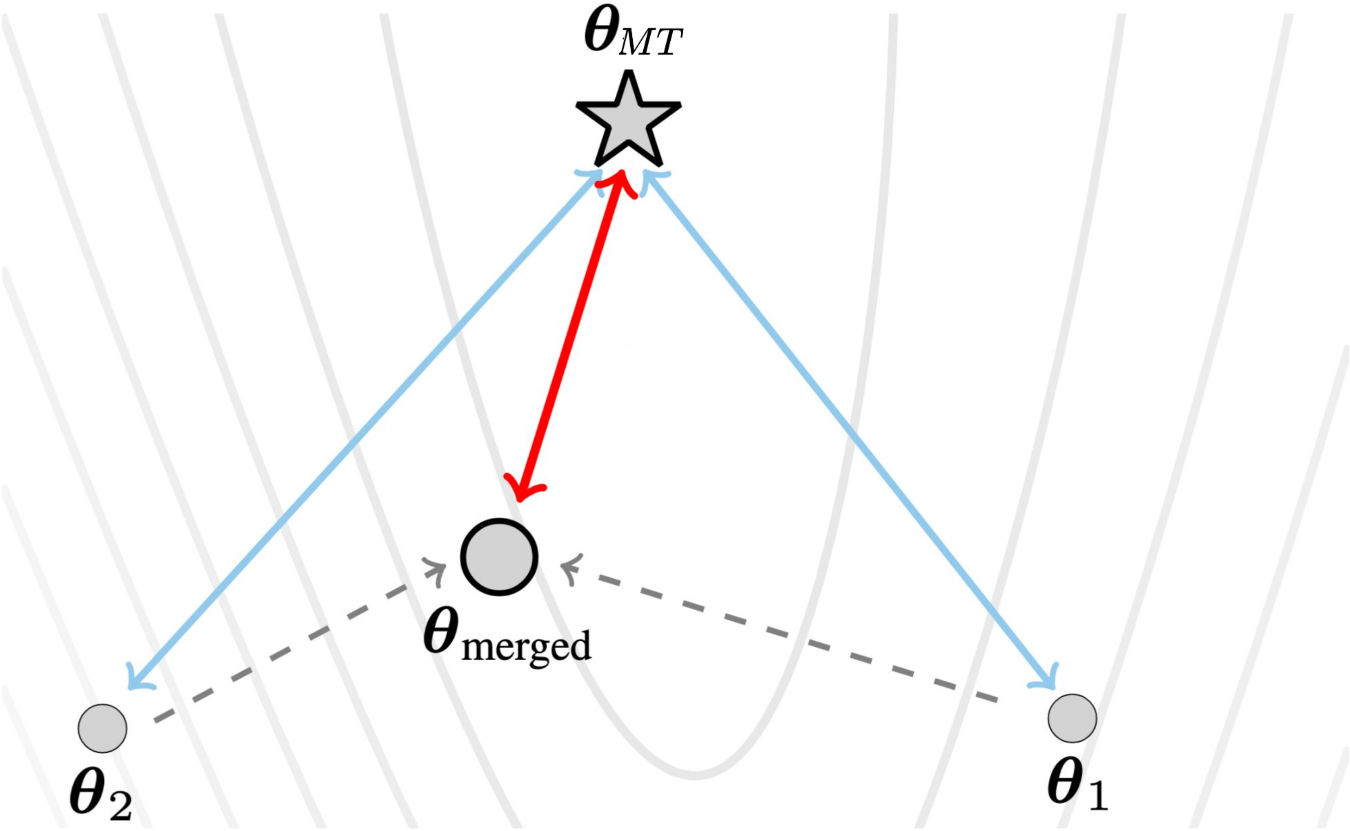


MNLI *SST-2*

From "Exploring Mode Connectivity for Pre-trained Language Models" by Qin et al. (2022)

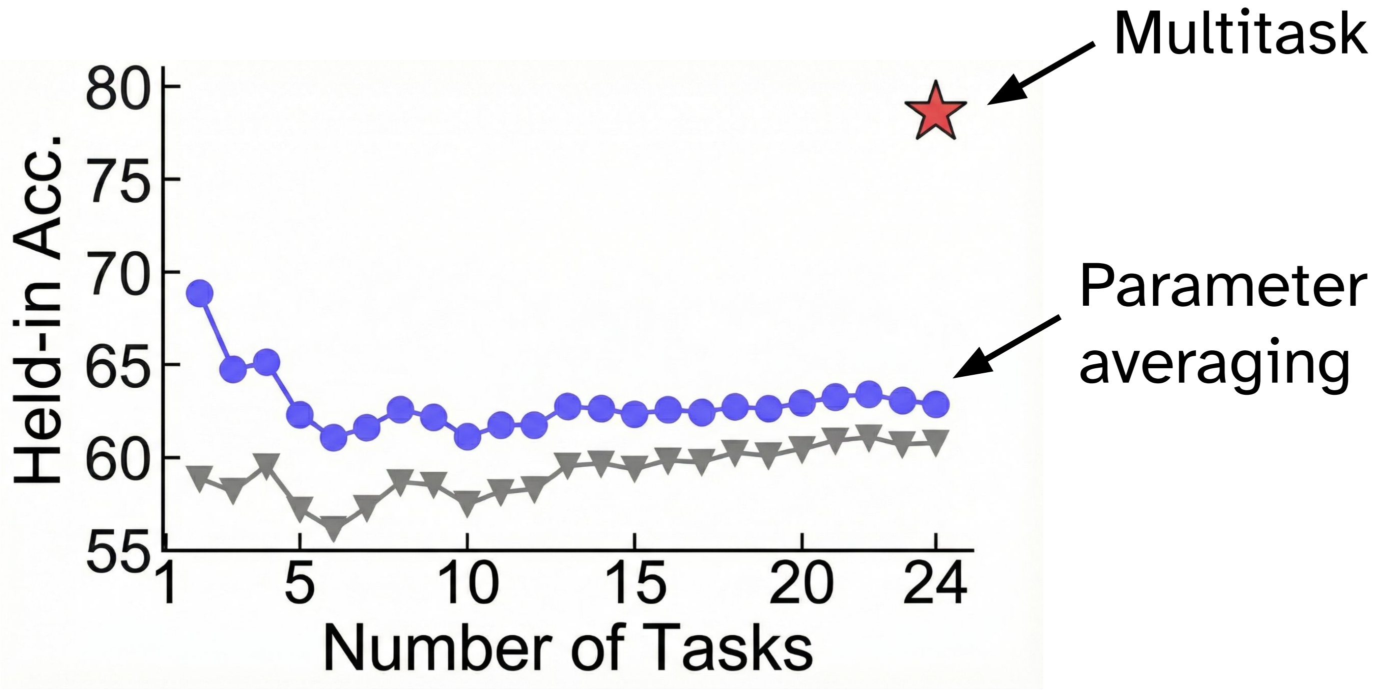
So, merging always works
when using a pre-trained model?

Model Merging vs. Multi-Task Learning



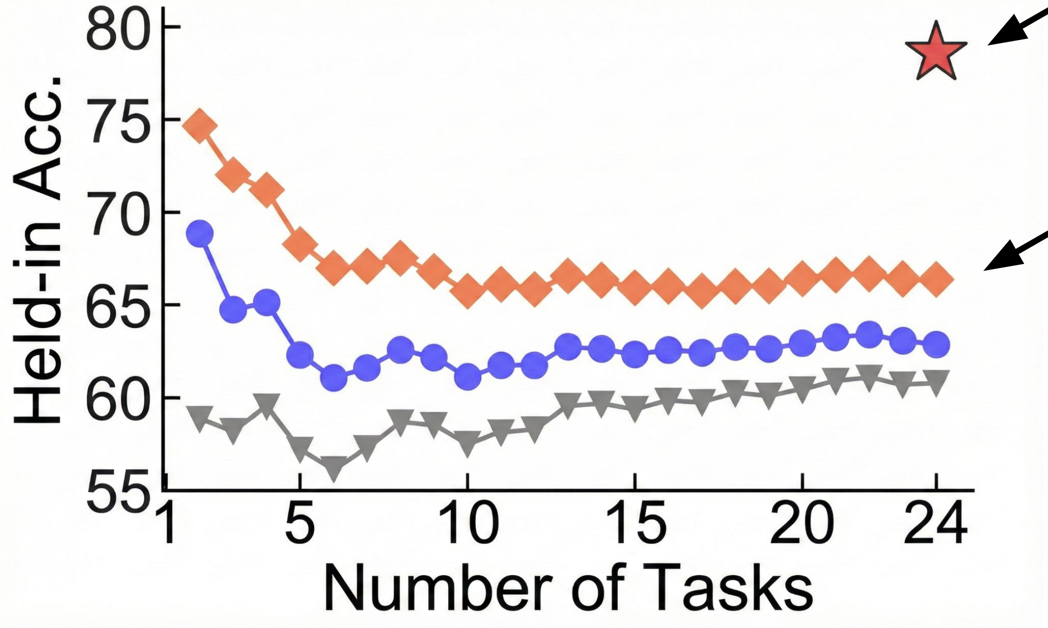
From "Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

Model Merging performance decreases with more tasks



From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

Better methods can improve performance - More on practice!



MTL

Better merging method!

From “Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

🛠️ *Practice*

What optimization problem does parameter averaging correspond to?

$$\theta^{(j)}_{\text{merged}} = \frac{\sum_{i=1}^{M} \lambda_i \theta_i^{(j)}}{\sum_{i=1}^{M} \lambda_i}$$

Hyperparameter controlling the importance of model i

From "Merging Models with Fisher-Weighted Averaging" by Matena and Raffel

Model merging as jointly maximizing model posteriors

$$\theta^{(j)}_{\text{merged}} = \frac{\sum_{i=1}^{M} \lambda_i \theta_i^{(j)}}{\sum_{i=1}^{M} \lambda_i}$$

$$\uparrow \theta \sim \mathcal{N}(\theta_i, \boldsymbol{I})$$

$$\arg\max_{\theta} \sum_{i=1}^{M} \lambda_i \underbrace{\log p(\theta|\mathcal{D}_i)}_{\text{Log posterior for model}}$$

From "Merging Models with Fisher-Weighted Averaging" by Matena and Raffel

Using the Laplace approximation produces Fisher merging

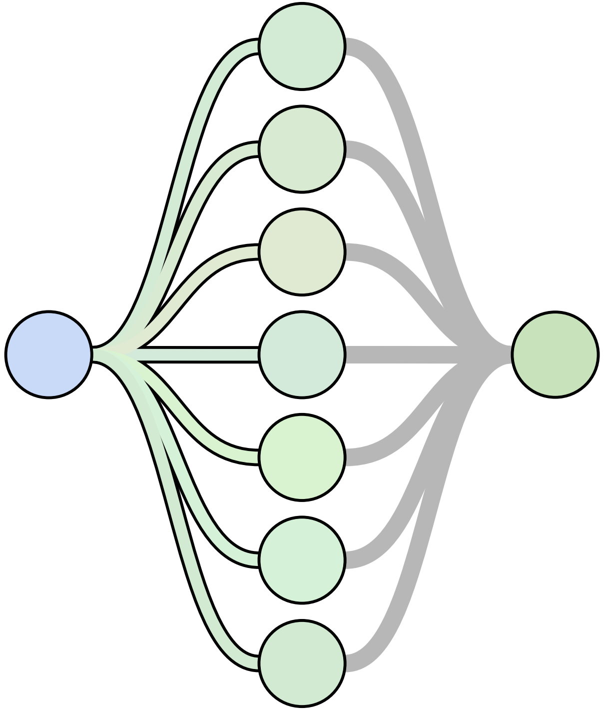
$$\boldsymbol{\theta}^{(j)}_{\text{merged}} = \frac{\sum_{i=1}^{M} \lambda_i \hat{\boldsymbol{F}}^{(j)}_i \boldsymbol{\theta}^{(j)}_i}{\sum_{i=1}^{M} \lambda_i \hat{\boldsymbol{F}}^{(j)}_i}$$

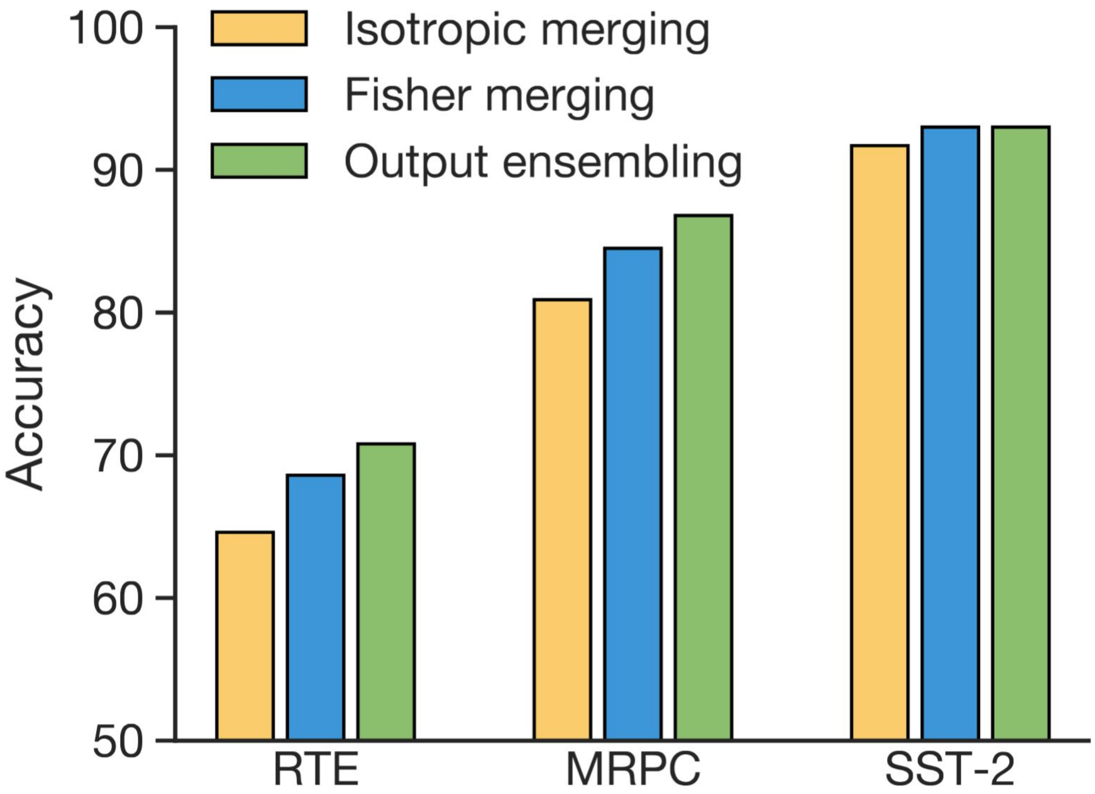
$$\Big\uparrow \theta \sim \mathcal{N}(\theta_i, \hat{F}_i^{-1})$$

$$\arg\max_{\theta} \sum_{i=1}^{M} \lambda_i \log p(\theta | \mathcal{D}_i)$$

From "Merging Models with Fisher-Weighted Averaging" by Matena and Raffel

Fisher merging outperforms "isotropic merging" (parameter averaging)





From "Merging Models with Fisher-Weighted Averaging" by Matena and Raffel

RegMean: Merging by maximizing activation agreement

$$\min_W \quad \|\underbrace{W^T \overbrace{X_1 - W_1^T X_1}^{\text{Model 1 weights and data}}\|^2 + \|W}_{\text{Merged weights}}{}^T \overbrace{X_2 - W_2^T X_2}^{\text{Model 2 weights and data}}\|^2$$

From "Dataless Knowledge Fusion by Merging Weights of Language Models" by Jin et al.

Merged weights via solving the linear regression problem

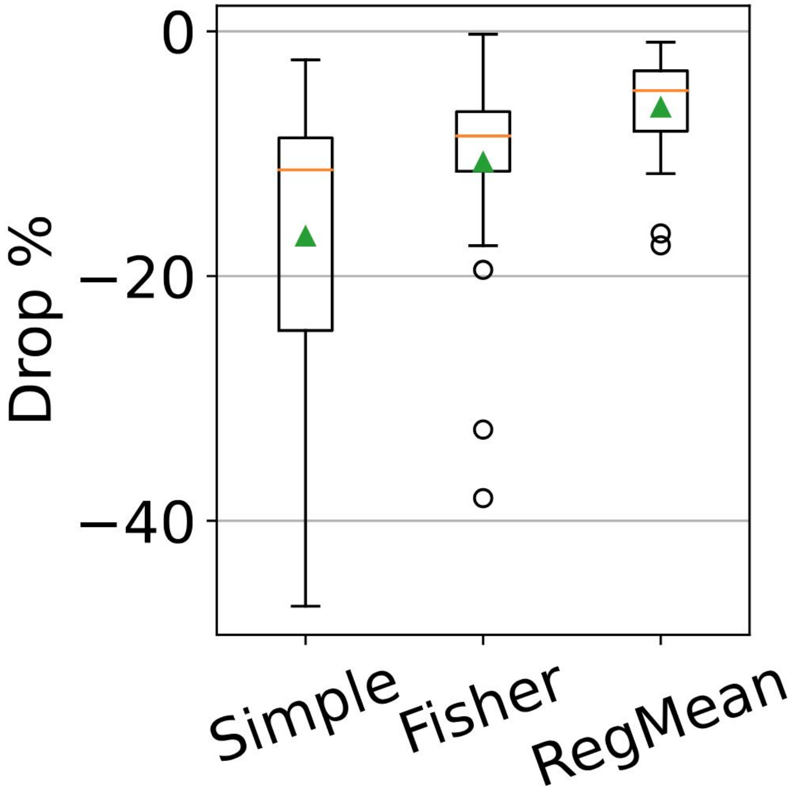
$$W_{\text{merged}} = \left(\sum_{i=1}^{M} X_i^\top X_i \right)^{-1} \sum_{i=1}^{M} \left(X_i^\top X_i W_i \right)$$

$$\uparrow$$

$$\min_{W} \quad \|W^T X_1 - W_1^T X_1\|^2 + \|W^T X_2 - W_2^T X_2\|^2$$

From "Dataless Knowledge Fusion by Merging Weights of Language Models" by Jin et al.

RegMean can outperform Fisher Merging



From "Dataless Knowledge Fusion by Merging Weights of Language Models" by Jin et al.

Connecting simple averaging, Fisher merging, and RegMean

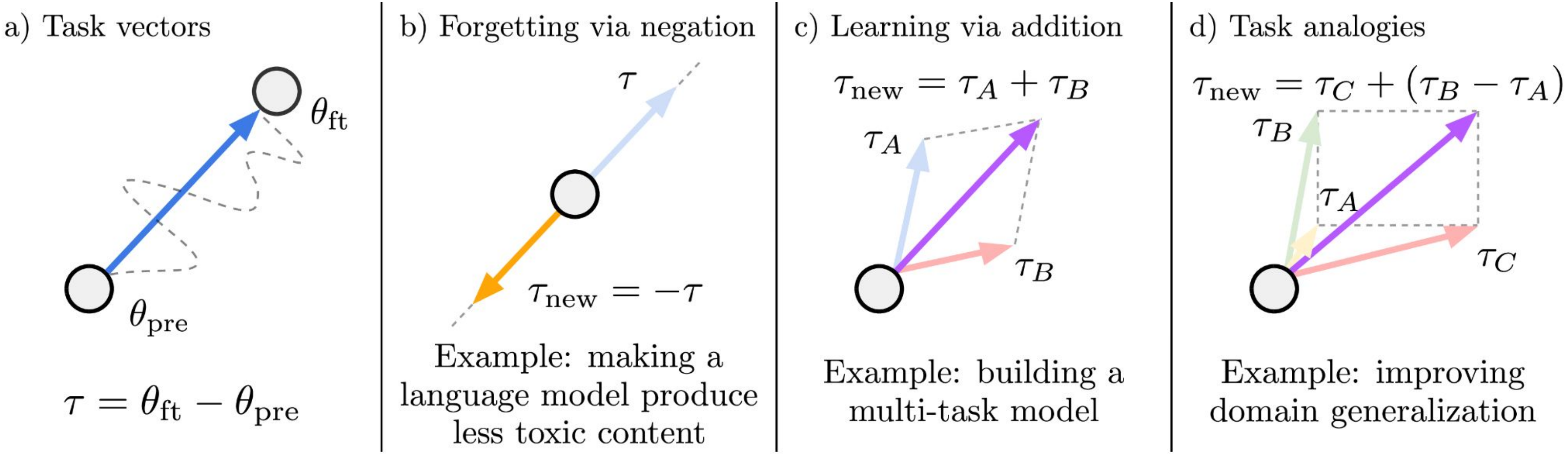
Simple averaging: $$\theta_{\text{merged}} = \left(\sum_{i=1}^{M} I\right)^{-1} \left(\sum_{i=1}^{M} I\theta_i\right)$$

Fisher merging: $$\theta_{\text{merged}} = \left(\sum_{i=1}^{M} \hat{F}_i\right)^{-1} \left(\sum_{i=1}^{M} \hat{F}_i\theta_i\right)$$

RegMean: $$\theta_{\text{merged}} = \left(\sum_{i=1}^{M} \frac{1}{N_i} Z_i^\top Z_i\right)^{-1} \left(\sum_{i=1}^{M} \frac{1}{N_i} (Z_i^\top Z_i)\theta_i\right)$$

From "Merging by Matching Models in Task Parameter Subspaces" by Tam et al.

Task vectors provide an alternative way of merging…



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

… that is, in a sense, equivalent to averaging

$$\tau_i = \theta_i - \theta_p$$

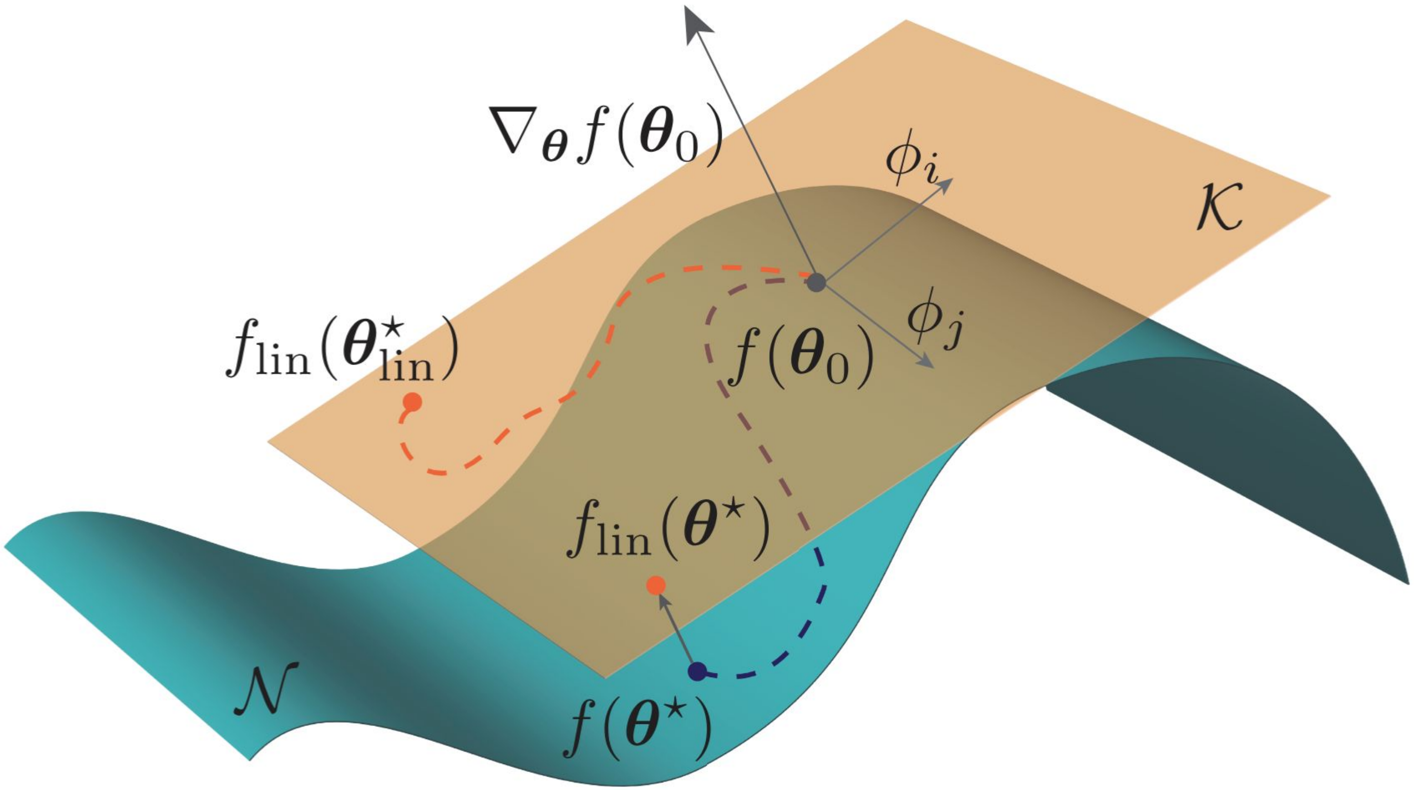
Task vectors: $$\theta_{\text{merged}} = \theta_p + \lambda \sum_{i=1}^{M} \tau_i$$

Simple averaging: $$\theta_{\text{merged}} = \frac{1}{\sum_i \lambda_i} \sum_{i=1}^{M} \lambda_i \theta_i$$

$$= \frac{1}{\sum_i \lambda_i} \sum_{i=1}^{M} \lambda_i (\theta_p + \tau_i)$$

$$= \theta_p + \frac{1}{\sum_i \lambda_i} \sum_{i=1}^{M} \lambda_i \tau_i$$

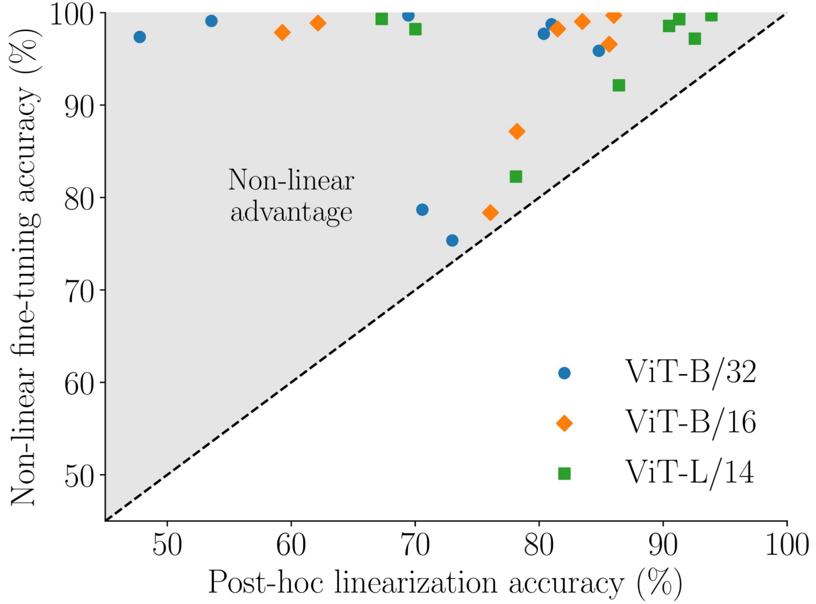
Do task vectors work because fine-tuning is inherently linear?



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

Not really!

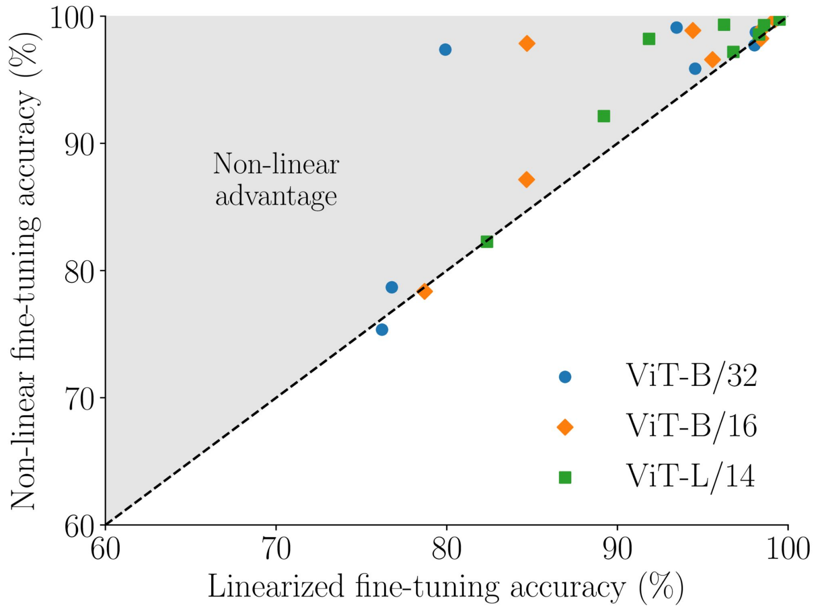
Method		ViT-B/32 Abs. (↑)	ViT-B/32 Norm. (↑)
Pre-trained	$f(\cdot\,; \boldsymbol{\theta}_0)$	48.4	–
Non-lin. FT	$f(\cdot\,; \boldsymbol{\theta}_0 + \boldsymbol{\tau})$	71.4	76.5
Post-hoc lin.	$f_{\text{lin}}(\cdot\,; \boldsymbol{\theta}_0 + \boldsymbol{\tau})$	57.1	81.9



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

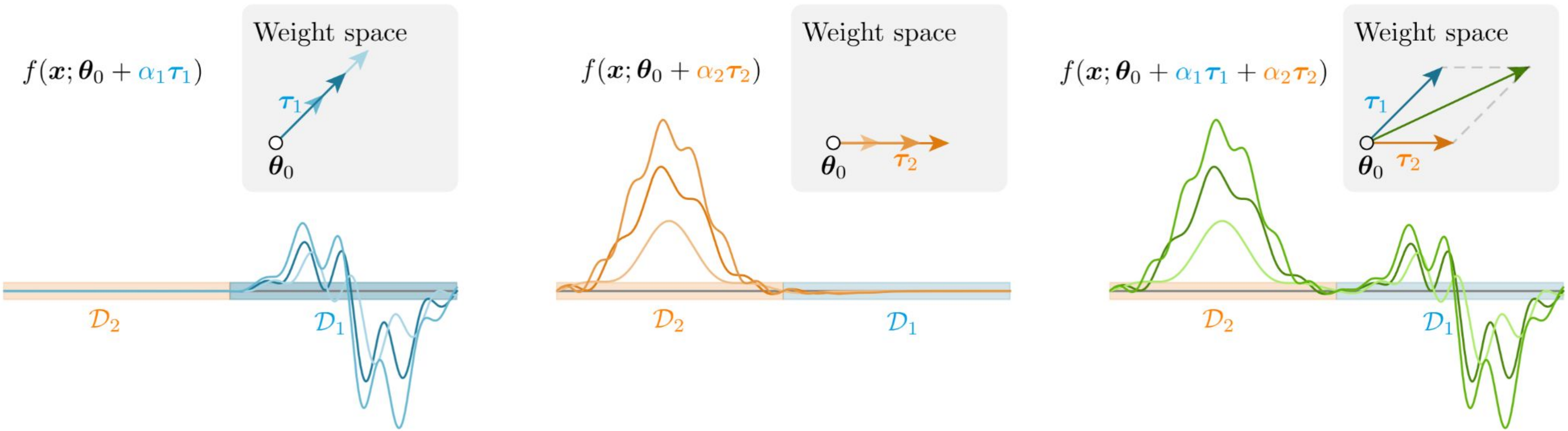
Linearized fine-tuning improves performance

Method		ViT-B/32	
		Abs. (↑)	Norm. (↑)
Pre-trained	$f(\cdot\,;\boldsymbol{\theta}_0)$	48.4	–
Non-lin. FT	$f(\cdot\,;\boldsymbol{\theta}_0+\boldsymbol{\tau})$	71.4	76.5
Post-hoc lin.	$f_{\text{lin}}(\cdot\,;\boldsymbol{\theta}_0+\boldsymbol{\tau})$	57.1	81.9
Linear. FT	$f_{\text{lin}}(\cdot\,;\boldsymbol{\theta}_0+\boldsymbol{\tau}_{\text{lin}})$	**76.5**	**85.4**



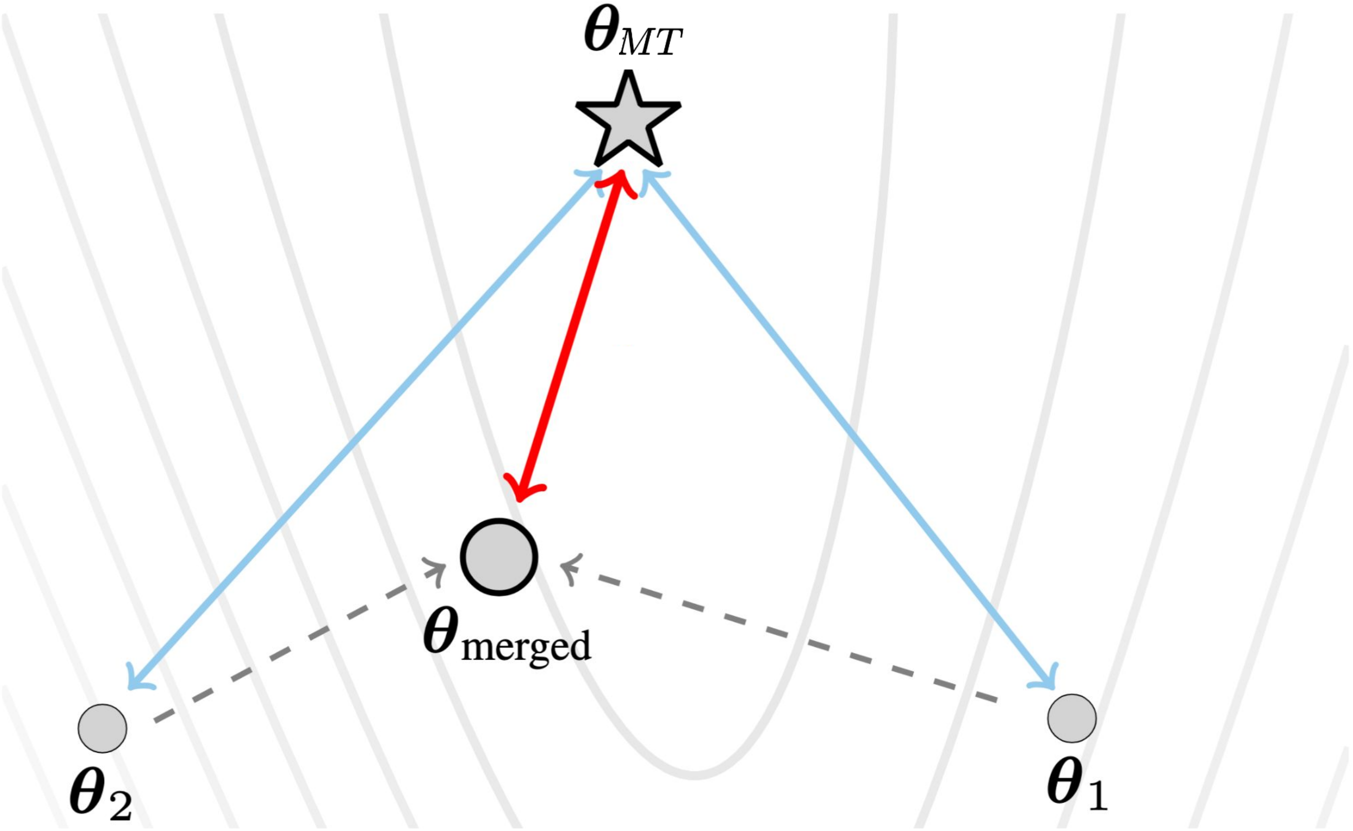
From "Task Arithmetic in the Tangent Space: Improved Editing of Pre-Trained Models" by Ortiz-Jimenez et al.

"*Weight disentanglement*" *provides an alternative view*



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

Characterizing and minimizing the merging error



From "Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

Quantifying merging error via "gradient mismatch"

$$\underbrace{\theta_{MT} - \left(\theta_p + \lambda \sum_{i=1}^{M} \tau_i\right)}_{\text{Merging error}} = -\lambda \sum_{i=1}^{M} \hat{F}_p^{-1} \overbrace{\left(\nabla \ell_i(\theta_{MT}) - \nabla \ell_i(\theta_i)\right)}^{\text{"Gradient mismatch"}}$$

Preconditioner for measuring distance from initialization

From “Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

Approximating the multitask model gradient

$$\theta_{MT} - \left(\theta_p + \lambda \sum_{i=1}^{M} \tau_i\right) = -\lambda \sum_{i=1}^{M} \hat{F}_p^{-1} \left(\nabla \ell_i(\theta_{MT}) - \nabla \ell_i(\theta_i)\right)$$

$$\nabla \ell_i(\theta_{MT}) \approx \nabla \ell_i(\theta_i) + \hat{F}_i(\theta_{MT} - \theta_i)$$

From "Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

Merging while minimizing gradient mismatch

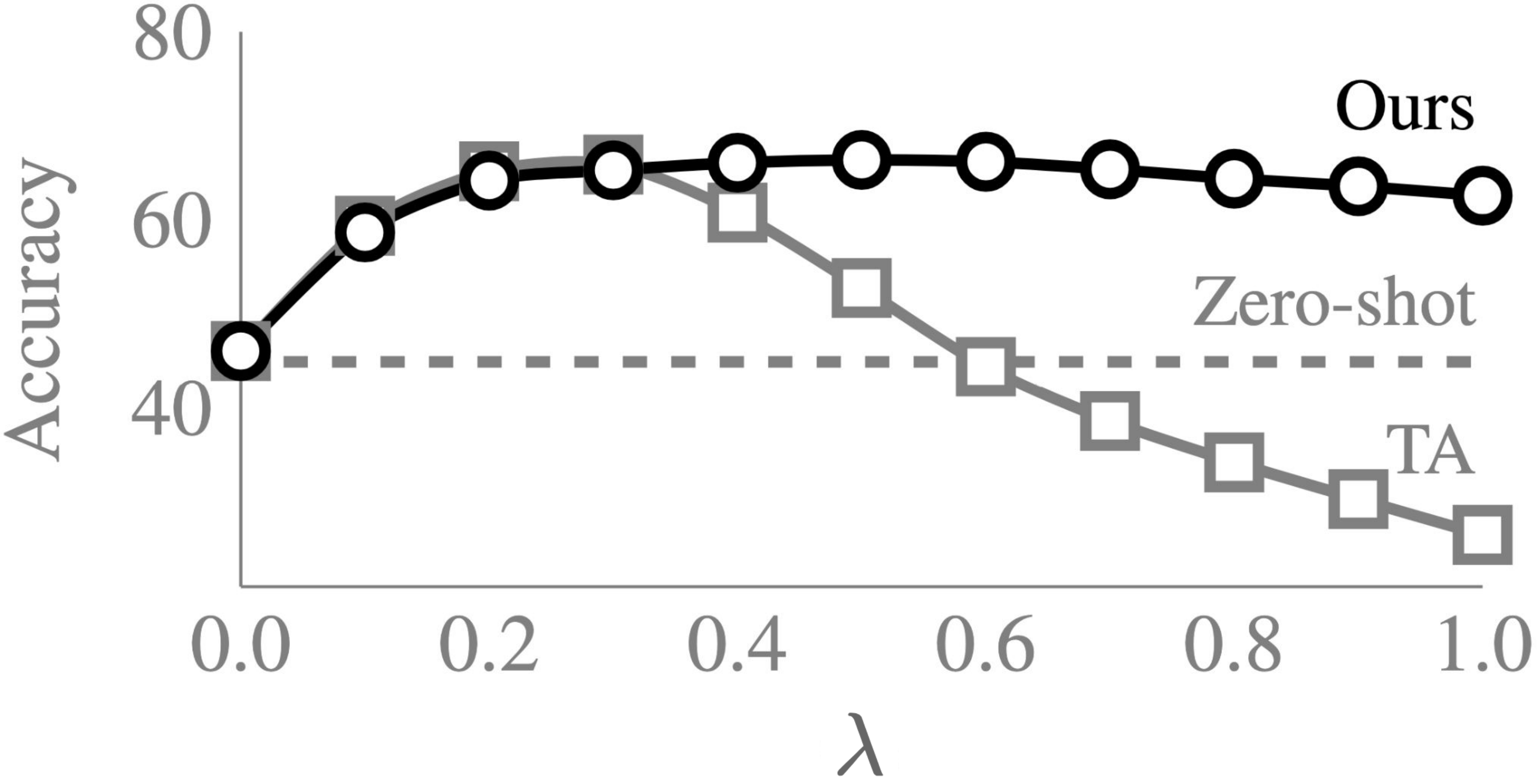
$$\theta_{MT} - \left(\theta_p + \lambda \sum_{i=1}^{M} \tau_i\right) = -\lambda \sum_{i=1}^{M} \hat{F}_p^{-1} \left(\nabla \ell_i(\theta_{MT}) - \nabla \ell_i(\theta_i)\right)$$

$$\nabla \ell_i(\theta_{MT}) \approx \nabla \ell_i(\theta_i) + \hat{F}_i(\theta_{MT} - \theta_i)$$

$$\theta_{\mathrm{merged}} = \theta_p + \lambda \left(\hat{F}_p + \lambda \sum_{i=1}^{M} \hat{F}_i\right)^{-1} \sum_{i=1}^{M} (\hat{F}_p + \hat{F}_i)\tau_i$$

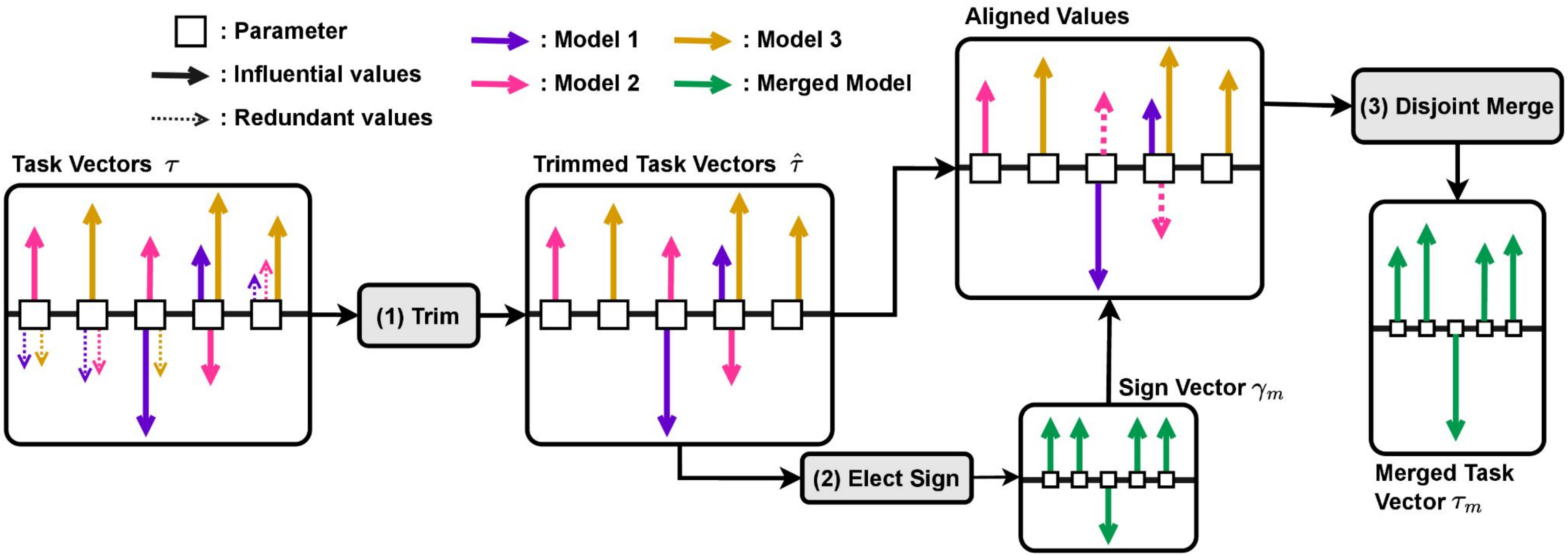
From "Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

Minimizing gradient mismatch reduces sensitivity of λ



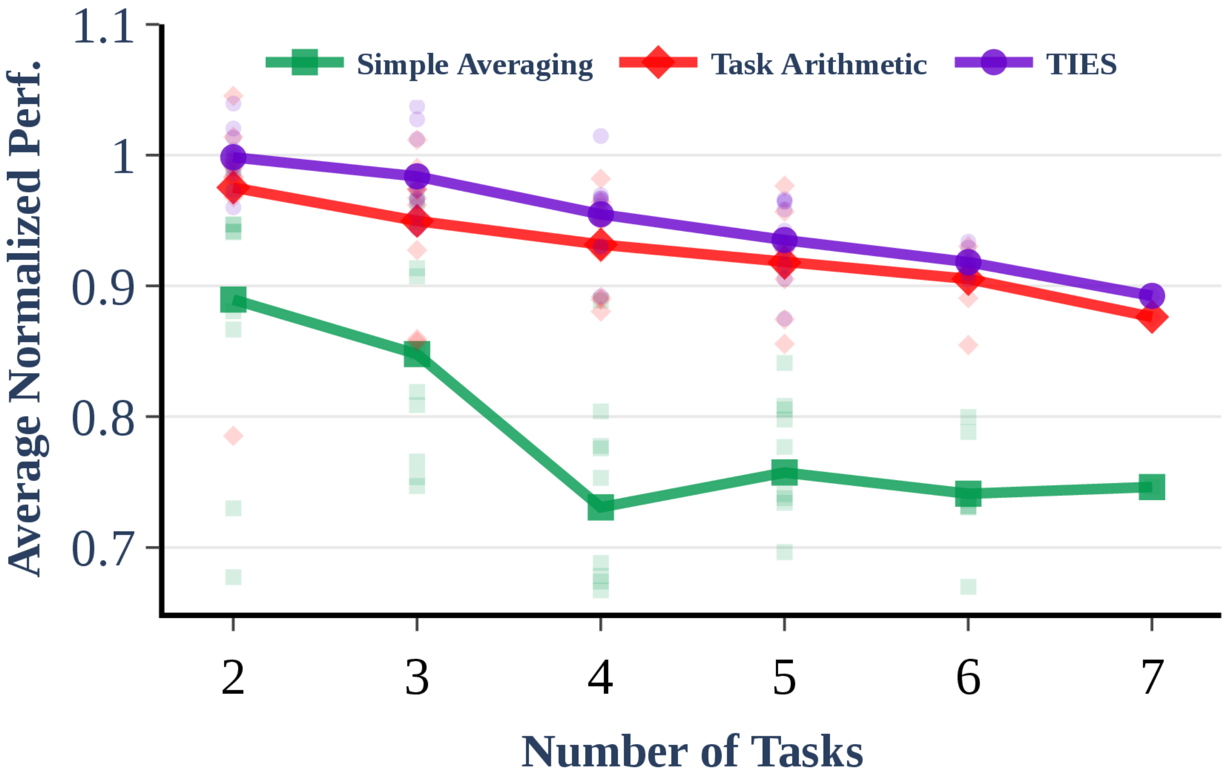
From "Model Merging by Uncertainty-Based Gradient Matching" by Daheim et al.

TIES Merging aims to resolve "interference" between task vectors



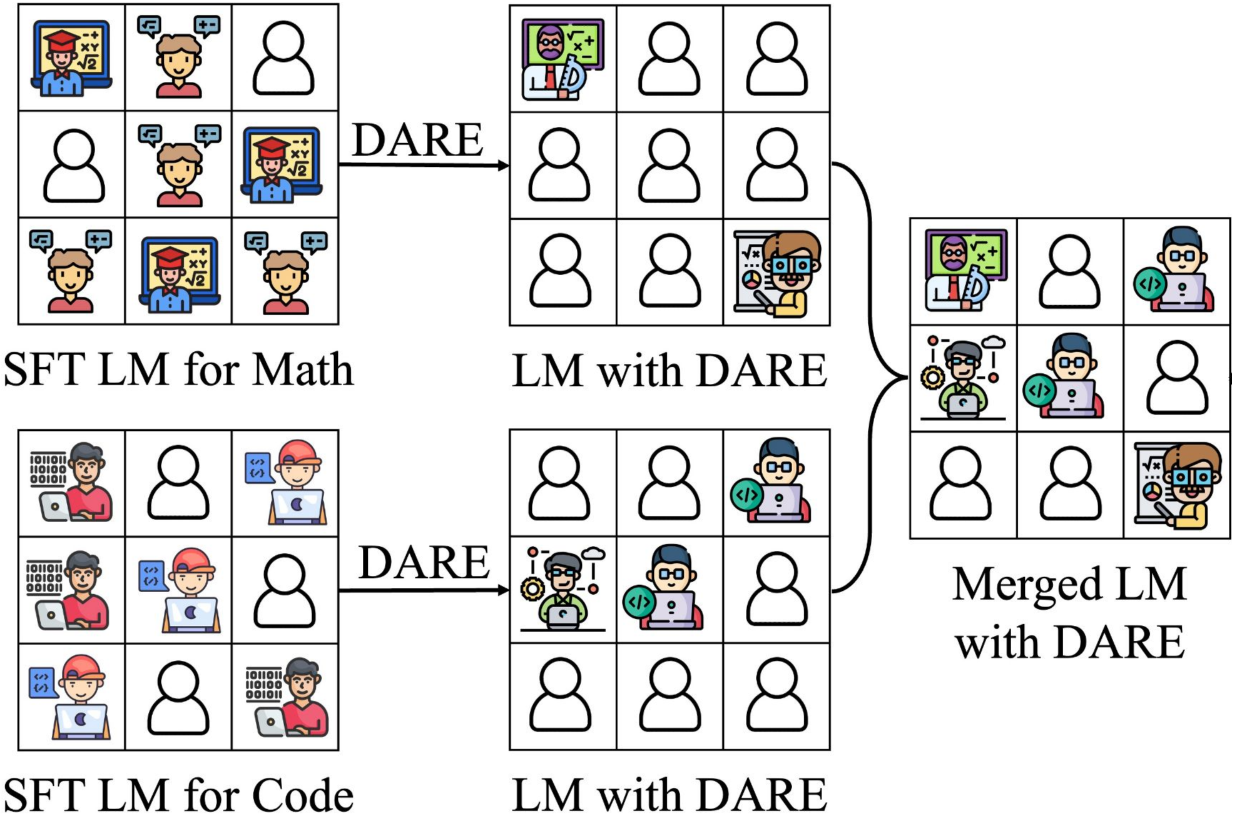
From "Resolving Interference When Merging Models" by Yadav et al.

TIES helps retain individual model performance



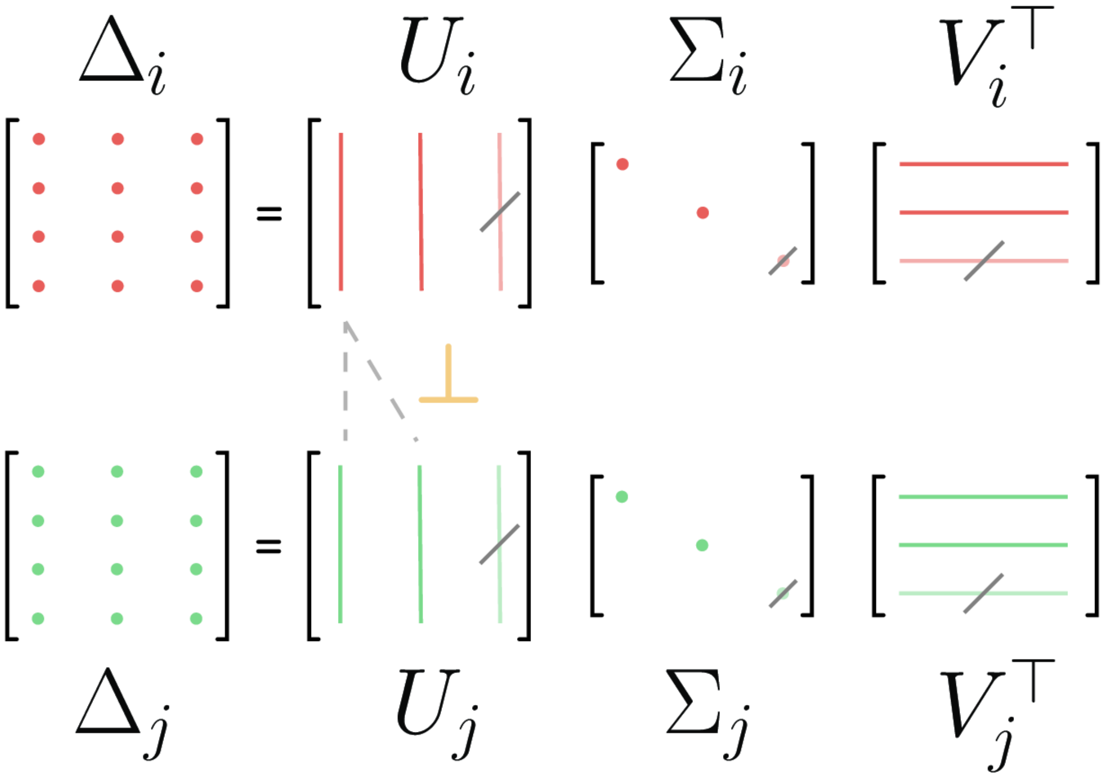
From “Resolving Interference When Merging Models" by Yadav et al.

Dropping parameter updates randomly can also help



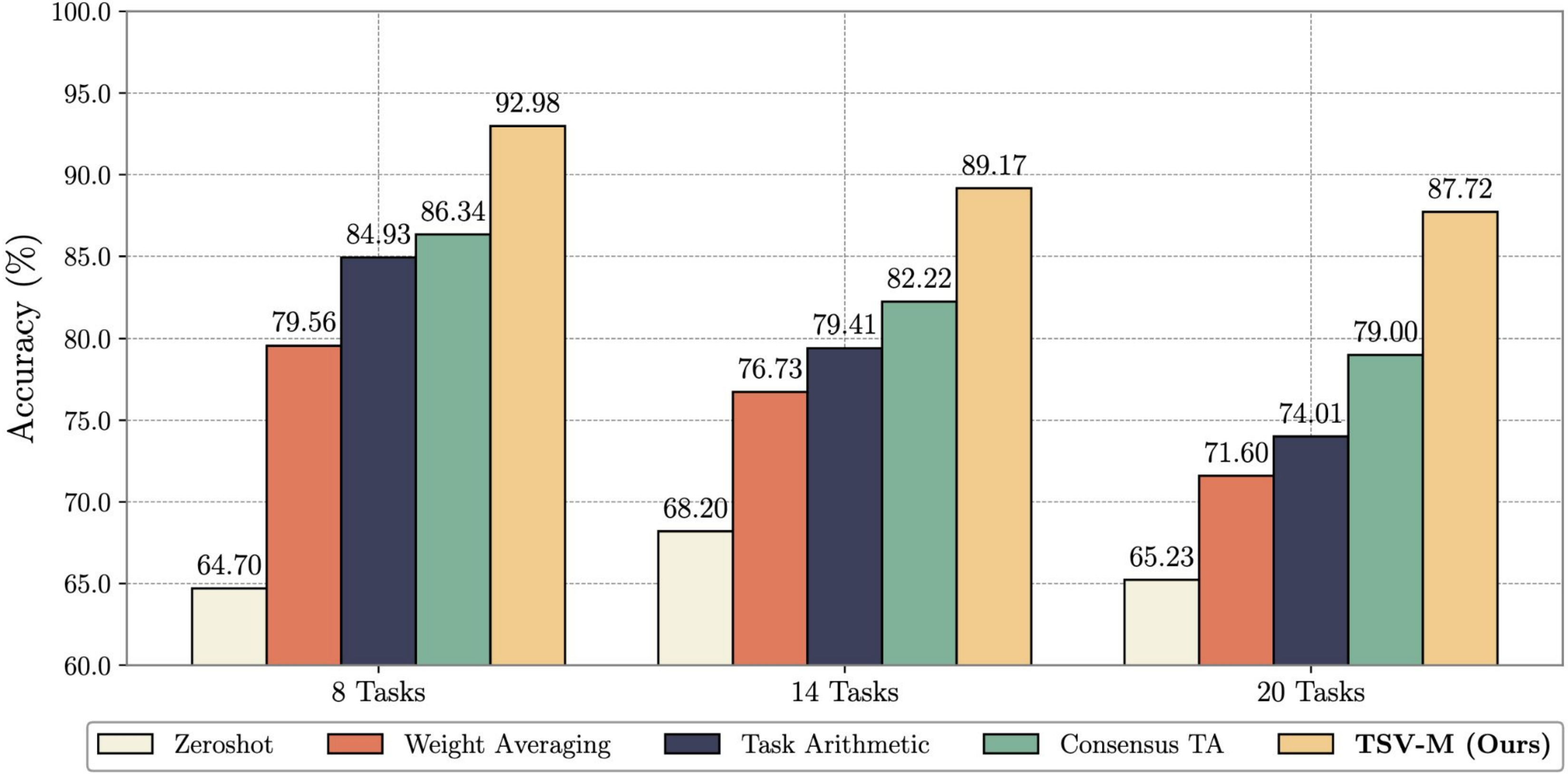
From "Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch" by Yu et al.

Avoiding interference in task singular vectors



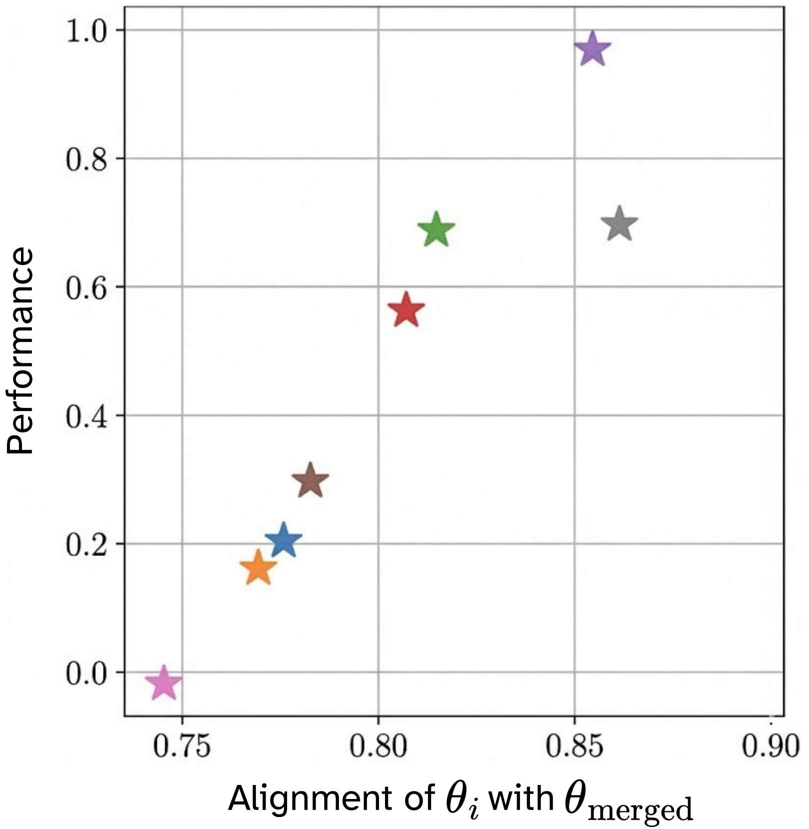
From “Task Singular Vectors: Reducing Task Interference in Model Merging" by Gargiulo et al.

Task singular vectors improve performance



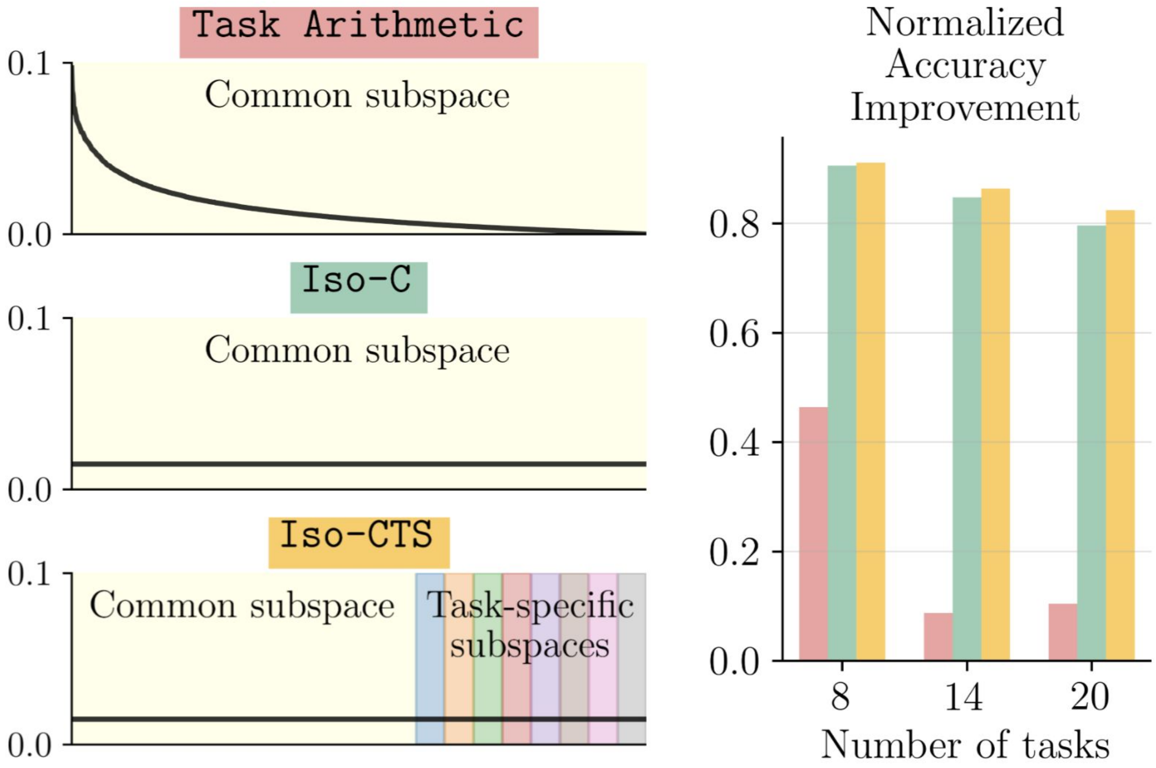
From "Task Singular Vectors: Reducing Task Interference in Model Merging" by Gargiulo et al.

Singular vector alignment predicts per-task performance



From "No Task Left Behind: Isotropic Model Merging with Common and Task-Specific Subspaces" by Marczak et al.

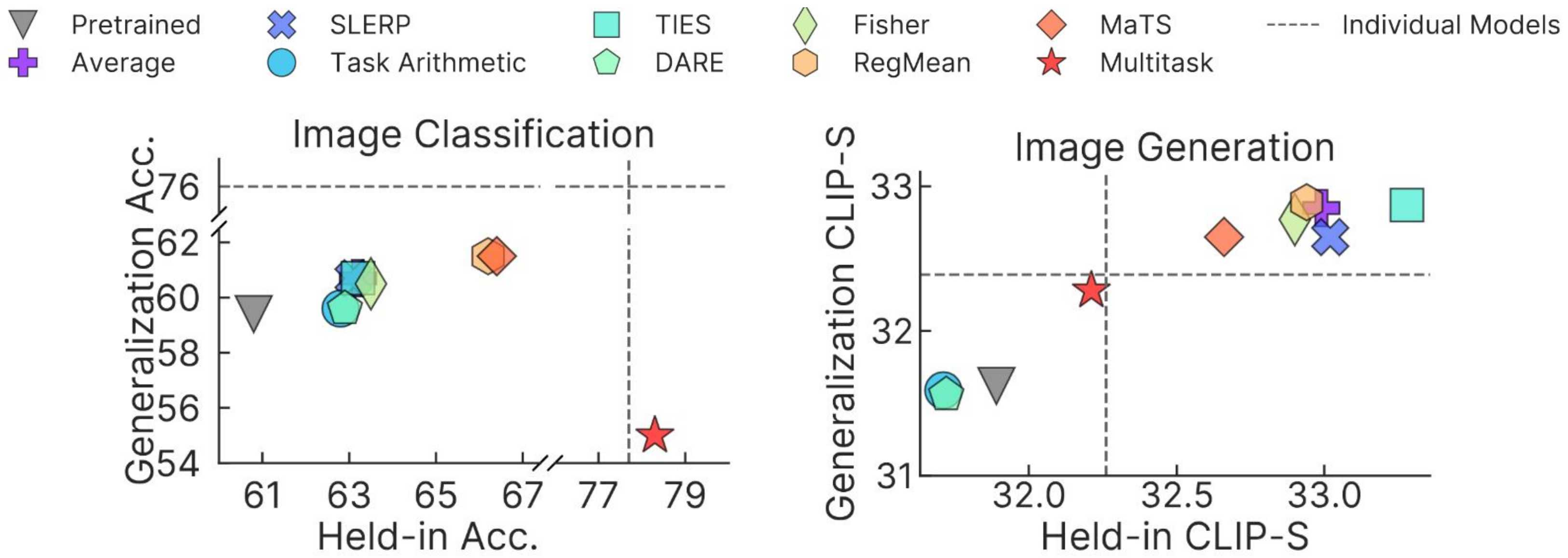
Enforcing a uniform spectrum of singular values can improves performance



From "No Task Left Behind: Isotropic Model Merging with Common and Task-Specific Subspaces" by Marczak et al.

OK, but which merging method should I use?

Multitask and generalization performance tends to correlate



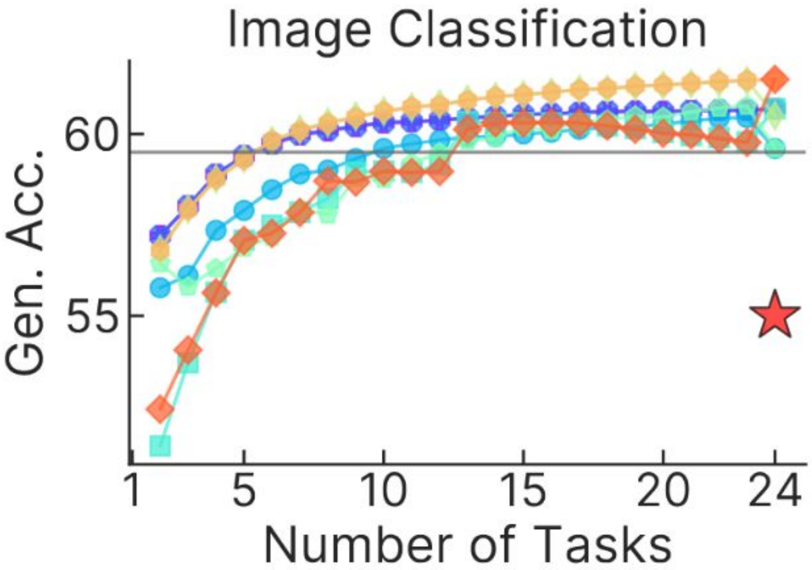
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

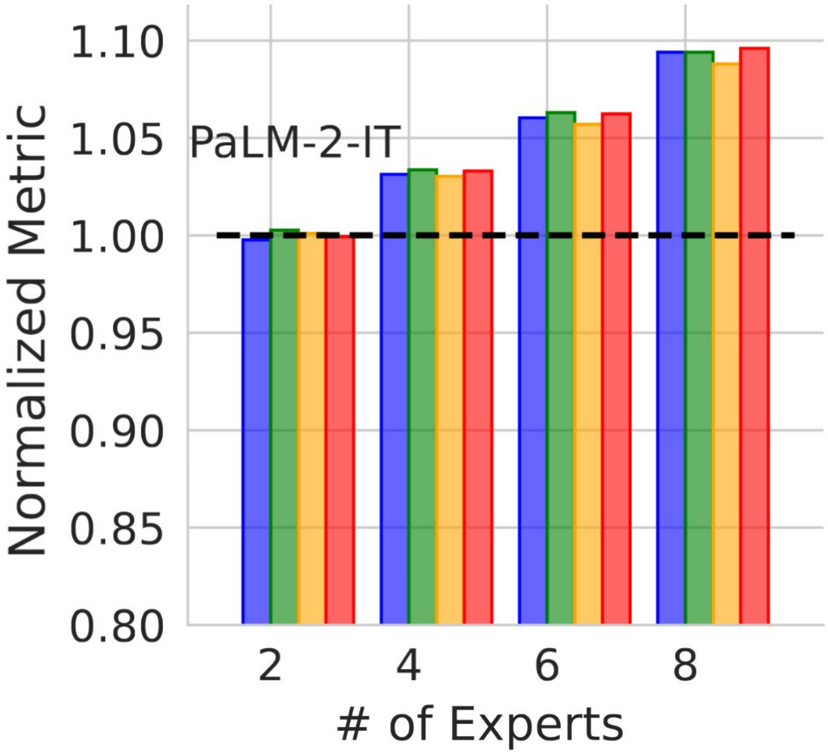
Merging methods have differing costs and requirements

	Prerequisites			Hparams?	Computational cost (FLOPs)	
	$\theta_{\mathbf{p}}$	Stats	Data		Merging	Statistics
Average					Mdk	
SLERP					$\mathcal{O}((5M-2)dk)$	
Task Arith.	✗		✗	✓	$(2M+1)dk$	
DARE	✗		✗	✓	$(6M+1)dk$	
TIES	✗		✗*	✓*	$(4M+1)dk$	$\mathcal{O}(MKdk)$
Fisher		✗*	✗*		$(3M-1)dk$	$4MTd^2k$
RegMean		✗*	✗*	✓	$\mathcal{O}((M+2)d^2k)$	MTd^2k
MaTS		✗*	✗*	✓	$\mathcal{O}((M+N)d^2k)$	$4MTd^2k$

From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

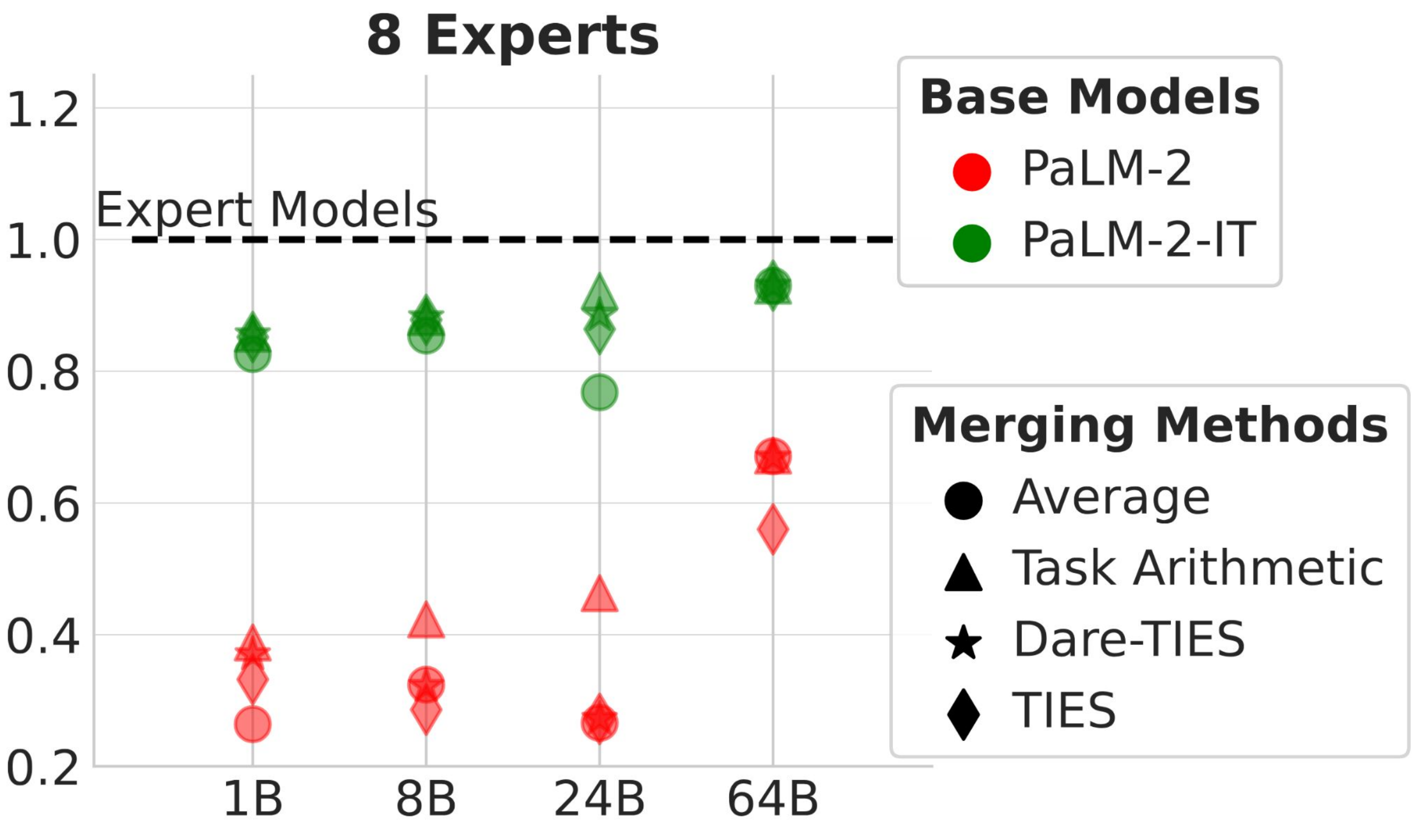
When merging more models, generalization improves





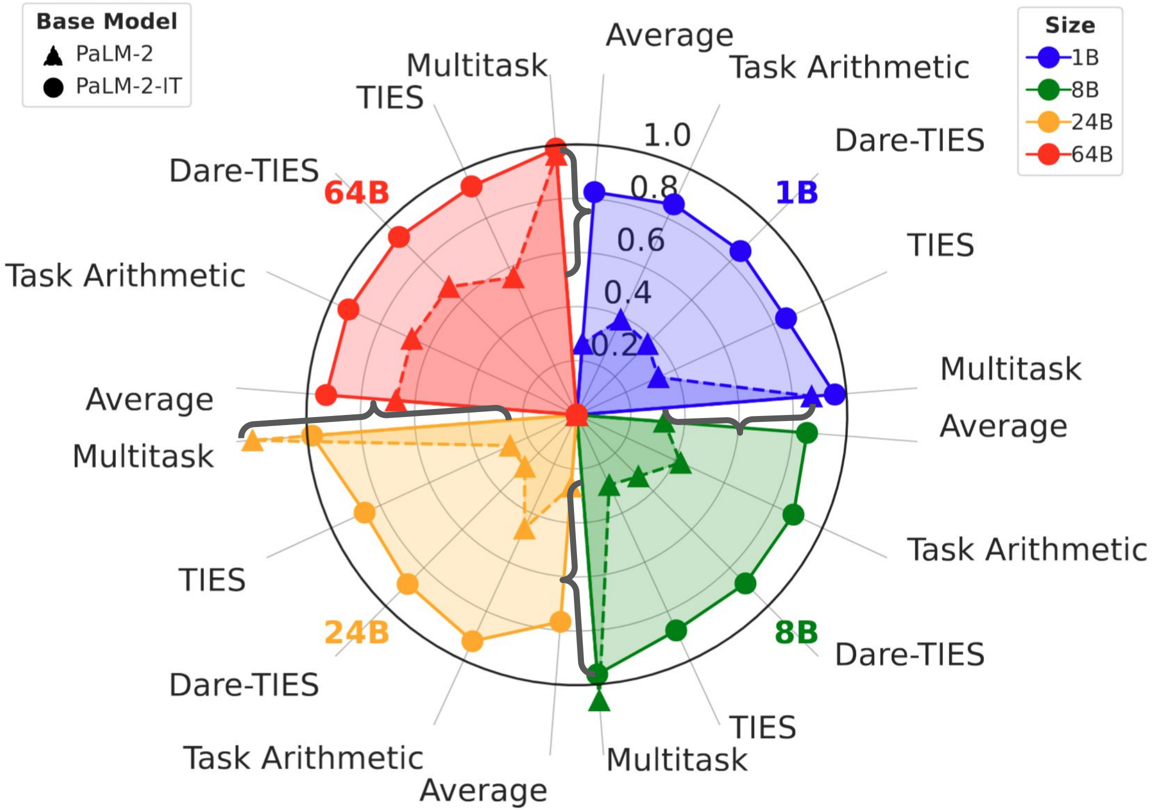
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al. and "What Matters for Model Merging at Scale?" by Yadav et al.

Merging methods perform similarly when the base model is stronger



From "What Matters for Model Merging at Scale?" by Yadav et al.

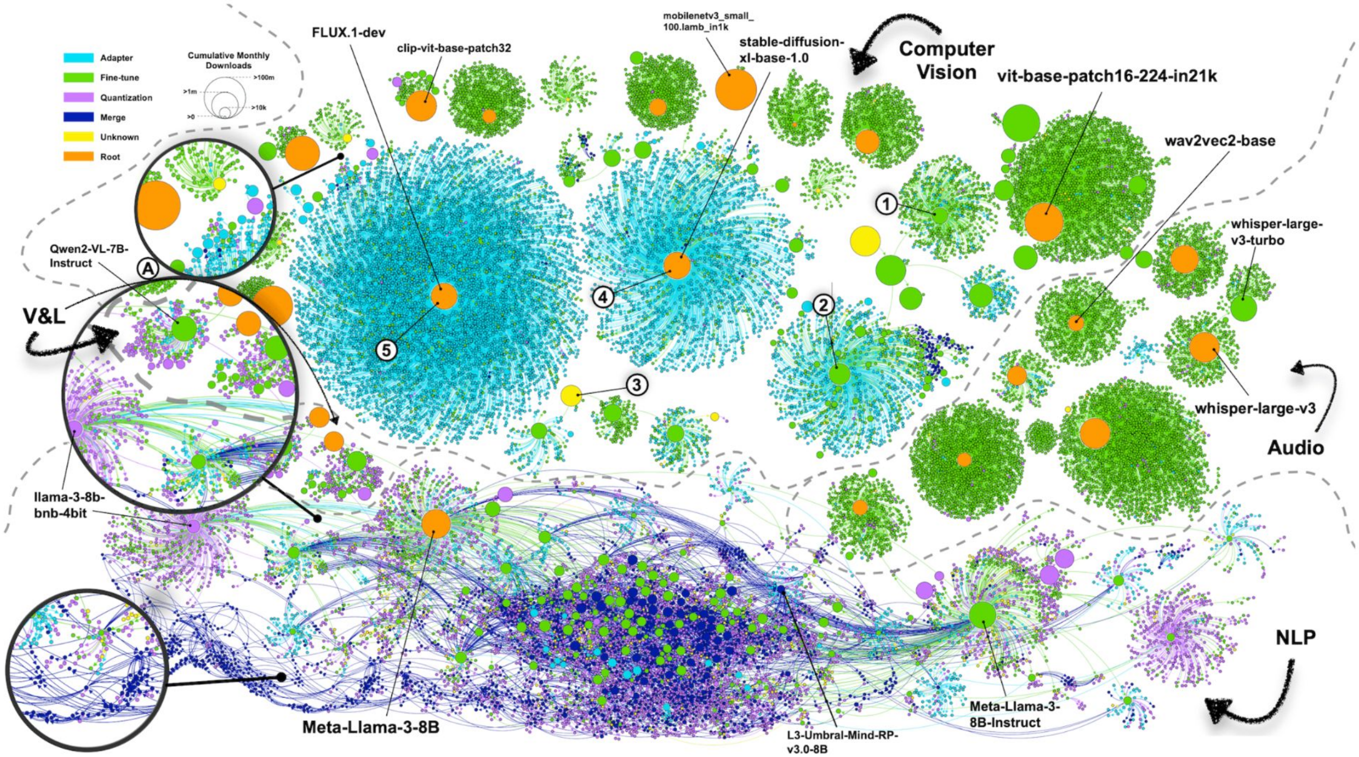
Merged performance is closer to multitask for larger models



From “What Matters for Model Merging at Scale?" by Yadav et al.

🚀 Applications

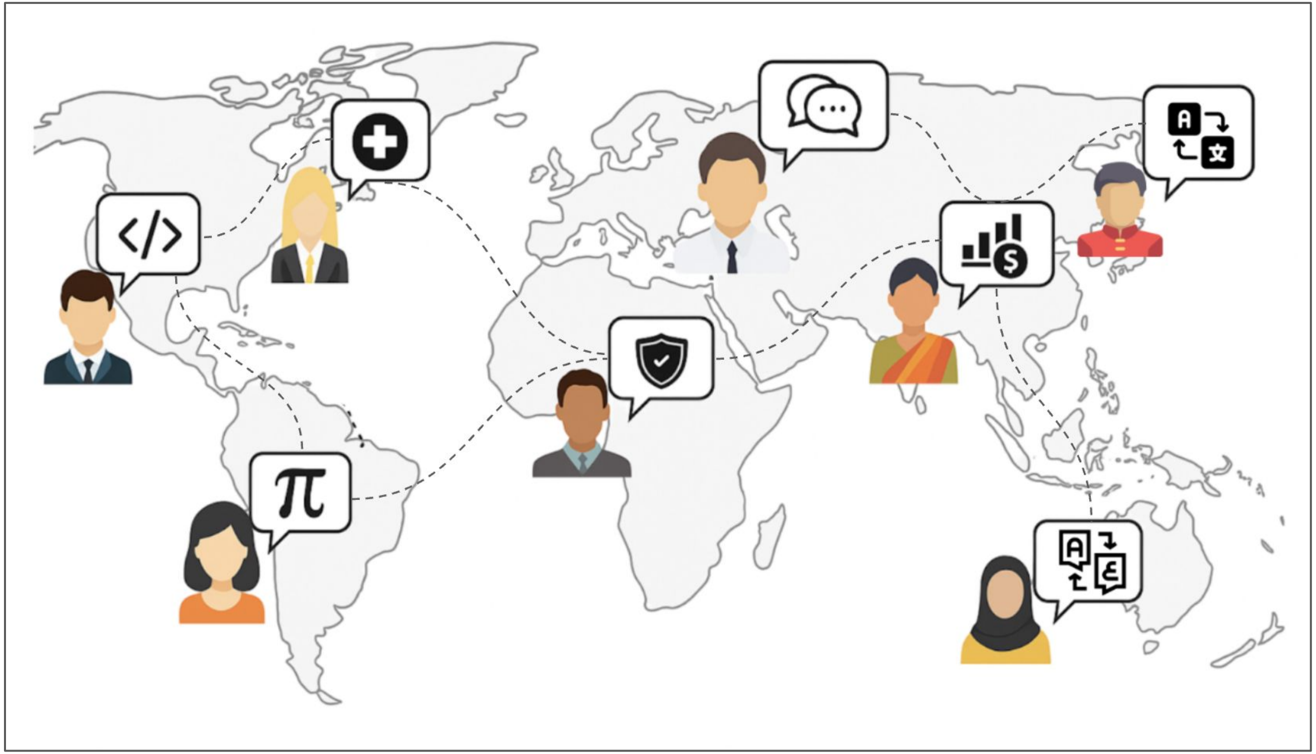
Merged LLMs Frequently Overtake Parent Models in Popularity on Hugging Face

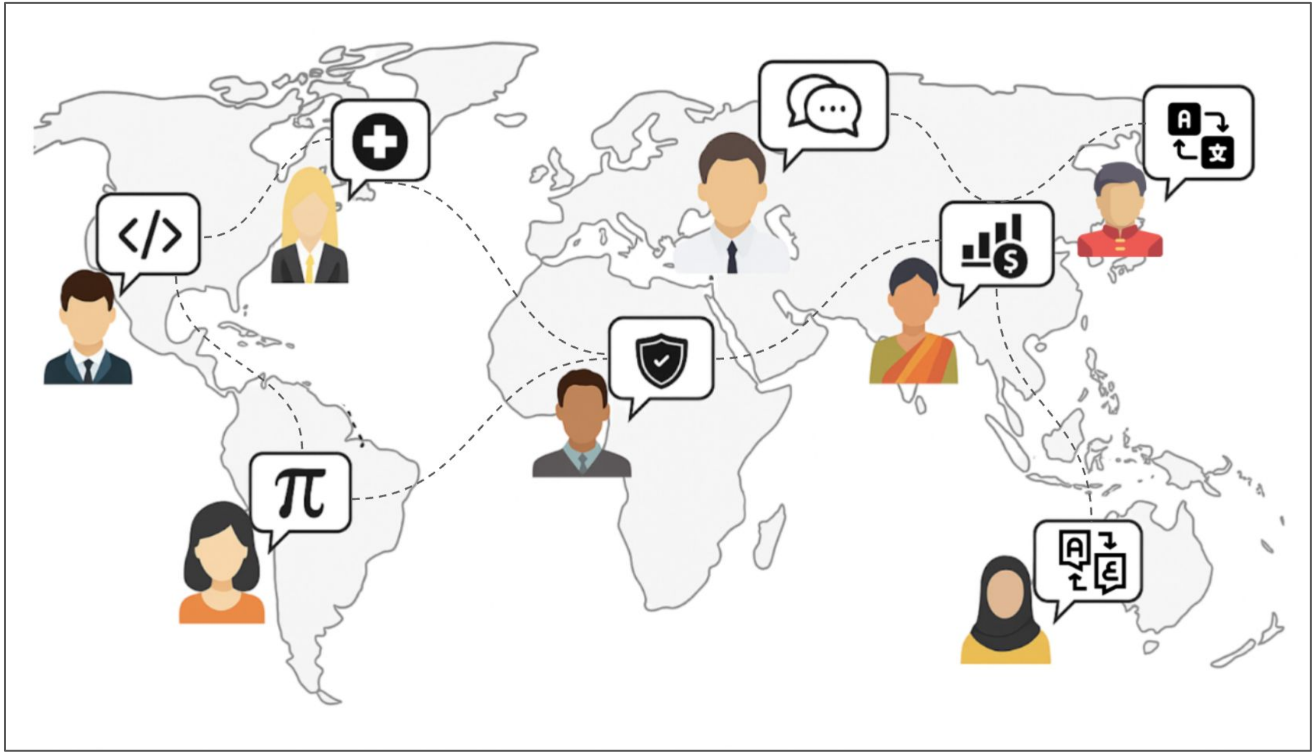


From “Charting and Navigating Hugging Face's Model Atlas" by Horwitz, Eliahu, et al.

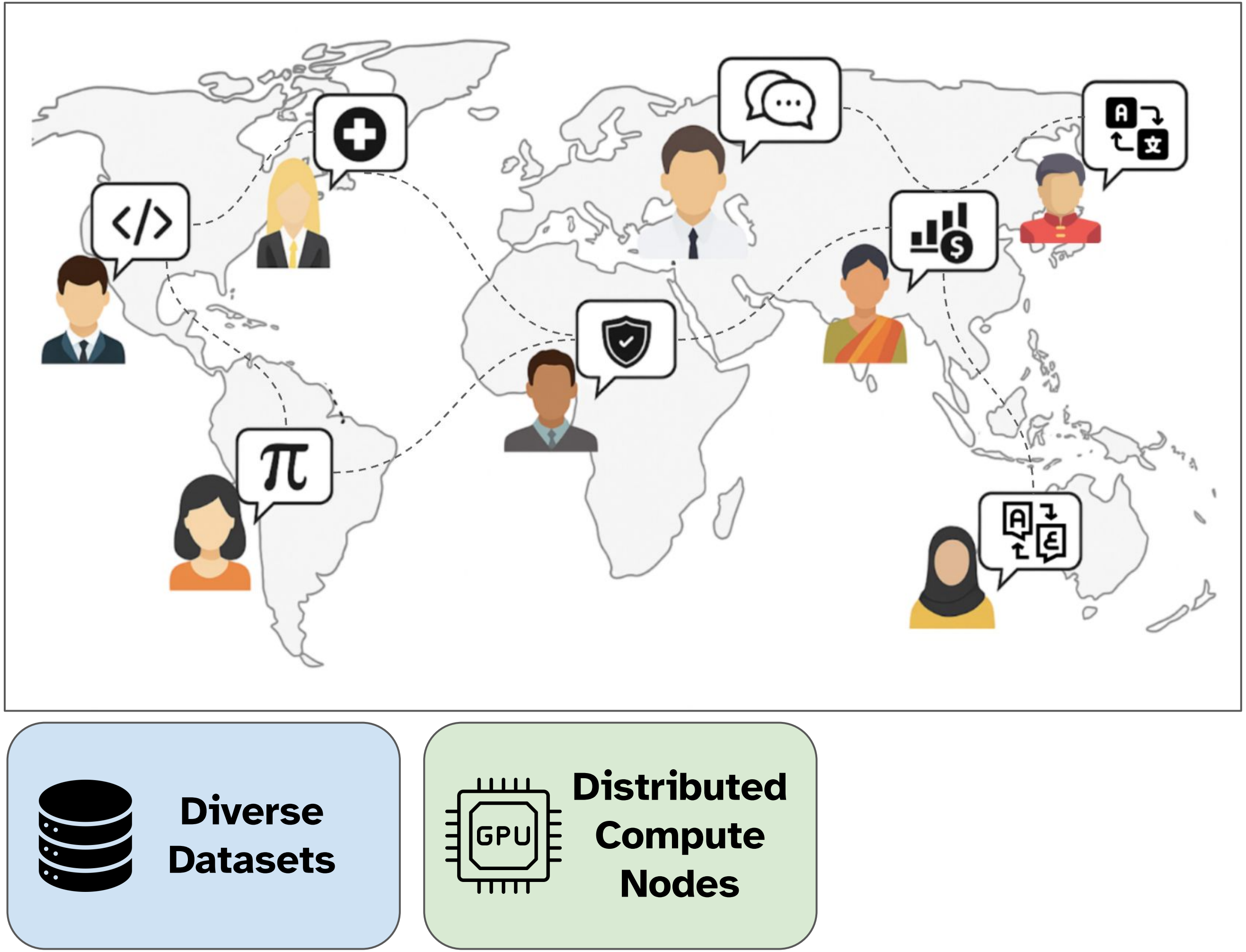
But Why?

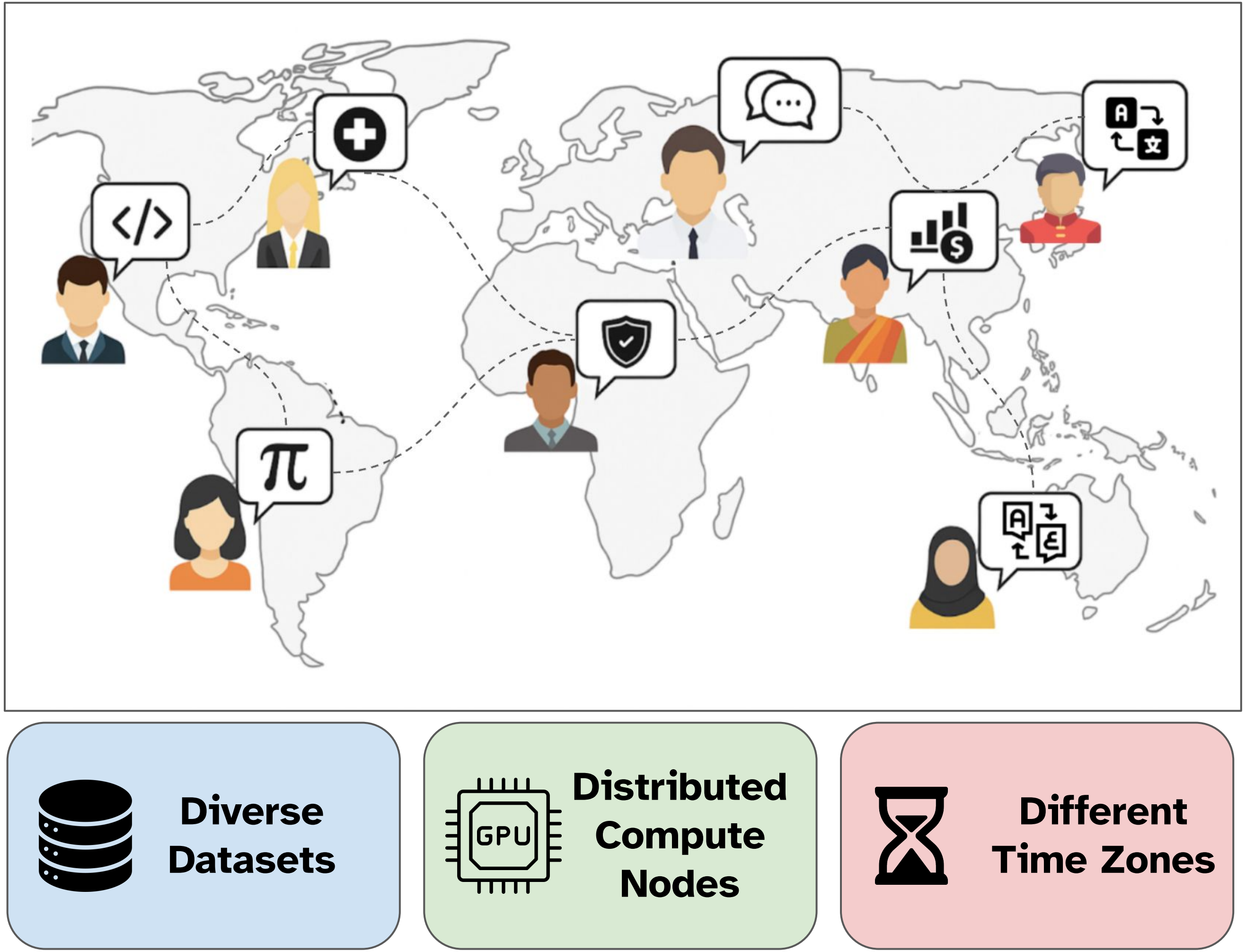
What applications are driving widespread adoption?



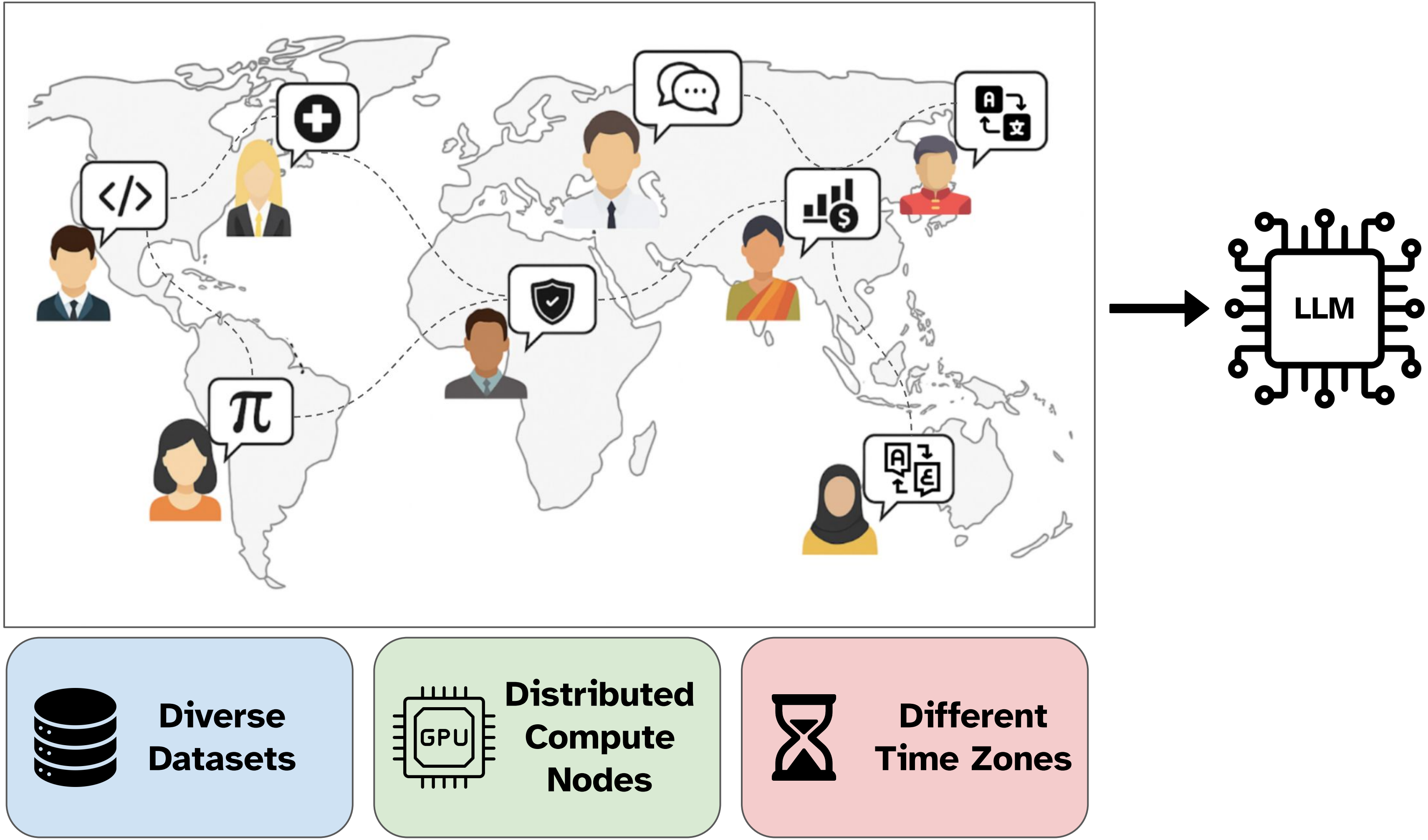


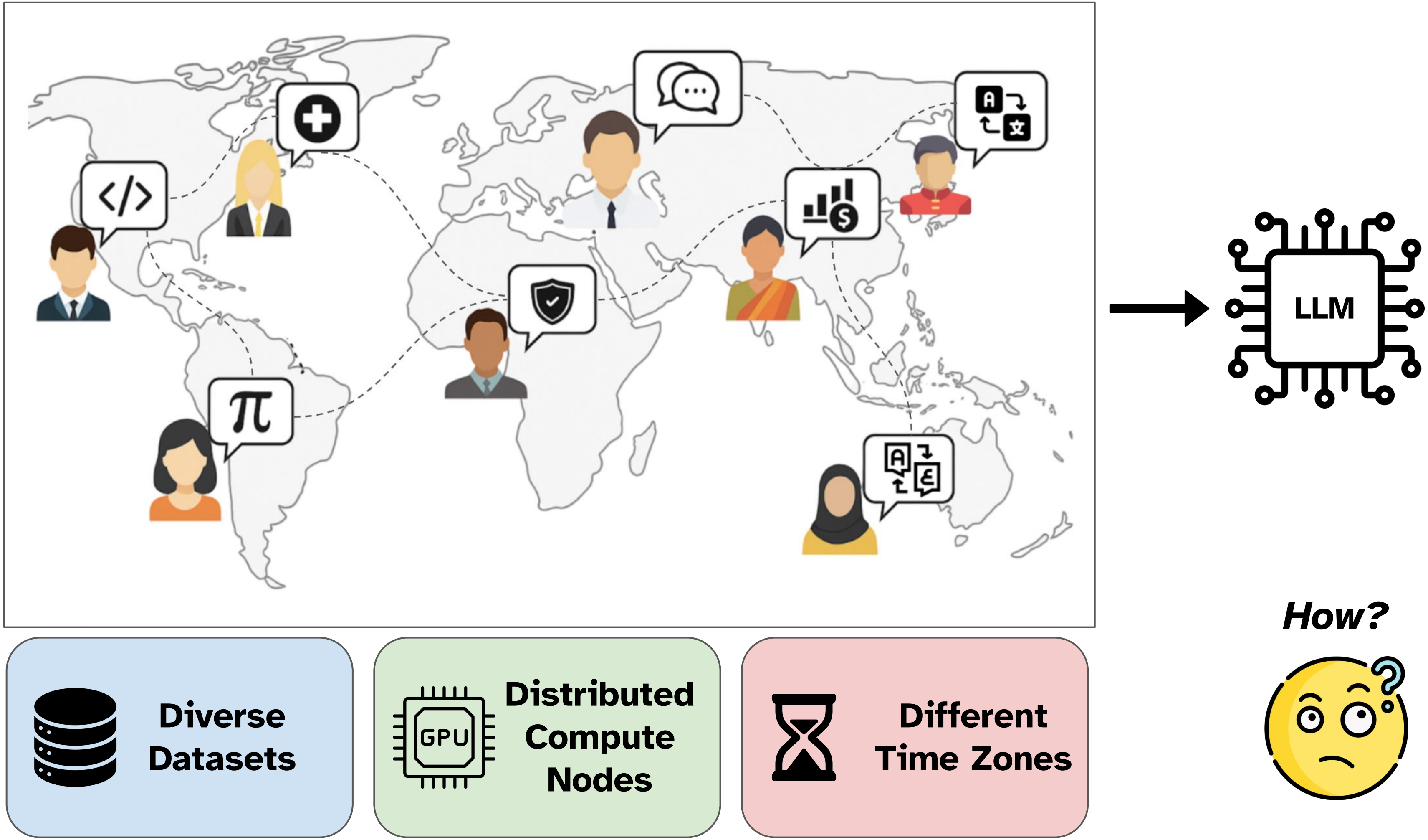
Diverse Datasets

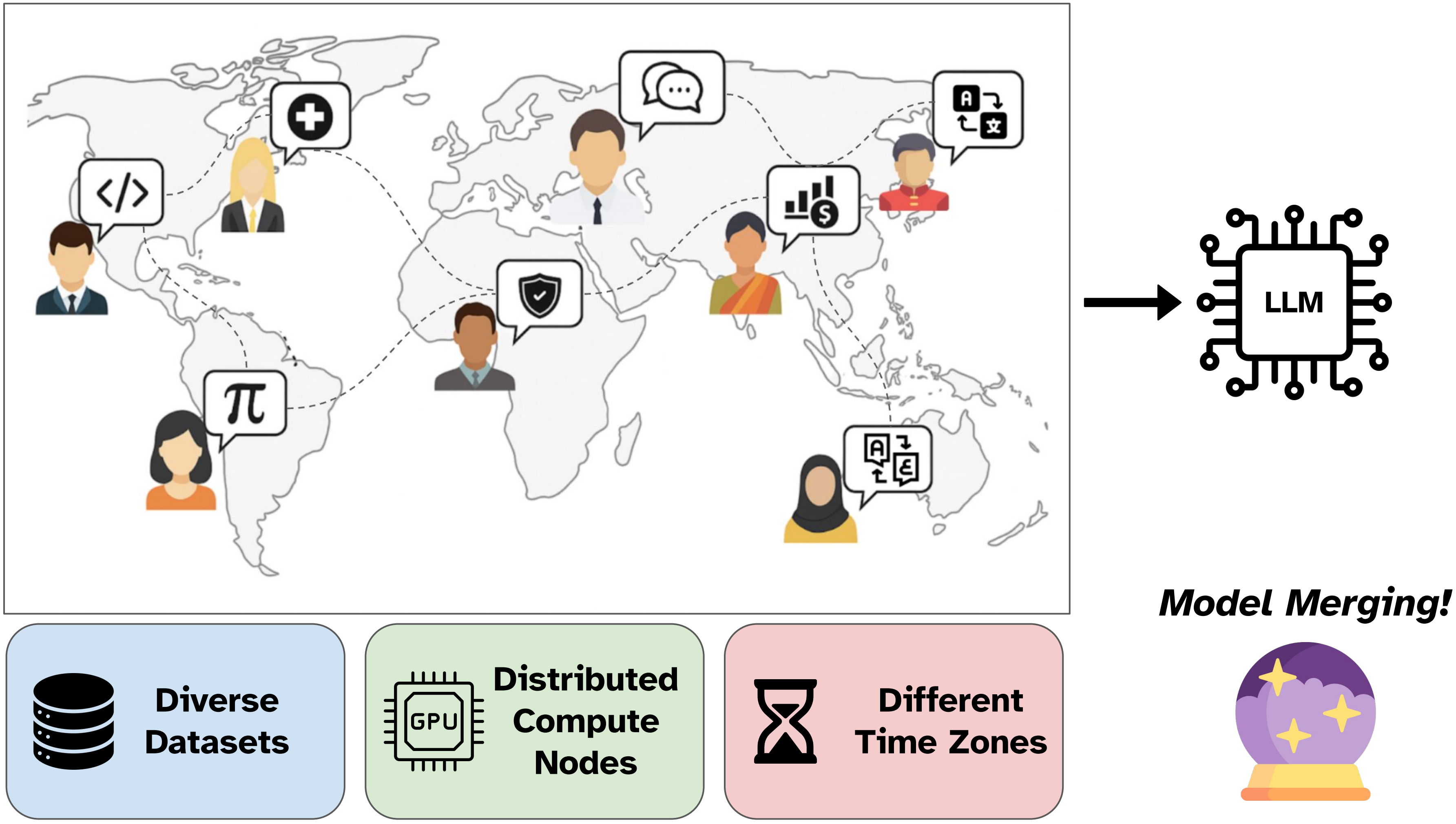

Diverse Datasets
GPU
Distributed Compute Nodes



Diverse Datasets
GPU
Distributed Compute Nodes
Different Time Zones


LLM
Diverse Datasets
GPU
Distributed Compute Nodes
Different Time Zones


LLM
How?
Diverse Datasets
GPU
Distributed Compute Nodes
Different Time Zones



LLM
Model Merging!
Diverse Datasets
GPU
Distributed Compute Nodes
Different Time Zones

Model Merging Helps Overcome Core Challenges across the Model development Lifecycle.

Combining Model Capabilities

Multiple tasks, languages, and modalities to integrate

Decentralized Training

Distributed compute and human resources across organizations

Continual Model Development

Continuous training and evolution of models

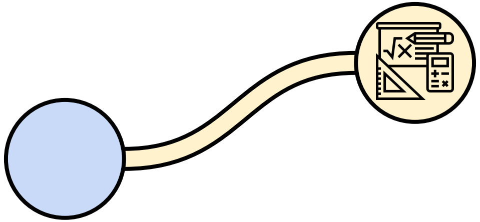
Pluralistic Alignment

Post-hoc multi-objective optimization for diverse stakeholders

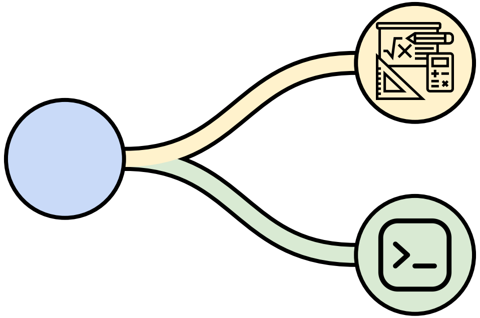
Generalization and Robustness

Ensuring robust, and compositionally generalizable behaviour in the real-world

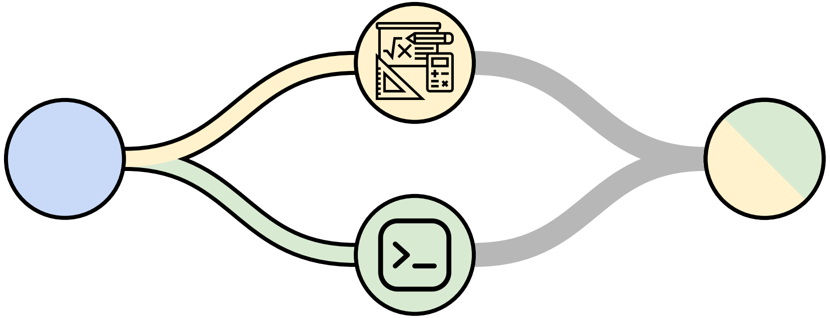
Task-specific models are merged to create multitask capability



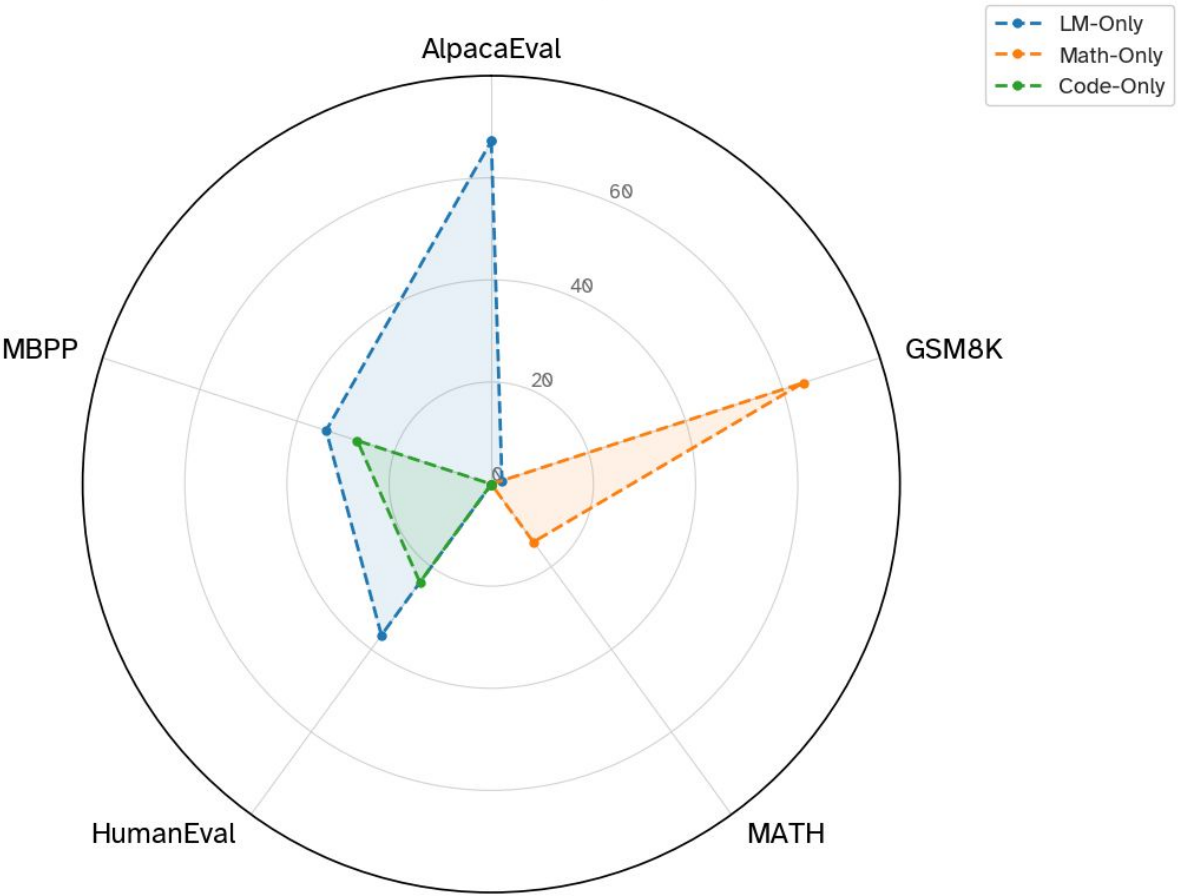
Task-specific models are merged to create multitask capability



Task-specific models are merged to create multitask capability

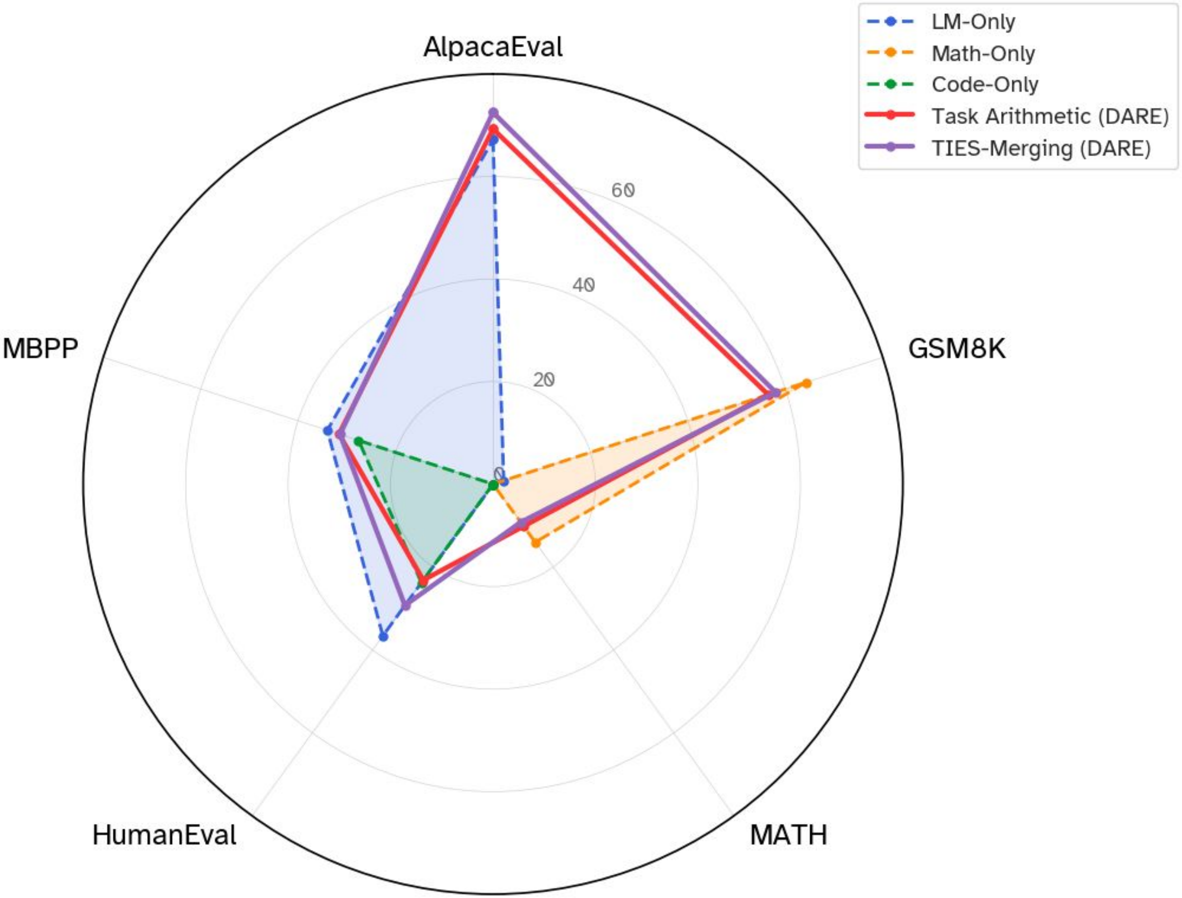


Task-specific models are merged to create multitask capability



From "Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch" by Yu, Le, et al.

Task-specific models are merged to create multitask capability



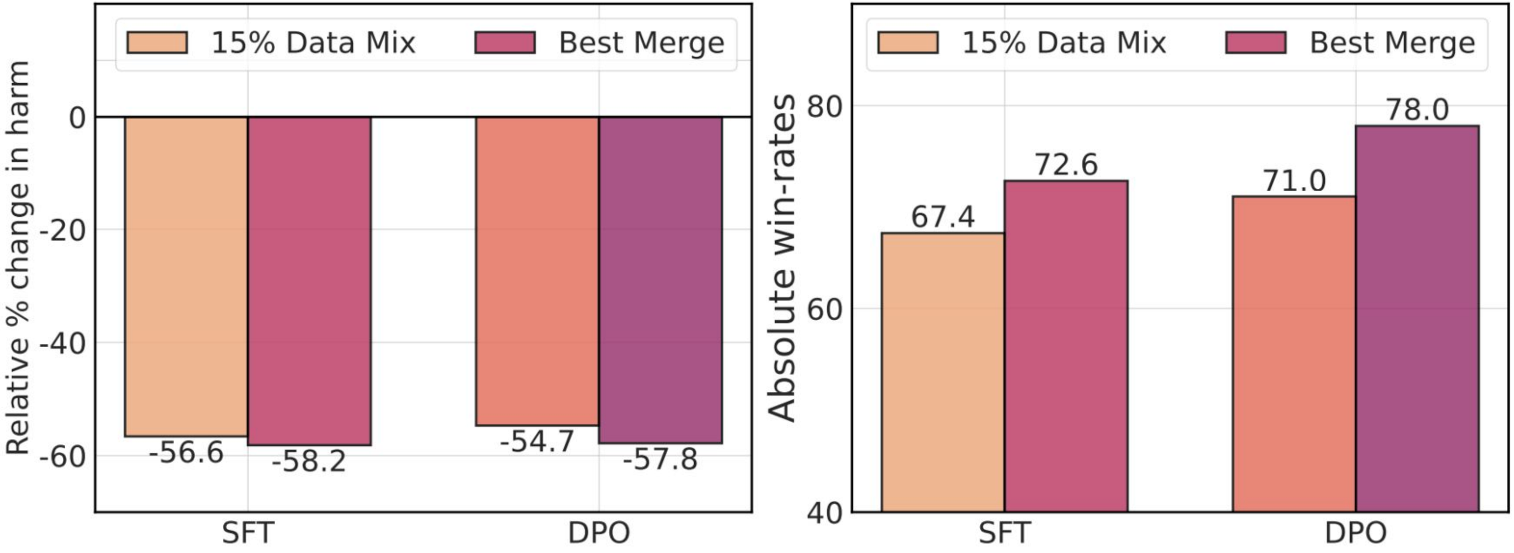
From "Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch" by Yu, Le, et al.

Task-specific models are merged to create multitask capability

Merging Methods	Models	Use DARE	AlpacaEval	GSM8K	MATH	HumanEval	MBPP
/	LM	No	67.20	2.20	0.04	36.59	34.00
	Math	No	/	64.22	14.02	/	/
	Code	No	/	/	/	23.78	27.60
Task Arithmetic	LM & Math & Code	No	69.03	58.45	9.88	18.29	29.80
	LM & Math & Code	Yes	69.28	56.48	10.16	23.17	31.60
TIES-Merging	LM & Math & Code	No	65.91	62.55	9.54	21.95	30.40
	LM & Math & Code	Yes	72.50	58.00	9.20	29.27	31.40

From "Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch" by Yu, Le, et al.

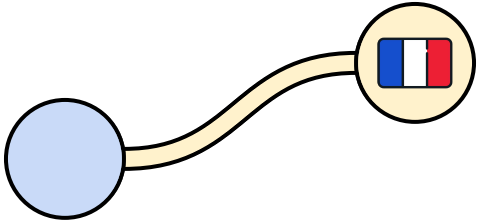
Model merging can beat data mixing for multitask learning



Safety Performance **General Performance**

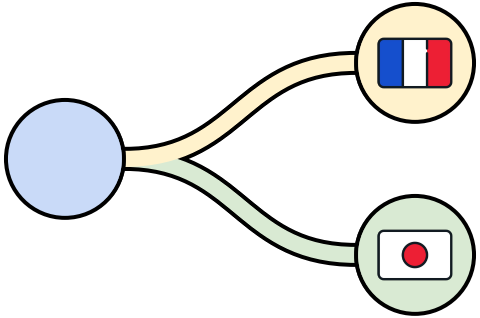
From "Mix Data or Merge Models? Optimizing for Diverse Multi-Task Learning" by Ahmadian, Arash, et al.

Monolingual models are merged to create multilingual capability



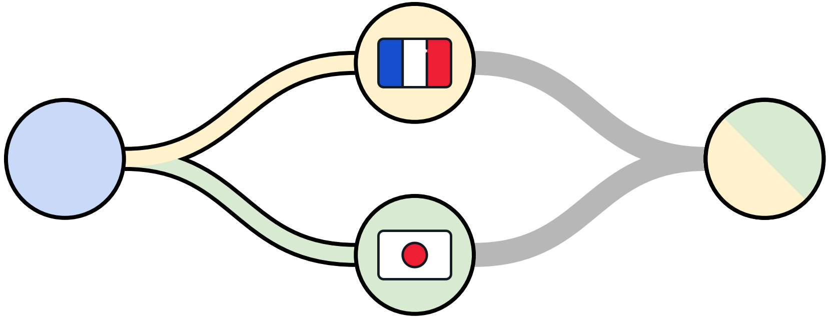
From "Mix Data or Merge Models? Optimizing for Diverse Multi-Task Learning" by Ahmadian, Arash, et al.

Monolingual models are merged to create multilingual capability



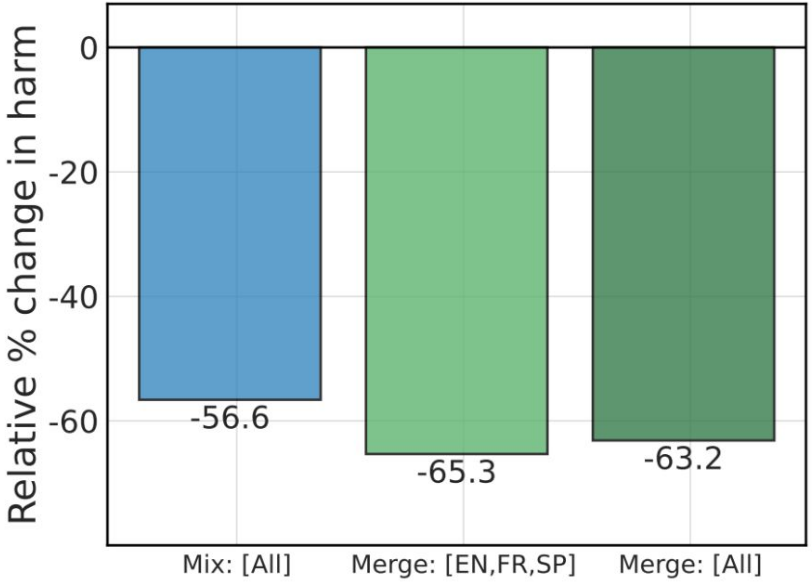
From “Mix Data or Merge Models? Optimizing for Diverse Multi-Task Learning" by Ahmadian, Arash, et al.

Monolingual models are merged to create multilingual capability

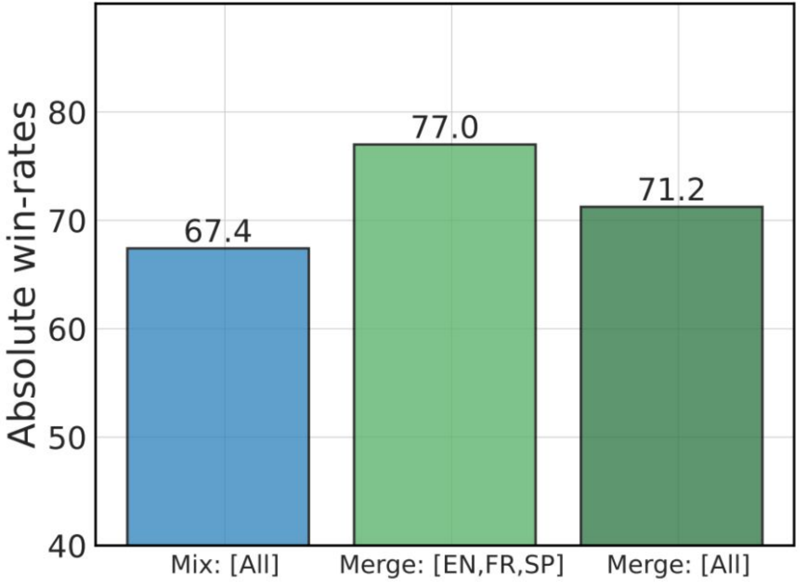


From "Mix Data or Merge Models? Optimizing for Diverse Multi-Task Learning" by Ahmadian, Arash, et al.

Monolingual models are merged to create multilingual capability



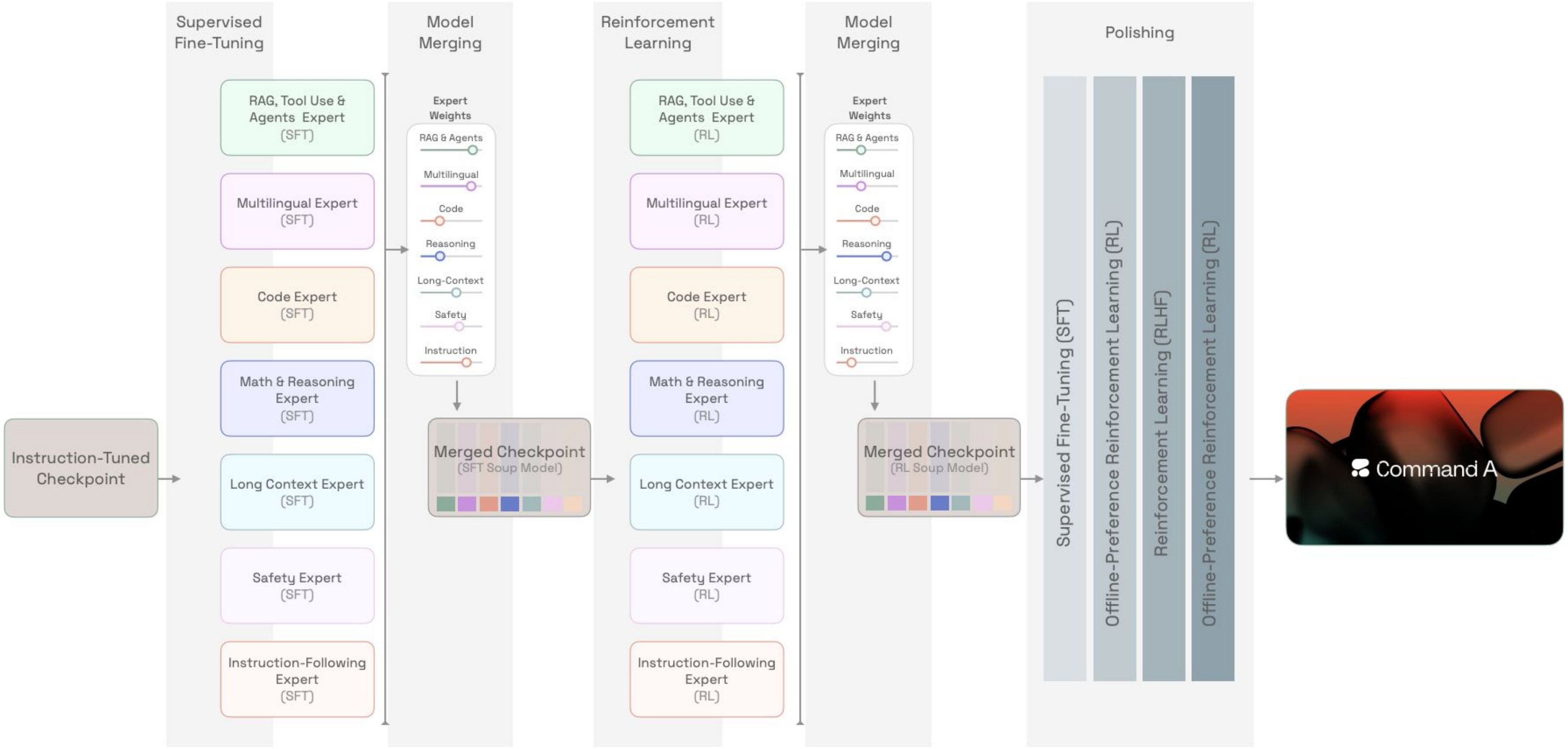
Safety Performance



General Performance

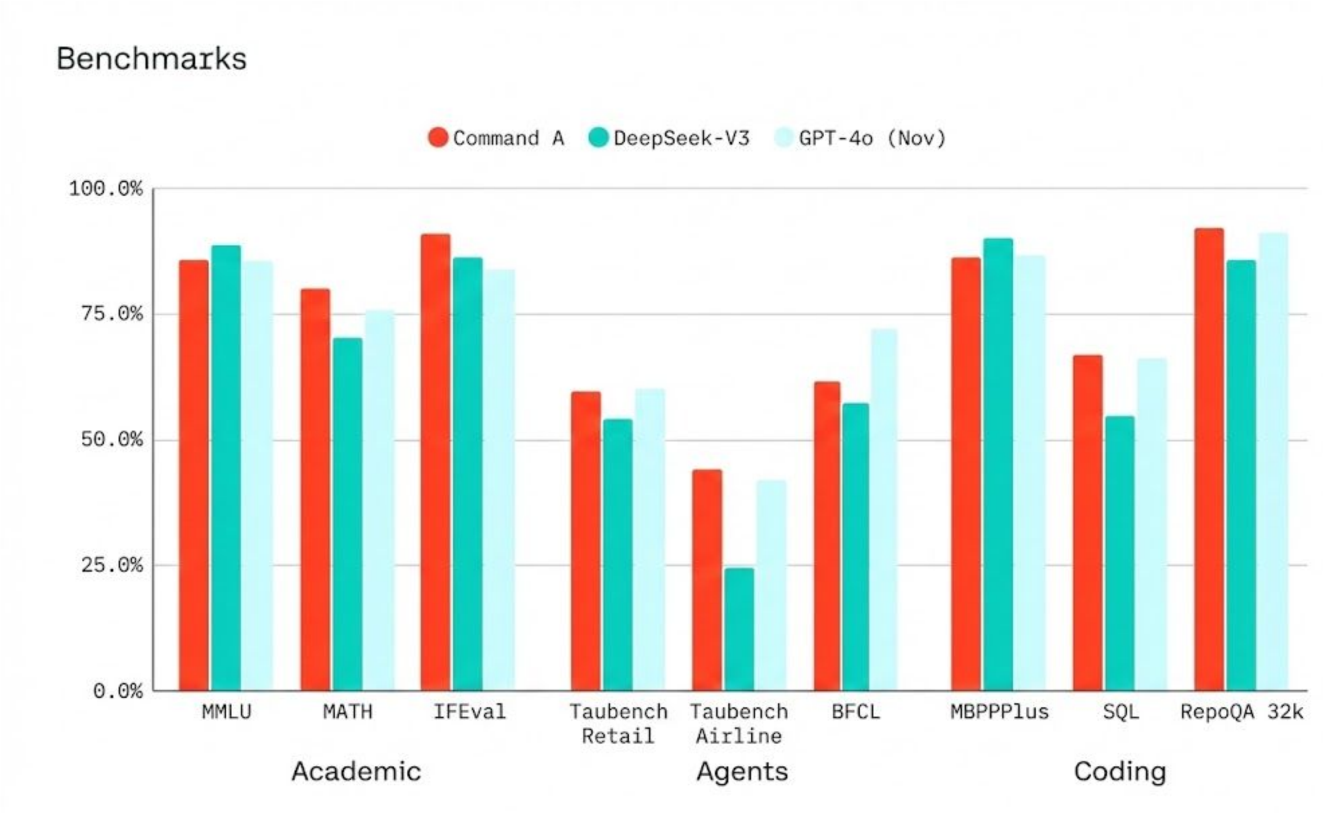
From "Mix Data or Merge Models? Optimizing for Diverse Multi-Task Learning" by Ahmadian, Arash, et al.

Command A is a successful real-world example of multitask and multilingual learning via model merging.



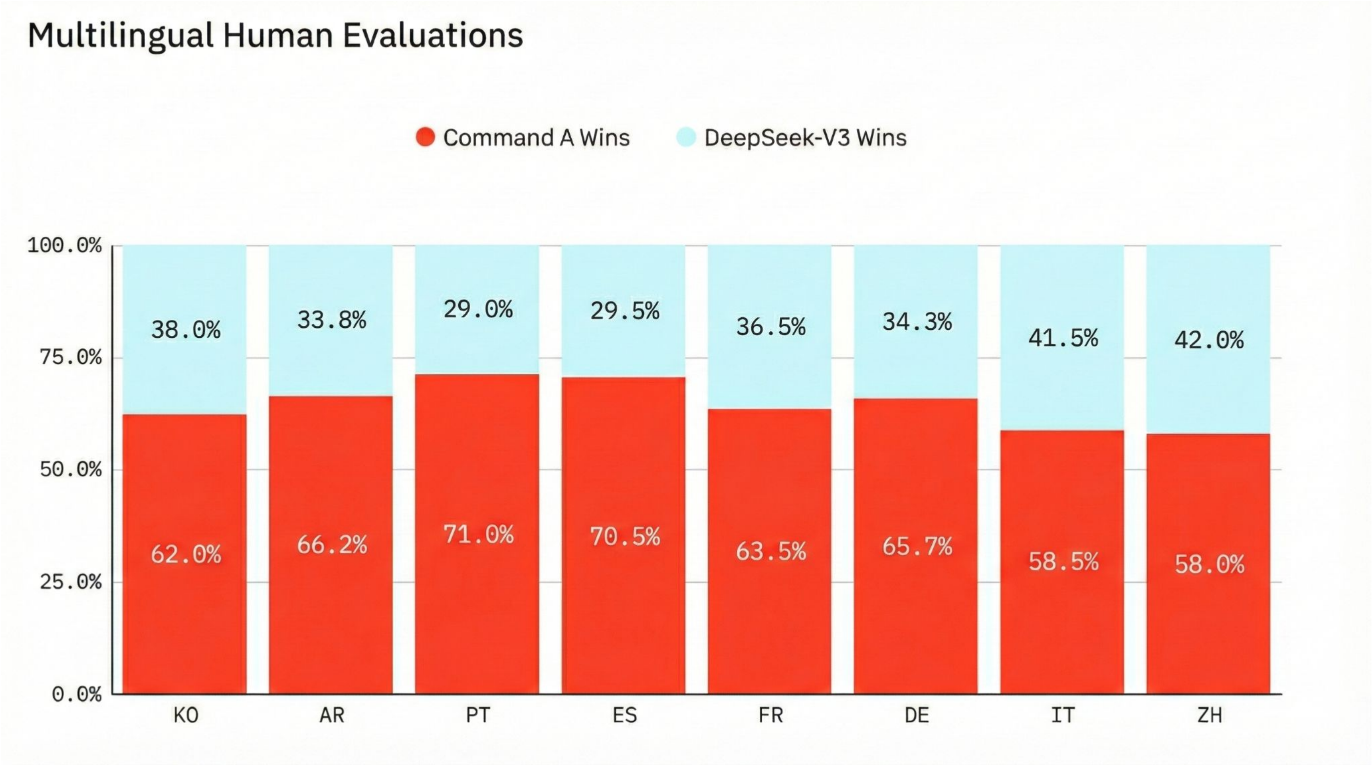
From "Command A: An Enterprise-Ready Large Language Model" by Ahmadian, Cohere, Team, et al.

Command A is on par or better than GPT-4o and DeepSeek-V3 across agentic enterprise tasks with expanded support for 23 languages



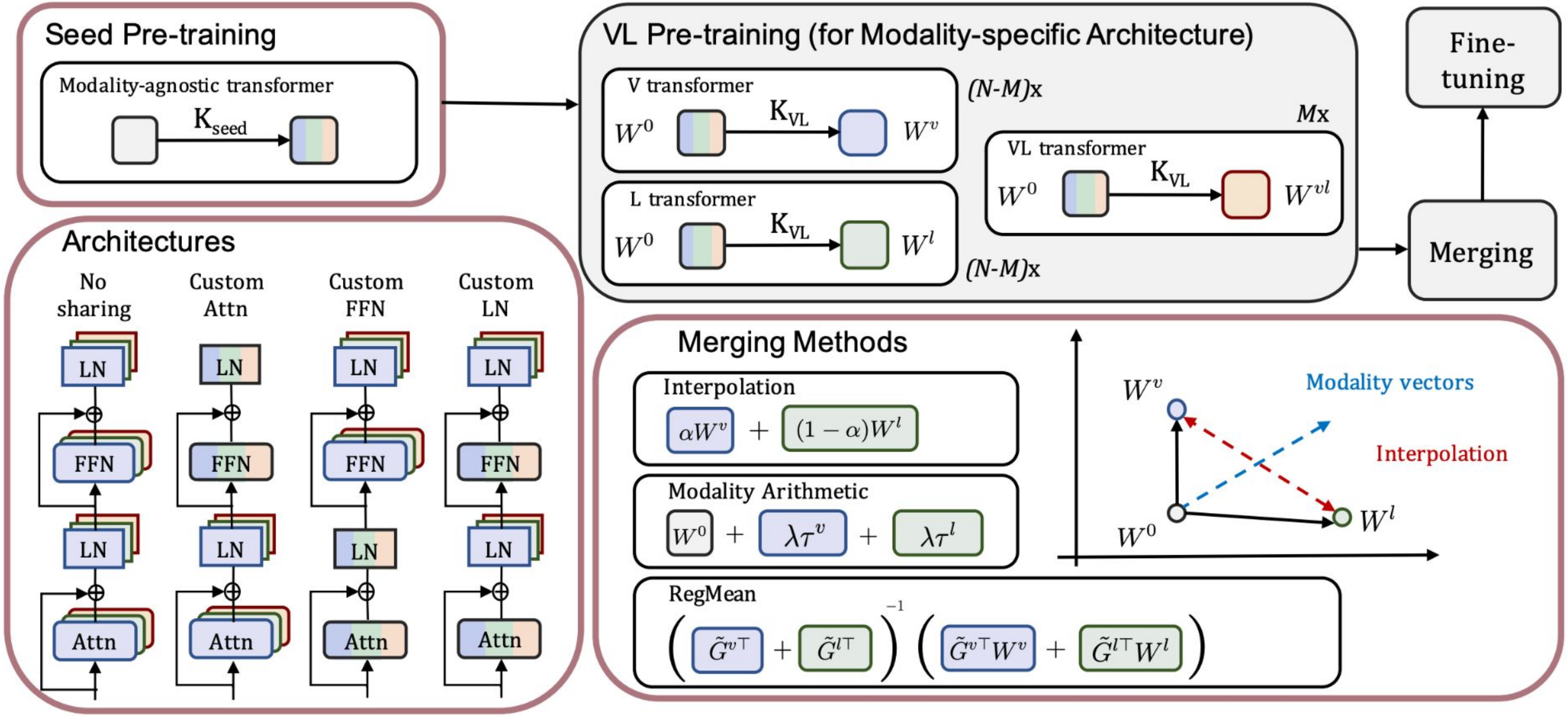
From "Command A: An Enterprise-Ready Large Language Model" by Ahmadian, Cohere, Team, et al.

Command A is on par or better than GPT-4o and DeepSeek-V3 across agentic enterprise tasks with expanded support for 23 languages



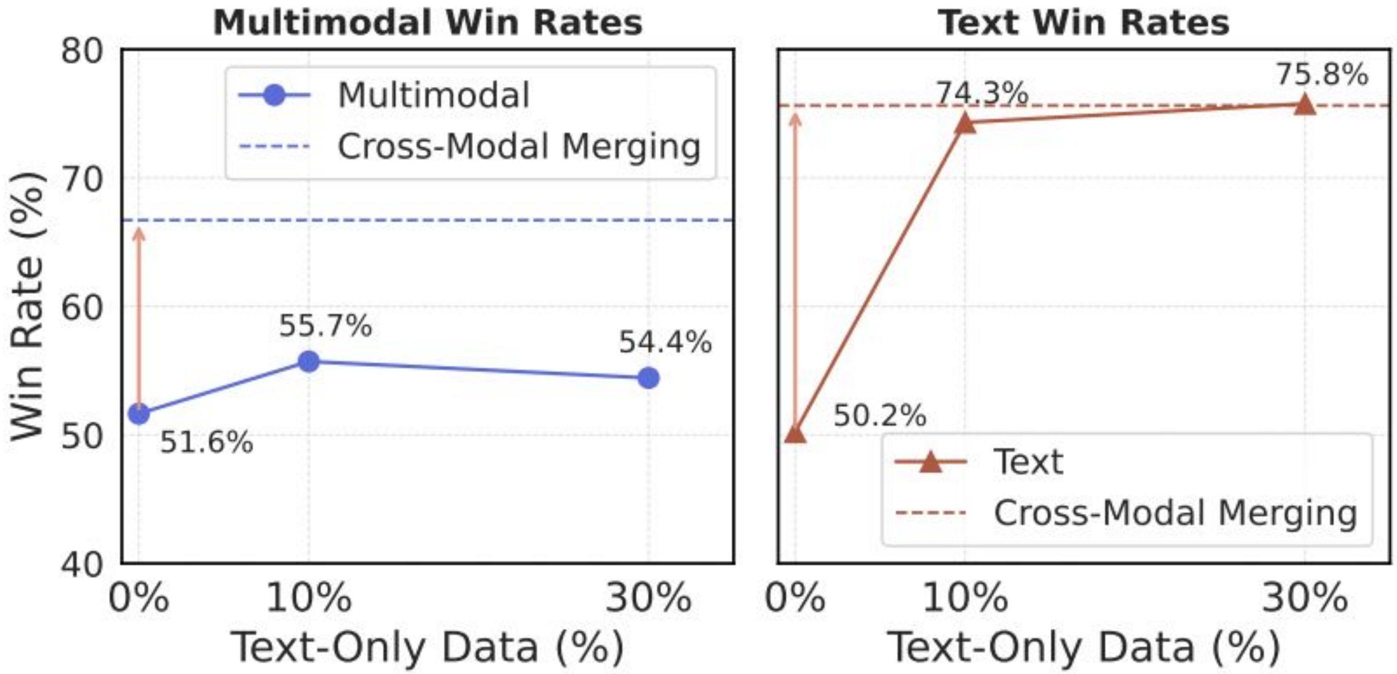
From "Command A: An Enterprise-Ready Large Language Model" by Ahmadian, Cohere, Team, et al.

Merging modality-specific models enables a simple path to multimodal learning



From "An Empirical Study of Multimodal Model Merging" by Sung, Yi-Lin, et al.

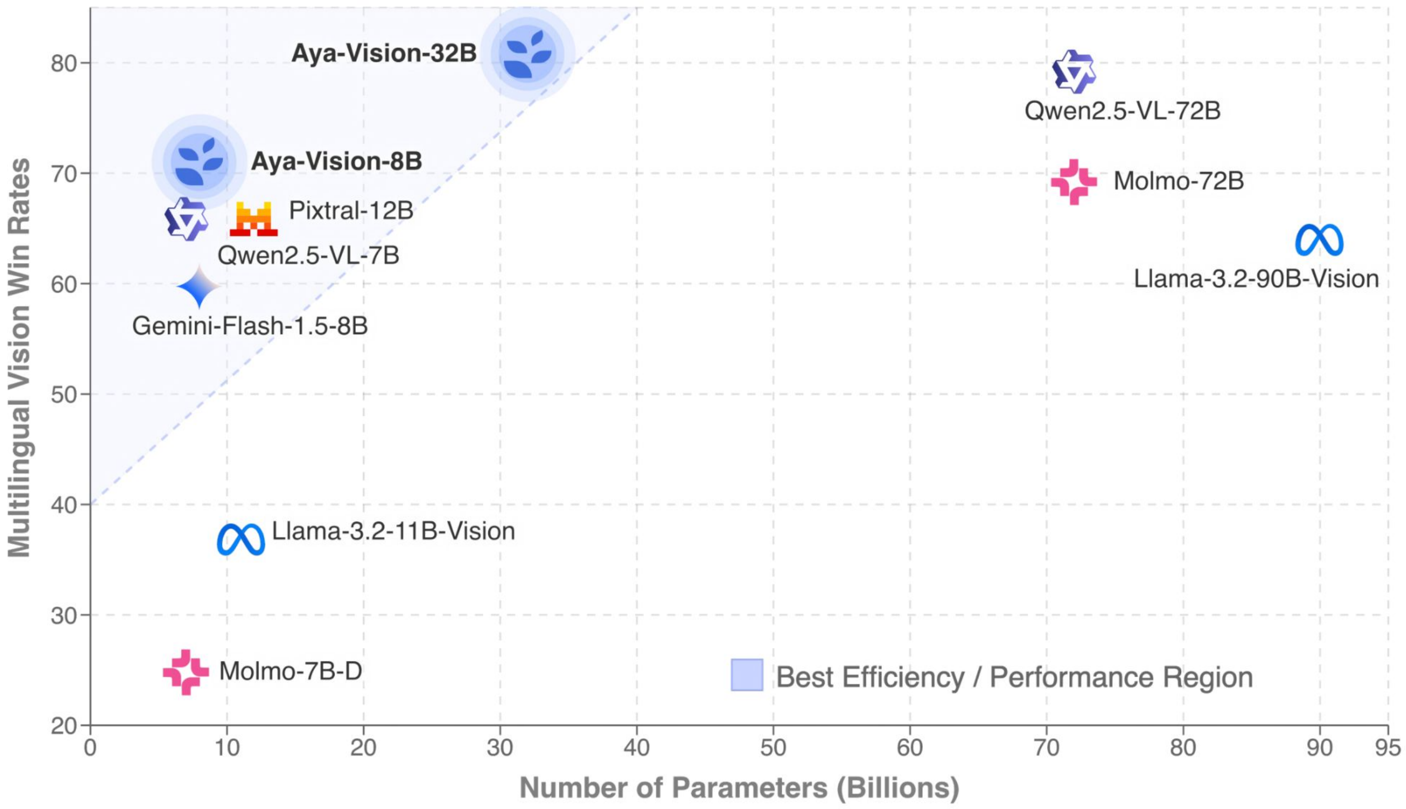
Cross-modal merging provides on-the-fly capability extension



$$W_{merged} = \alpha \cdot W_{mm-LLM} + (1-\alpha) \cdot W_{text-LLM}$$

From "Aya Vision: Advancing the Frontier of Multilingual Multimodality" by Dash, Saurabh, et al.

Aya Vision is a successful example of multimodal learning via model merging



From "Aya Vision: Advancing the Frontier of Multilingual Multimodality" by Dash, Saurabh, et al.

Model Merging Helps Overcome Core Challenges across the Model development Lifecycle.

Combining Model Capabilities

Multiple tasks, languages, and modalities to integrate

Decentralized Training

Distributed compute and human resources across organizations

Continual Model Development

Continuous training training and evolution of models

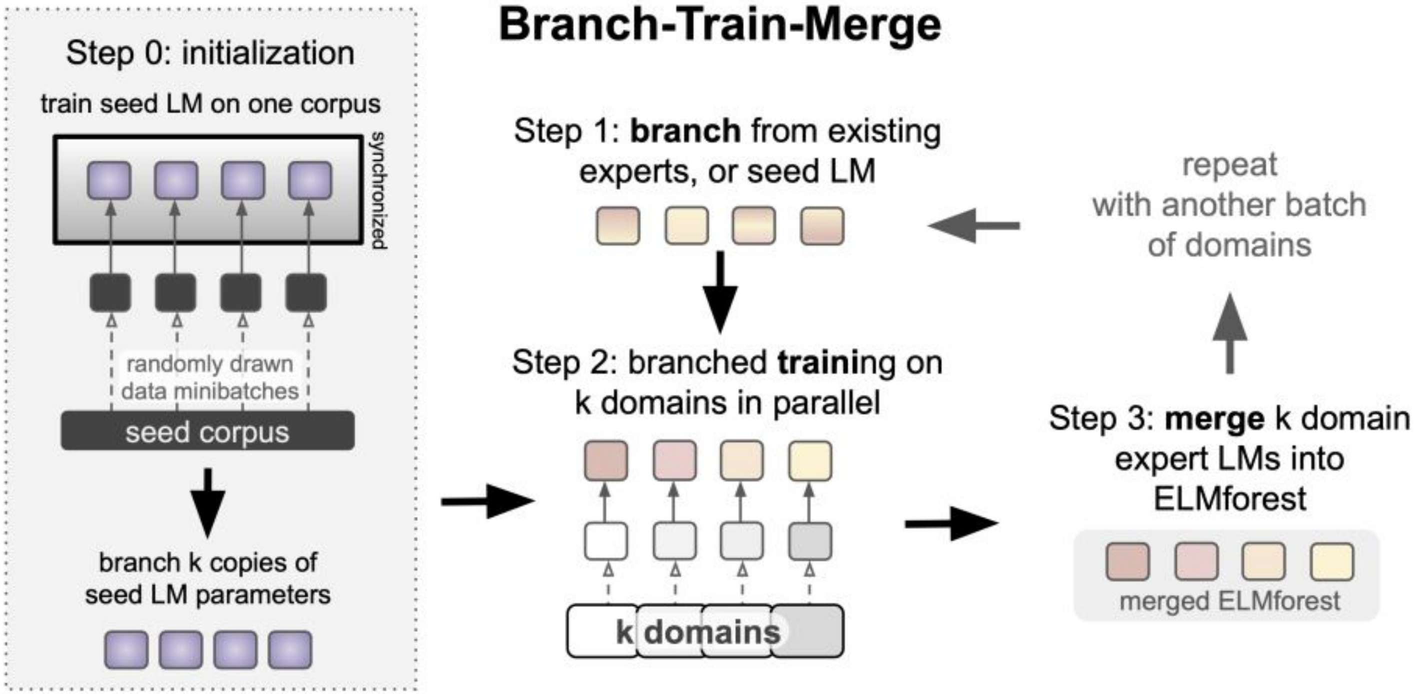
Pluralistic Alignment

Post-hoc multi-objective optimization for diverse stakeholders

Generalization and Robustness

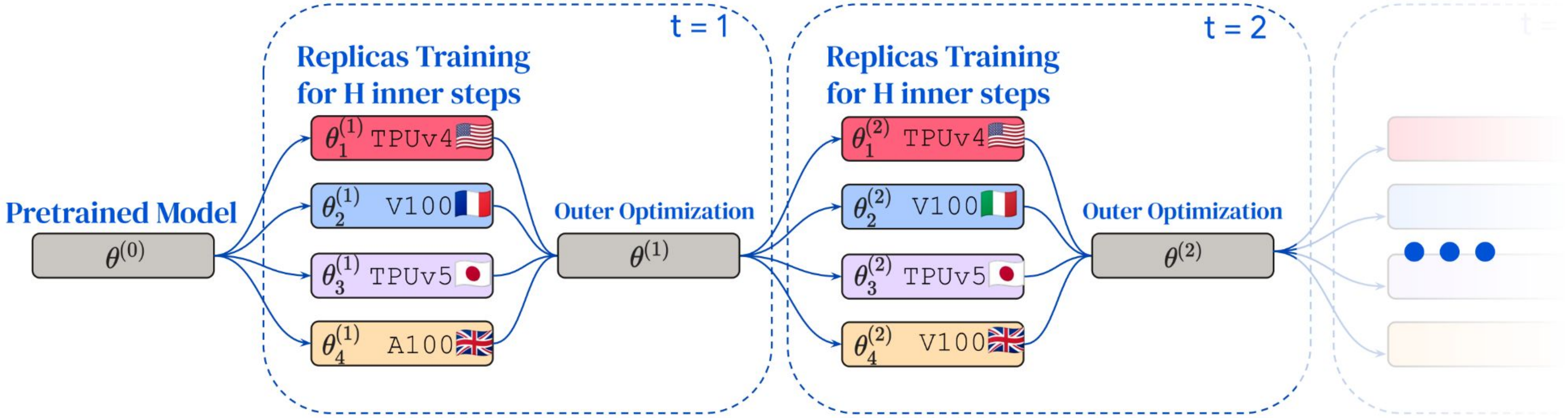
Ensuring robust, and compositionally generalizable behaviour in the real-world

Merging enables massive parallel training of specialized models



From "Branch-Train-Merge: Embarrassingly Parallel Training of Expert Language Models" by Li, Margaret, et al.

Periodic merging allows decentralized training with low-communication



From "DiLoCo: Distributed Low-Communication Training of Language Models" by Douillard, Arthur, et al.

INTELLECT-1 is a successful real-world example of decentralized trained model using DiLoCo

INTELLECT-1: The First Decentralized Trained 10B Model, trained on 1 trillion tokens using up to 14 concurrent nodes distributed across 3 continents, with contributions from 30 independent compute providers.

From "INTELLECT-1 Technical Report" by Jaghouar, Sami, et al.

Model Merging Helps Overcome Core Challenges across the Model development Lifecycle.

Combining Model Capabilities

Multiple tasks, languages, and modalities to integrate

Decentralized Training

Distributed compute and human resources across organizations

Continual Model Development

Continuous training training and evolution of models

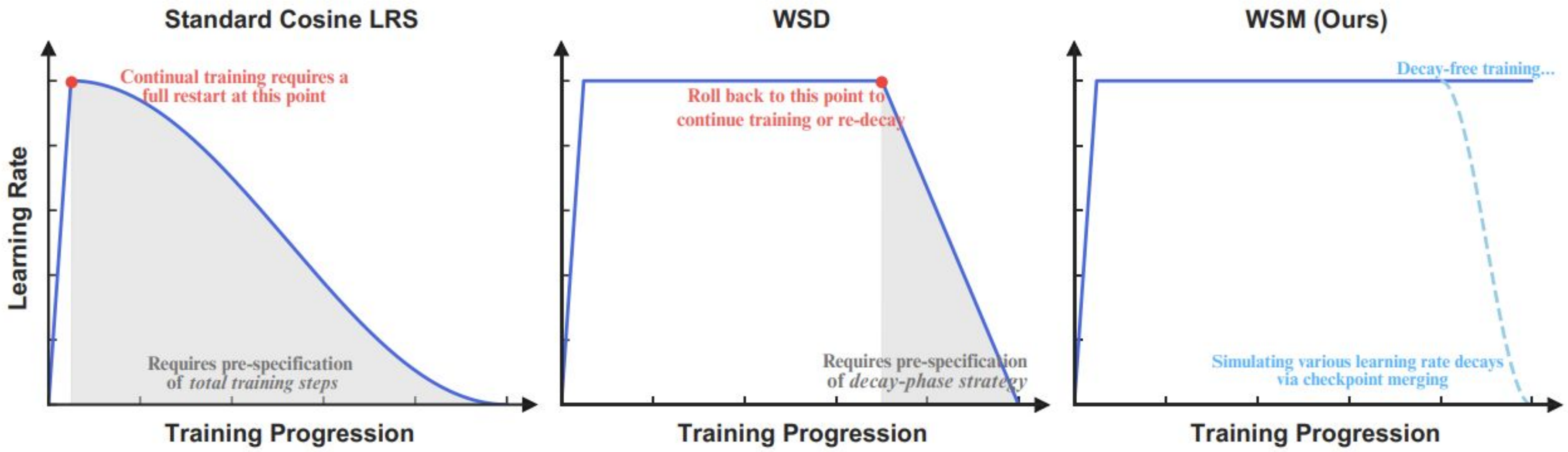
Pluralistic Alignment

Post-hoc multi-objective optimization for diverse stakeholders

Generalization and Robustness

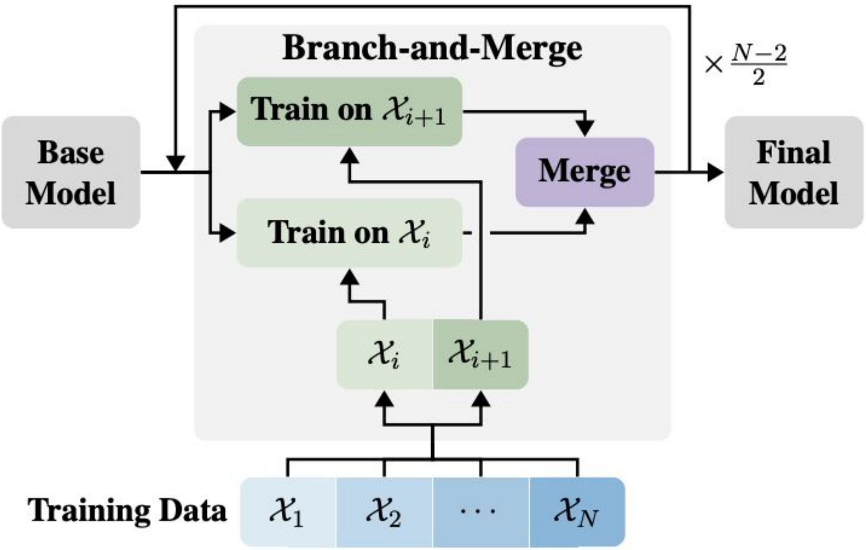
Ensuring robust, and compositionally generalizable behaviour in the real-world

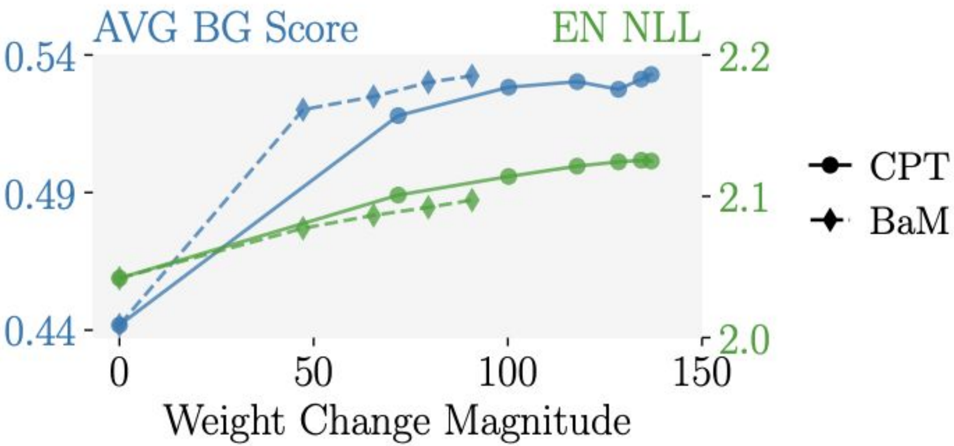
Model merging enables a decay-free LR schedule by checkpoint merging for LLM pretraining



From “WSM: Decay-Free Learning Rate Schedule via Checkpoint Merging for LLM Pre-training" by Tian, Changxin, et al.

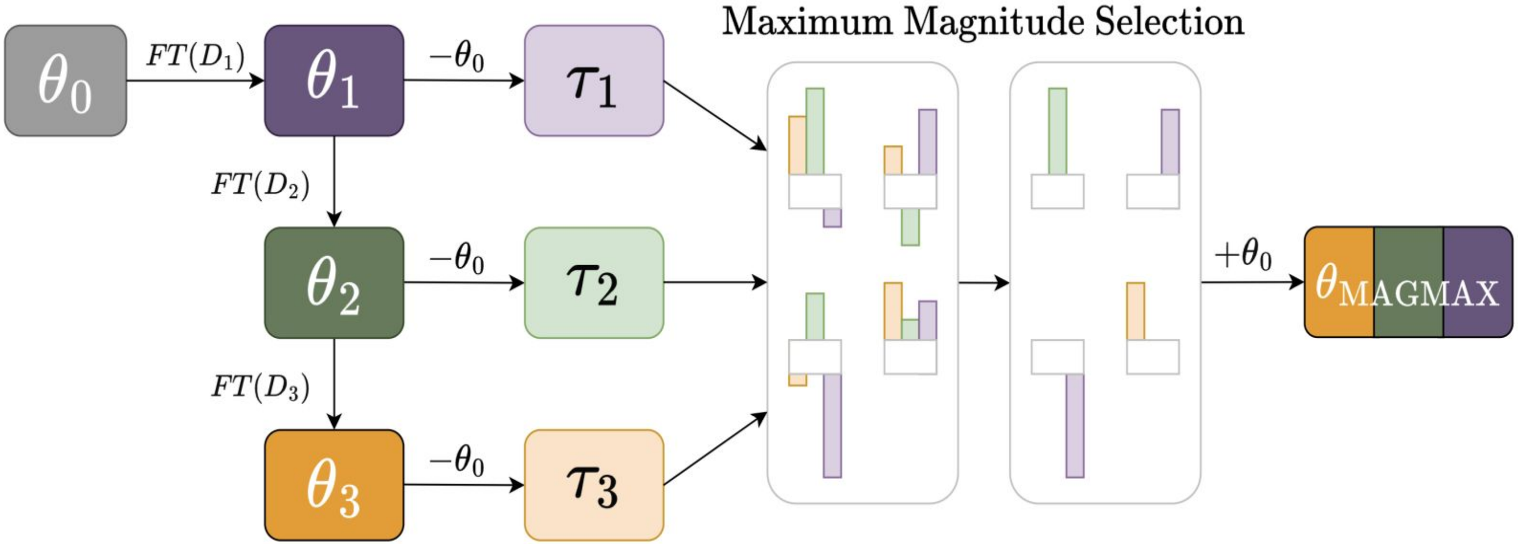
Branch-and-merge helps mitigate catastrophic forgetting in language transfer





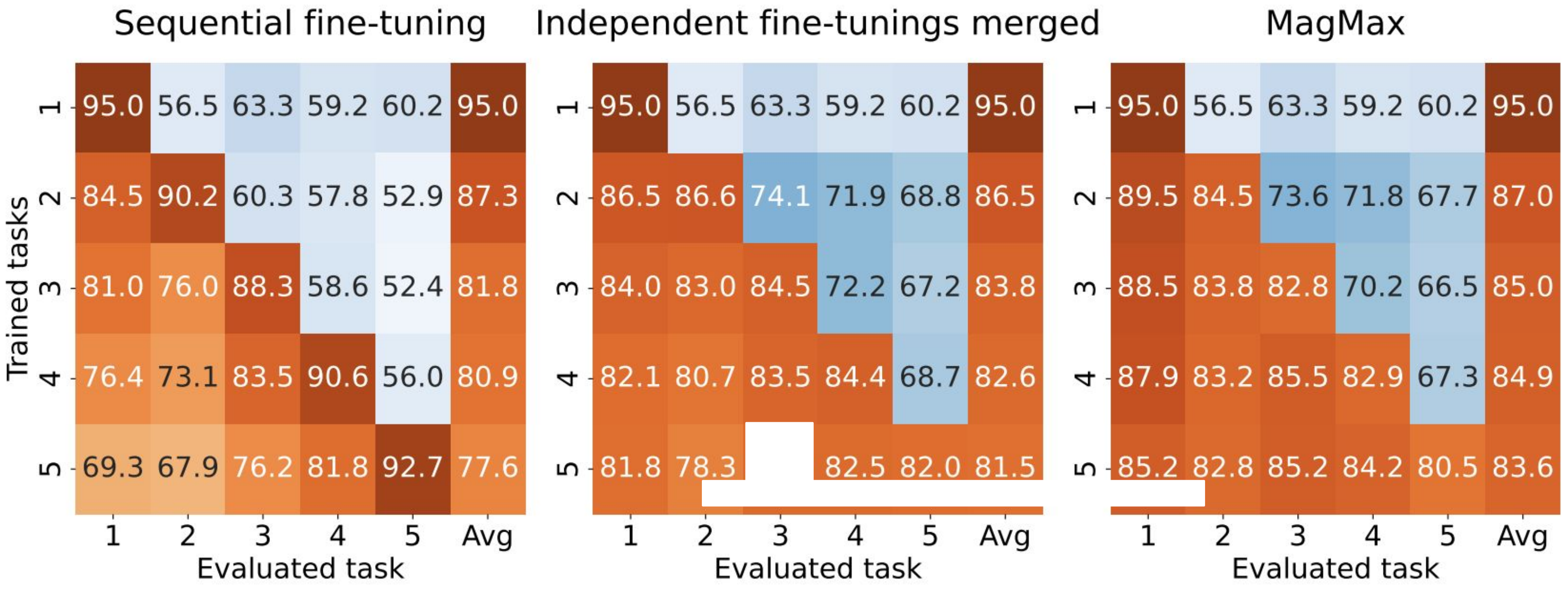
From "Mitigating Catastrophic Forgetting in Language Transfer via Model Merging" by Alexandrov, Anton, et al.

Task-vector merging using maximum-magnitude selection helps consolidate knowledge and mitigate forgetting



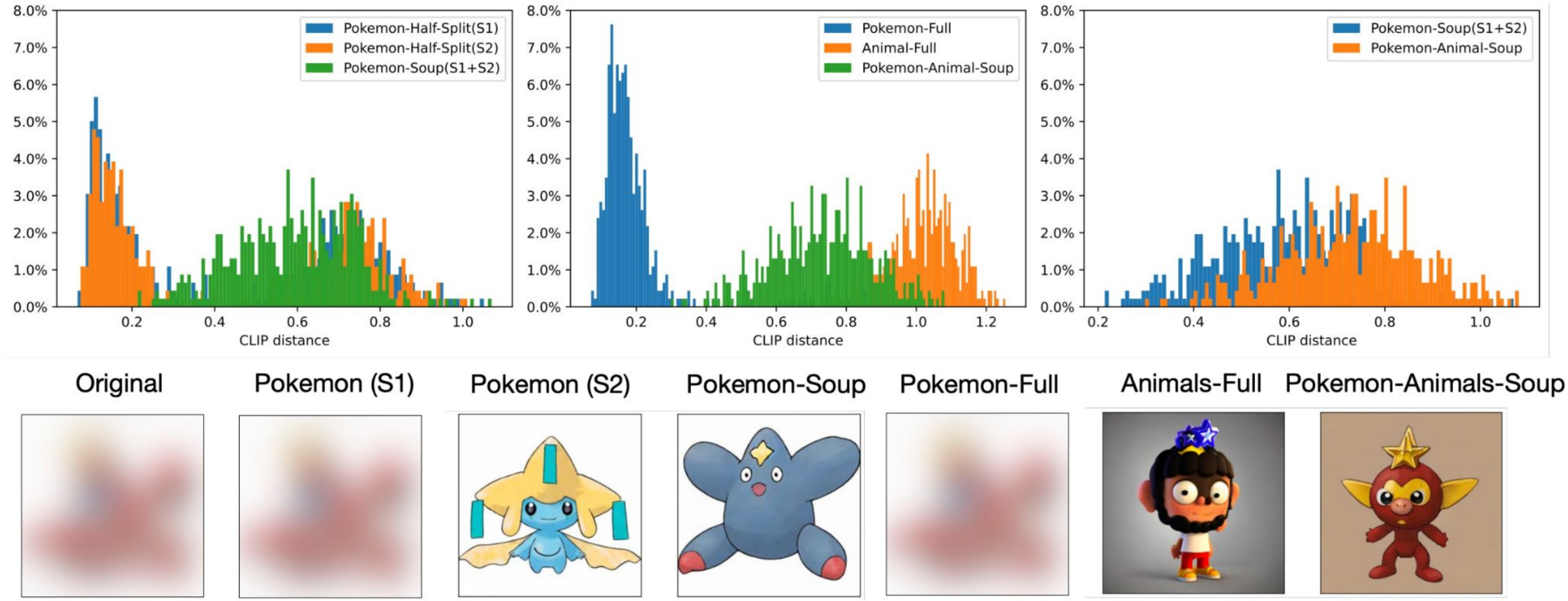
From "MagMax: Leveraging Model Merging for Seamless Continual Learning" by Marczak, Daniel, et al.

Task-vector merging using maximum-magnitude selection helps consolidate knowledge and mitigate forgetting



From "MagMax: Leveraging Model Merging for Seamless Continual Learning" by Marczak, Daniel, et al.

Model merging can help reduce training-image memorization while preserving underlying visual information.



From "Diffusion Soup: Model Merging for Text-to-Image Diffusion Models" by Biggs, Benjamin, et al.

Model Merging Helps Overcome Core Challenges across the Model development Lifecycle.

Combining Model Capabilities

Multiple tasks, languages, and modalities to integrate

Decentralized Training

Distributed compute and human resources across organizations

Continual Model Development

Continuous training and evolution of models

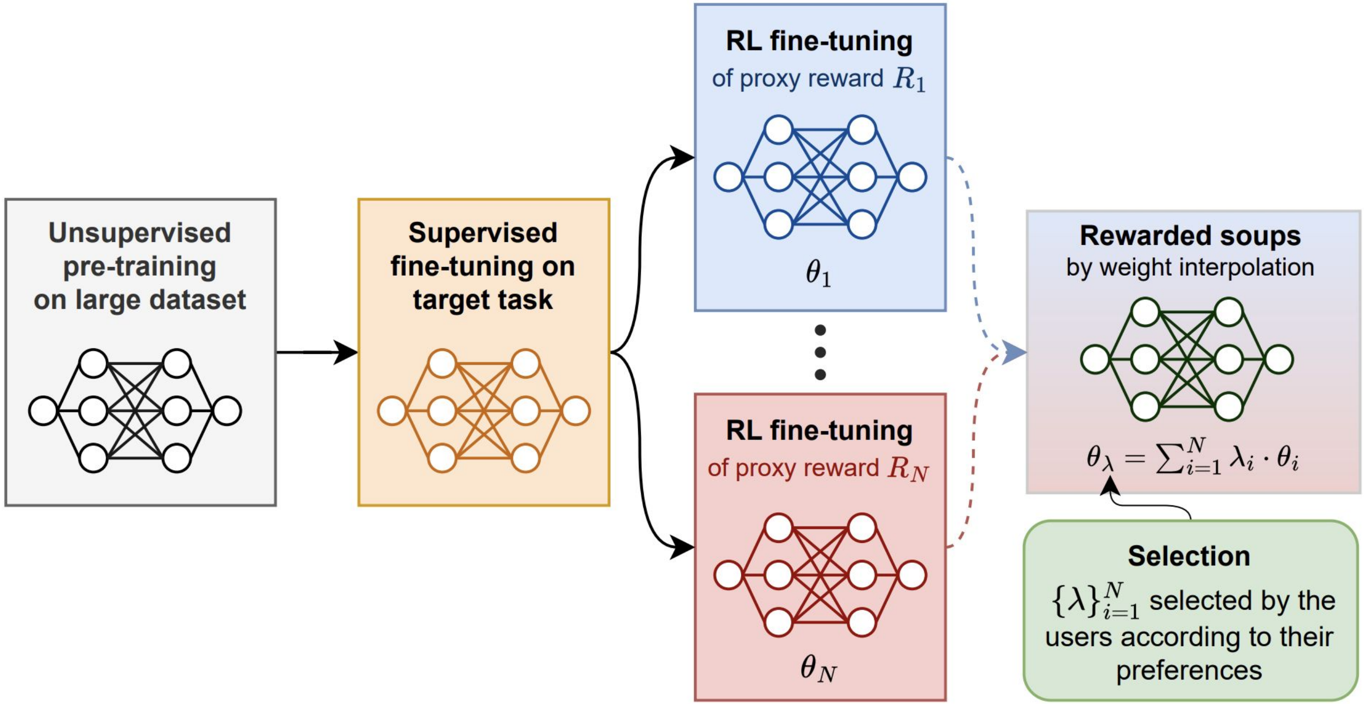
Pluralistic Alignment

Post-hoc multi-objective optimization for diverse stakeholders

Generalization and Robustness

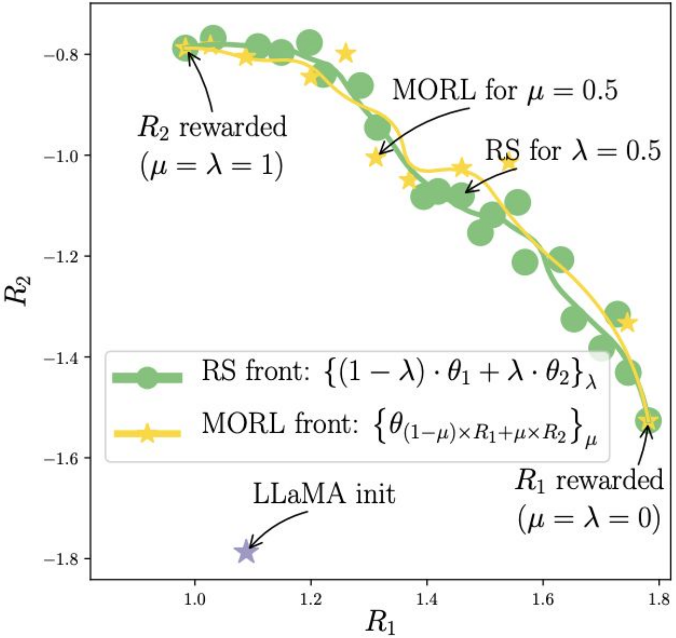
Ensuring robust, and compositionally generalizable behaviour in the real-world

Rewarded Soups uses weight interpolation to resolve diverse and conflicting rewards, yielding Pareto-optimal alignment



From "Rewarded soups: towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards" by Rame, Alexandre, et al.

Rewarded Soups can be a cost-efficient alternative of Multi-Objective Reinforcement Learning (MORL)



LLaMA RLHF for News Summarization

* RS matches the costly yellow front of multi-objective (MORL)

Feature	MORL	RS
Timing of Combination	**A Priori:** Objectives are combined before training	**A Posteriori:** Policies (weights) are combined after training
What is Interpolated	**Rewards**: Linearly interpolates the proxy rewards to create a single objective: $\boldsymbol{\Sigma\mu_i^j R_i}$	**Weights**: Linearly interpolates the expert network weights: $\boldsymbol{\Sigma\lambda_i\theta_i}$
Efficiency/Cost	Requires $\boldsymbol{M \gg N}$ trainings (where M is the number of preference weightings) to cover the Pareto front	Requires only $\boldsymbol{N}$ trainings (one for each proxy reward)
Scalability	**Unscalable** in deep learning due to high computational, memory, and engineering costs.	**Highly scalable and efficient**.

From "Rewarded soups: towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards" by Rame, Alexandre, et al.

Model Merging Helps Overcome Core Challenges across the Model development Lifecycle.

Combining Model Capabilities

Multiple tasks, languages, and modalities to integrate

Decentralized Training

Distributed compute and human resources across organizations

Continual Model Development

Continuous training and evolution of models

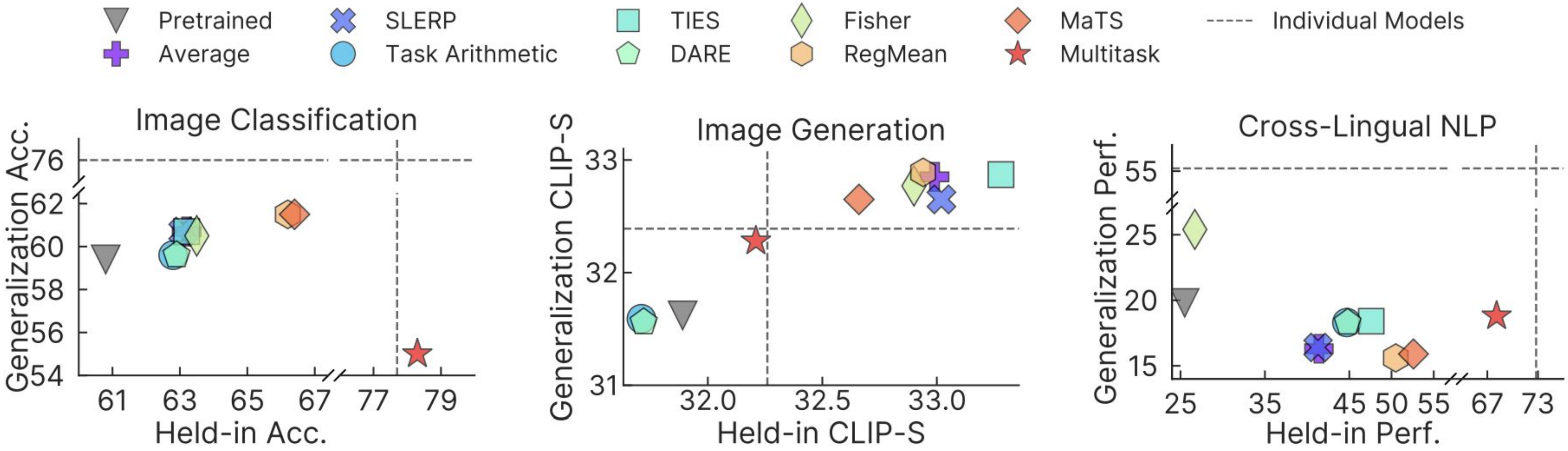
Pluralistic Alignment

Post-hoc multi-objective optimization for diverse stakeholders

Generalization and Robustness

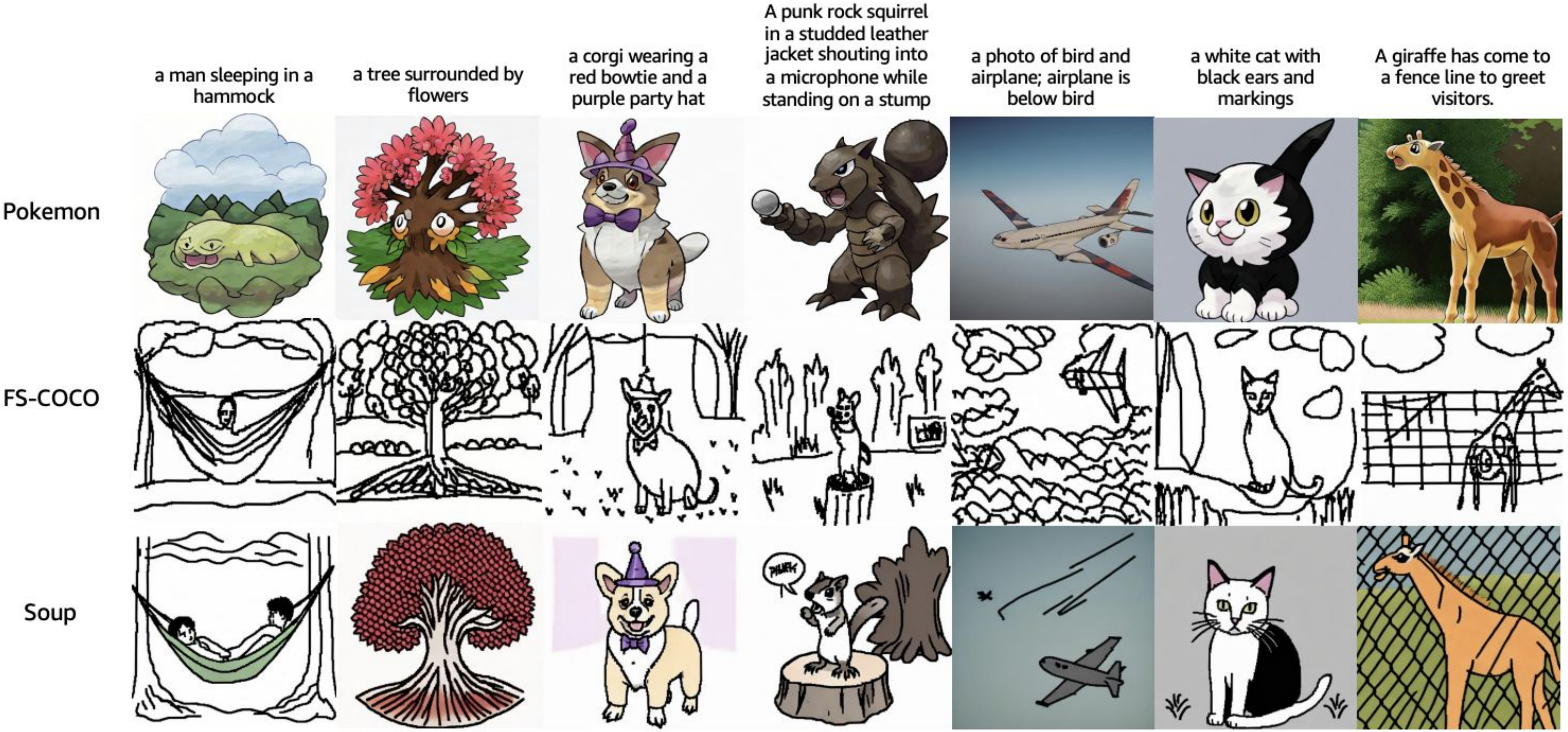
Ensuring robust, and compositionally generalizable behaviour in the real-world

Model merging can help models generalize compositionally, with effectiveness varying by domain



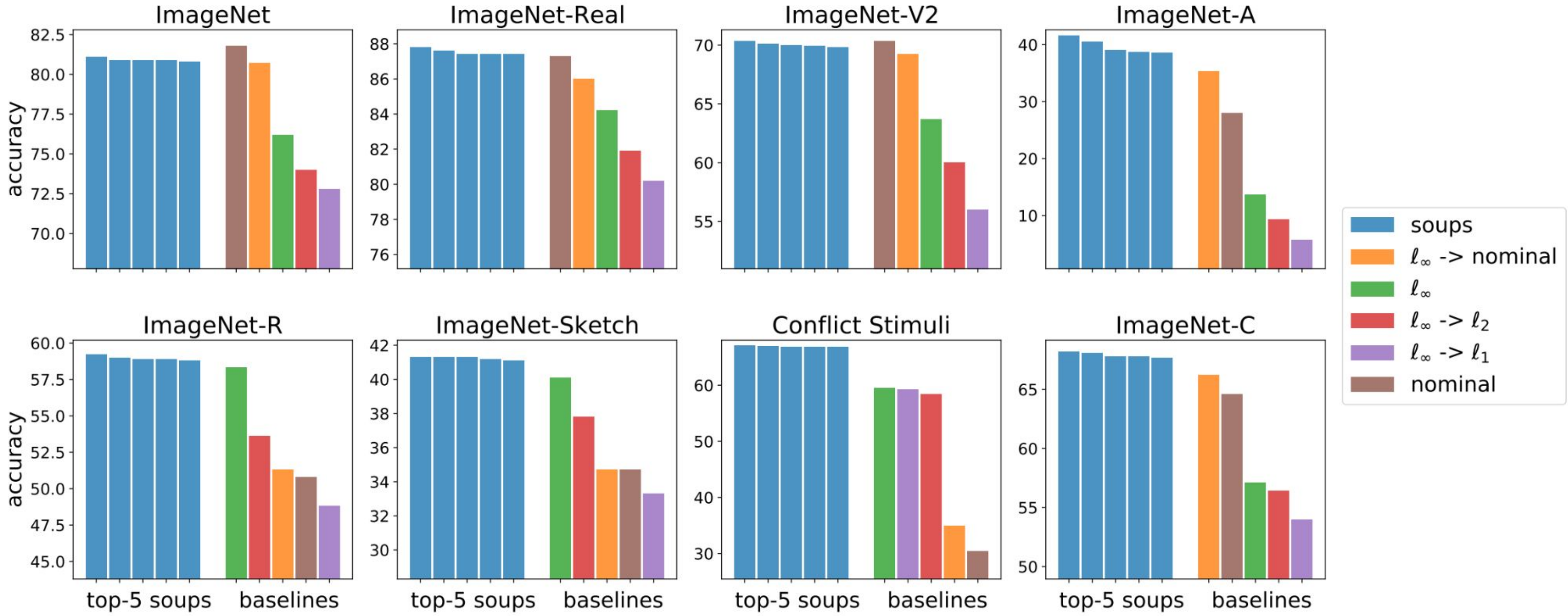
From "Realistic evaluation of model merging for compositional generalization" by Tam, Derek, et al.

Model merging can blend disparate artistic styles into novel hybrid outputs



From "Diffusion Soup: Model Merging for Text-to-Image Diffusion Models" by Biggs, Benjamin, et al.

Model merging unifies robustness to diverse adversarial attacks in one model



From "Seasoning Model Soups for Robustness to Adversarial and Natural Distribution Shifts" by Croce, Francesco, et al.

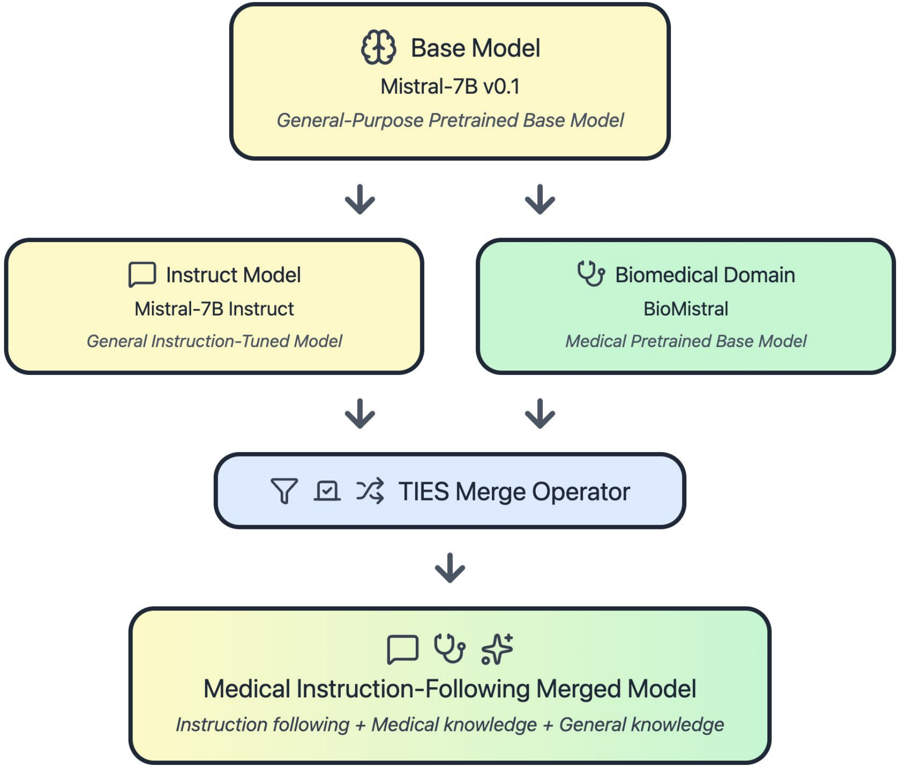
Merging Toolkits in Practice

MergeKit: A user-friendly toolkit for merging large language models, runnable on CPU or accelerated with as little as 8 GB of VRAM

Example YAML file for TIES-Merging:

```yaml
models:
  - model: mistralai/Mistral-7B-v0.1
    #no parameters necessary for base model
  - model: mistralai/Mistral-7B-Instruct-v0.2
    parameters:
      density: 0.5
      weight: 0.5
  - model: BioMistral/BioMistral-7B
    parameters:
      density: 0.5
      weight: 0.5

merge_method: ties
base_model: mistralai/Mistral-7B-v0.1
parameters:
  normalize: false
  int8_mask: true
dtype: float16
```



https://github.com/arcee-ai/mergekit

From "Arcee's Mergekit: A Toolkit for Merging Large Language Models" by Goddard, Charles, et al.

Hugging Face’s Parameter-Efficient Fine-Tuning (PEFT) Adapter Merging

```
pipe.unet.add_weighted_adapter(
    ["teapot","watercolour"],
    [1.0, 1.0],
    "merge",
    combination_type="ties_svd",
    density=0.5
)
```

https://huggingface.co/blog/peft_merging

Git-Theta: A Git Extension for Collaborative Development of Machine Learning Models

θ Git-Theta

A Git extension for collaborative, continual, and communal development of machine learning models.

Merging updates

Git-Theta can also handle merging of models trained with differing updates. For example, if an existing model is further trained on a new branch called `alternate-training`:

```
git checkout -b alternate-training
# After performing training...
git add model.pt
git commit
```

and is separately trained on the main branch:

```
git checkout main
# After some other training...
git add model.pt
git commit
```

We then can then merge the updates from the `alternate-training` branch via a standard `git merge`:

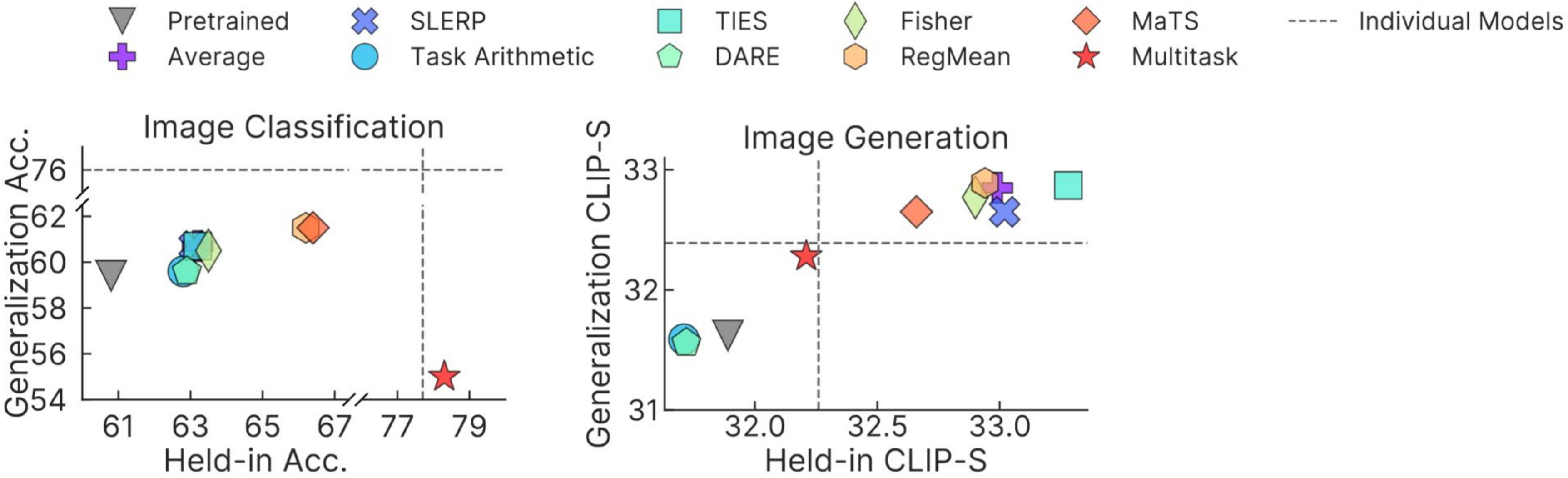
```
git merge alternate-training
```

Git-Theta supports various methods for automatically merging models, including parameter averaging. The merge tools shows us each parameter that is different between the two models and asks what merge operation to perform.

From "Git-Theta: A Git Extension for Collaborative Development of Machine Learning Models" by Kandpal, Nikhil, et al.

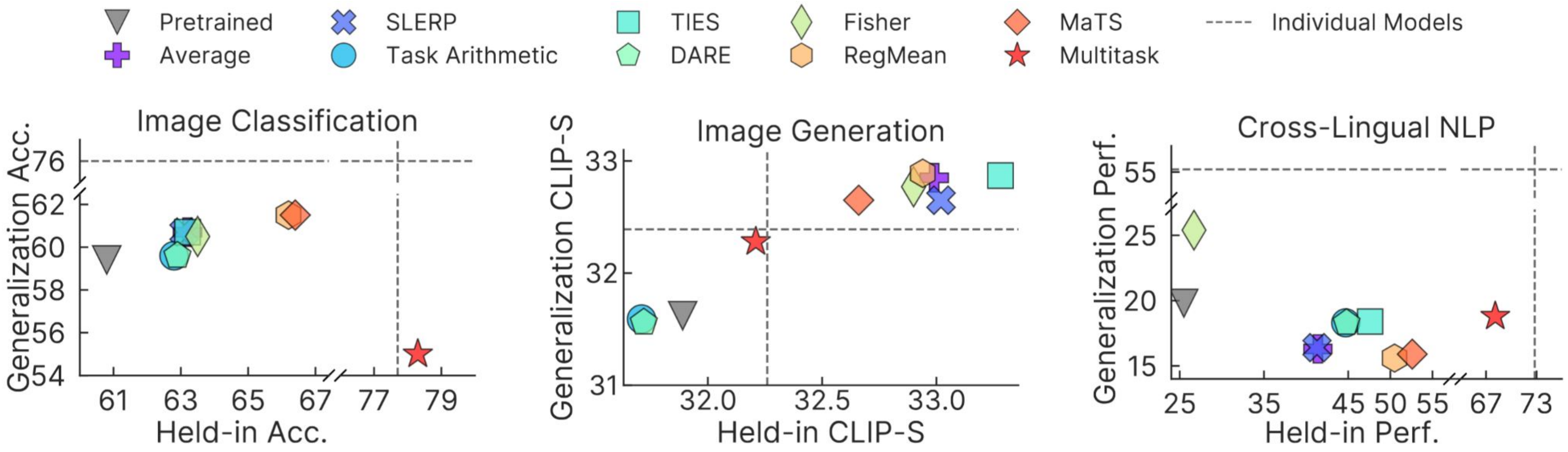
Open problems

Merging can generalize better than multitask learning…



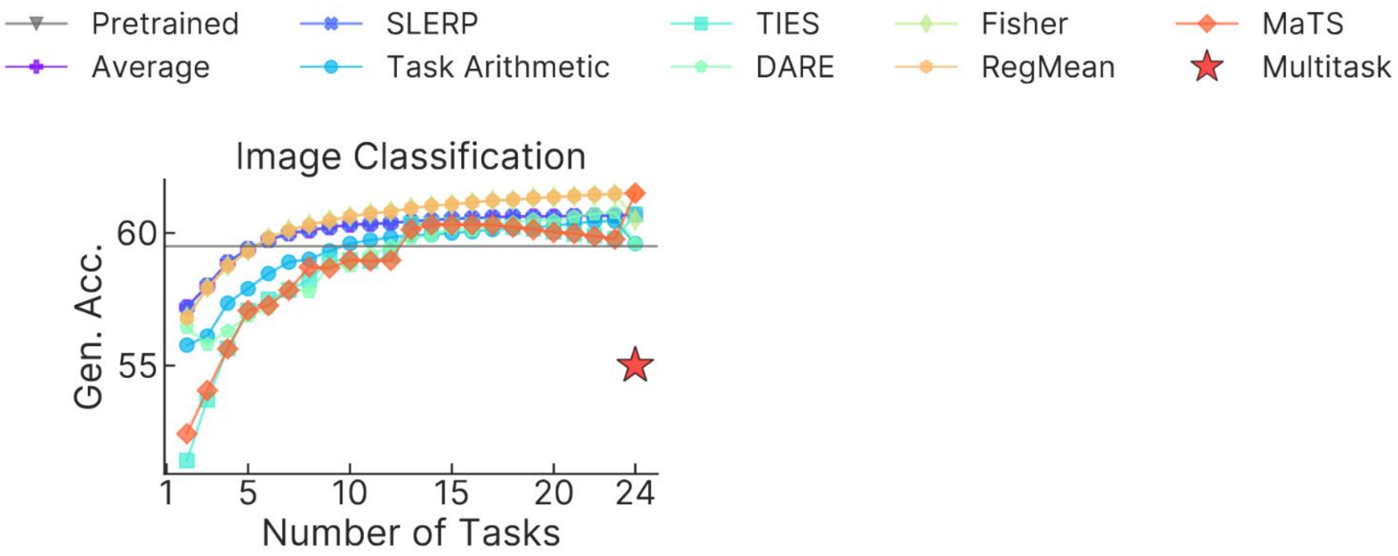
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

... *but it doesn't always*



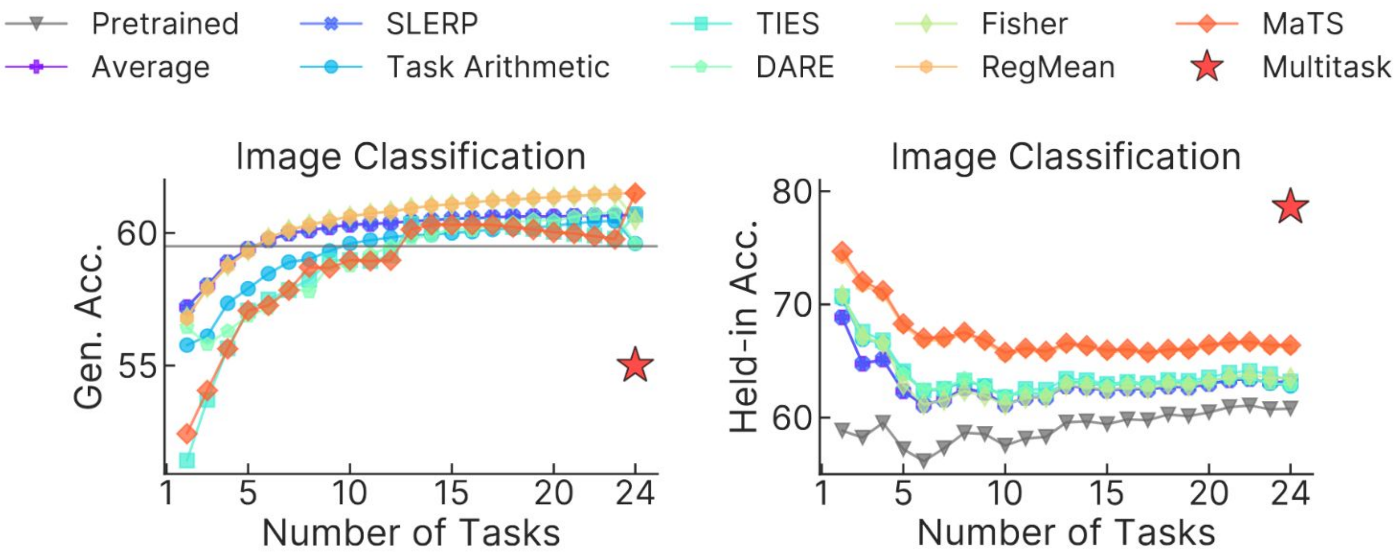
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

Merging can generalize better than multitask learning…



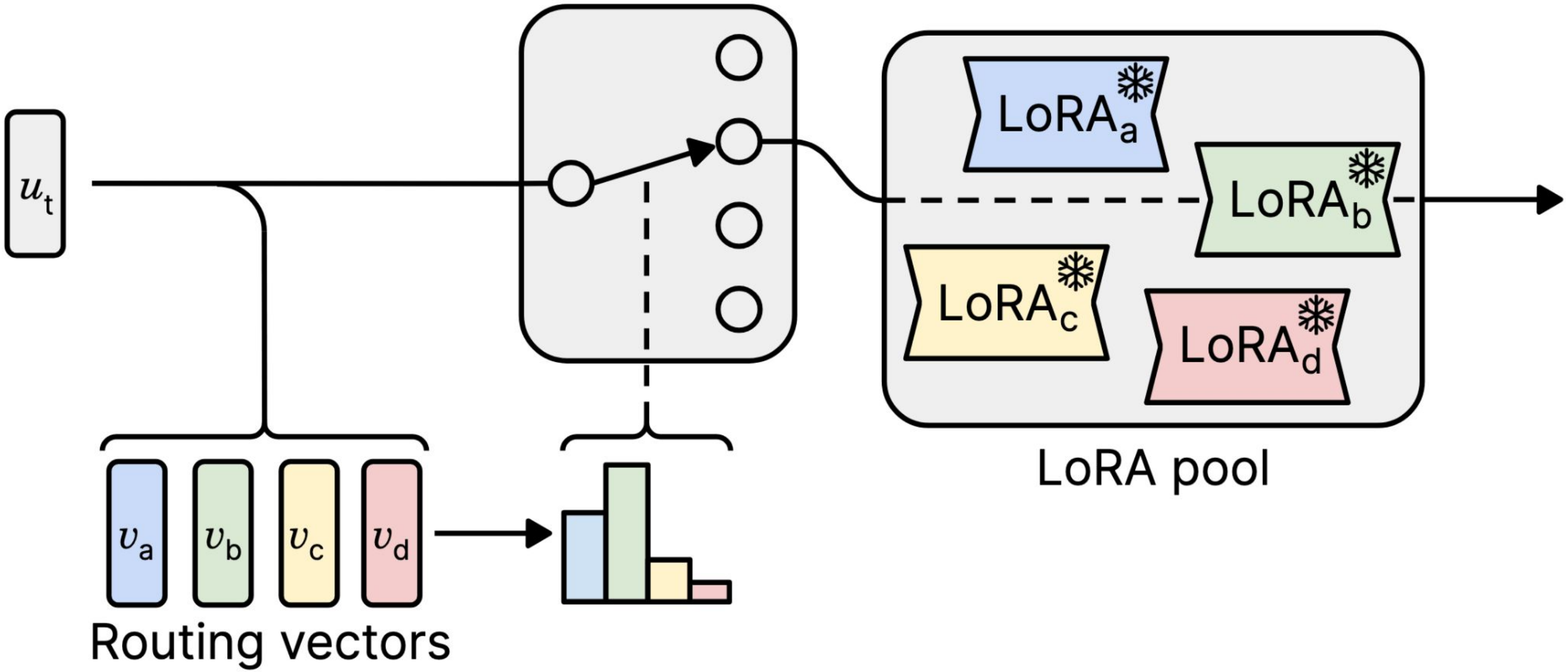
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

... *but the gap on "held-in" tasks increases as the number of models increases*



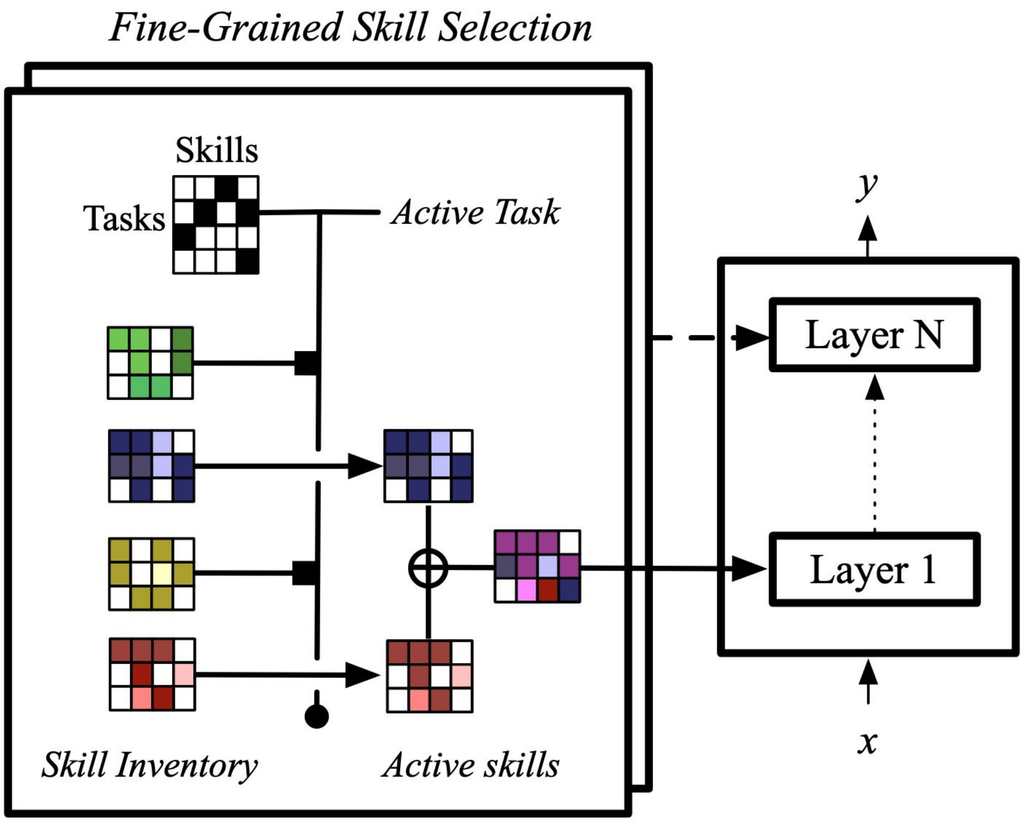
From "Realistic Evaluation of Model Merging for Compositional Generalization" by Tam et al.

MoErging as an alternative



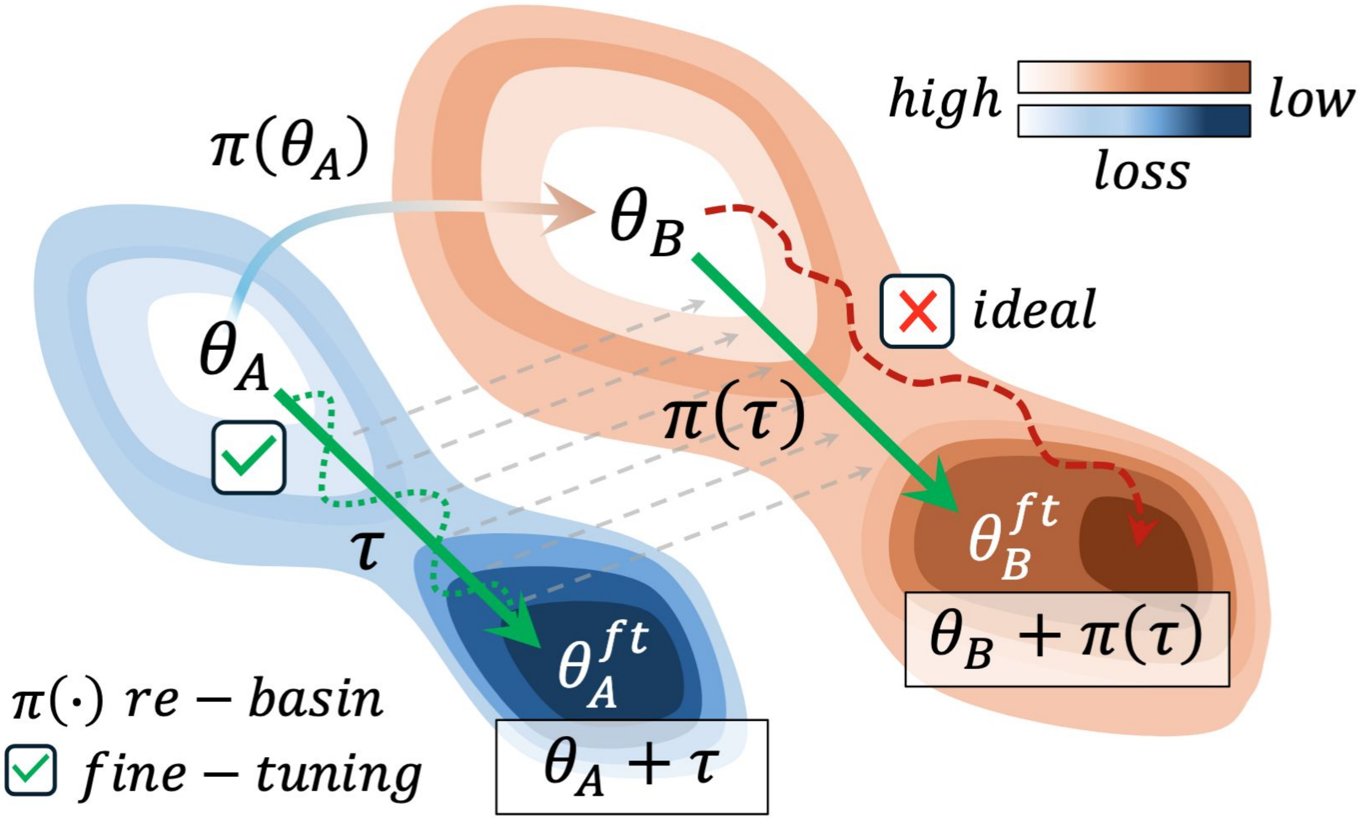
From “Learning to Route Among Specialized Experts for Zero-Shot Generalization" by Muqeeth et al.

Alternative architectures for better modularity



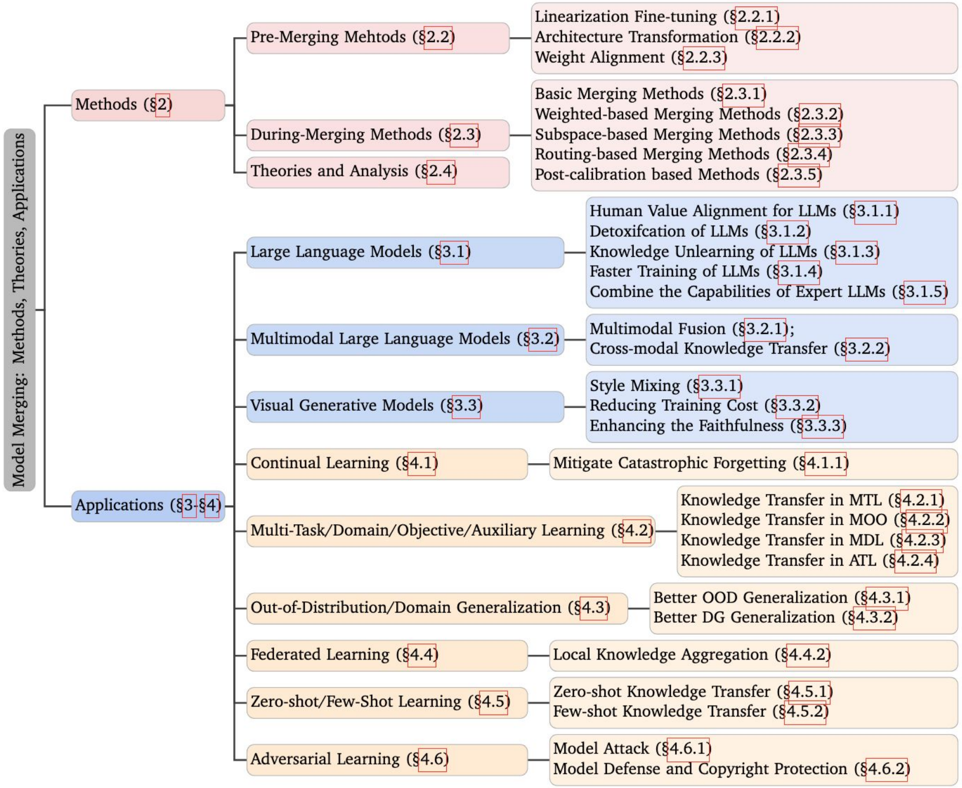
From "Combining Modular Skills in Multitask Learning" by Ponti et al.

Transferring task vectors across models



From "Update Your Transformer to the Latest Release: Re-Basin of Task Vectors" by Rinaldi et al.

More references!



https://github.com/EnnengYang/Awesome-Model-Merging-Methods-Theories-Applications

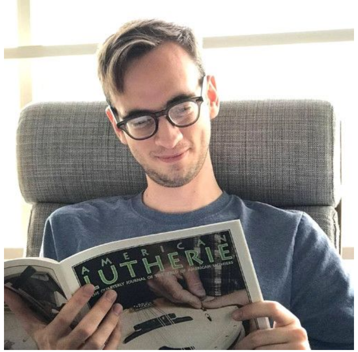
From "Model Merging in LLMs, MLLMs, and Beyond: Methods, Theories, Applications and Opportunities" by Yang et al.

Our panelists!

Margaret Li
Ph.D student
University of Washington

Edoardo Ponti
Assistant Professor
University of Edinburgh

Alexandra Chronopoulou
Research Scientist
Google DeepMind



Charles Goddard
Chief of Frontier Research
Arcee.ai

Sara Hooker
Co-founder
Adaptation