LLMTailor: A Layer-wise Tailoring Tool for Efficient Checkpointing of Large Language Models

Minqiu Sun, Dong Dai

Xin Huang, Kento Sato

Luanzheng (Lenny) Guo, Nathan R. Tallent

University of Delaware

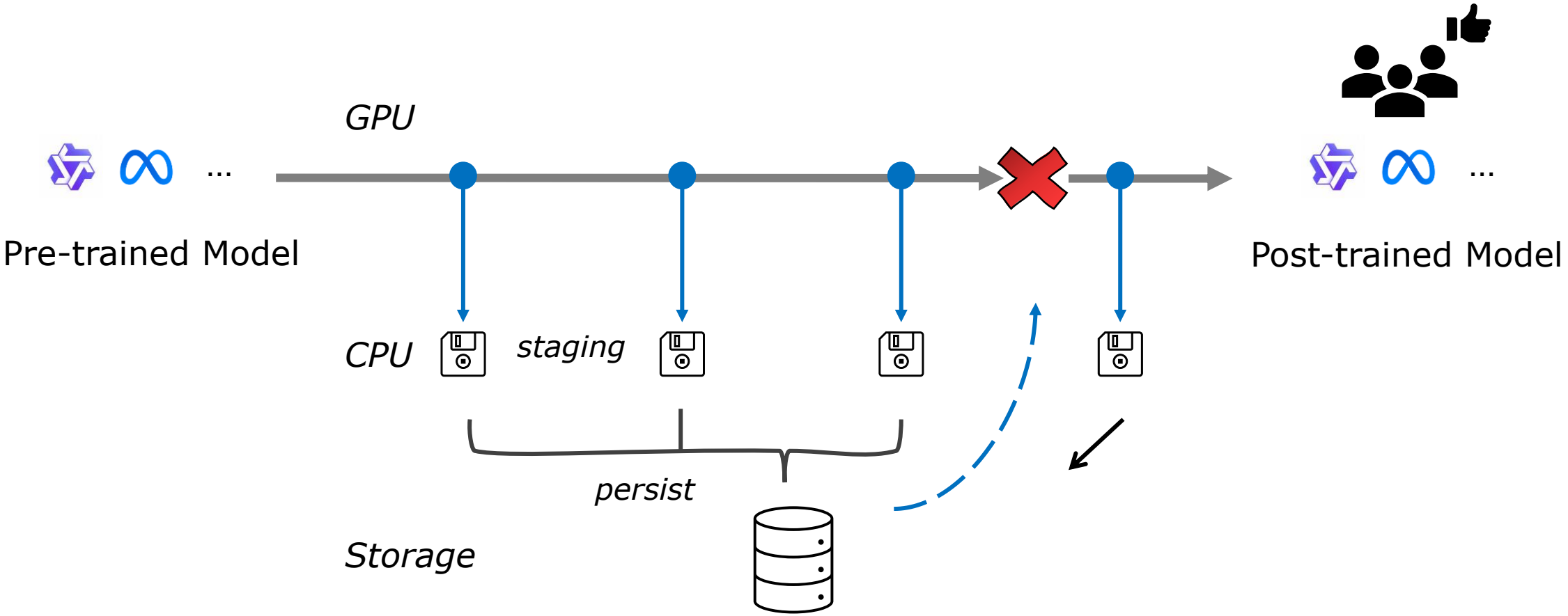
RIKEN

Pacific Northwest National Laboratory

DIRLAB

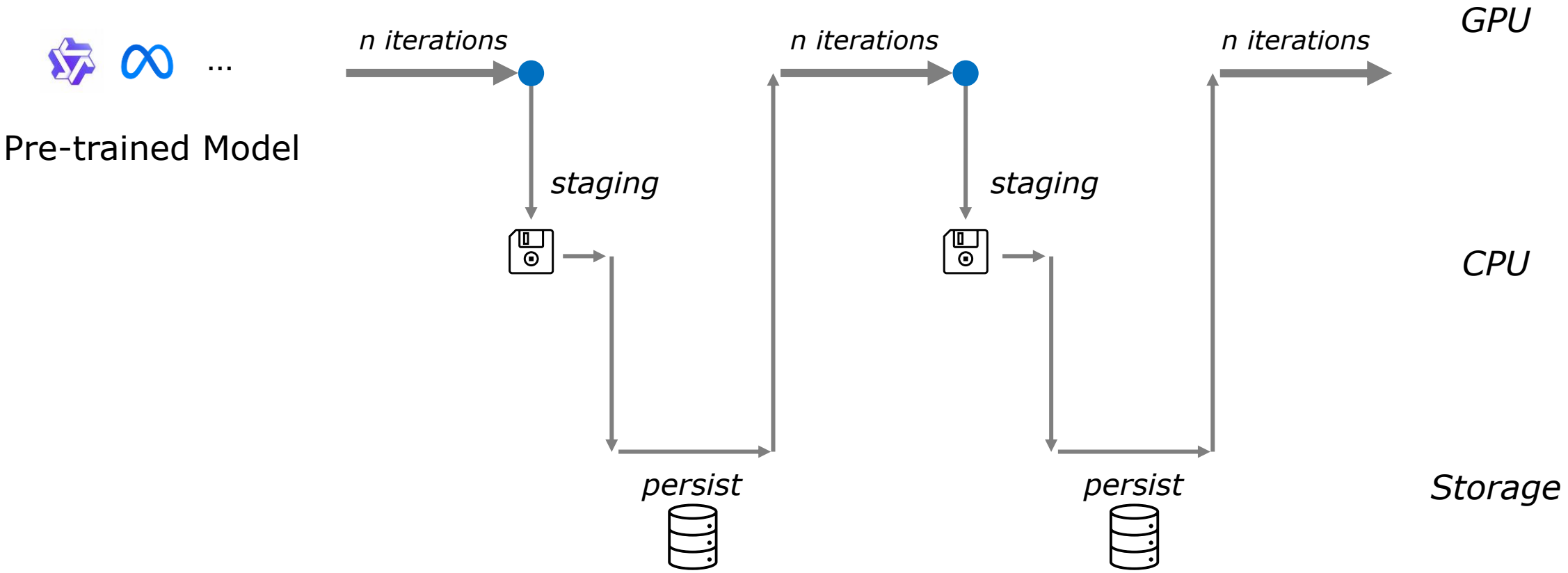
DIRLAB

Checkpoint in Post-training



DIRLAB

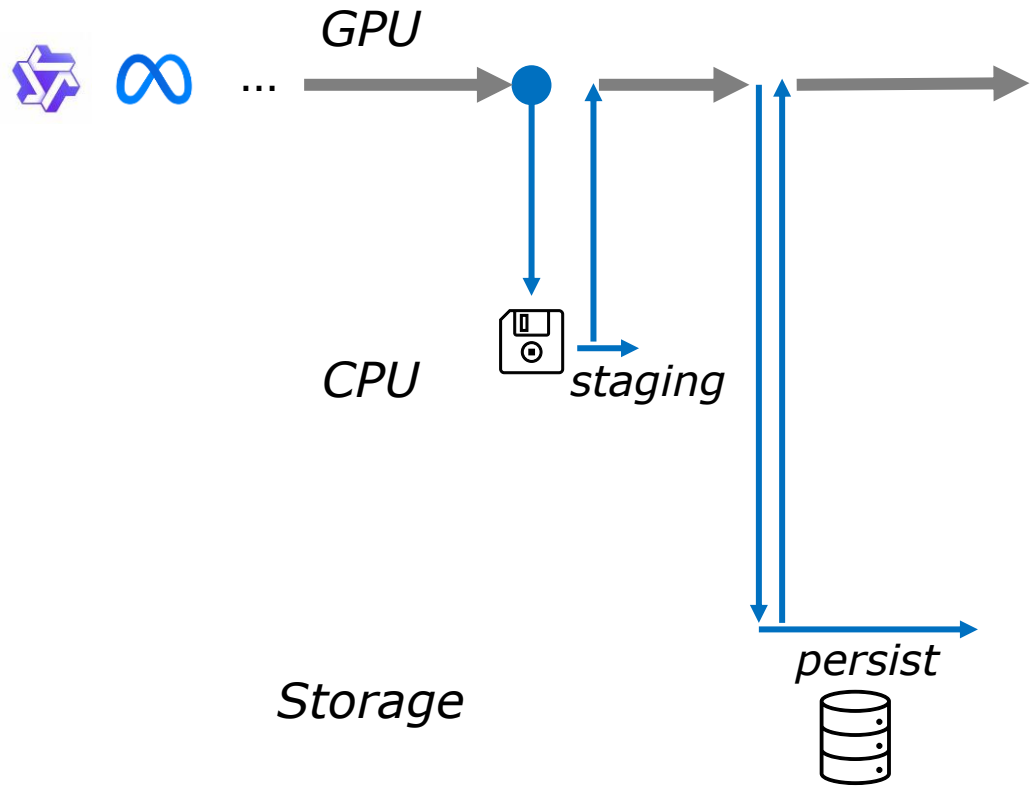
Checkpoint Overhead



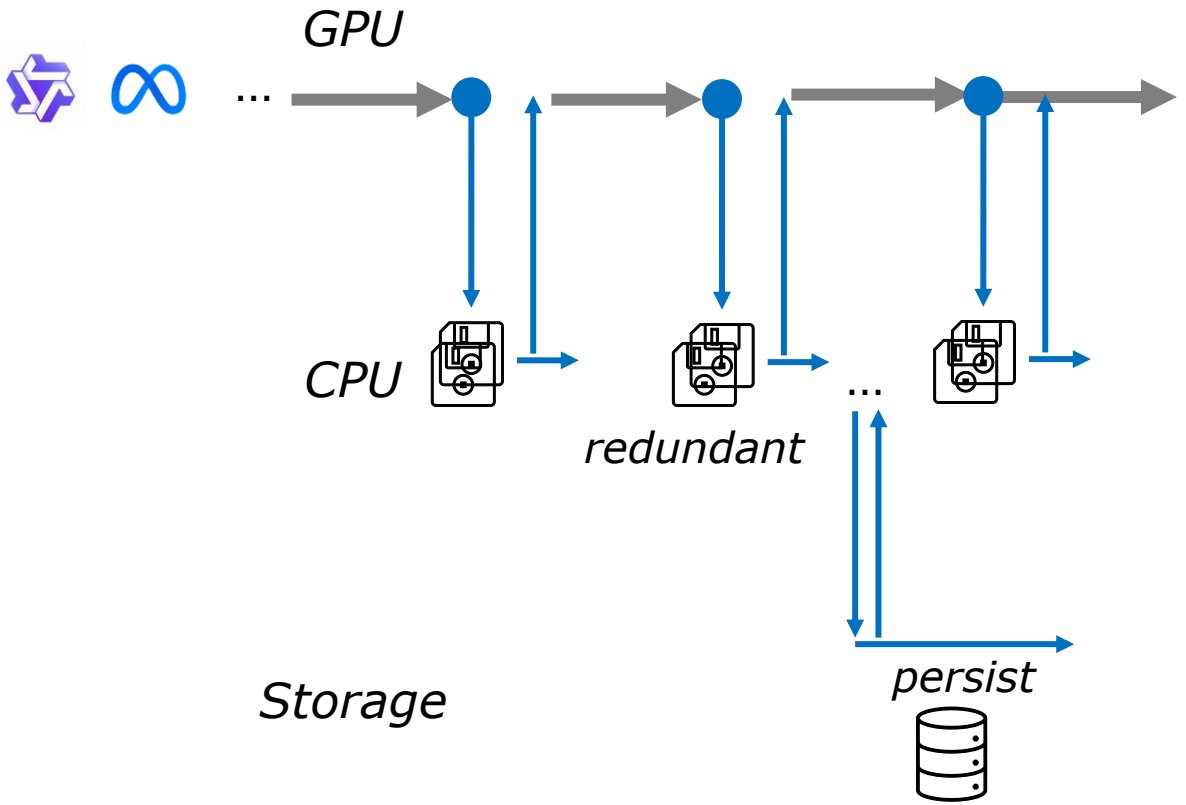
DIRLAB

Prior optimizations

Interleaving

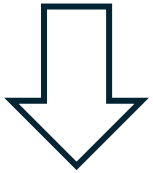


In-memory



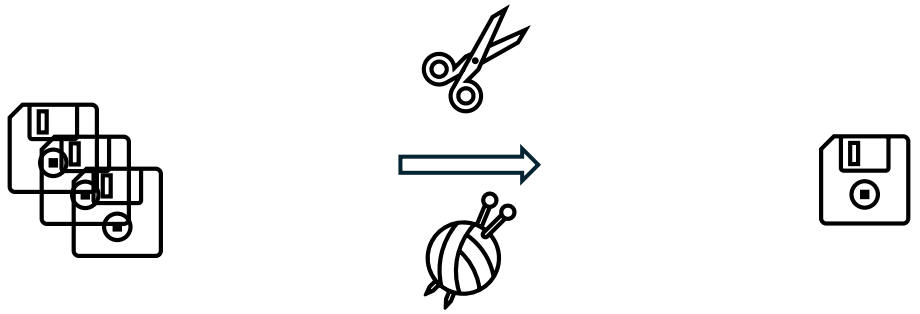
DIRLAB

No optimization to the checkpoint itself!



Utilize the architecture of the checkpoint to minimize overhead at the source!

DIRLAB

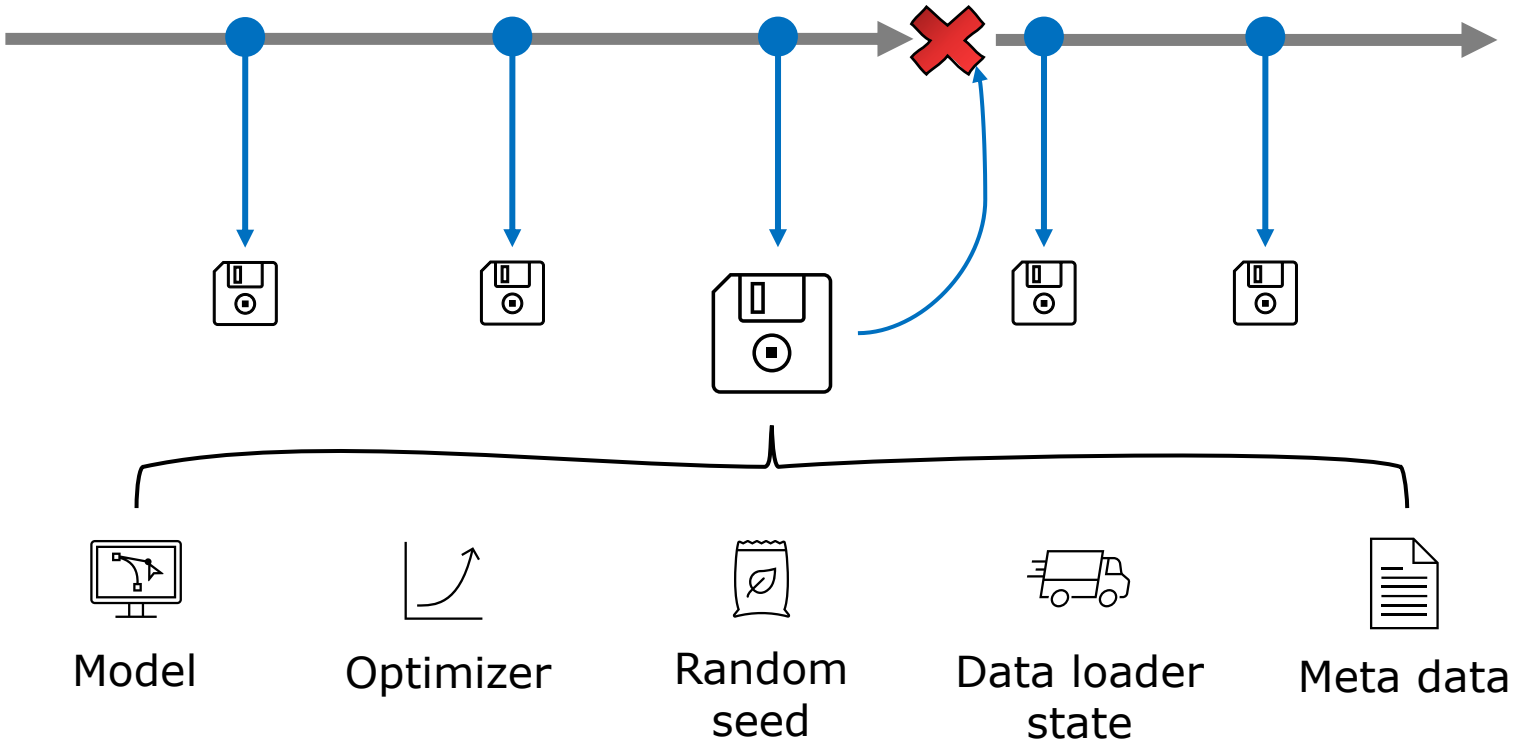


LLMTailor: A checkpoint-merging framework that filters and assembles layers from different checkpoints to form a composite checkpoint.

We enable saving only part of the original checkpoint to reduce checkpoint overhead. (**up to 4.3 times smaller storage size** and **3.0× speedup in checkpointing time**)

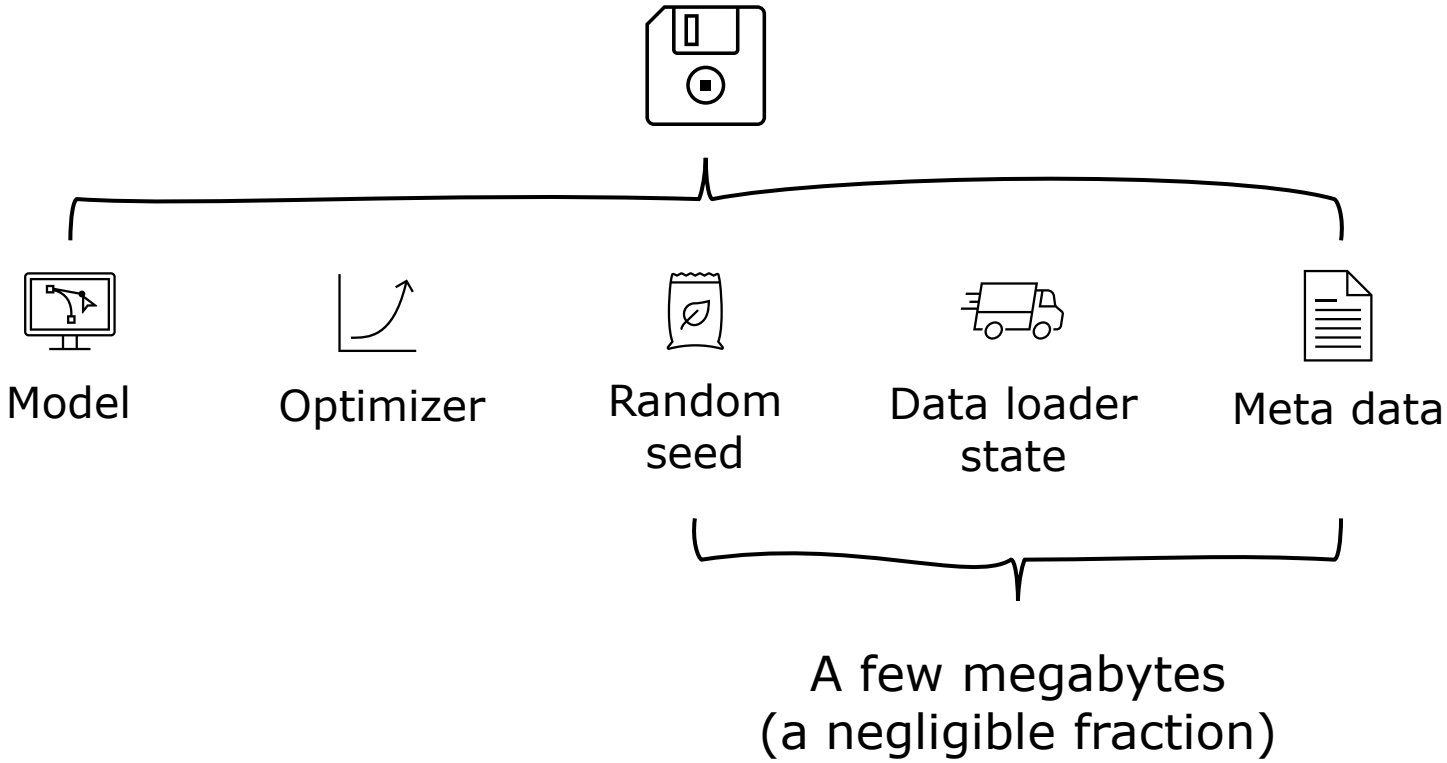
DIRLAB

What is included in the checkpoint for resuming training?



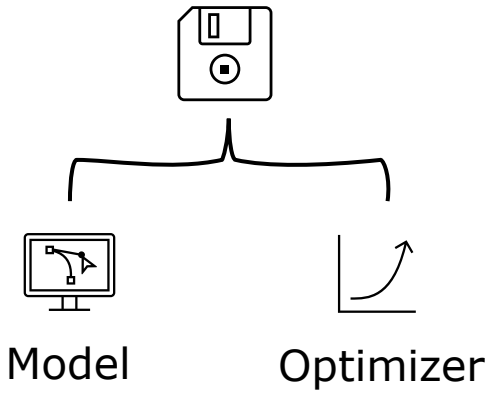
DIRLAB

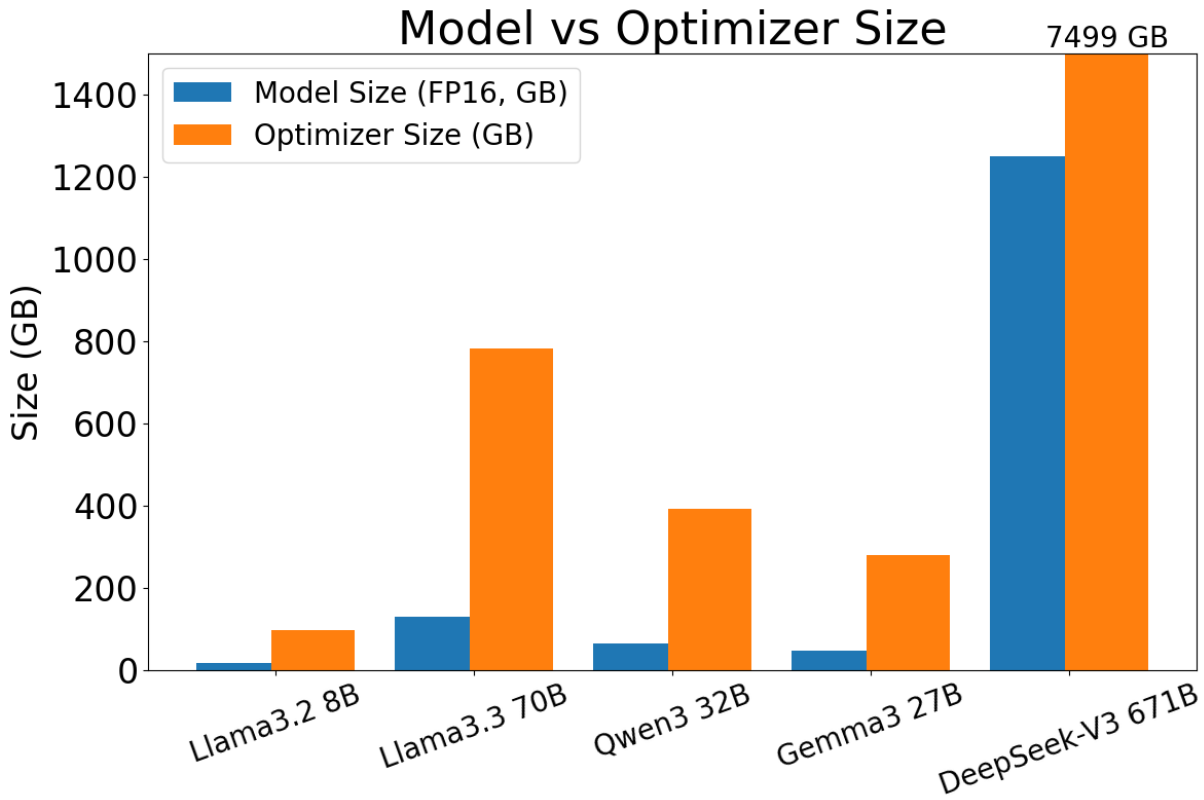
How large is a checkpoint?



DIRLAB

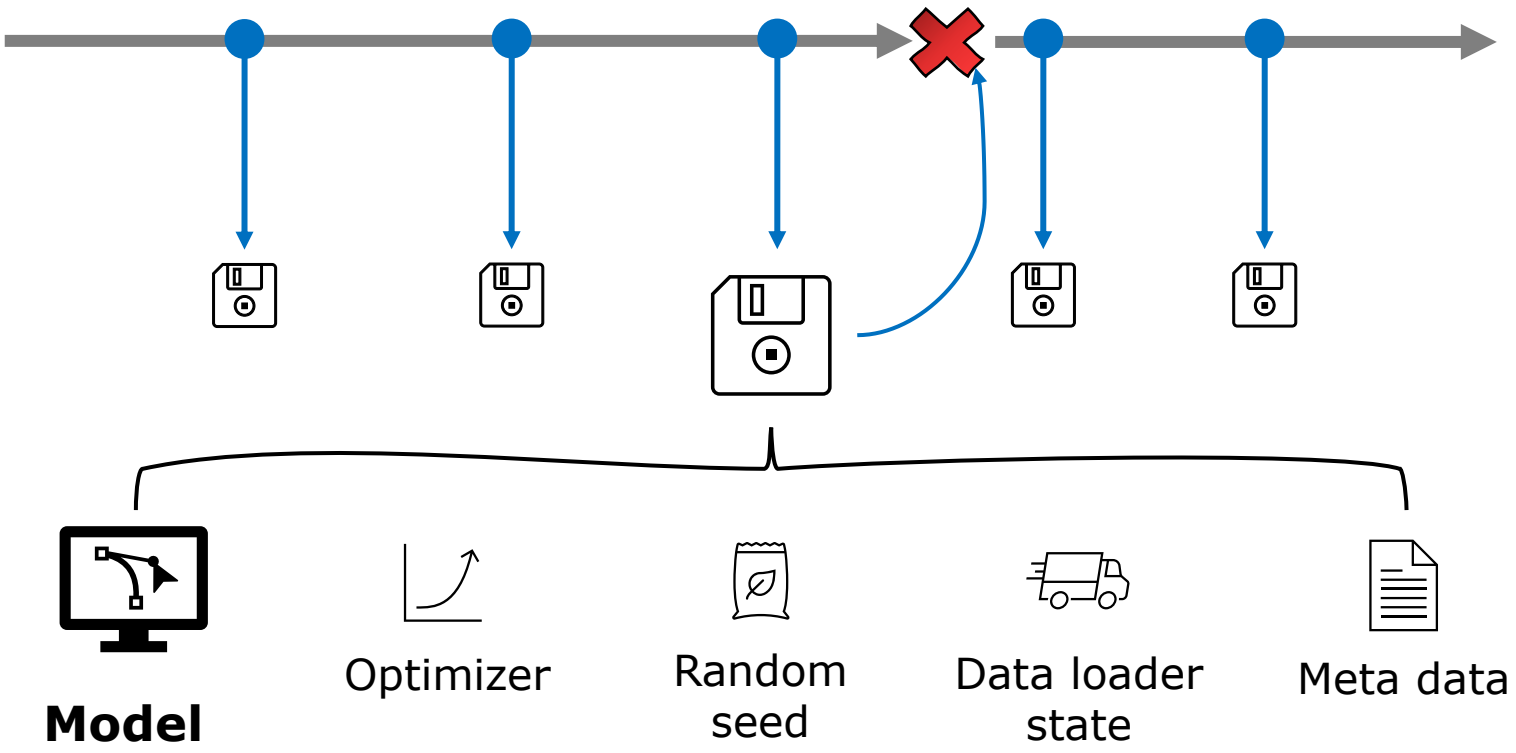
How large is a checkpoint?





PS: The optimizer size is based on the calculation

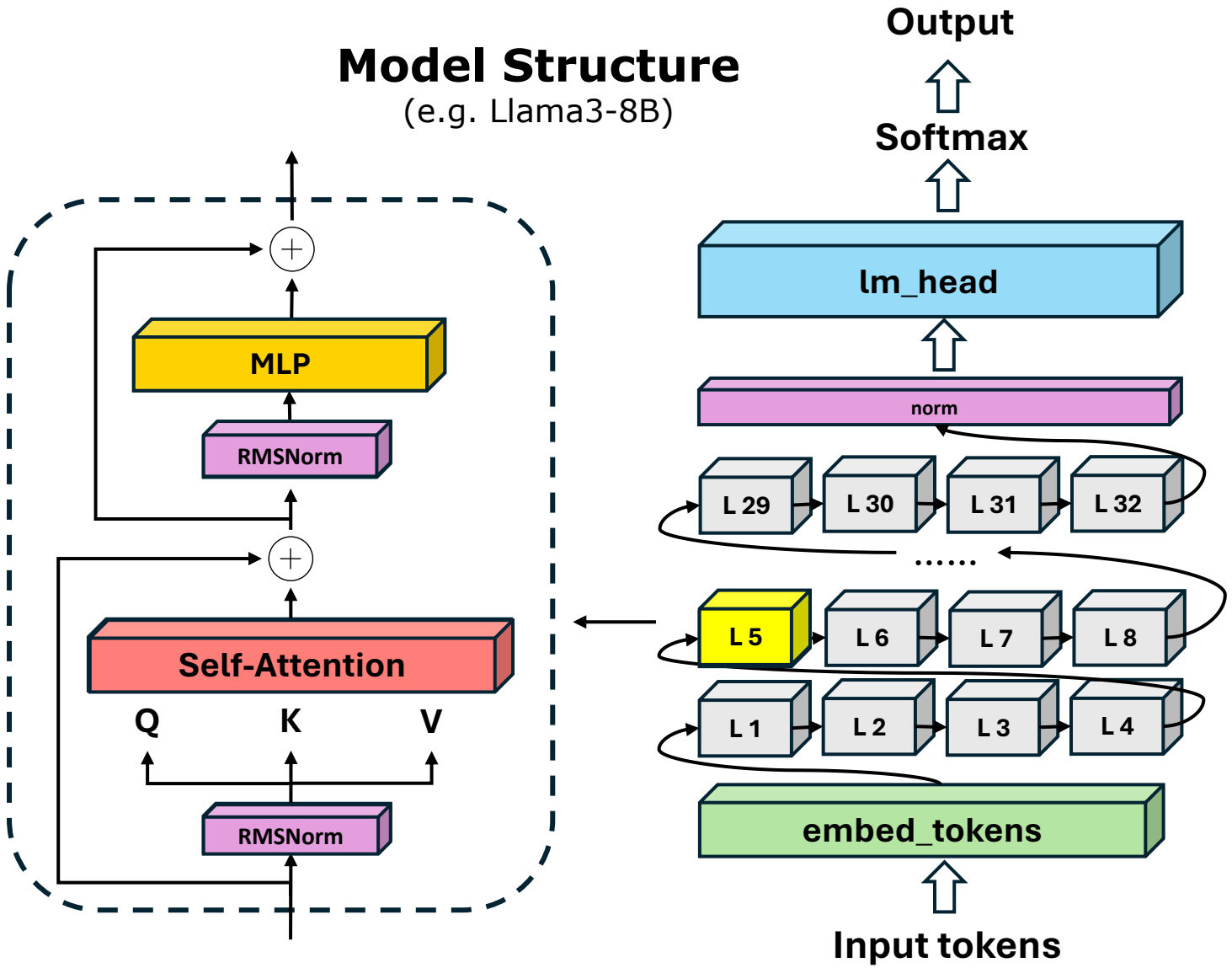
DIRLAB

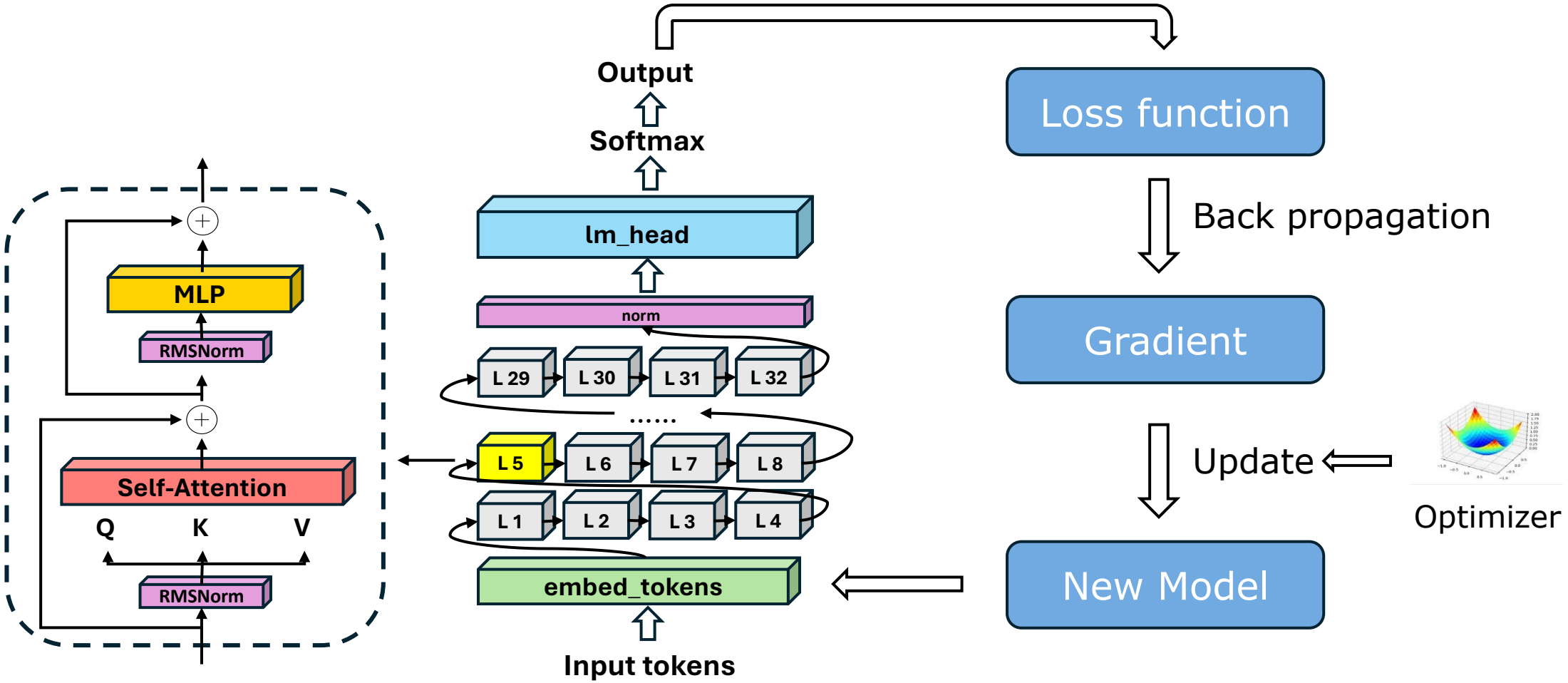


DIRLAB

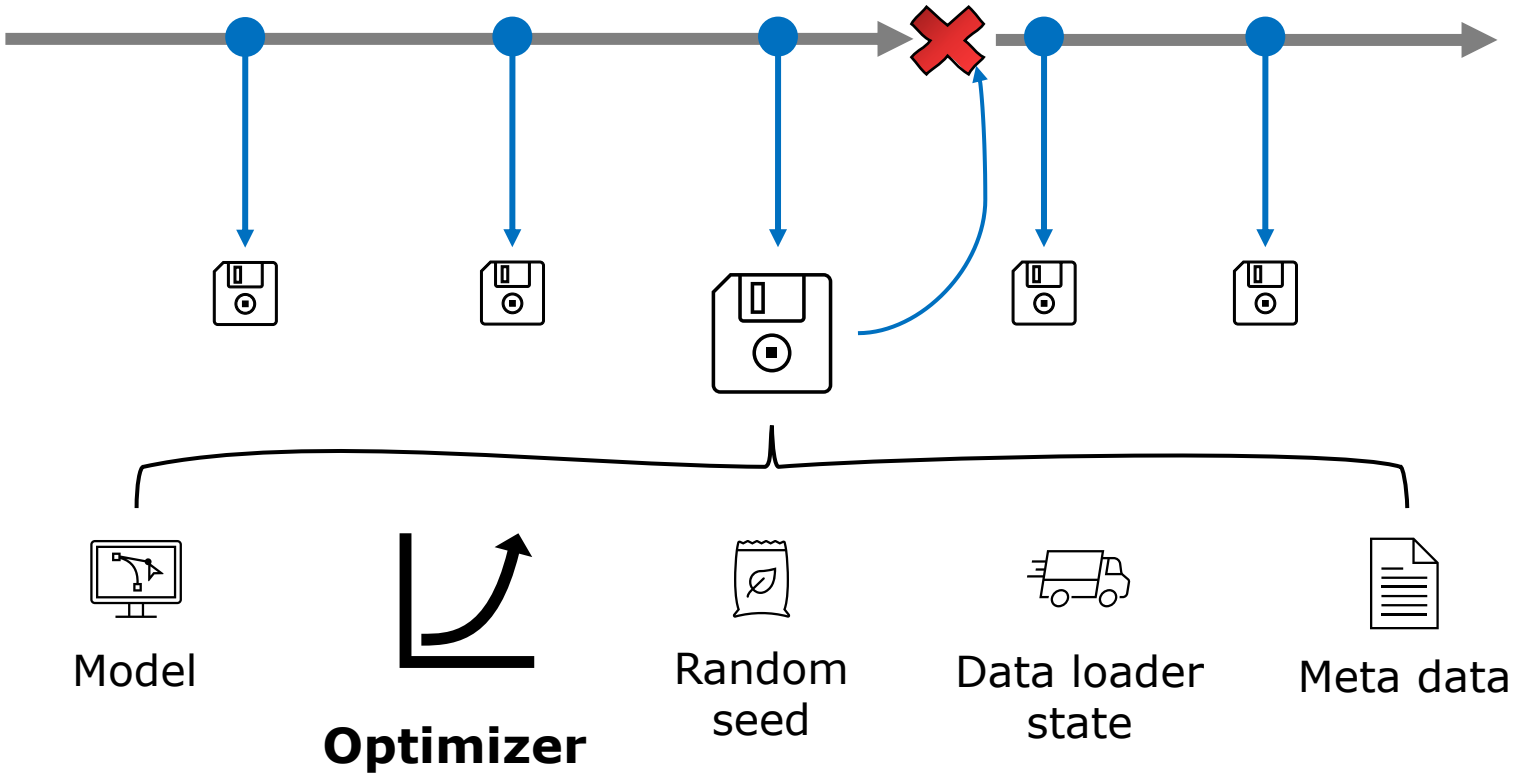
Model Structure

(e.g. Llama3-8B)



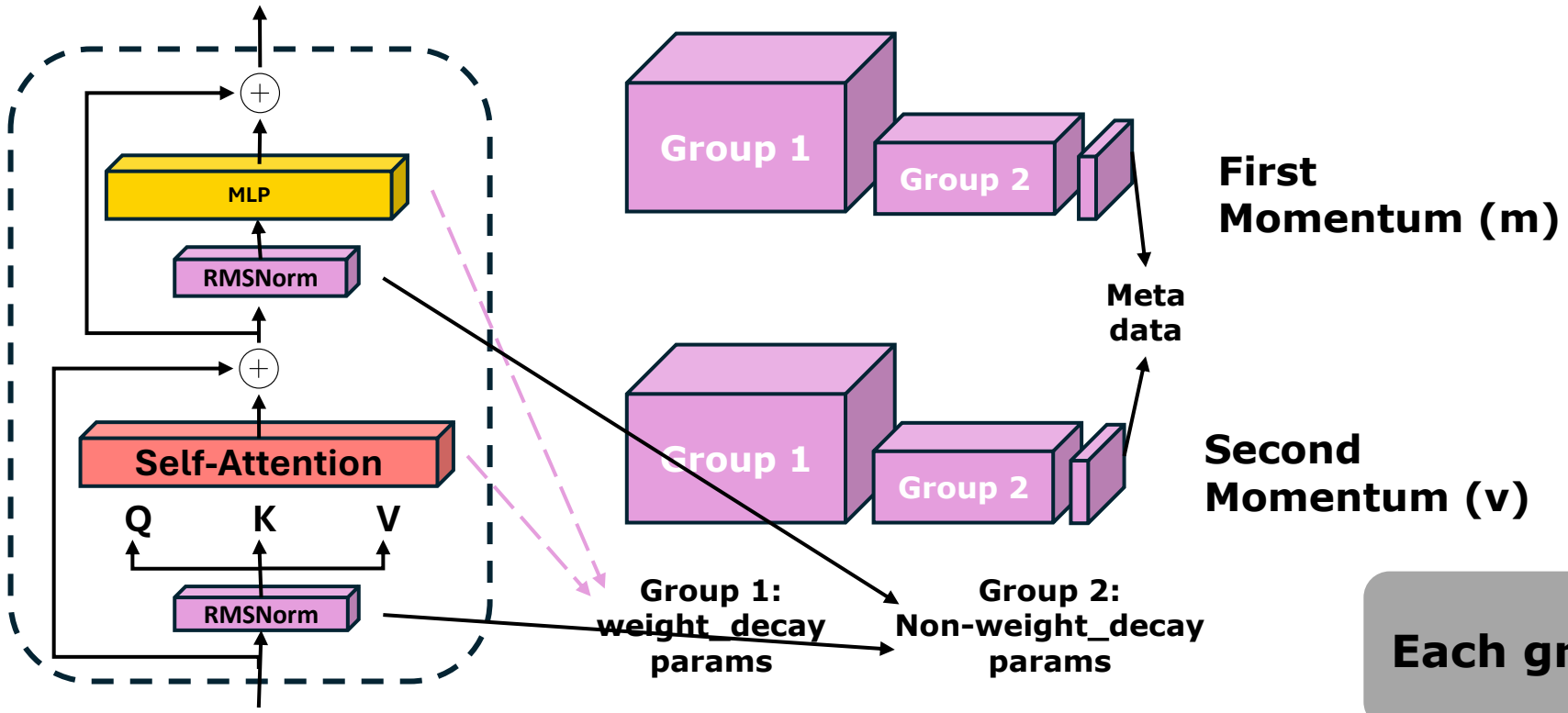

Output
Softmax
lm_head
norm
L 29
L 30
L 31
L 32
.....
L 5
L 6
L 7
L 8
L 1
L 2
L 3
L 4
embed_tokens
Input tokens
MLP
RMSNorm
Self-Attention
Q
K
V
RMSNorm
Loss function
Back propagation
Gradient
Update
Optimizer
New Model

DIRLAB



DIRLAB

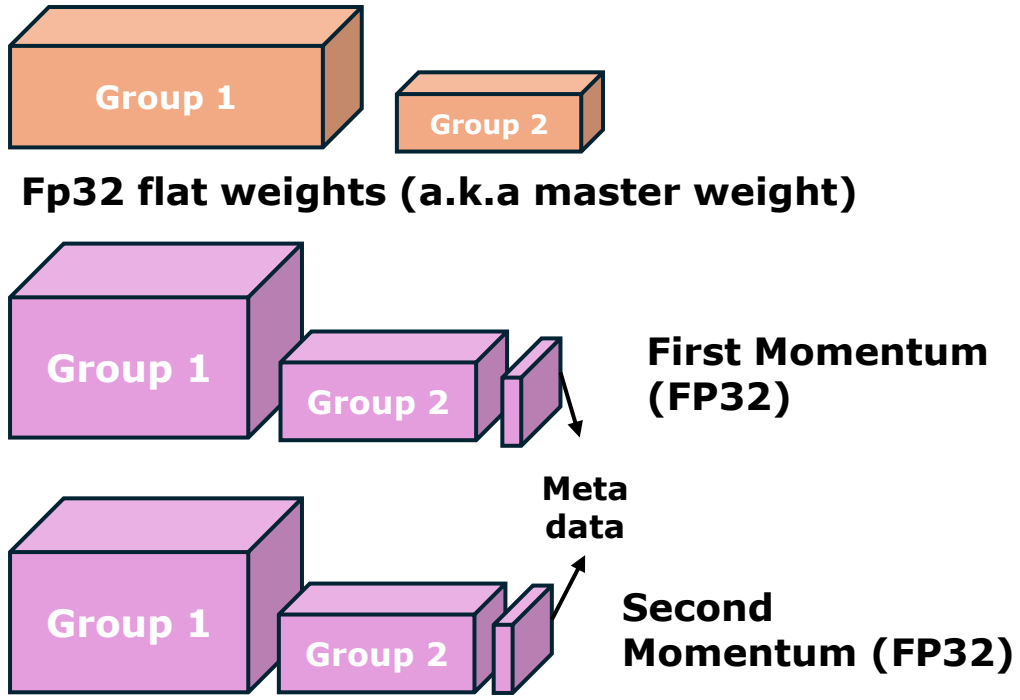
1. Two momentum terms in the **Adam Series Optimizer**

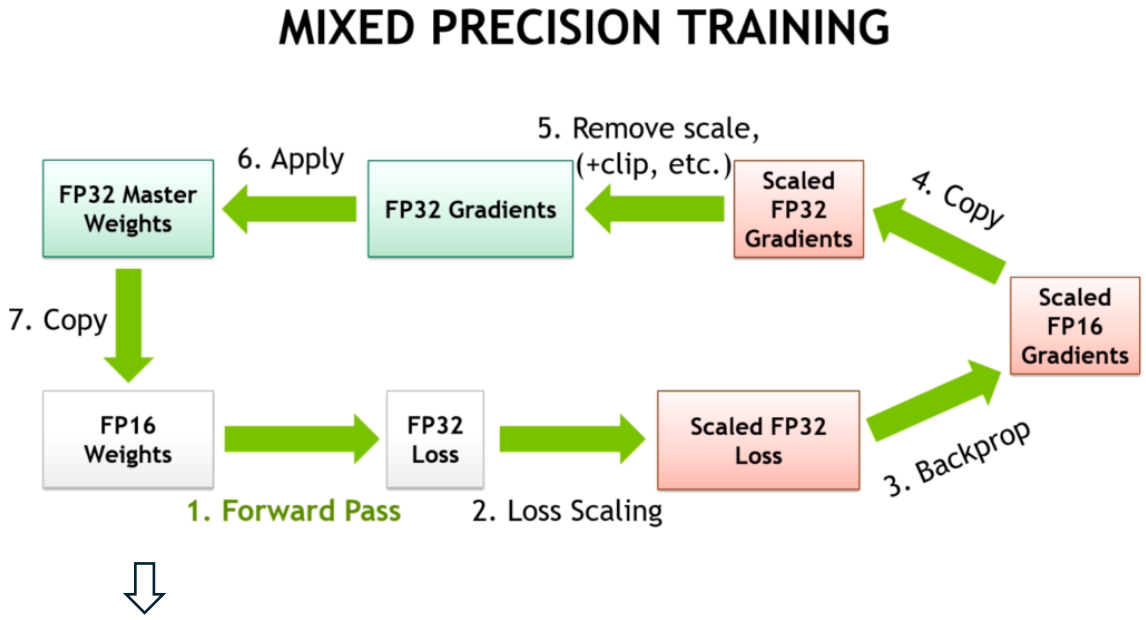


Each group is a flat tensor

DIRLAB

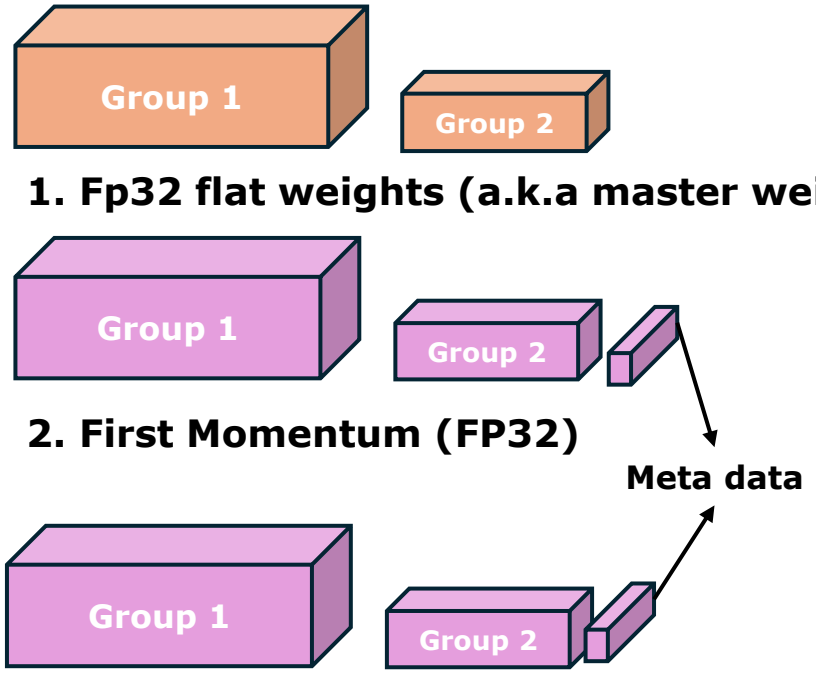
2. FP32 precision & Master weight





PS: For quick reference or validation tests, the FP16 model is typically saved as well.

DIRLAB

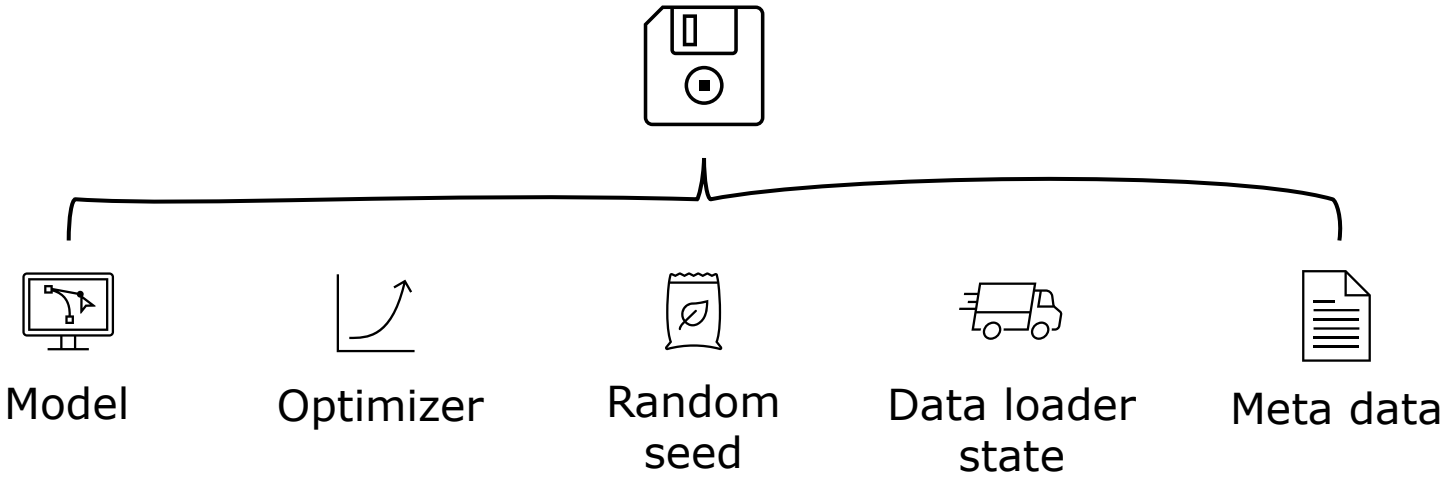


$\times 3 \times 2 = \times 6$

AdamW Optimizer Structure

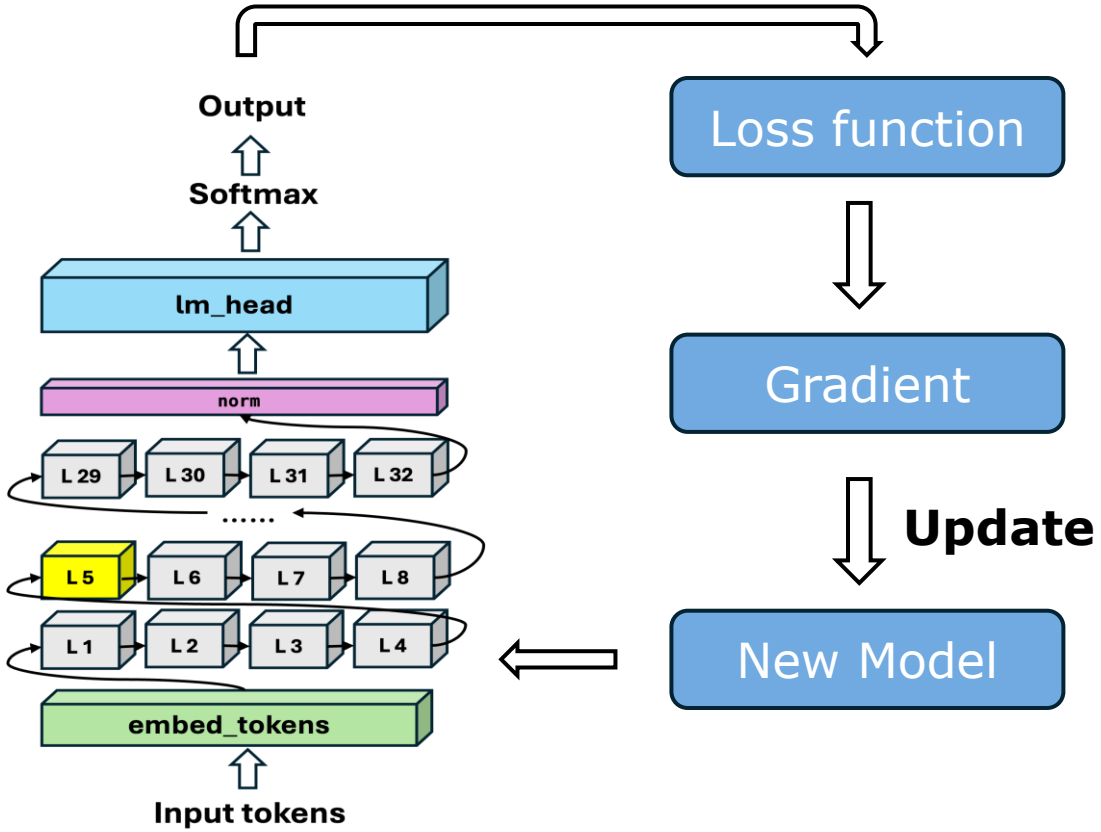
DIRLAB

Question 1: Do we really need to save everything every time?



DIRLAB

Question 1: Are the weights of all layers uniformly updated each time the model is saved?

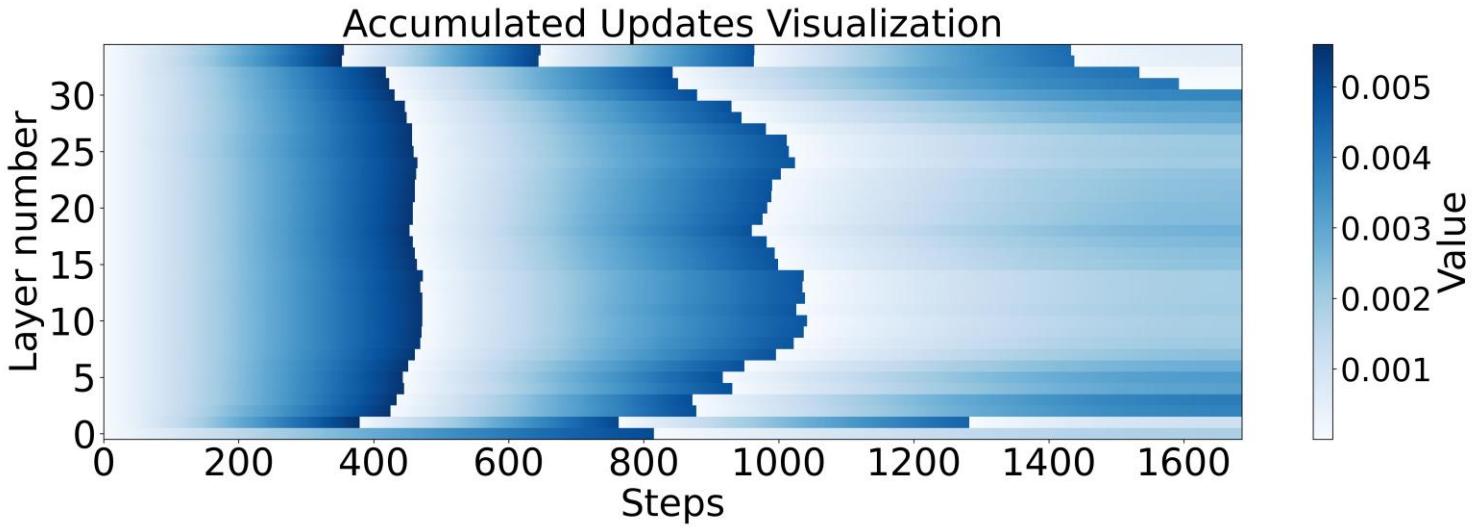


DIRLAB

Question 1: Are the weights of all layers uniformly updated each time the model is saved?

Answer: No.

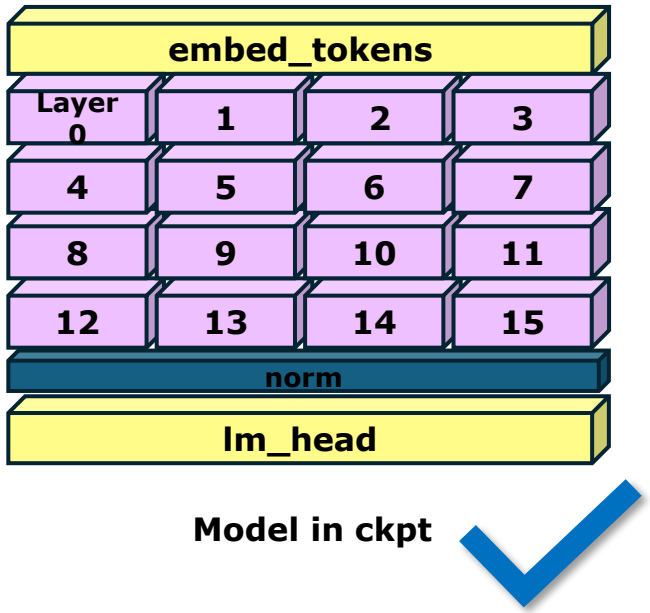
Therefore, we can checkpoint a subset of all layers at different checkpoints based on the magnitude of updates.

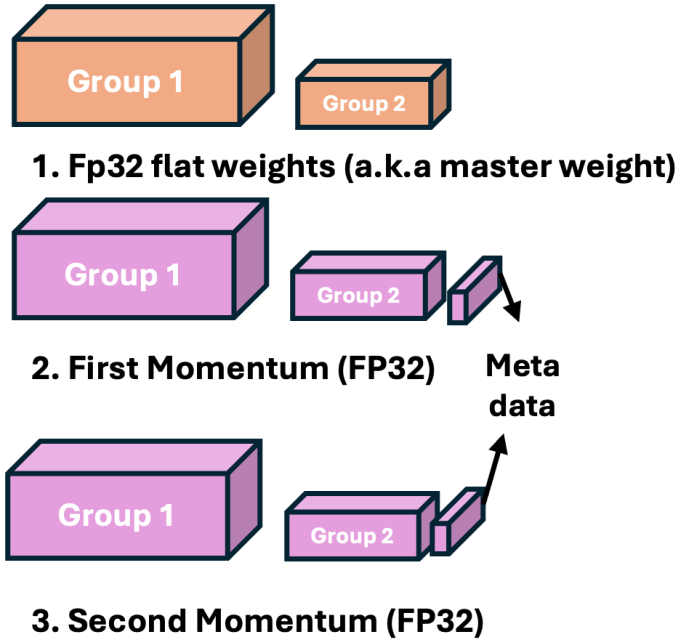


DIRLAB

Checkpoint a subset of all layers

Question 2: Can we split and merge the checkpoints layer-wise?





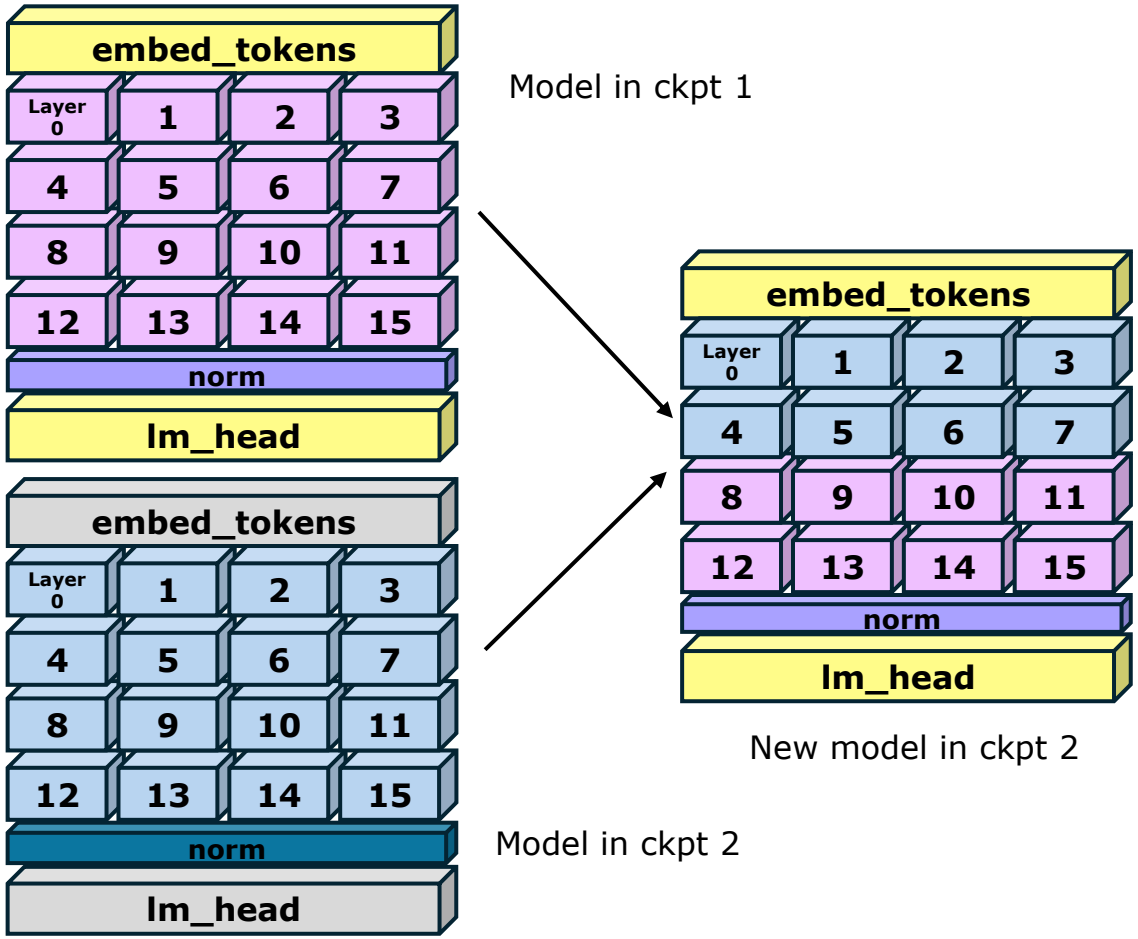
DIRLAB

Question 2: Can we split and merge the checkpoints layer-wise?

Answer: For the models, we already have a toolkit - mergekit

mergekit [1]

- mergekit is a toolkit for merging pre-trained language models.
- Merges can be run entirely on CPU or accelerated with as little as 8 GB of VRAM.
- Many merging algorithms are supported.

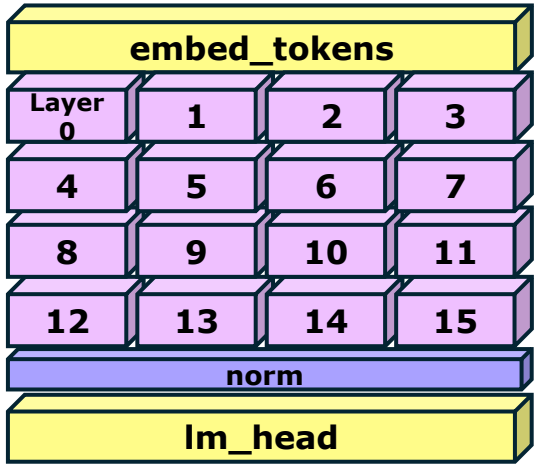


[1] Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024.Arcee's MergeKit: A Toolkit for Merging Large Language Models. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina (Eds.). Association for Computational Linguistics, Miami, Florida, US, 477–485. doi:10.18653/v1/2024.emnlp-industry.3

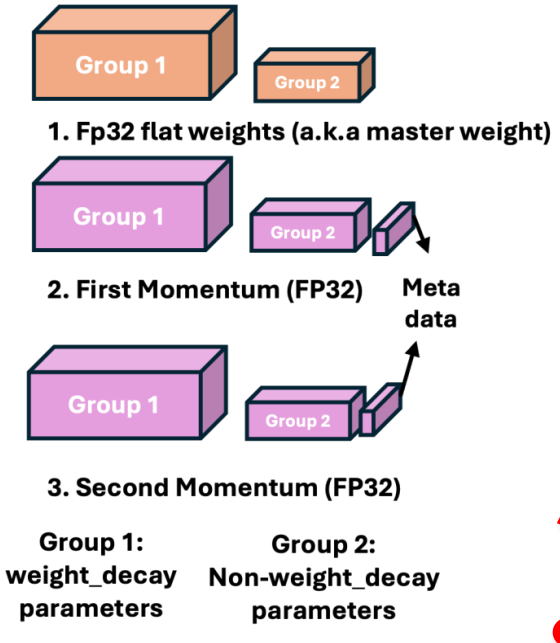
DIRLAB

Checkpoint a subset of all layers

Question 2: Can we split and merge the checkpoints layer-wise?



Model in ckpt

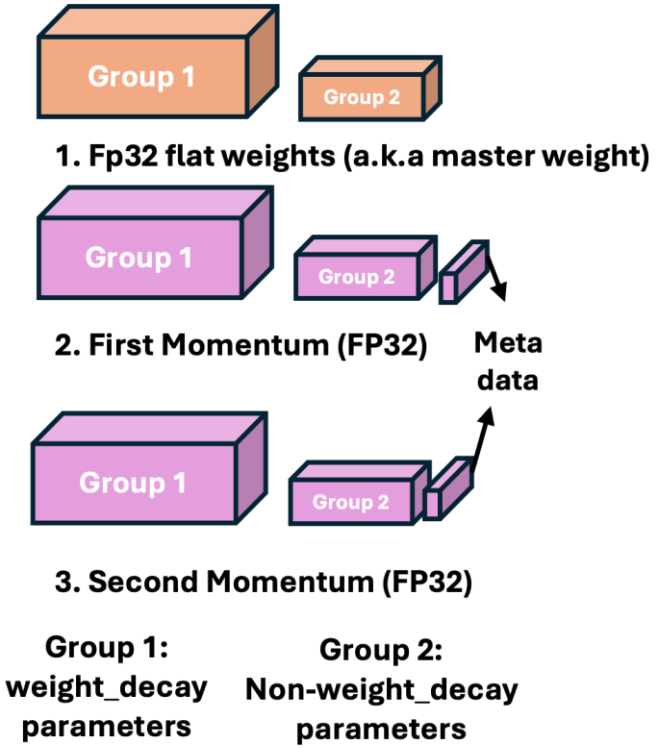


Group 1: weight_decay parameters

Group 2: Non-weight_decay parameters

DIRLAB

Challenge 1: How to divide tensors in the optimizer?

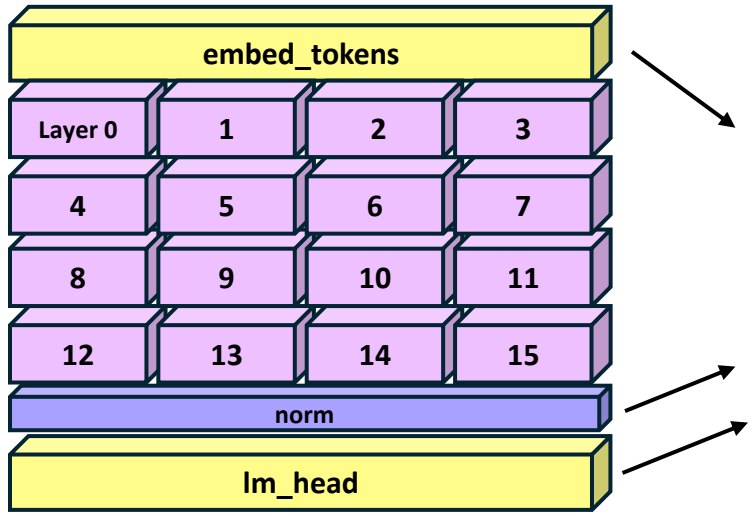


AdamW Optimizer Structure

DIRLAB

Challenge 2: How to divide auxiliary layers in the model?

mergekit can only manipulate **hidden layers**

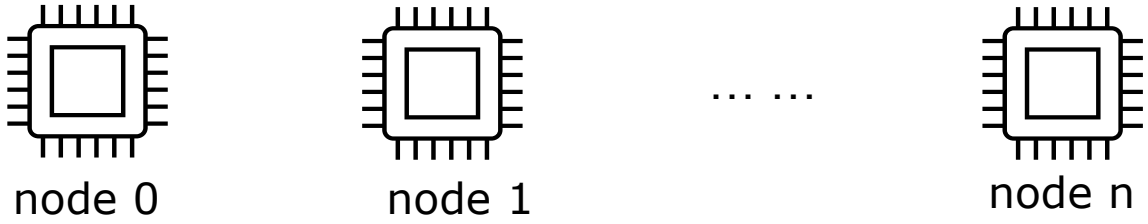


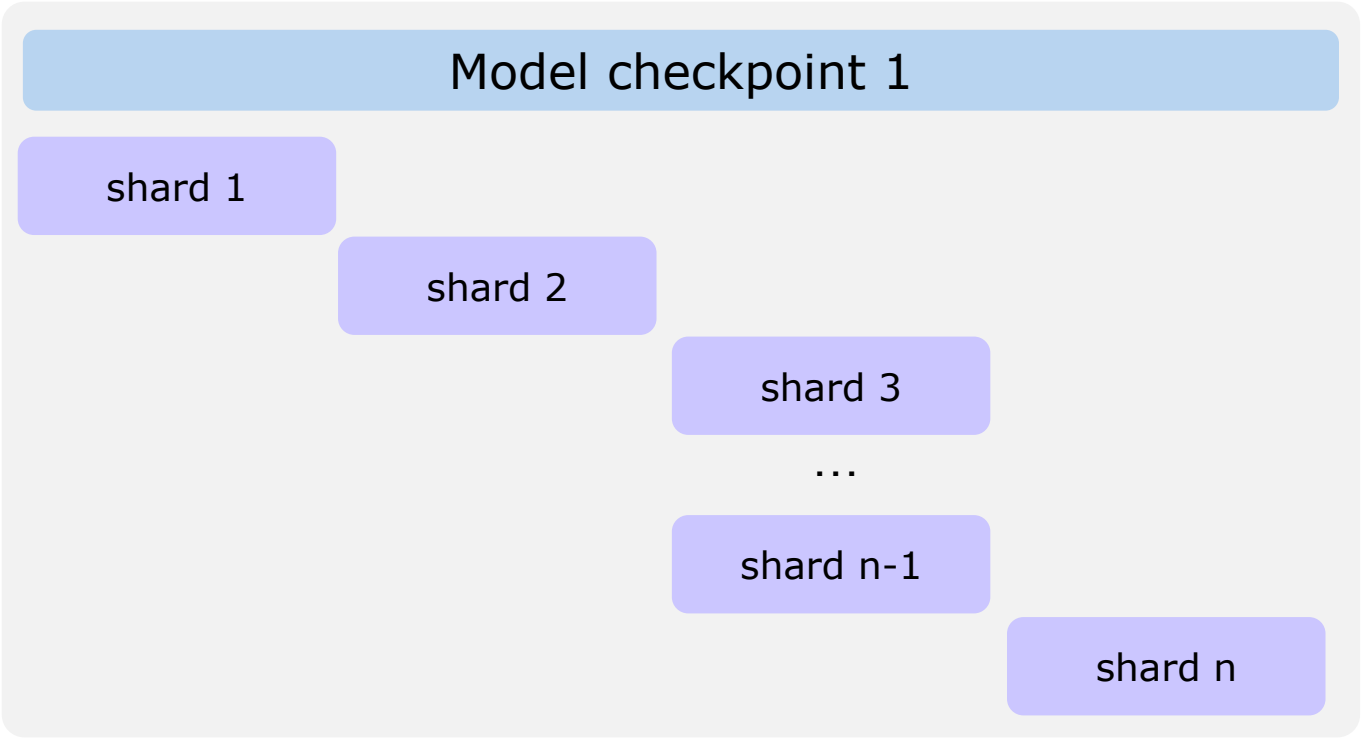
Model in ckpt

Auxiliary layers can only be copied from one checkpoint when merging

DIRLAB

Challenge 3: How to adapt distributed training ?

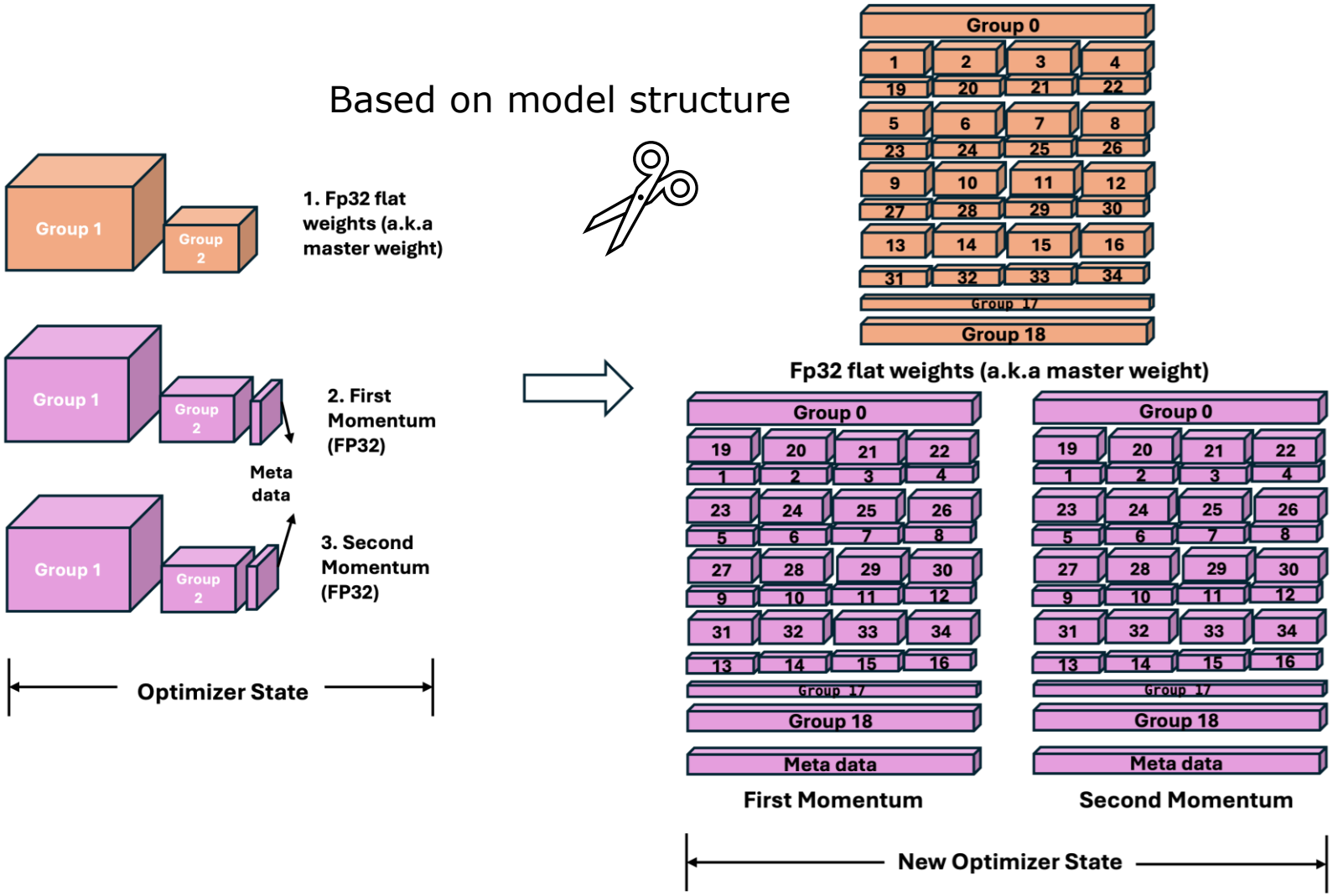




DIRLAB

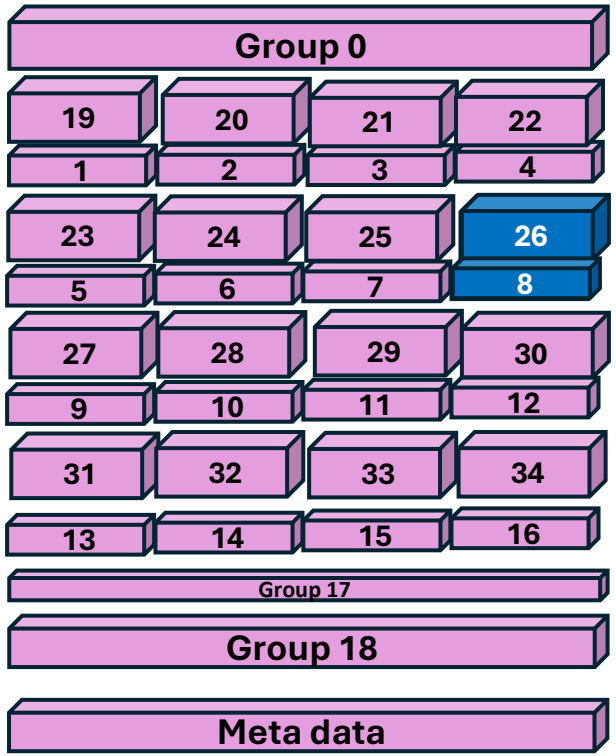
Challenge 1: Unindexable tensors in the optimizer

Solution: Divide the training group to divide the optimizer naturally



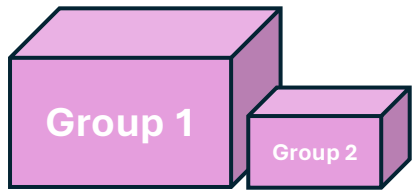
Challenge 1: Unindexable tensors in the optimizer

Solution: Divide the training group to divide the optimizer naturally



First Momentum

Same structure as



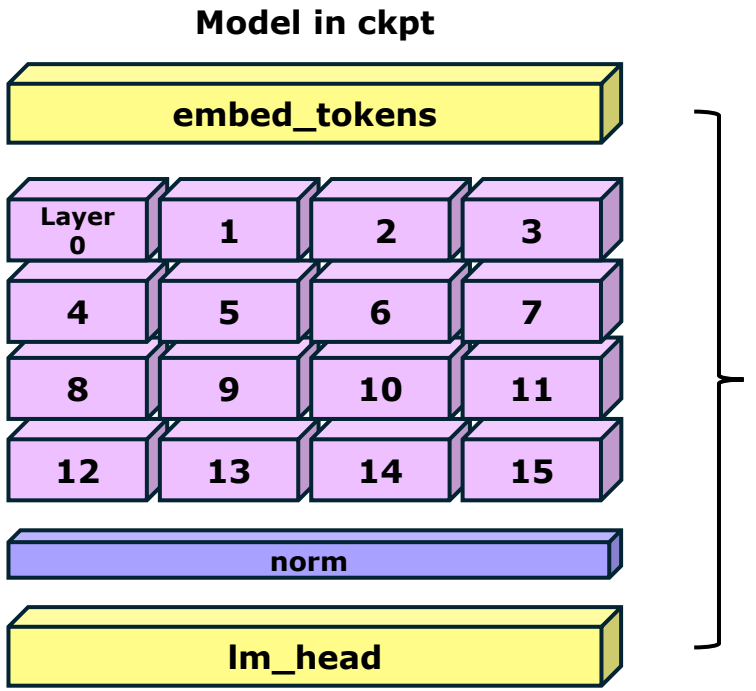
weight_decay

Non-weight_decay

DIRLAB

Challenge 2: Unindexable auxiliary layers in the model

Solution: Improve mergekit functionality

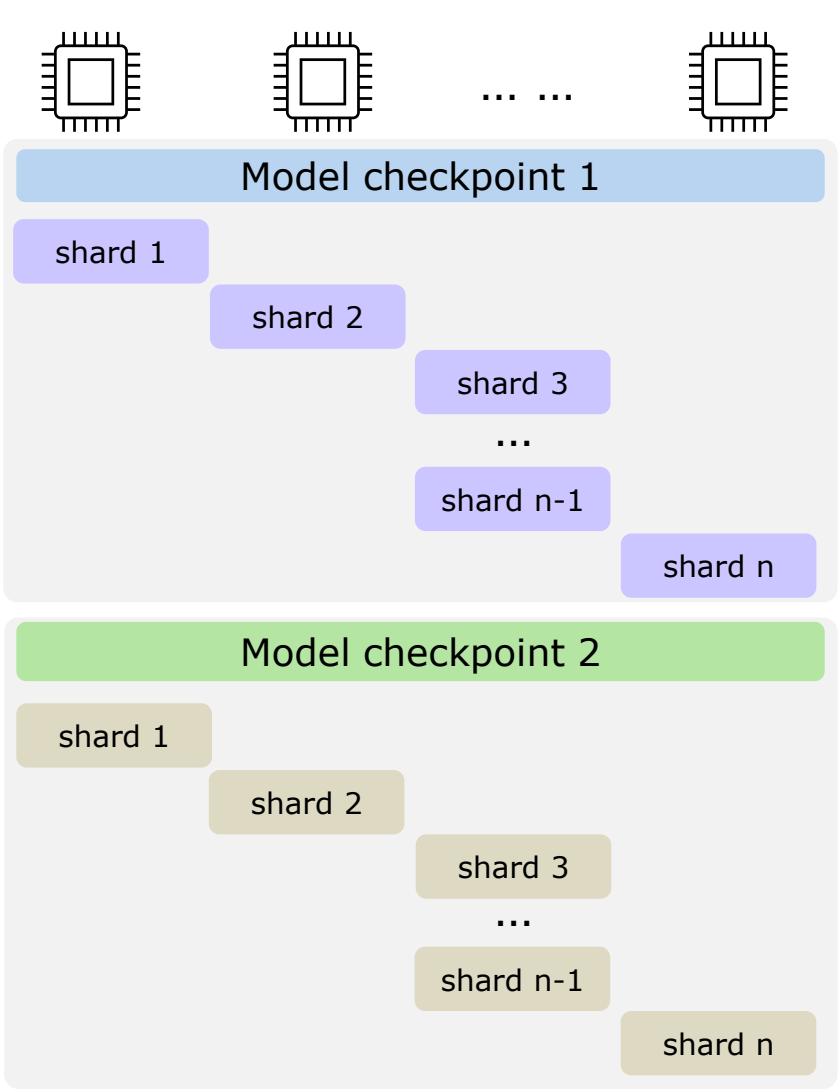


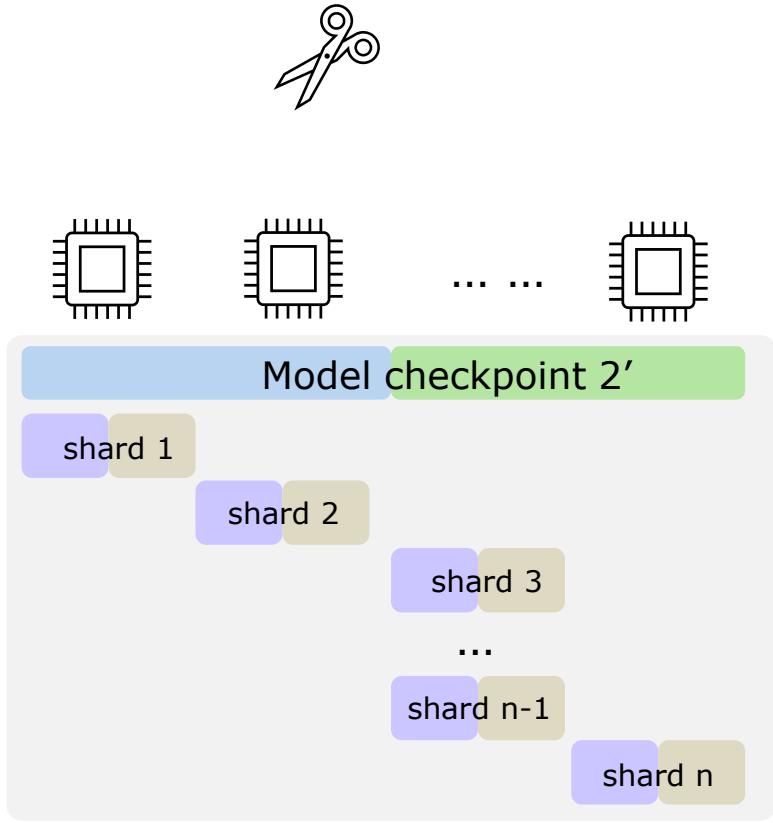
- All the layers are divisible and independent.

DIRLAB

Challenge 3: Distributed checkpoints when merging

Solution: Merge with node index





LLMTailor: Usage

- YAML-driven configuration (Inherited from mergekit)

```yaml
# YAML
slices:
- sources:
- model: /path_to_ckpt
layer_range: [0, 0]
…
- sources:
- model: //path_to_ckpt
layer_range: [34, 34]
…
pre_weights:
- name: model.norm.weight
model: /path_to_ckpt
```

- Simple setup

```bash
# bash
OUTPUT_PATH="/path"
CONFIG_YML="/path_to_yaml."
COPY_TOKENIZER=true
NUM_GPU=8
FAILURE_STEP=100
TOTAL_LAYERS=28
```

DIRLAB

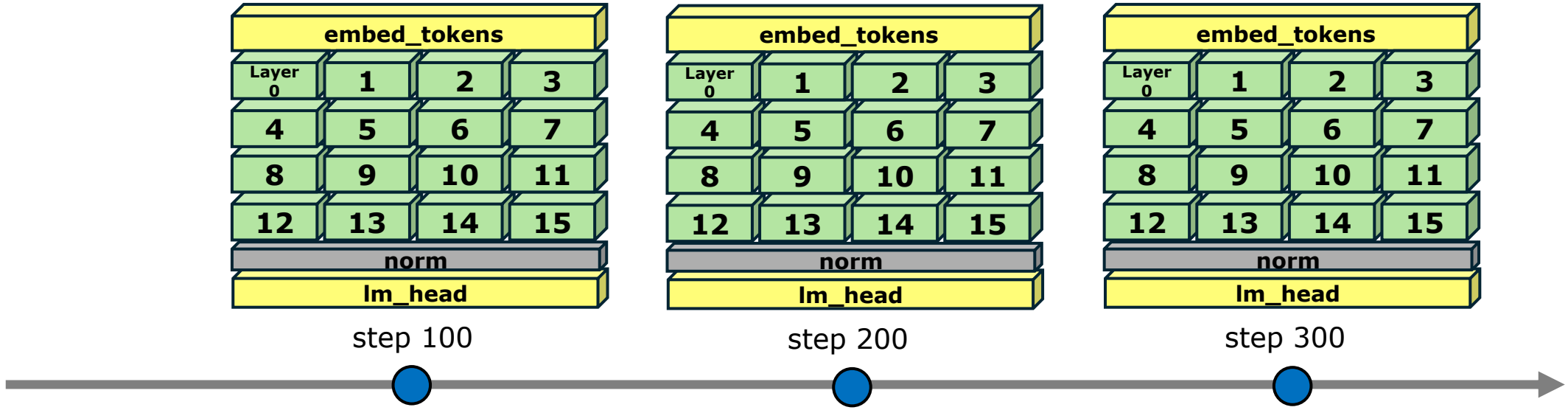
Setup

Configuration	**Supervised Fine-Tuning (SFT)**		**Continual Pre-Training (CPT)**	
GPUs	8 x A100			
Dataset	medical-dialogue-to-soap-summary **(Instruct Dataset)**		pubmed-summarization (**Plain Text**)	
Models	Llama3.1 8B	Qwen2.5 7B	Llama3.1 8B	Qwen2.5 7B

DIRLAB

Use case 1 : Parity Merge

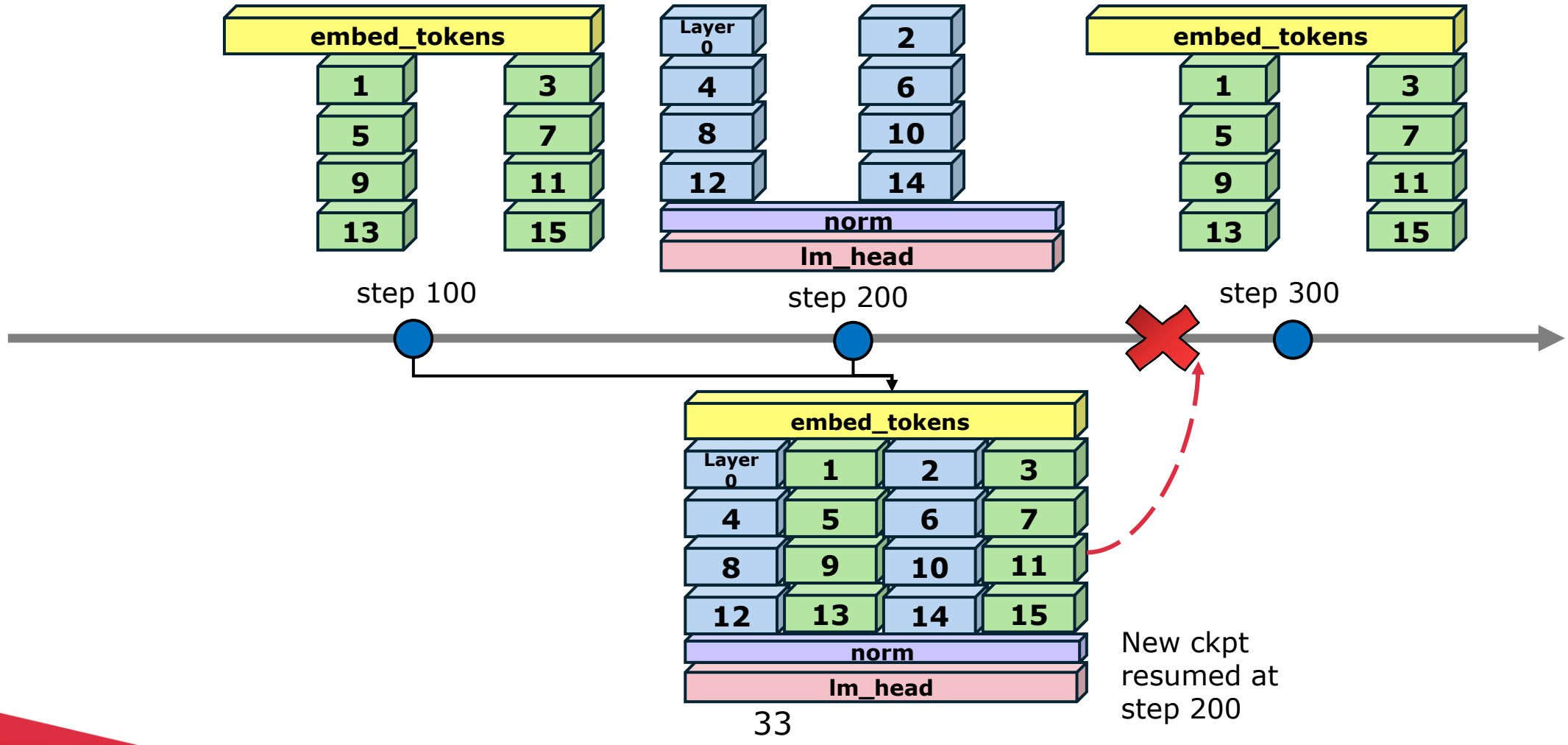
The original checkpoint logic: *save_step = 100*



DIRLAB

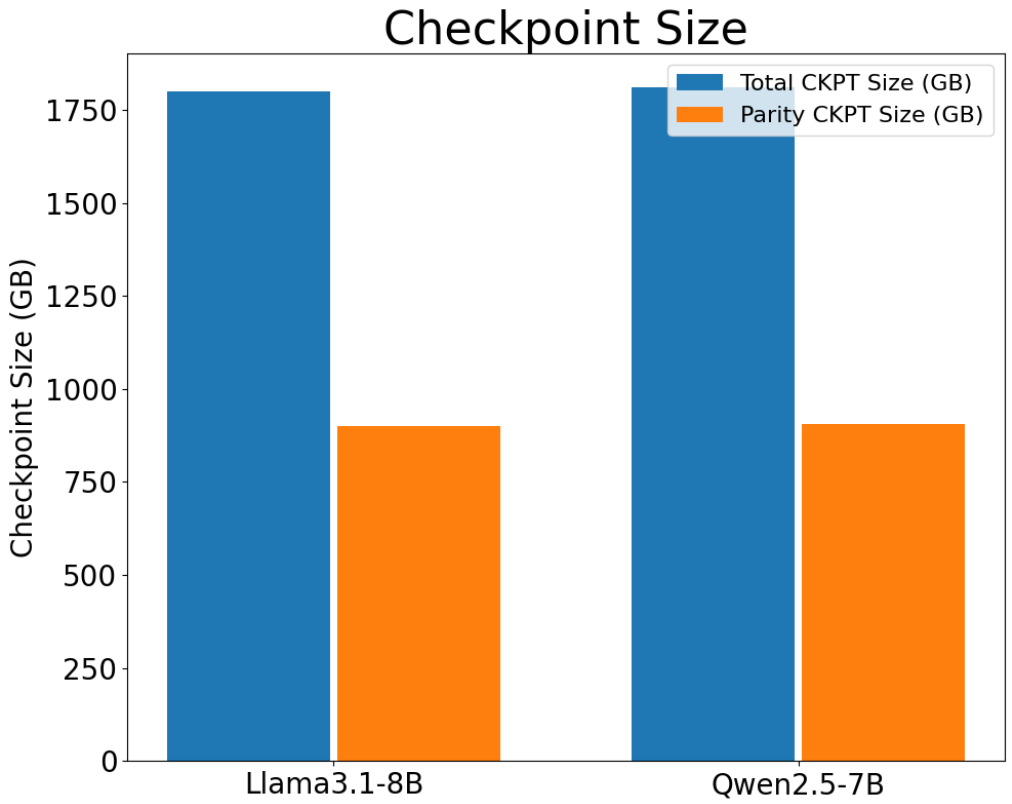
Use case 1 : Parity Merge

with LLMTailor:

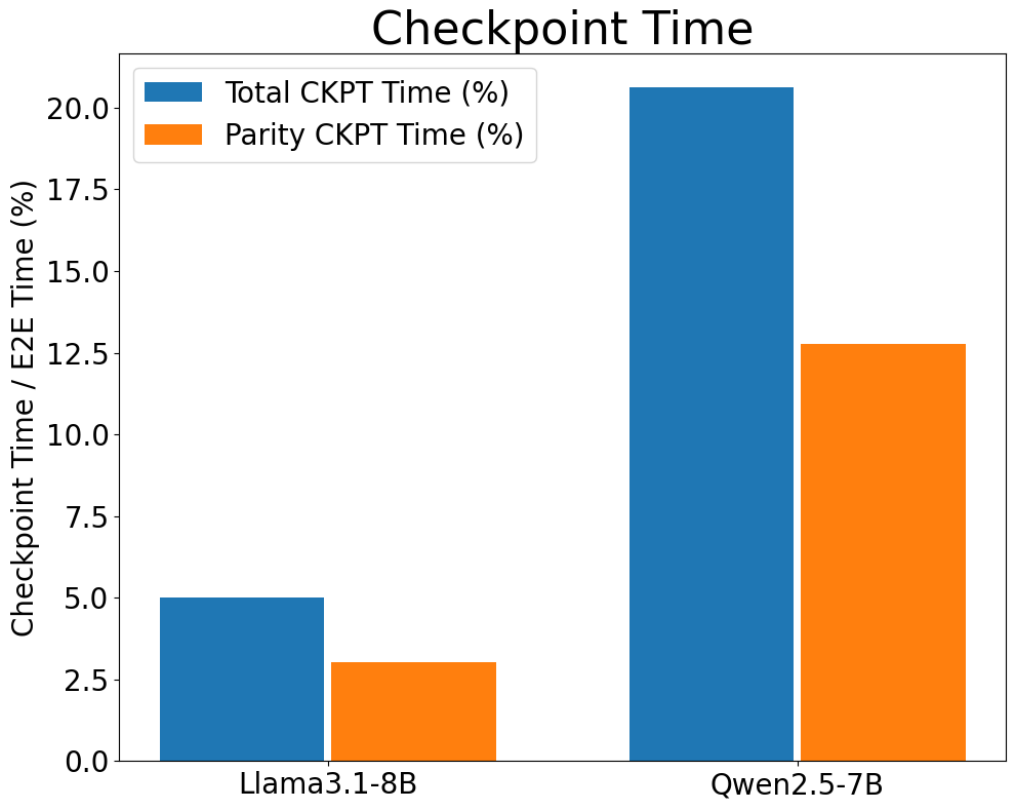


DIRLAB

1. Checkpointing Efficiency



× 2



× 1.6

DIRLAB

2. *Post-trained Model Quality*

We compare the model quality of the finally trained models resumed from different checkpointing methods:

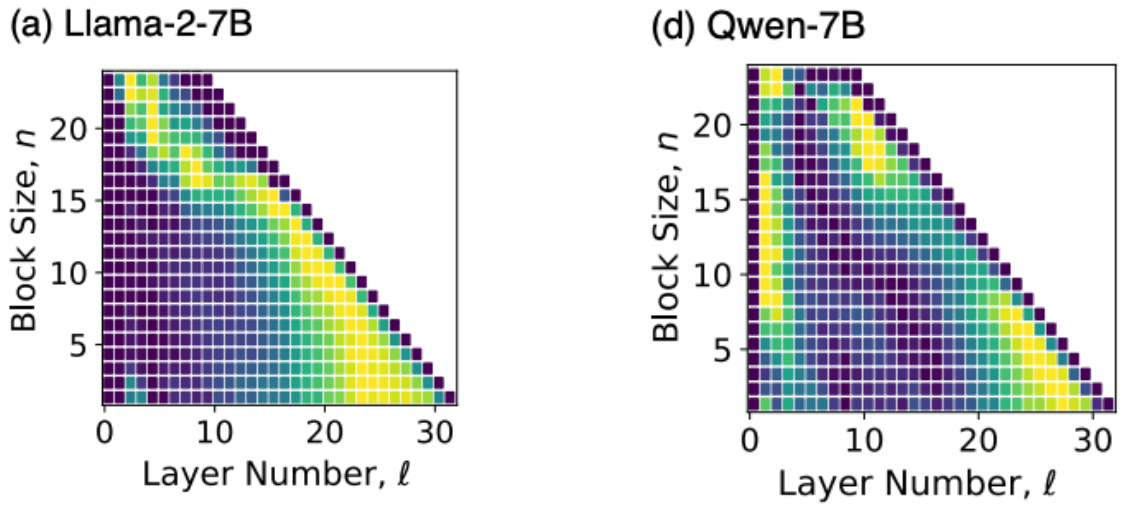
Task	Checkpoint method	MMLU	MMLUmed	MedMCQA	MedQA	PubMedQA
SFT	Default	73.14	89.00	60.75	64.02	75.20
	Parity merged	72.89	87.00	60.58	**64.10**	**76.20**
CPT	Default	60.00	75.00	53.10	55.15	77.20
	Parity merged	**60.03**	72.00	**53.12**	54.36	76.60

Benchmark Evaluation Results

DIRLAB

Use case 2: Filter layers

Previous work shows that the first several layers and the last two layers of the model have a greater impact on model reasoning.[1]

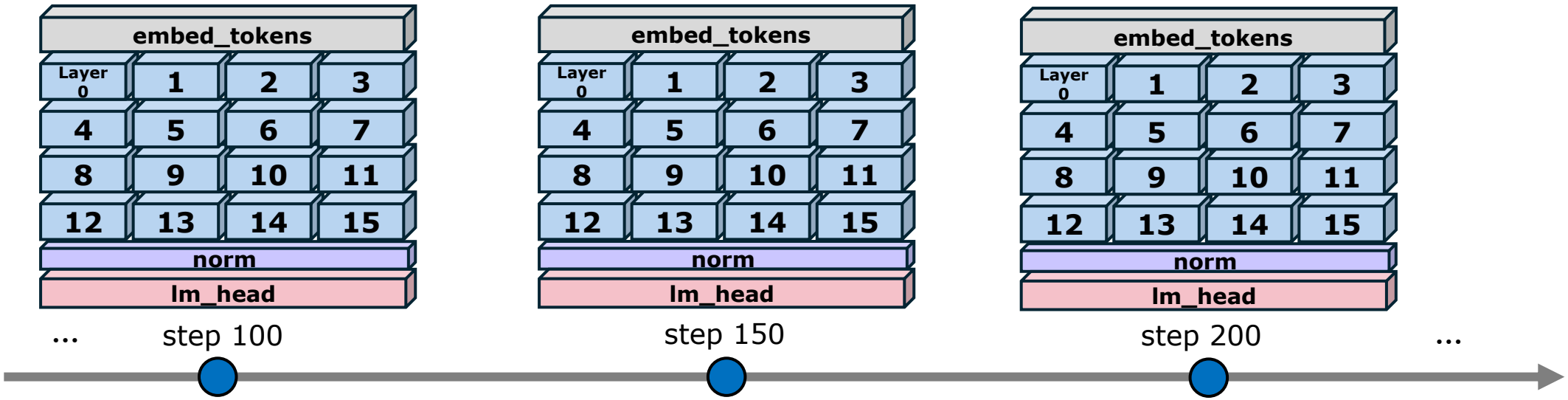


[1] Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. 2025. The Unreasonable Ineffectiveness of the Deeper Layers. arXiv:2403.17887 [cs.CL] https://arxiv.org/abs/2403.17887

DIRLAB

Use case 2: Filter layers

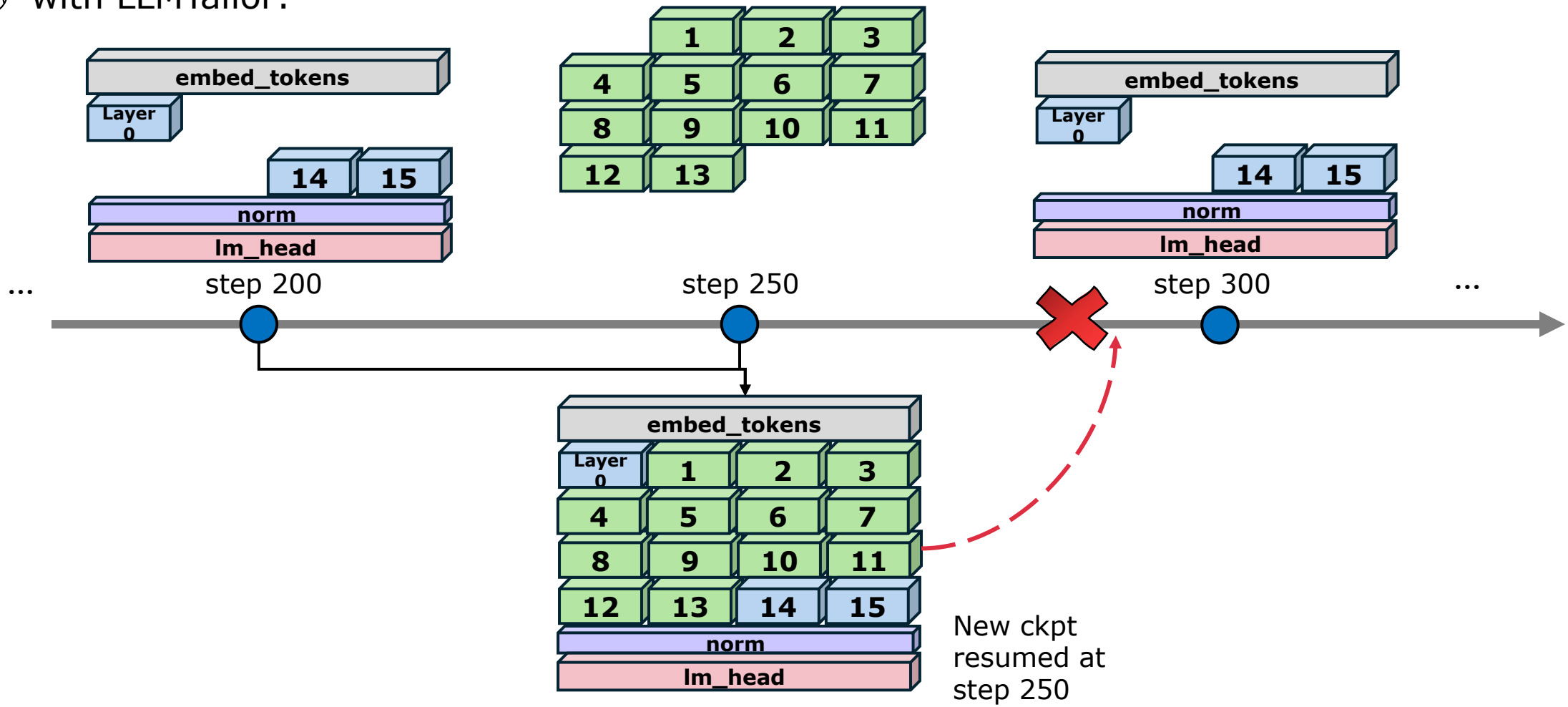
The original checkpoint logic: *save_step = 50*



DIRLAB

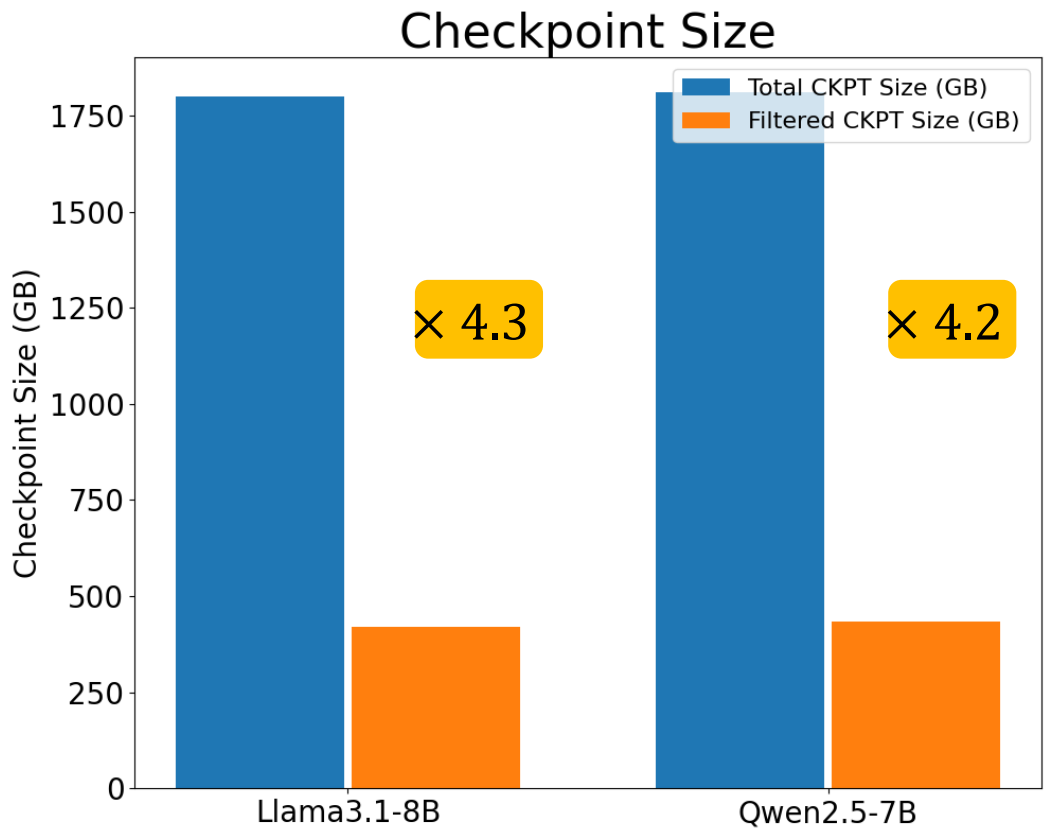
Use case 2: Filter layers

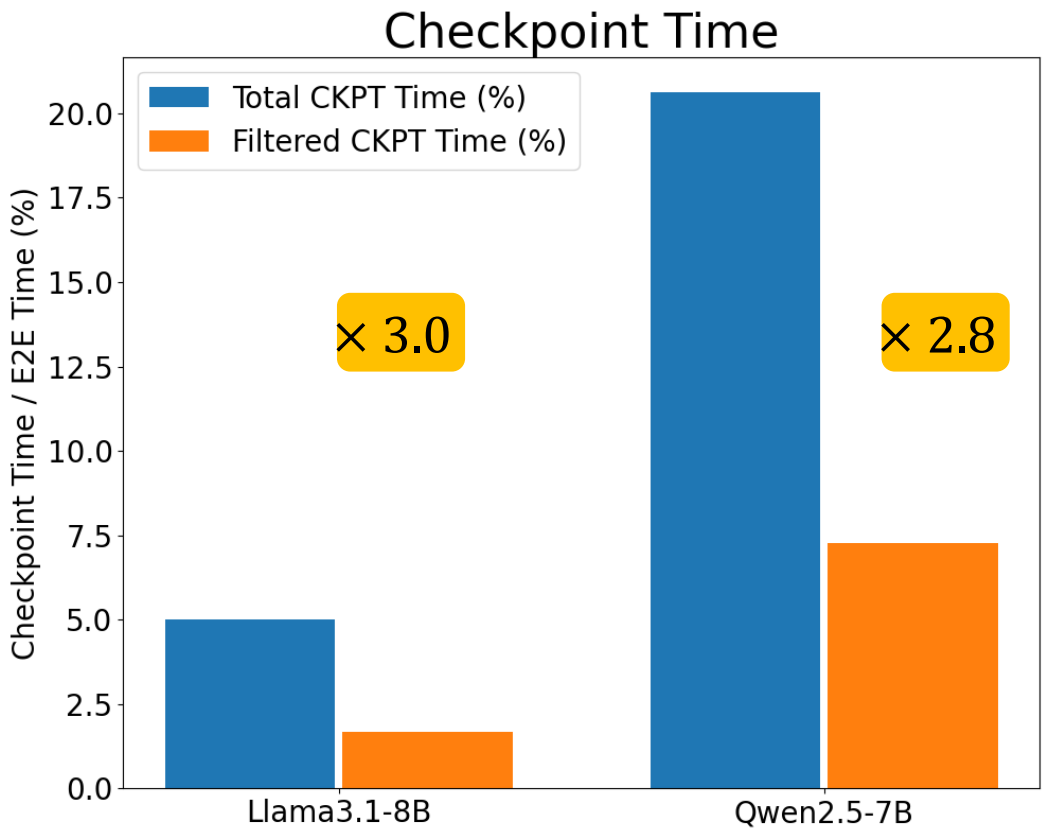
with LLMTailor:



DIRLAB

1. Checkpointing Efficiency





DIRLAB

2. *Post-trained Model Quality*

We compare the model quality of the finally trained models resumed from different checkpointing methods:

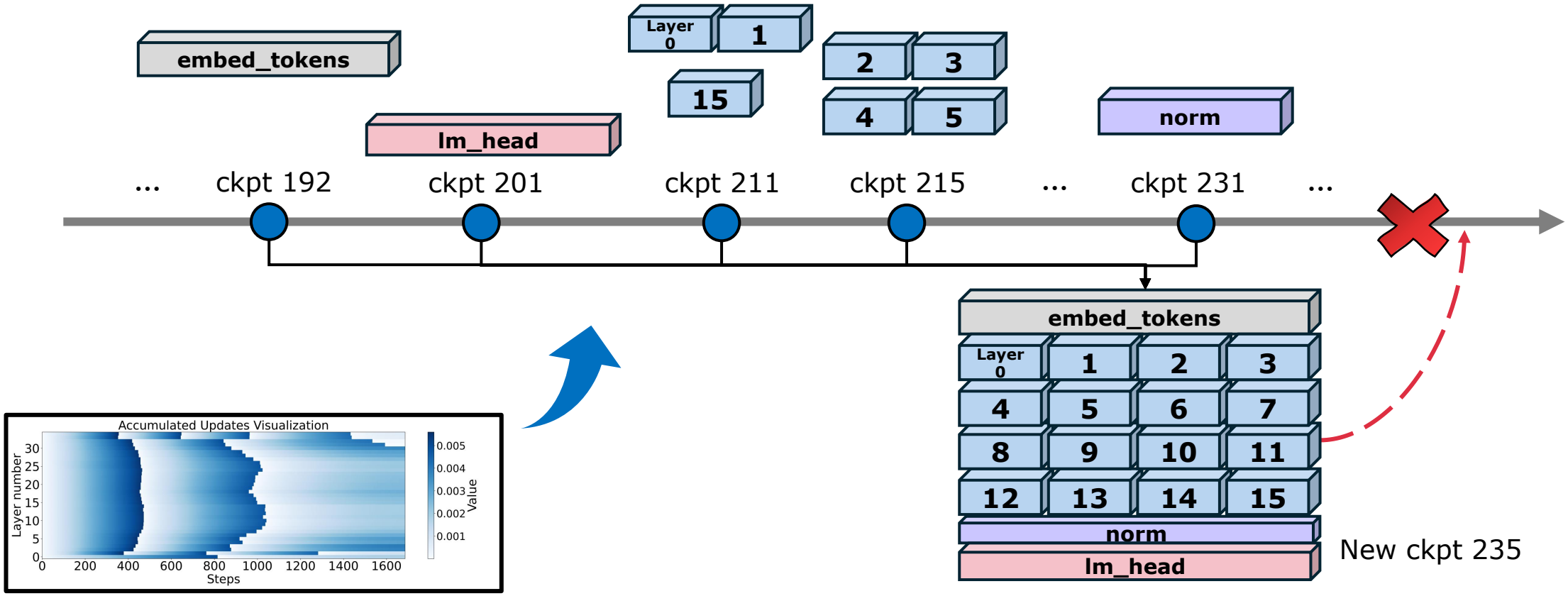
Task	Checkpoint method	MMLU	MMLUmed	MedMCQA	MedQA	PubMedQA
SFT	Default	73.14	89.00	60.75	64.02	75.20
	Parity merged	71.64	84.00	59.50	62.06	**75.60**
CPT	Default	60.00	75.00	53.10	55.15	77.20
	Parity merged	**62.06**	**77.00**	**53.45**	54.91	**78.00**

Benchmark Evaluation Results

DIRLAB

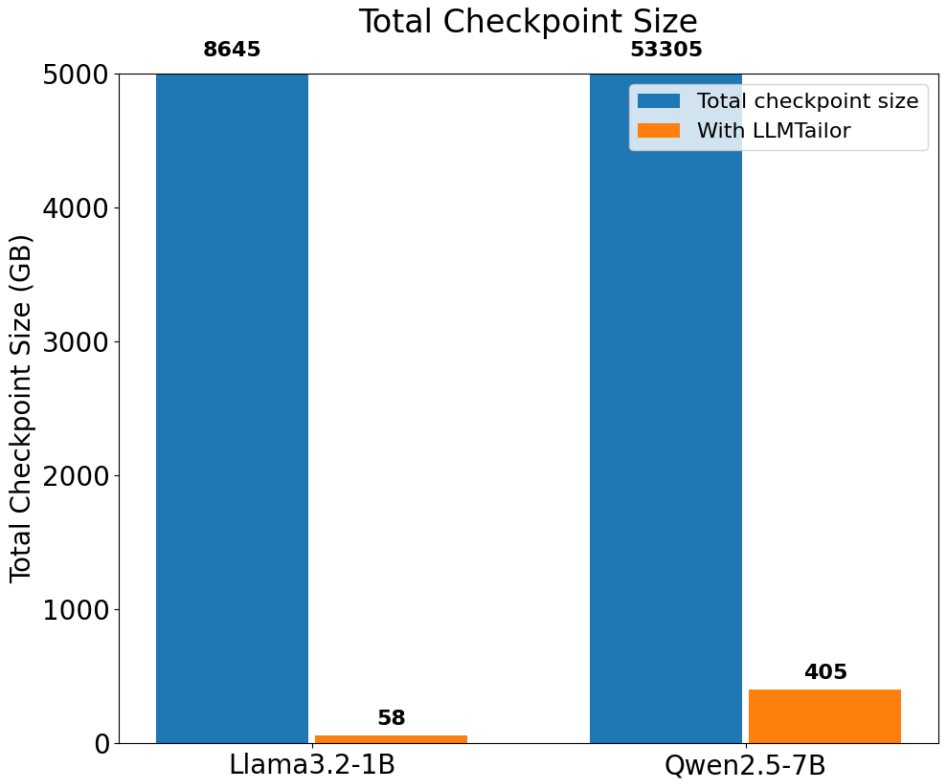
Ongoing: Tailoring by monitoring model updates

with LLMTailor:

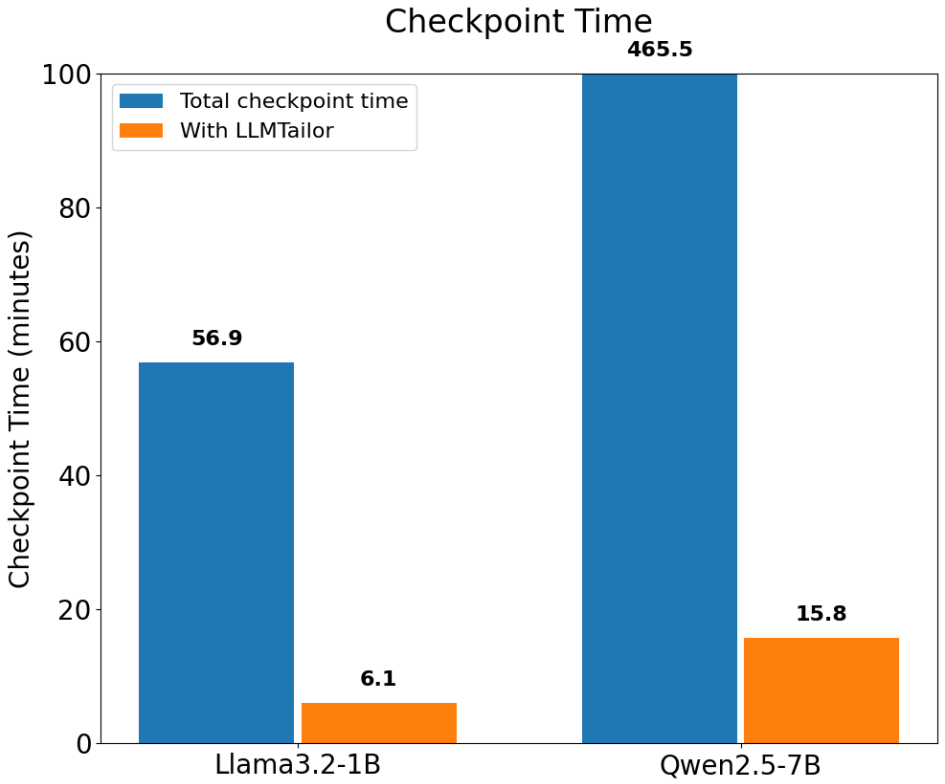


DIRLAB

1. Preliminary results: Checkpointing Efficiency



Up to × 149



Up to × 29

DIRLAB

2. *Preliminary results: Post-trained Model Quality*

We compare the model quality of the finally trained models resumed from different checkpointing methods:

Checkpoint method	Final train loss
Default	1.80
with LLMTailor	1.80

Qwen2.5-7B in an SFT task.

Task	Checkpoint method	MMLU	MMLUmed	MedMCQA	MedQA	PubMedQA
SFT	Default	71.68	88.00	59.74	62.14	74.40
	with LLMTailor	71.44	**89.00**	**59.77**	**63.00**	74.40

Benchmark Evaluation Results

LLMTailor

- Assembling a resumable "Frankenstein" checkpoint
- Substantially lower storage requirements & checkpoint time
- Adaptively checkpointing by monitoring model update

Minqiu Sun (mqsun@udel.edu)

https://zenodo.org/records/16909083

Github Repo

DIRLAB

Loading time of different method of merging

Model Name	Checkpoint Size (G)	Total layers	CKPTs included	Time (s)
Llama3-1B	17.29	18	Baseline: 1	0.80
			2	117
			parity (2)	233.6
			8	60.4
			18	62.5
Llama3-8B	112.47	35	Baseline: 1	16.8
			2	332.4
			parity (2)	1027.5
			8	279.2
			35	264.3

Engineering Issue