Model Merging with Adaptive Drop Rate based on Weight Importance

강채아

Contents

01 Introduction

Disadvantages of separate fine-tuned model

- Separate model stored and deployed
- Cannot improve in-domain performance or out-of-domain generalization

$\Rightarrow$ Model merging

Problems of merging models: Parameter interference

- Redundant parameters
- Sign conflict

02 Background

Given: set of tasks $\{t_1, \ldots, t_n\}$, pre-trained model

Task vector

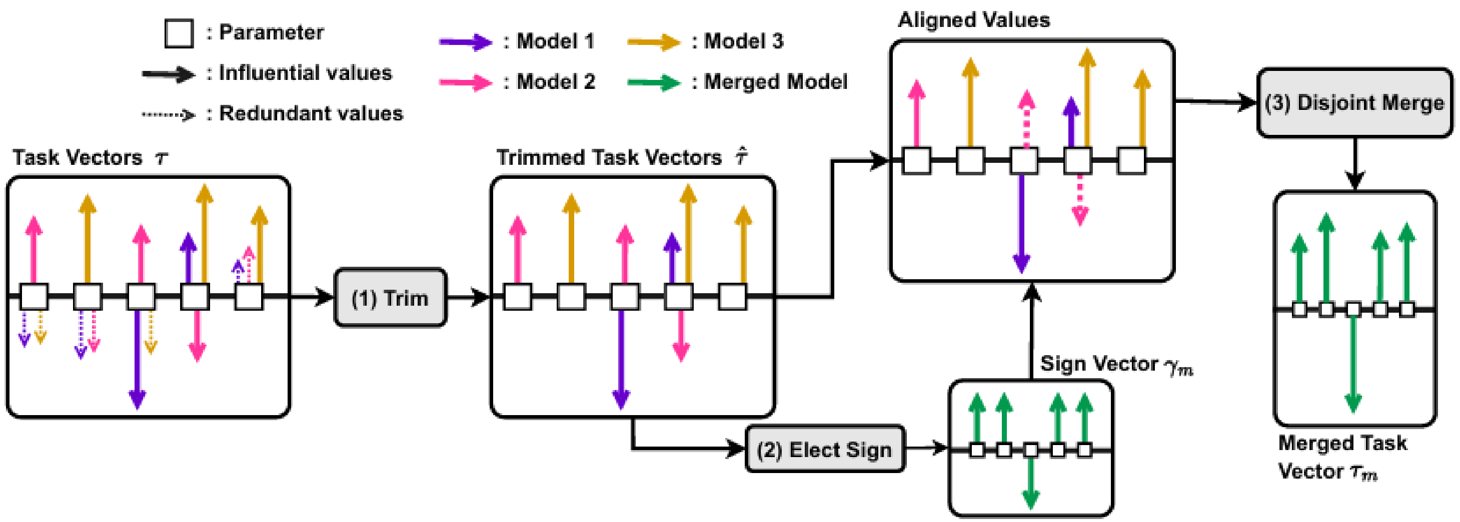
$$\tau_t = \theta_{ft}^t - \theta_{init}^t$$

$$(\tau_t \in \mathbb{R}^d, \;\; \theta : trainable\ parameters,$$

$$\theta_{init}^t : initialization, \;\; \theta_{ft}^t : finetuned\ parameters)$$

TIES-Merging

- Trim, Elect sign, Disjoint merge
- Solve parameter redundancy and sign disagreement
- Remove redundant parameters



02 Background

DARE

- Drop, Rescale
- Random Drop delta parameter with drop rate p
- Rescaling factor $1/(1-p)$
- Drop redundant parameters

$$\boldsymbol{m}^t \sim \text{Bernoulli}(p),$$
$$\widetilde{\boldsymbol{\delta}}^t = (\mathbf{1} - \boldsymbol{m}^t) \odot \boldsymbol{\delta}^t,$$
$$\hat{\boldsymbol{\delta}}^t = \widetilde{\boldsymbol{\delta}}^t / (1-p).$$

02 Background

DELLA-Merging

- Drop(MagPrune), Elect, Fuse
- Align delta parameter based on magnitude size
- Assign drop rate sequentially
- Remove insignificant parameters

$$\{r_1, r_2, \ldots, r_n\} = \text{rank}(\{\delta_1, \delta_2, \ldots \delta_n\})$$

$$\Delta_i = \frac{\epsilon}{n} * r_i$$

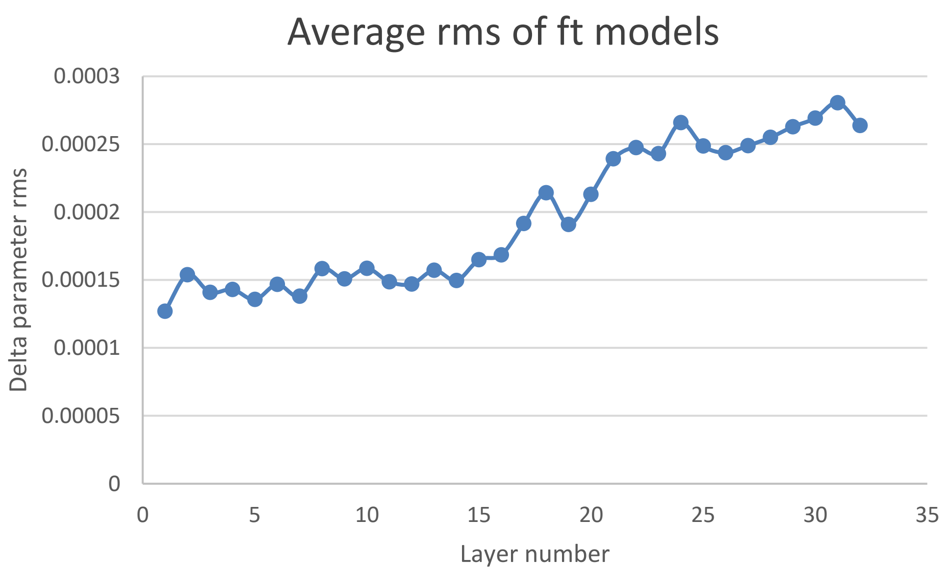
$$p_i = p_{min} + \Delta_i$$

02 Background

Problems of previous merging methods : Consider only parameter level, not layer level

Weight importance

- More delta parameter changes of later layer than early layer in ft models
- Different importance between layers should be considered



03 Adaptive Drop Rate Merging

Remove redundant parameters based on relative weight importance

SFT delta parameters

- $\theta_{PRE} \in \mathbb{R}^d$: parameters of pre-trained LM
- $\theta_{SFT}^t \in \mathbb{R}^d$: parameters of fine-tuned LM
- $\delta^t = \theta_{SFT}^t - \theta_{PRE} \in \mathbb{R}^d$: delta parameters

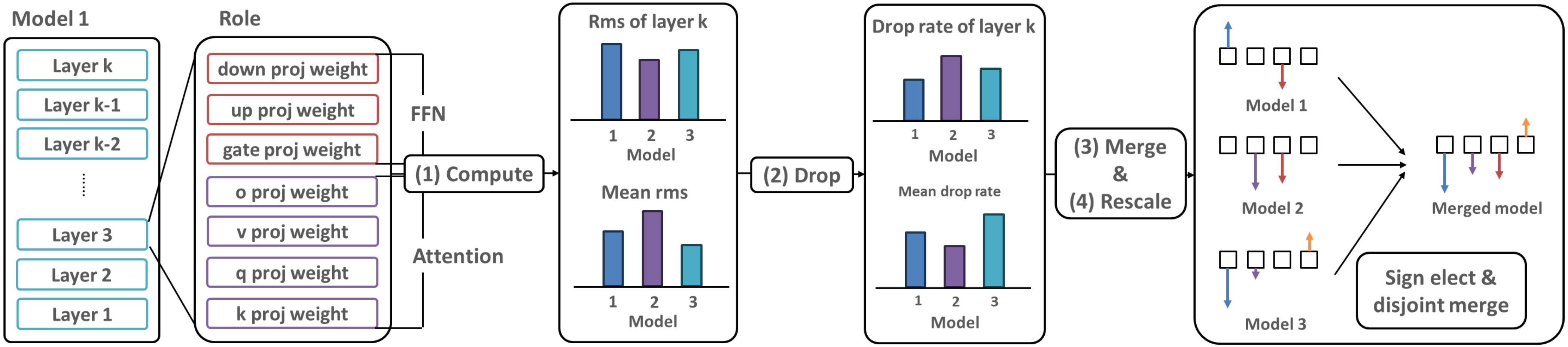
Merge fine-tuned models with same base model

03 Adaptive Drop Rate Merging

4 steps

Compute, Drop, Merge, Rescale

1. Compute role rms(root mean square) of each delta matrix
2. Random drop on δ_r^t based on role drop rate p_r^t
3. Merge each delta matrix with elect sign & disjoint merge
4. Rescale delta matrix by using smallest drop rate

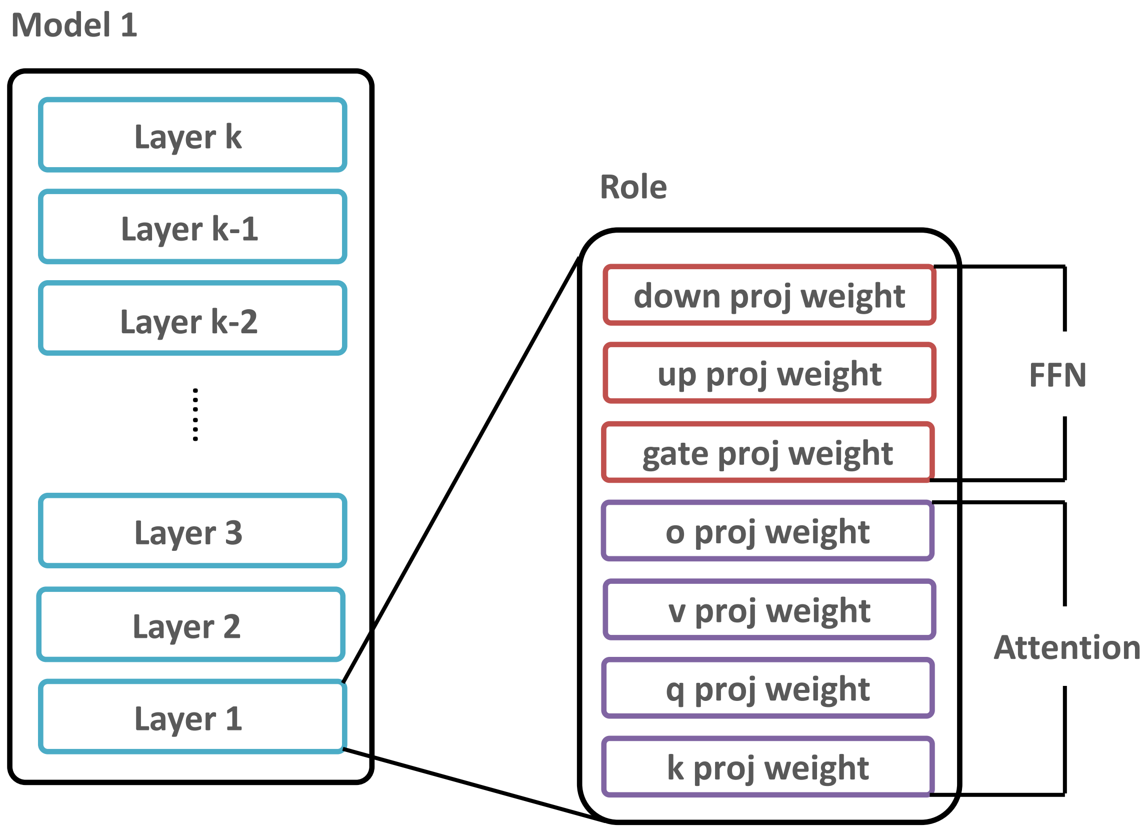


03 Adaptive Drop Rate Merging

Delta parameters δ_k^t of model t, layer k, role r

1. Compute

- Compute rms ($||W||_F/N$) of each weight matrix and group by role r
- Compute CV(Coefficient of Variance, $\frac{\sigma}{\mu}$) of rms
- Role : Attention weight, FFN weight



03 Adaptive Drop Rate Merging

Delta parameters δ_k^t of model t, layer k, role r

2. Drop

- Compute role drop rates using default drop rate $p_{default}$ and drop scale λ, considering rms distribution

$$p_k^t = \begin{cases} p_{default} - \lambda(rms_k^t - rms_{mean}), & CV \leq 50 \\ p_{default} - \lambda(rms_k^t - rms_{mean}^t), & CV > 50 \end{cases}$$

- Random drop parameters in role r matrix with drop rate p_k^t

$$m^t \sim \mathrm{Bernoulli}(p_k^t)$$

$$\tilde{\delta}_k^t = (1 - m^t) \odot \delta_k^t$$

03 Adaptive Drop Rate Merging

Delta parameters δ_k^t of model t, layer k, role r

3. Merge

- Merge each weight matrix with elect sign & disjoint merge
- Elect sign of each parameter and compare
- Disjoint merge parameters with elected sign

03 Adaptive Drop Rate Merging

Delta parameters δ_k^t of model t, layer k, role r

4. Rescale

- Rescale with the smallest drop rate
- Merge delta parameters with base model

$$\hat{\delta}_k^t = \tilde{\delta}_k^t / (1 - p_k^t)$$
$$\theta_{DARE}^t = \hat{\delta}^t + \theta_{PRE}$$

Role r

Red: Smallest drop rate

Model 1	Model 2	Model 3		Rescaling factor
p_{k-1}^1	p_{k-1}^2	p_{k-1}^3	$\longrightarrow$	$1/(1 - p_{k-1}^2)$
p_k^1	p_k^2	p_k^3	$\longrightarrow$	$1/(1 - p_k^3)$
p_{k+1}^1	p_{k+1}^2	p_{k+1}^3	$\longrightarrow$	$1/(1 - p_{k+1}^1)$

04 Experimental Setup

Baselines

TIES-Merging

- Top-50% parameters in task vector (trim 50%)
- $\lambda = 1$

DARE

- Drop rate 0.7, 0.9

Base model

- Llama 2 7b hf

Fine-tuned models

LoRA module fine-tuned only with q, v attention weight

05 Results

Merging similar tasks

	BoolQ	CoQA	QQP	Average
BoolQ Challenge fine-tuned	0.8205	0.7647	0.5193	0.7015
CoQA fine-tuned	0.7976	0.8082	0.5111	0.7056
QQP fine-tuned	0.7798	0.7749	0.532	0.6956
Sign elect & disjoint merge	0.8150	0.8054	0.5066	0.7090
TIES-Merging	**0.8235**	0.8055	<u>0.5132</u>	<u>0.7141</u>
DARE + TIES 0.7 drop	0.8150	0.8054	0.5127	0.7110
DARE + TIES 0.9 drop	0.8171	0.8022	0.5122	0.7105
Adaptive drop rate 0.7	0.8165	<u>0.8100</u>	0.5124	0.7130
Adaptive drop rate 0.9	<u>0.8190</u>	**0.8115**	**0.5185**	**0.7163**

05 Results

Merging different tasks (n=3)

	CoQA	HellaSwag	RTE	Average
CoQA fine-tuned	0.8082	0.7603	0.6534	0.7406
HellaSwag fine-tuned	0.7629	0.7594	0.5668	0.6964
RTE fine-tuned	0.7790	0.7624	0.5884	0.7099
Sign elect & disjoint merge	0.7956	0.7609	0.5993	0.7186
TIES-Merging	<u>0.7999</u>	0.761	0.5957	0.7189
DARE + TIES 0.7 drop	0.7967	0.7603	0.6065	0.7212
DARE + TIES 0.9 drop	0.7946	<u>0.7613</u>	0.6029	0.7196
Adaptive drop rate 0.7	0.7983	**0.7608**	<u>0.6173</u>	<u>0.7255</u>
Adaptive drop rate 0.9	**0.8006**	0.7602	**0.6390**	**0.7333**

05 Results

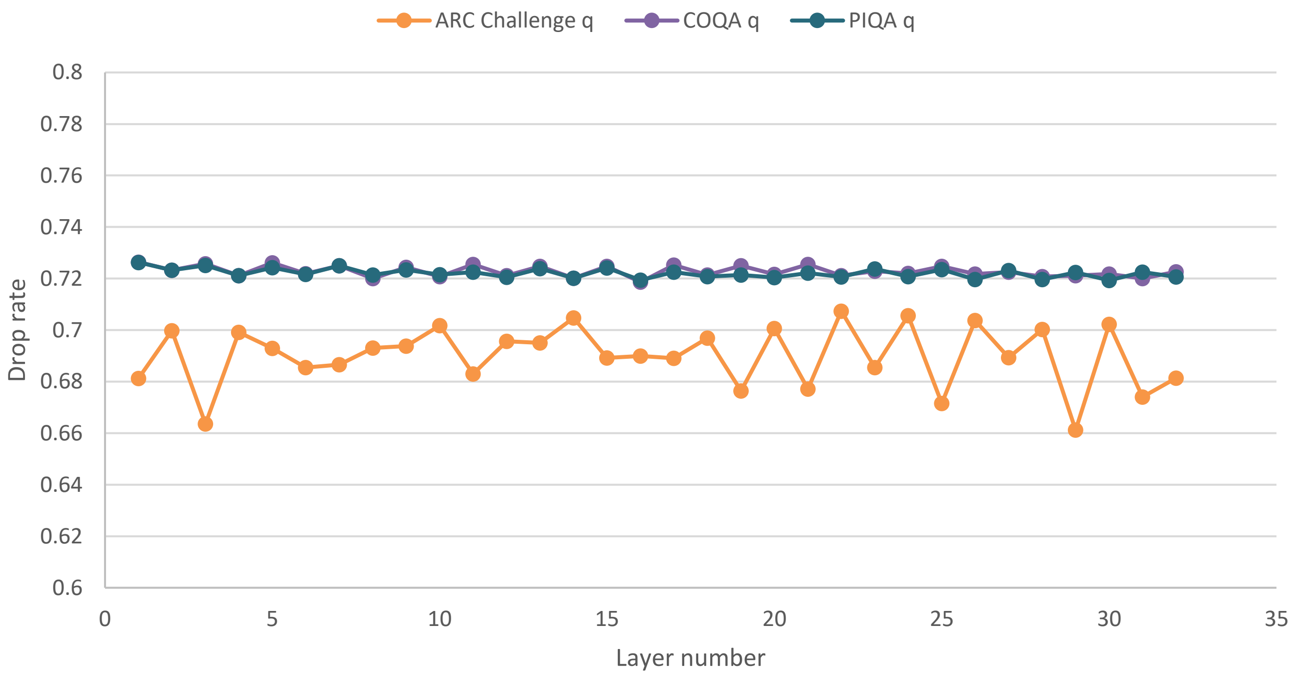
Merging different tasks (n=5)

	Average
Sign elect & disjoint merge	0.6950
TIES-Merging	0.6952
DARE + TIES 0.7 drop	0.6949
DARE + TIES 0.9 drop	0.6941
Adaptive drop rate 0.7	<u>0.6960</u>
Adaptive drop rate 0.9	**0.6980**

- Merge 5 tasks (BoolQ, CoQA, HellaSwag, RTE, QQP)

05 Results

Merging without considering rms distribution

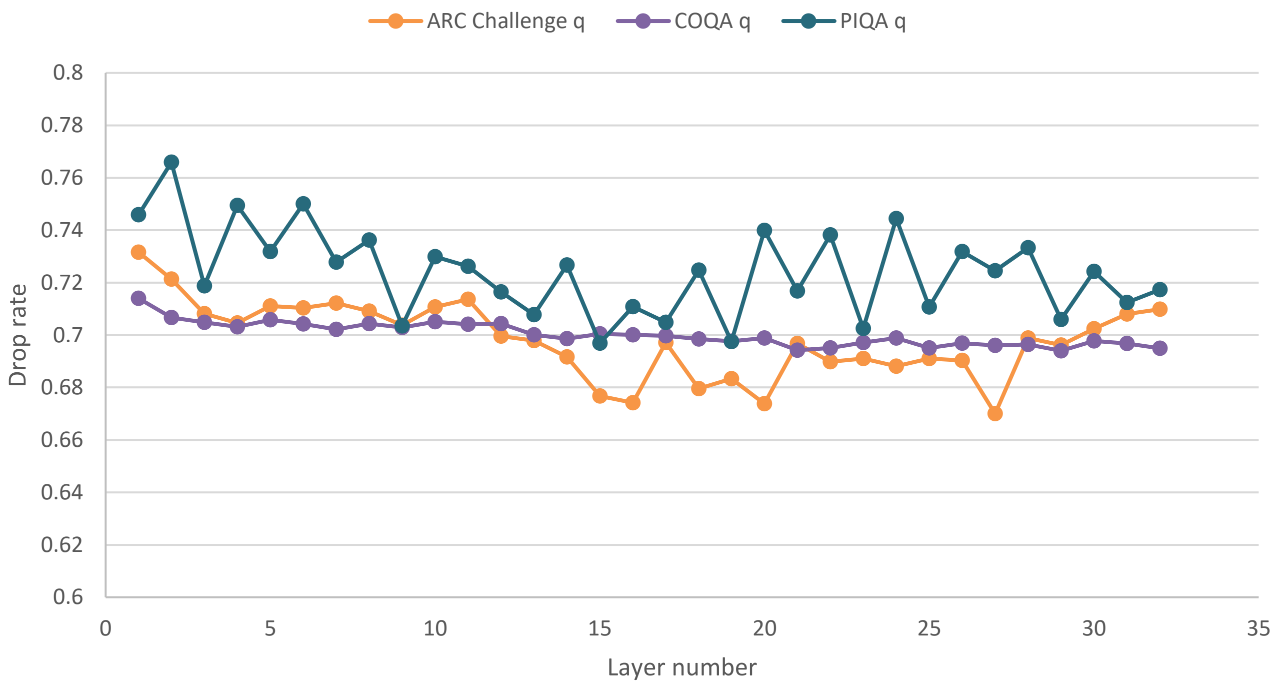


FT Model Task	Average rms
ARC Challenge	8.499E-4
CoQA	1.243E-4
PIQA	1.296E-4

Average rms(μ)	0.3679
SD(σ)	0.4174
CV(σ/μ)	113.4455

05 Results

Rms distribution-aware merging



FT Model Task	Average rms
ARC Challenge	8.499E-4
CoQA	1.243E-4
PIQA	1.296E-4

Average rms(μ)	0.3679
SD(σ)	0.4174
CV(σ/μ)	113.4455

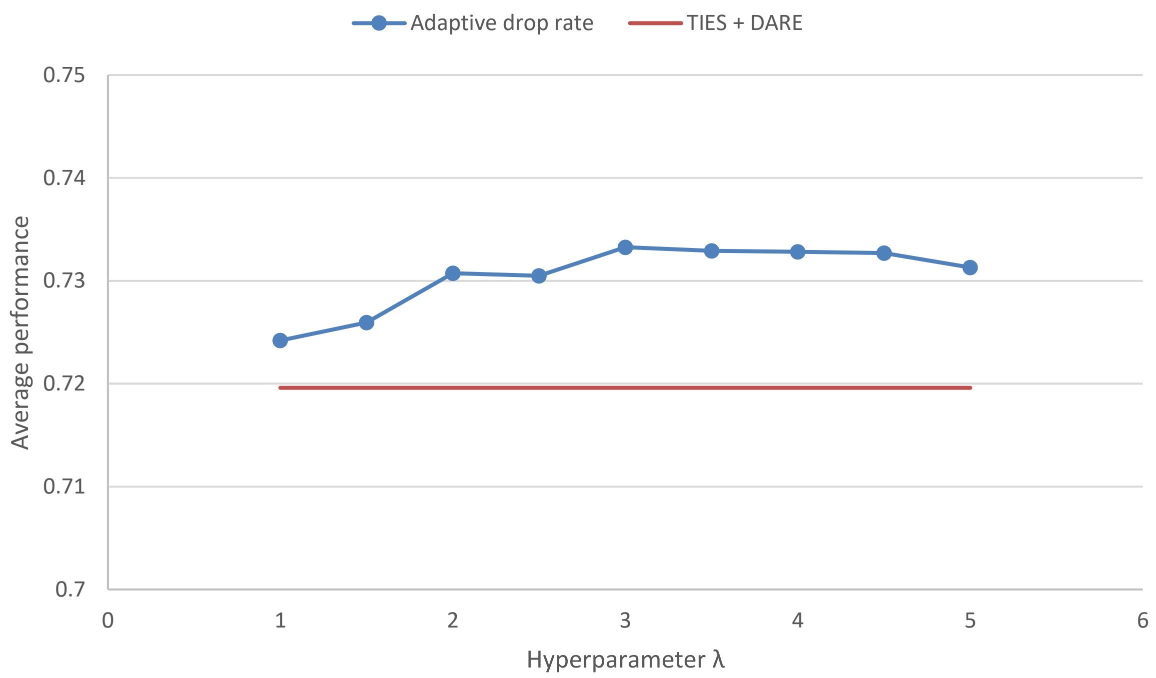
05 Results

Merging high rms dispersion models

	ARC Challenge	CoQA	PIQA	Average
ARC Challenge fine-tuned	0.4198	0.7398	0.7818	0.647133
CoQA fine-tuned	0.4693	0.8082	0.7916	0.6897
PIQA fine-tuned	0.4616	0.7605	0.7933	0.6718
DARE + TIES 0.7 drop	0.494	0.7715	<u>0.7889</u>	<u>0.6848</u>
DARE + TIES 0.9 drop	**0.4983**	0.7688	0.7856	0.6842
Rms distribution-unaware 0.7	0.4889	0.7678	0.7884	0.6817
Rms distribution-unaware 0.9	0.4787	0.7659	0.7862	0.6769
Rms distribution-aware 0.7	<u>0.4966</u>	<u>0.7742</u>	**0.7922**	**0.6877**
Rms distribution-aware 0.9	0.4838	**0.7808**	<u>0.7889</u>	0.6845

05 Results

Hyperparameter sensitivity



05 Results

Different drop methods

	CoQA	HellaSwag	RTE	Average
DARE + TIES 0.7 drop	0.7967	0.7603	0.6065	0.7212
DARE + TIES 0.9 drop	0.7946	0.7613	0.6029	0.7196
Adaptive drop rate 0.7	0.7983	0.7608	0.6173	0.7255
Adaptive drop rate 0.9	**0.8006**	0.7602	**0.639**	**0.7333**
Other drop version 0.7	<u>0.7992</u>	<u>0.7611</u>	0.5884	0.7162
Other drop version 0.9	0.7979	0.7599	<u>0.6282</u>	<u>0.7287</u>

- Other drop version: Drop based on threshold, pruning small parameters

05 Results

Correlation between rms and merge performance

	Rms Mean (SD)	ΔP Mean (SD)	PCC
CoQA/Hella/RTE 0.7	0.1102 (0.0125)	0.0154 (0.0161)	+0.8966
CoQA/Hella/RTE 0.9	0.1102 (0.0125)	0.0121 (0.0208)	+0.8917
BoolQ/COQA/QQP 0.7	0.1005 (0.0357)	0.0176 (0.0259)	+0.8786
BoolQ/COQA/QQP 0.9	0.1005 (0.0357)	0.0098 (0.0084)	+0.8587
BoolQ/CoQA/Hella/QQP/RTE 0.7	0.1013 (0.0270)	0.0059 (0.0059)	+0.8523
BoolQ/CoQA/Hella/QQP/RTE 0.9	0.1013 (0.0270)	0.0079 (0.0094)	+0.8654

05 Results

Average merge time

Method	Average merge time(s)
DARE + TIES	281.1663
Adaptive drop rate	259.6078

Ablation of adaptive drop rate

Method	Average
Adaptive drop rate	**0.7333**
-Elect	0.7229
-Rescale	0.7295

06 Conclusion

Adaptive drop rate merging

- Consider layer-level parameter interference
- Compute role rms, drop redundant parameters, merge models with sign elect & disjoint merge, rescale weight matrix
- Better performance than TIES-Merging and DARE

Limitations

- Other importance metric instead of using rms
- Need to calculate rms of all models