JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

TIES-MERGING: Resolving Interference When Merging Models

Tianwei Zhao, Ziyao Mou

Why Model Merging

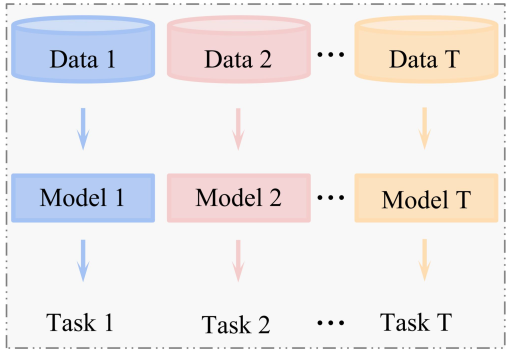
Most popular pre-trained models were created by large corporations, without involvement from most of the research community.

Even worse, most pre-trained models are never updated, it's hard for them to learn new information to improve their performance.

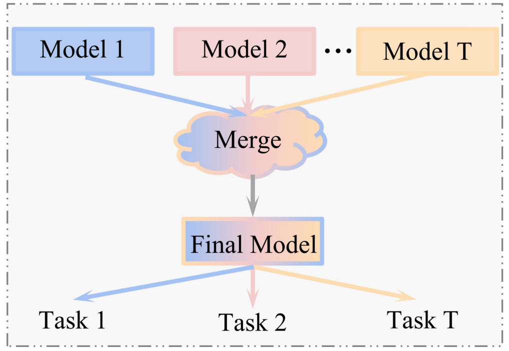
Build pre-trained models in the same way that we build open-source software offers a solution to these challenges.

Models could be developed by a large community who continually update them.

Models could be be modularized, and easy to reuse and improve.



Separate Training



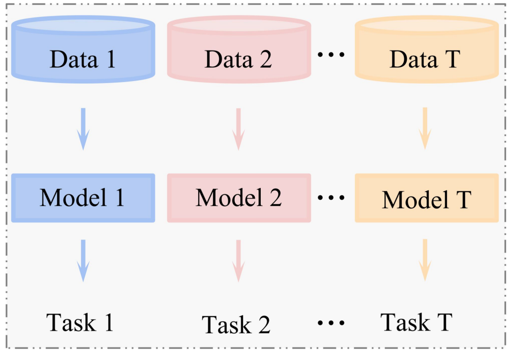
Model Merging

Why Model Merging

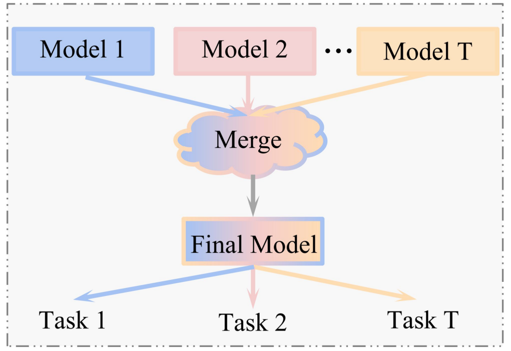
In distributed open-source software development, “merge conflicts” occur when contributors introduce conflicting changes to the same part of the codebase.

Model merging has the similar problem.

Therefore, research in how to merge model better become valuable.



Separate Training



Model Merging

Prior Work: Basic Methods

Model Soups (Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time)

Directly average the parameters of multiple models.

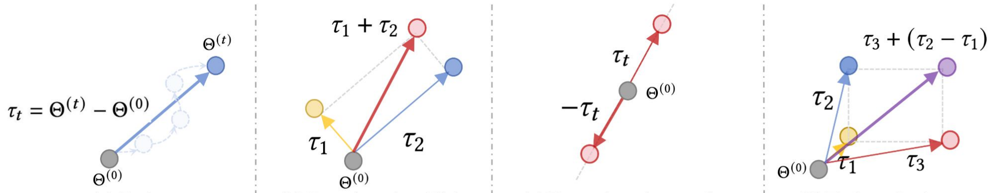
Wortsman, Mitchell, et al. "Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time."

Task Arithmetic (Editing models with task arithmetic)

Introduced the concept of Task Vector, which represents the model parameter Θ(t) fine-tuned on task t subtract the pre-trained model parameter Θ(0).

Multitask learning can be accomplished by adding task vectors, forgetting can be achieved by subtracting task vectors, and task analogies can be performed using analogous task vectors.

Ilharco, Gabriel, et al. "Editing models with task arithmetic."



(a) Task vector (b) Learning via addition (c) Forgetting via negation (d) Task analogies

Prior Work: Weighted-based Methods

Fisher Merging (Merging Models with Fisher-Weighted Averaging)

The method performs model merging based on the parameters importance in each model.

A diagonal approximation of the Fisher Information Matrix: $\hat{F}_\theta = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{y \sim p_\theta(y|x_i)} (\nabla_\theta \log p_\theta(y|x_i))^2$

is used to measure the importance of each parameter for the task.

Michael, Raffel. "Merging models with fisher-weighted averaging."

Regression Mean (Dataless Knowledge Fusion by Merging Weights of Language Models)

Describe model merging as a linear regression optimization $\min_W \quad \|W^T X_1 - W_1^T X_1\|^2 + \|W^T X_2 - W_2^T X_2\|^2$

which has a closed form solution: $W_M = (X_1^T X_1 + X_2^T X_2)^{-1}(X_1^T X_1 W_1 + X_2^T X_2 W_2)$

Jin, Xisen, et al. "Dataless knowledge fusion by merging weights of language models."

Challenges in Prior Merging Methods

While above model parameters averaging methods has proven effective for merging.

All of these methods ignore that values may interfere across models, thereby harming the performance of the merged model.

The paper demonstrates that interference can stem from two major causes:

1. Interference from redundant parameter.
2. Interference from sign disagreement.

Redundant Parameter

During fine-tuning, many model parameters can change but have a small impact on performance, we call this kind of parameters "redundant".

The influential value may be obscured by the redundant values, lowering the overall model performance.

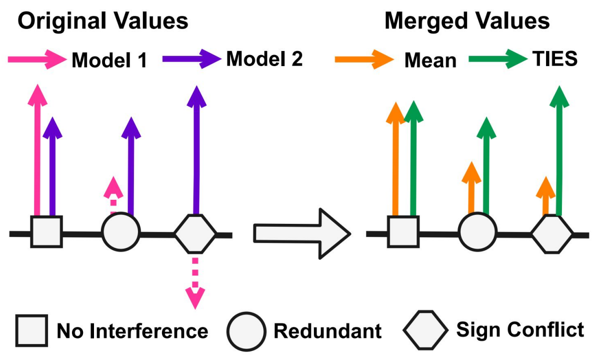


Figure 2: Different types of conflict and merged outputs produced by either averaging or TIES-MERGING. The parameters causing interference are denoted by dotted arrows.

The parameters causing interference are denoted by dotted arrows.

Johns Hopkins Whiting School of Engineering

Redundant Parameter

The experiment shows that removing redundant parameters does not affect the performance of the task.

In the experiment, for each time, keep the top-k% largest-magnitude parameters and resetting the rest to their initial value.

Keeping only the top-20% parameters delivers comparable results to keeping all parameters.

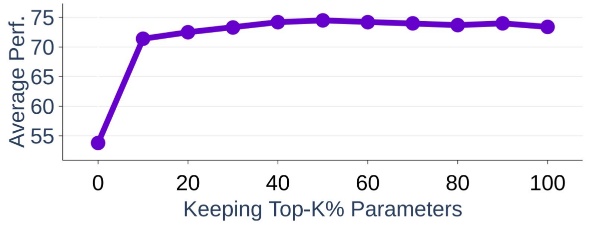


Figure 3: **Performance depends on a small fraction of high-magnitude parameters.** For each task vector, we keep only the largest - top-$k\%$ parameters and plot the average performance across eleven tasks. Keeping only the top-20% of the parameter does not degrade the performance.

perform experiment on the eleven $(IA)^3$ models

Sign Disagreement

A given parameter might have a positive value for some models and a negative value for others.

Employing simple average might compromise the performance on both tasks.

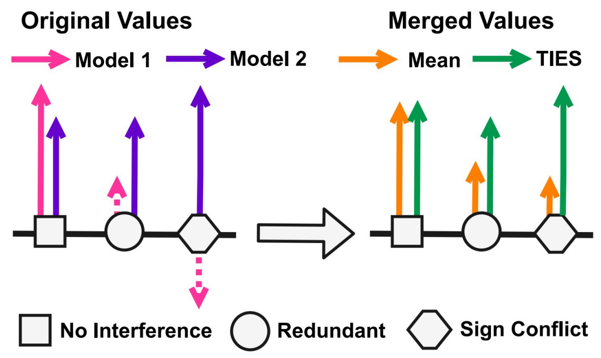


Figure 2: Different types of conflict and merged outputs produced by either averaging or TIES-MERGING. The parameters causing interference are denoted by dotted arrows.

Sign Disagreement

The experiment shows that the likelihood of sign disagreement increases with the number of models being merged.

Sign disagreement occur even when merging only 2 models from different tasks.

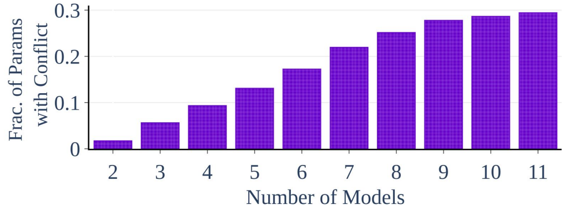
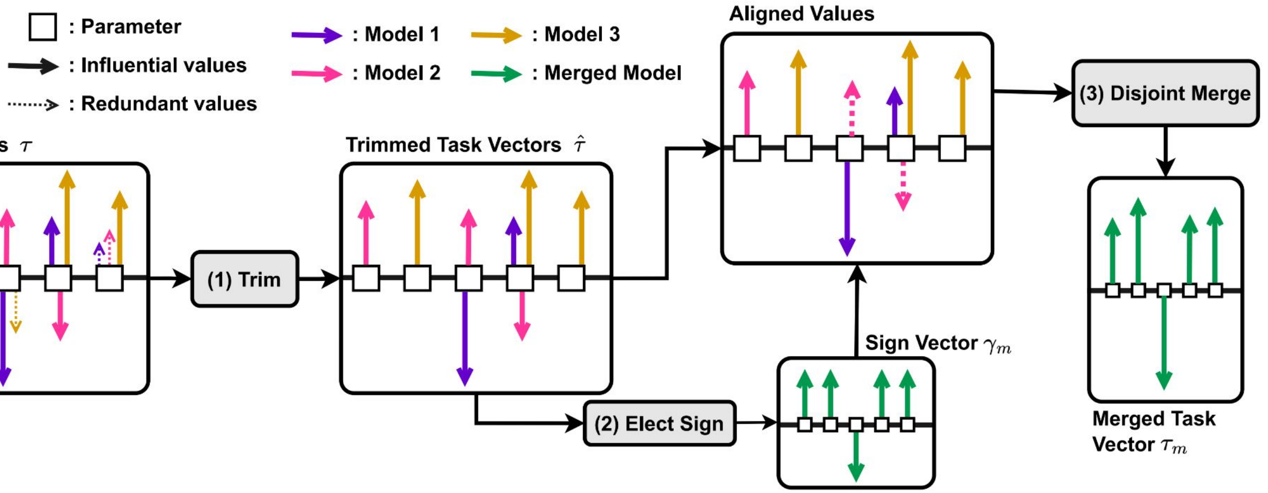


Figure 4: **Sign conflicts occur even after trimming and increase with the number of models.** We plot the fraction of parameters that have a sign conflict after trimming versus the number of models being merged.

Use T0-3B as the base model and finetune models with $(IA)^3$ methods on different datasets.

TIES-MERGING

To address the interference issue, paper propose TRIM, ELECT SIGN & MERGE (TIES-MERGING) method.



Preliminaries

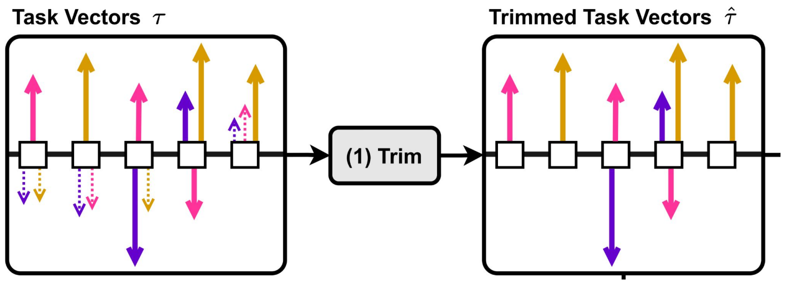
- For a task t, the task-vector $\tau_t \in \mathbb{R}^d$ is defined as $\tau_t = \theta^t_{\text{ft}} - \theta^t_{\text{init}}$.
- A task vector τ_t can be decomposed into a sign vector $\gamma_t \in \mathbb{R}^d$ and a magnitude vector $\mu_t \in \mathbb{R}^d$.
- The decomposition formula is $\tau_t = \gamma_t \odot \mu_t$, where $\odot$ represents the elementwise product.

Trim

To resolve redundant parameters, for each task, the method only keep the top-k% values according to their magnitude, and setting the value of rest parameters in task vector as 0.

Before Trim: Task_Vector = [1, -2, 2, 1, 3]

Only keep top-20% Value: Task_Vector' = [0, 0, 0, 0, 3]



Algorithm 1 TIES-MERGING Procedure.

```
Input: Fine-tuned models $\{\theta_t\}_{t=1}^n$, Initialization $\theta_{\text{init}}$,
       $k$, and $\lambda$.
Output: Merged Model $\theta_m$
forall $t$ in $1, ..., n$ do
    ▷ Create task vectors.
    $\tau_t = \theta_t - \theta_{\text{init}}$
    ▷ Step 1: Trim redundant parameters.
    $\hat{\tau}_t \leftarrow \text{keep\_topk\_reset\_rest\_to\_zero}(\tau_t, k)$
    $\hat{\gamma}_t \leftarrow sgn(\hat{\tau}_t)$
    $\hat{\mu}_t \leftarrow |\hat{\tau}_t|$
end
▷ Step 2: Elect Final Signs.
$\gamma_m = sgn(\sum_{t=1}^n \hat{\tau}_t)$
▷ Step 3: Disjoint Merge.
forall $p$ in $1, ..., d$ do
    $\mathcal{A}^p = \{t \in [n] \mid \hat{\gamma}_t^p = \gamma_m^p\}$
    $\tau_m^p = \frac{1}{|\mathcal{A}^p|} \sum_{t \in \mathcal{A}^p} \hat{\tau}_t^p$
end
▷ Obtain merged checkpoint
$\theta_m \leftarrow \theta_{\text{init}} + \lambda * \tau_m$
return $\theta_m$
```

JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

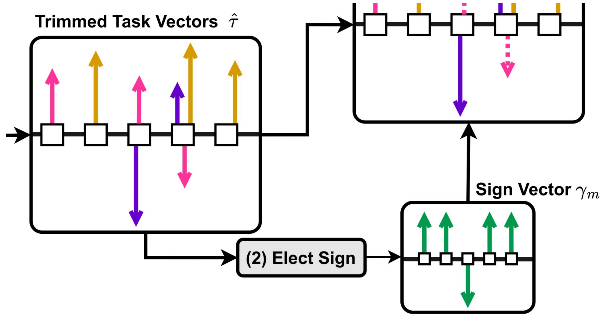
Elect

To resolve the sign disagreements, the method creates an aggregate elected sign vector.

Task_Vector_1 = [1, -2, 2, 1, 3]

Task_Vector_2 = [-2, 0, 1, 0, -1]

Aggregate_Elected_Sign_Vector = [-1, -1, +1, +1, +1]



Algorithm 1 TIES-MERGING Procedure.

```
Input: Fine-tuned models $\{\theta_t\}_{t=1}^n$, Initialization $\theta_{\text{init}}$,
       $k$, and $\lambda$.
Output: Merged Model $\theta_m$
forall $t$ in 1, ..., $n$ do
    ▷ Create task vectors.
    $\tau_t = \theta_t - \theta_{\text{init}}$
    ▷ Step 1: Trim redundant parameters.
    $\hat{\tau}_t \leftarrow$ keep_topk_reset_rest_to_zero$(\tau_t, k)$
    $\hat{\gamma}_t \leftarrow sgn(\hat{\tau}_t)$
    $\hat{\mu}_t \leftarrow |\hat{\tau}_t|$
end
▷ Step 2: Elect Final Signs.
$\gamma_m = sgn(\sum_{t=1}^n \hat{\tau}_t)$
▷ Step 3: Disjoint Merge.
forall $p$ in 1, ..., $d$ do
    $\mathcal{A}^p = \{t \in [n] \mid \hat{\gamma}_t^p = \gamma_m^p\}$
    $\tau_m^p = \frac{1}{|\mathcal{A}^p|} \sum_{t \in \mathcal{A}^p} \hat{\tau}_t^p$
end
▷ Obtain merged checkpoint
$\theta_m \leftarrow \theta_{\text{init}} + \lambda * \tau_m$
return $\theta_m$
```

Disjoint Merge

For each parameter p, the method computes a disjoint mean, which is obtained by only keeping the parameter values whose signs are the same as the aggregated elected sign and calculate their mean.

Task_Vector_1 = [1, -2, 2, 1, 3]

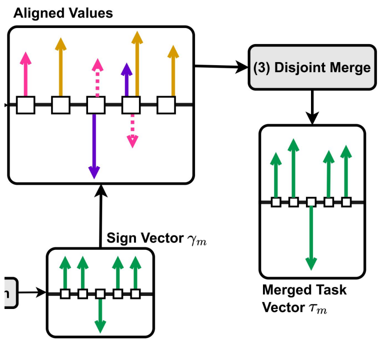
Task_Vector_2 = [-2, 0, 1, 0, -1]

Task_Vecotor_3 = [-1, 1, -2, 0, -1]

Aggregate_Elected_Sign_Vector =

[-1, -1, +1, +1, +1]

Disjoint_Mean = [-1.5, -2, 1.5, 1, 3]



Algorithm 1 TIES-MERGING Procedure.

```
Input: Fine-tuned models $\{\theta_t\}_{t=1}^n$, Initialization $\theta_{\text{init}}$,
       $k$, and $\lambda$.
Output: Merged Model $\theta_m$
forall $t$ in $1, ..., n$ do
    ▷ Create task vectors.
    $\tau_t = \theta_t - \theta_{\text{init}}$
    ▷ Step 1:  Trim redundant parameters.
    $\hat{\tau}_t \leftarrow \text{keep\_topk\_reset\_rest\_to\_zero}(\tau_t, k)$
    $\hat{\gamma}_t \leftarrow sgn(\hat{\tau}_t)$
    $\hat{\mu}_t \leftarrow |\hat{\tau}_t|$
end
▷ Step 2:  Elect Final Signs.
$\gamma_m = sgn(\sum_{t=1}^n \hat{\tau}_t)$
▷ Step 3:  Disjoint Merge.
forall $p$ in $1, ..., d$ do
    $\mathcal{A}^p = \{t \in [n] \mid \hat{\gamma}_t^p = \gamma_m^p\}$
    $\tau_m^p = \frac{1}{|\mathcal{A}^p|} \sum_{t \in \mathcal{A}^p} \hat{\tau}_t^p$
end
▷ Obtain merged checkpoint
$\theta_m \leftarrow \theta_{\text{init}} + \lambda * \tau_m$
return $\theta_m$
```

Experiment Setup: Baseline

Four baseline merging methods:

- Simple Averaging
- Fisher Merging
- RegMean
- Task Arithmetic

Experiment Setup: With Validation Set

Our general methods:

- Perform full fine-tuning on both vision models (ViT) and language models (T5)
- Perform PEFT fine-tuning $(IA)^3$ on T0-3B
- Finetune models on the train split of datasets and merged datasets
- Keep the top-20% parameters in the task vector resetting the rest to 0
    - For full fine-tuning, it is for all parameter in models
    - For PEFT such as $(IA)^3$, it is for all scaling factors
- Merge them, using each of the four baseline and our own and try to tune λ.

Experiment Setup: Without Validation Set

How about saving all training data for each task that are the same size as the original model?(Reg Mean proposed)

Storage and transmission cost a lot!!!

Our general methods:

- apply it to full fine-tuning on ViT (vision) and T5 (language) models
- keep the top-20% parameters in the task vector resetting the rest to 0
- sets $\lambda = 1$.

Main Results: Merging Models

λ=1

Merging PEFT Models:

Base model: T0-3B

Finetune method: $(\mathrm{IA})^3$

Task Datasets: 11 datasets

Merging Fully Fine-tuned NLP Models:

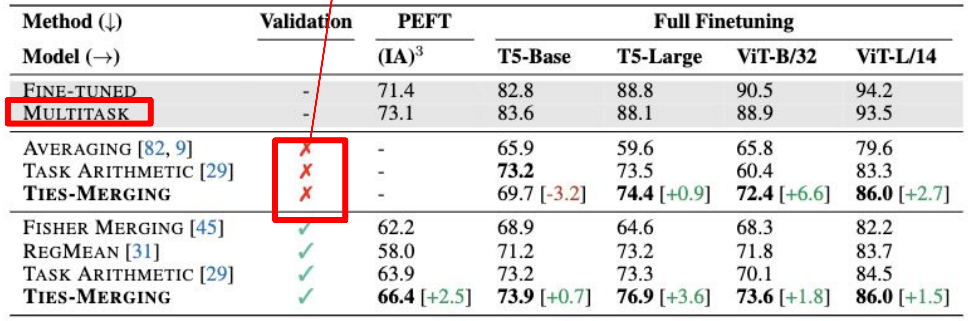
Base model: T5-base T5-large

Task Datasets: 7 datasets (question answering, Paraphrase Identification…)

Merging Fully Fine-tuned Vision Models:

Base model: ViT-B/32 ViT-L/14(CLIP)

Task Datasets: 8 datasets (remote sensing, traffic classification, and satellite imagery recognition)



Method (↓)	Validation	PEFT	Full Finetuning			
Model (→)		$(\mathrm{IA})^3$	T5-Base	T5-Large	ViT-B/32	ViT-L/14
FINE-TUNED	-	71.4	82.8	88.8	90.5	94.2
MULTITASK	-	73.1	83.6	88.1	88.9	93.5
AVERAGING [82, 9]	✗	-	65.9	59.6	65.8	79.6
TASK ARITHMETIC [29]	✗	-	**73.2**	73.5	60.4	83.3
TIES-MERGING	✗	-	69.7 [-3.2]	**74.4** [+0.9]	**72.4** [+6.6]	**86.0** [+2.7]
FISHER MERGING [45]	✓	62.2	68.9	64.6	68.3	82.2
REGMEAN [31]	✓	58.0	71.2	73.2	71.8	83.7
TASK ARITHMETIC [29]	✓	63.9	73.2	73.3	70.1	84.5
TIES-MERGING	✓	**66.4** [+2.5]	**73.9** [+0.7]	**76.9** [+3.6]	**73.6** [+1.8]	**86.0** [+1.5]

Table 1: Comparing model merging methods across multiple fine-tuning settings and modalities (NLP and Vision) with and without the availability of a validation set.

Merging Different Number of Tasks

(1) As the number of merged tasks increases, the performance of all methods decreases.

(2) Task Arithmetic and TIES-MERGING, which begin to perform better than simple averaging

(3)Task Arithmetic's merging performance drops faster than TIES-MERGING

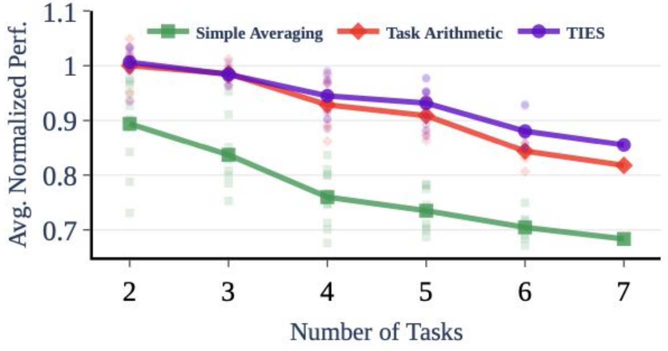
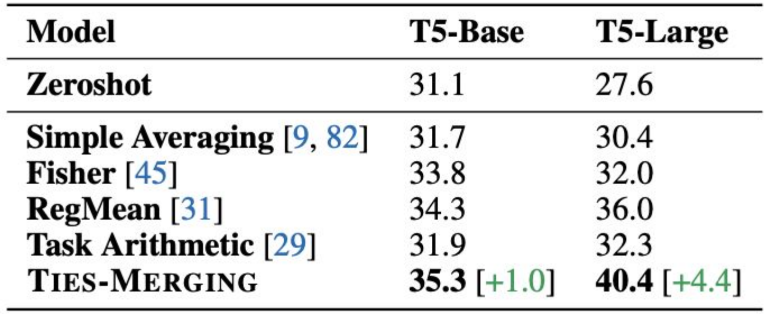


Figure 5: **TIES-MERGING scales better.** Average performance when merging a different number of tasks.

Out-of-Domain Generalization

Use the T5-base and T5-large models merged on the seven in-domain datasets from the previous experiments

Evaluate them on six held-out datasets from T0 mixture

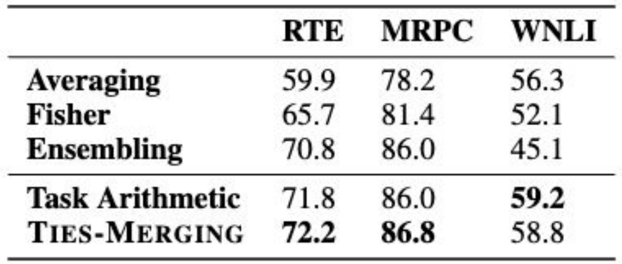


Model	T5-Base	T5-Large
Zeroshot	31.1	27.6
Simple Averaging [9, 82]	31.7	30.4
Fisher [45]	33.8	32.0
RegMean [31]	34.3	36.0
Task Arithmetic [29]	31.9	32.3
TIES-MERGING	**35.3** [+1.0]	**40.4** [+4.4]

Table 2: **TIES-MERGING generalizes better.** Out of Distribution Generalization for T5-Base and T5-Large on six held-out tasks.

Merging Models for better Robustness

Merge multiple checkpoints trained on the same task to see if it can improve robustness



	RTE	MRPC	WNLI
Averaging	59.9	78.2	56.3
Fisher	65.7	81.4	52.1
Ensembling	70.8	86.0	45.1
Task Arithmetic	71.8	86.0	**59.2**
TIES-MERGING	**72.2**	**86.8**	58.8

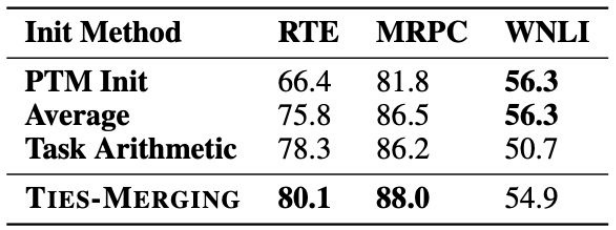
Table 3: **Model soups experimental setup. TIES improves performance when merging checkpoints on the same tasks.** For each task, we merge 10 checkpoints from Huggingface hub and evaluate on the one task they were trained on.

Merging Models for Better Initialization

Merge checkpoints from different tasks for a better initialization when fine-tuning on a downstream task

Fine-tuned bert-base-uncased checkpoints for 8 GLUE tasks (wnli, sst2, rte, qnli, mrpc, cola, mnli, qqp)

Merge 7 expert models' checkpoints



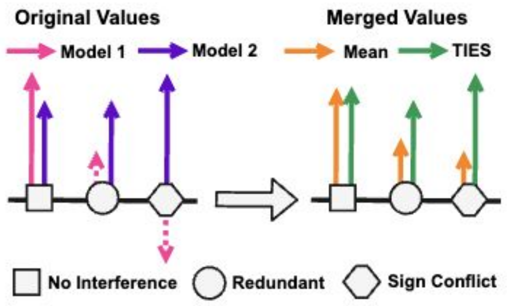
Init Method	RTE	MRPC	WNLI
PTM Init	66.4	81.8	**56.3**
Average	75.8	86.5	**56.3**
Task Arithmetic	78.3	86.2	50.7
TIES-MERGING	**80.1**	**88.0**	54.9

Table 4: **A TIES-merged model is a better initialization for finetuning.** For each task, we merge the checkpoints from the 7 other GLUE tasks and then finetune and evaluate on the selected task.

Additional Results

noise

Importance of Removing Redundant Parameters and Resolving Sign Interference



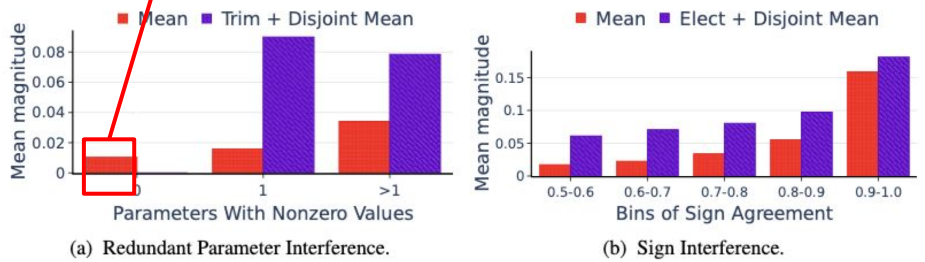


Figure 6: **Trimming Parameters and Electing Signs prevents interference.** Demonstration of parameter interference between different models and its impact on parameter values. The Standard Mean (red) shrinks magnitudes and does it more when there is less agreement on the sign (right) or the parameter is influential for multiple tasks (left).

Signs of the Top-k% Parameters

Use $(\mathrm{IA})^3$ to quantify,

Flip the direction of each of the top-k% parameters (by magnitude) with a probability p by multiplying the parameters with −1

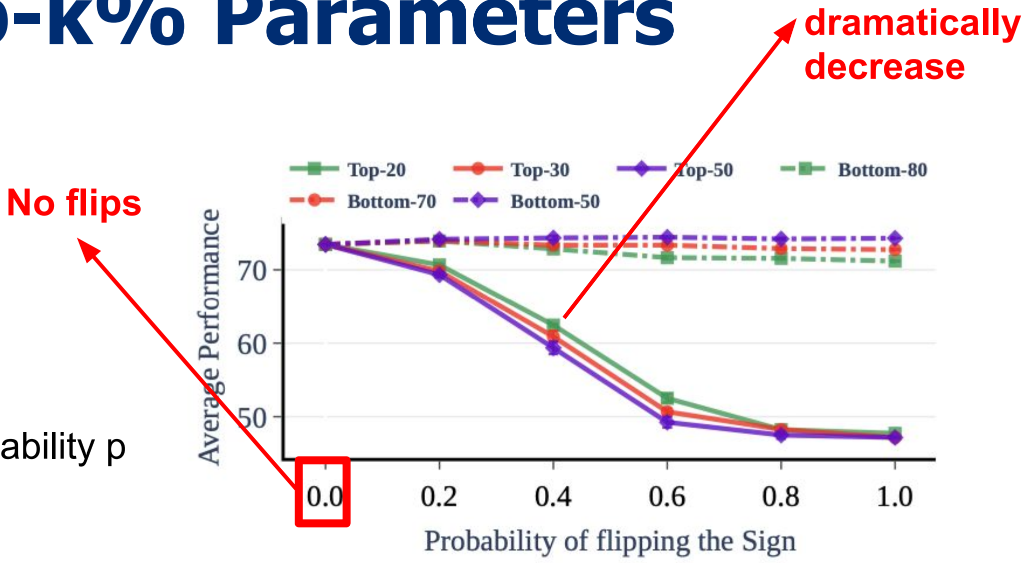


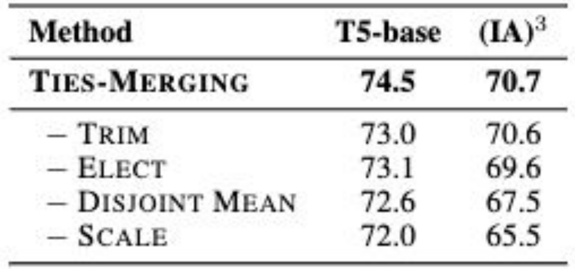
Figure 7: **Flipping the signs of high magnitude parameters leads to catastrophic performance drops.** Average Performance when flipping the directions of Top-k% and Bottom-k% parameters for each task. We report the results averaged over eleven $(\mathrm{IA})^3$ tasks.

Ablation of Components

Removing elect: taking the mean of values with signs +1 and −1 but not including the 0 values of the trimmed task vectors in the mean

Removing disjoint mean: taking the mean of the values with the elected signs and the 0 for the trimmed values.

Removing scaling: using λ = 1.



Method	T5-base	$(IA)^3$
TIES-MERGING	**74.5**	**70.7**
– TRIM	73.0	70.6
– ELECT	73.1	69.6
– DISJOINT MEAN	72.6	67.5
– SCALE	72.0	65.5

Table 6: Ablation on all the steps of TIES-MERGING.

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

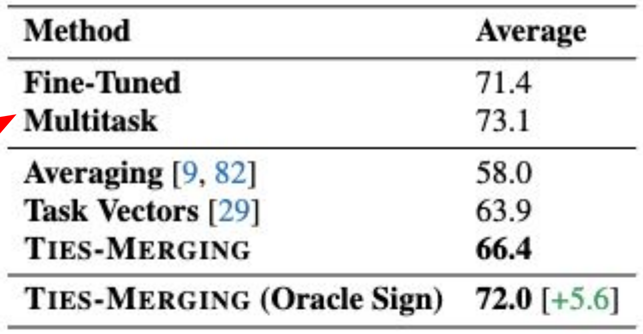
Estimating Correct Signs

Train a multitask $(\text{IA})^3$ model

Create the multitask vector τ_{mult}, the multitask sign vector γ_{mult}

Directly set $\gamma_m = \gamma_{\text{mult}}$

Closer



Method	Average
Fine-Tuned	71.4
Multitask	73.1
Averaging [9, 82]	58.0
Task Vectors [29]	63.9
TIES-MERGING	**66.4**
TIES-MERGING (Oracle Sign)	**72.0** [+5.6]

Table 5: **TIES-MERGING can perform close to multitask models if the signs can be estimated correctly.** We use the signs from the multitask vector as the elected sign and perform merging and report the performance.

Conclusion

(1) The paper Introduce TIES-MERGING considering redundant parameters and sign disagreement, which improves multi-task merging model performance

(2) Merging individual-task models to create a multitask still lags behind the simultaneous multitask training. The performance may still improve if weight disentanglement is considered.

(3) This method can only merge models with the same base model; merging models with different base models may be more challenging.

JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

THANK YOU ! ! !

Appendix

How to Rank Parameter Magnitude in Reality?

The gradient of the loss function L(W) with respect to the parameters W points in the direction of the greatest change in the loss function. It applied in sparse fine-tuning.

1. Accumulate the weighted gradient norm across steps

$$W^{(t+1)} = W^{(t)} - \eta \sum_{i=1}^{N} \nabla_W L_i(W)$$

2. Select the top k% of parameters with the greatest amount of variation. These parameters are the most important parameters in the task.

Appendix

Sample Combinations for Tasks: Optimized λ in T5 model

(2 tasks) TIES → [1.7, 1.9, 2, 2, 1.1, 1.5, 1.6, 1.8, 1.9, 1., 5]

(2 tasks) Task Arithmetic → [1, 0.9, 1, 1, 0.9, 1, 0.9, 0.9, 0.9, 1]

(3 tasks) TIES → [1.2, 2, 1.5, 1.9, 1.8, 1.7, 1.4, 2, 3, 1.9]

(3 tasks) Task Arithmetic → [1, 0.7, 0.7, 1, 1, 0.9, 0.7, 0.7, 0.9, 1]

(4 tasks) TIES → [1.5, 1.3, 1.3, 1.8, 2.3, 1.7, 1.8, 1.7, 1.9, 1.5]

(4 tasks) Task Arithmetic → [0.8, 0.7, 0.7, 0.7, 0.6, 0.7, 0.7, 0.8, 0.6, 0.7]

(5 tasks) TIES → [2, 2, 2, 1.8, 1.7, 2, 1.6, 2.1, 1.6, 1.3]

(5 tasks)Task Arithmetic → [0.7, 0.8, 0.6, 0.8, 0.7, 0.6, 0.6, 0.6, 0.6, 0.7]

(6 tasks) TIES → [1.6, 1.7, 1.7, 1.2, 1.7, 1.7, 1.5]

(6 tasks) Task Arithmetic → [0.6, 0.5, 0.5, 0.5, 0.7, 0.5, 0.6]

(7 tasks) TIES → [1.7]

(7 tasks) Task Arithmetic → [0.5]

$$sampleCount = Min(\binom{7}{T}, 10)$$