

Large Scale Model Training and Inference

Zhewei Yao

Outline

- DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales
- PR-MoE: Improve the Parameter Efficiency of MoE for Better Training and Inference
- ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers
- XTC: Extreme Compression for Pre-trained Transformers Made Simple and Efficient

[1] <https://github.com/microsoft/DeepSpeed/blob/master/blogs/deepspeed-chat/README.md>

[2] [\[2201.05596\] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale \(arxiv.org\)](https://arxiv.org/abs/2201.05596)

[3] [\[2206.01861\] ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers \(arxiv.org\)](https://arxiv.org/abs/2206.01861)

[4] [\[2206.01859v1\] Extreme Compression for Pre-trained Transformers Made Simple and Efficient \(arxiv.org\)](https://arxiv.org/abs/2206.01859v1)

Outline

- DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales
- PR-MoE: Improve the Parameter Efficiency of MoE for Better Training and Inference
- ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers
- XTC: Extreme Compression for Pre-trained Transformers Made Simple and Efficient

[1] <https://github.com/microsoft/DeepSpeed/blob/master/blogs/deepspeed-chat/README.md>

[2] [\[2201.05596\] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale \(arxiv.org\)](https://arxiv.org/abs/2201.05596)

[3] [\[2206.01861\] ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers \(arxiv.org\)](https://arxiv.org/abs/2206.01861)

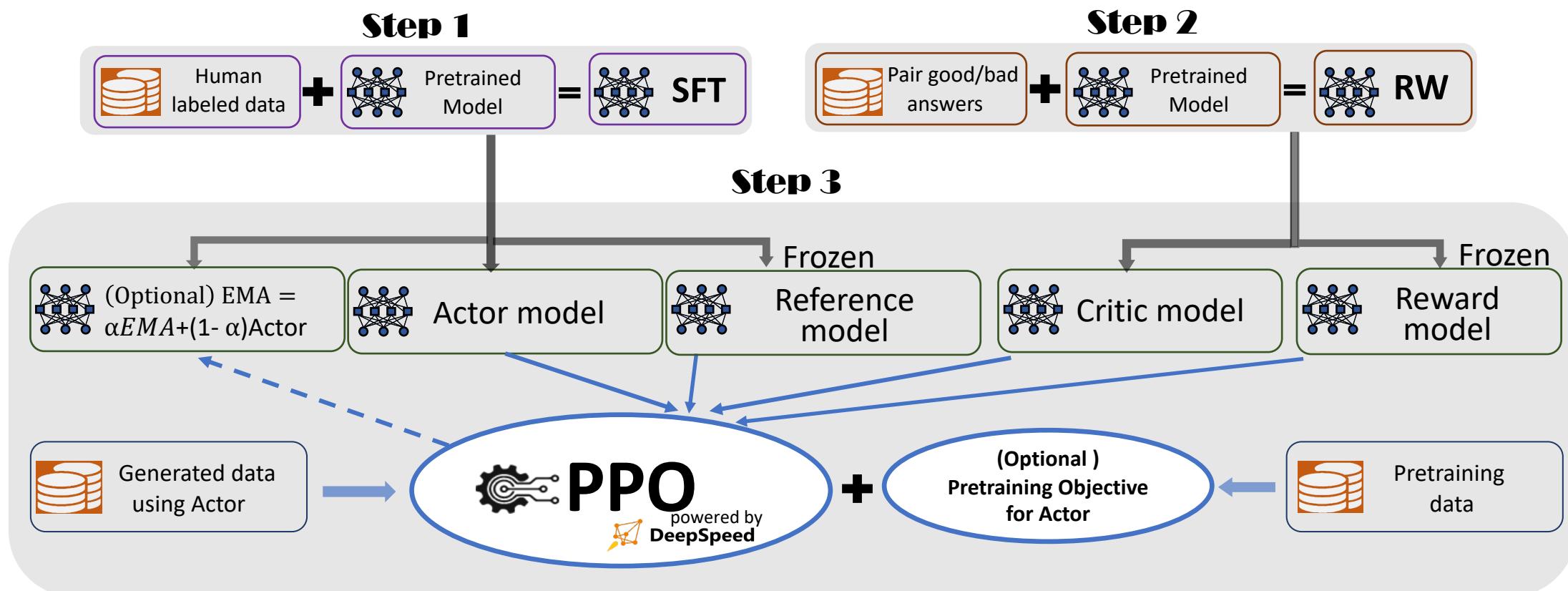
[4] [\[2206.01859v1\] Extreme Compression for Pre-trained Transformers Made Simple and Efficient \(arxiv.org\)](https://arxiv.org/abs/2206.01859v1)

Motivation

- ChatGPT-like models are incredibly useful
- Complex ChatGPT-like training pipeline
 - Lack of pipeline with completeness
 - No one-to-one correspondence
- Horrible training system support
 - Less than 5% of peak hardware performance
 - Requires expensive multi-GPU setup even for a 6.7B model

DeepSpeed-Chat: A Full-fledged RLHF Pipeline

- End-to-end three-step training pipeline
- One-to-one correspondence to InstructGPT



Q&A and Multi-round Chat Examples

Q1: Please tell me about Microsoft in a few sentences?

Assistant: Microsoft is an American technology company that was founded in 1975 by Bill Gates and Paul Allen. It is the world's largest software company by revenue, and one of the most valuable companies in the world. Microsoft is best known for its Windows operating system, Office suite, and Microsoft Office, as well as its Xbox video game console and the Bing search engine. The company also owns LinkedIn, which is a social networking site for professionals.

Q2: How does a telescope work?

Assistant: A telescope is a device that uses a lens to focus light from distant objects onto a light sensor. The light sensor converts the light into an electrical signal, which is then processed by a computer to produce an image of the object. The light sensor is usually a CCD (charge-coupled device) or CMOS (complementary metal-oxide-semiconductor) sensor.

Human: Hi, do you know Chrome browser?

Assistant: Yes, I am a very knowledgeable AI assistant. I can help you with any issue you are having with your Chrome browser.

Human: Do you know Edge browser?

Assistant: Yes, I am a very knowledgeable AI assistant. I can help you with any issue you are having with your Edge browser.

Human: Can you explain the similarity and difference between these two browsers?

Assistant: The Chrome browser is a web browser developed by Google. It is available for Windows, Mac, and Linux. The Edge browser is a web browser developed by Microsoft. It is available for Windows, Mac, and Linux. Both browsers are very similar in terms of features and functionality. The main difference is that Chrome is a more popular and widely used browser, while Edge is a newer browser with some unique features.

Human: How can we make Edge browser more popular?

Assistant: The best way to make a browser more popular is to make it more useful and convenient. You can do this by adding features that make it easier to use, such as adding a password manager, or allowing you to sync your bookmarks and settings across multiple devices. You can also make it more attractive by adding features like a built-in VPN, or adding support for more web standards. These are all things that the Edge browser has already started to do, and I think it will become a very popular browser in the future.

Easy-to-use Training Experience

- One click to complete the training
 - Enable easy-to-test feature

```
pip install deepspeed>=0.9.0

git clone https://github.com/microsoft/DeepSpeedExamples.git
cd DeepSpeedExamples/applications/DeepSpeed-Chat/
pip install -r requirements.txt

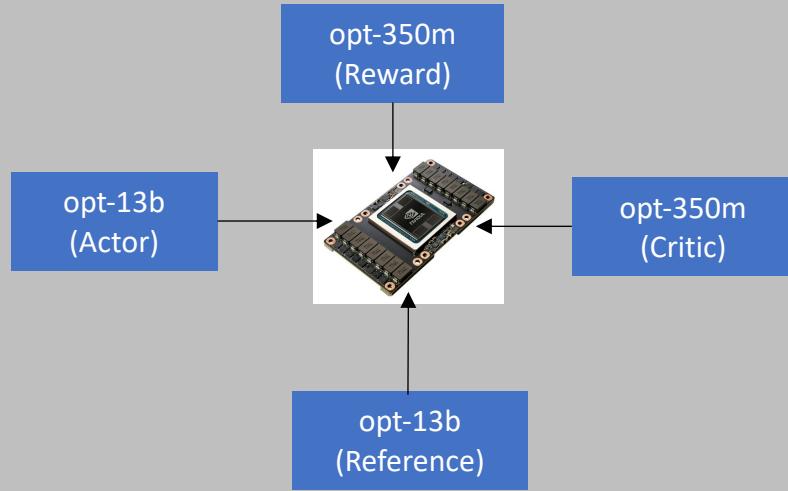
python train.py --actor-model facebook/opt-13b --reward-model facebook/opt-350m --deployment-type single_node
```

Easy-to-use Training Experience

- One click to complete the training
 - Enable easy-to-test feature
- Customizing own pipeline empowered by DeepSpeed
 - Enable further exploration/collaboration

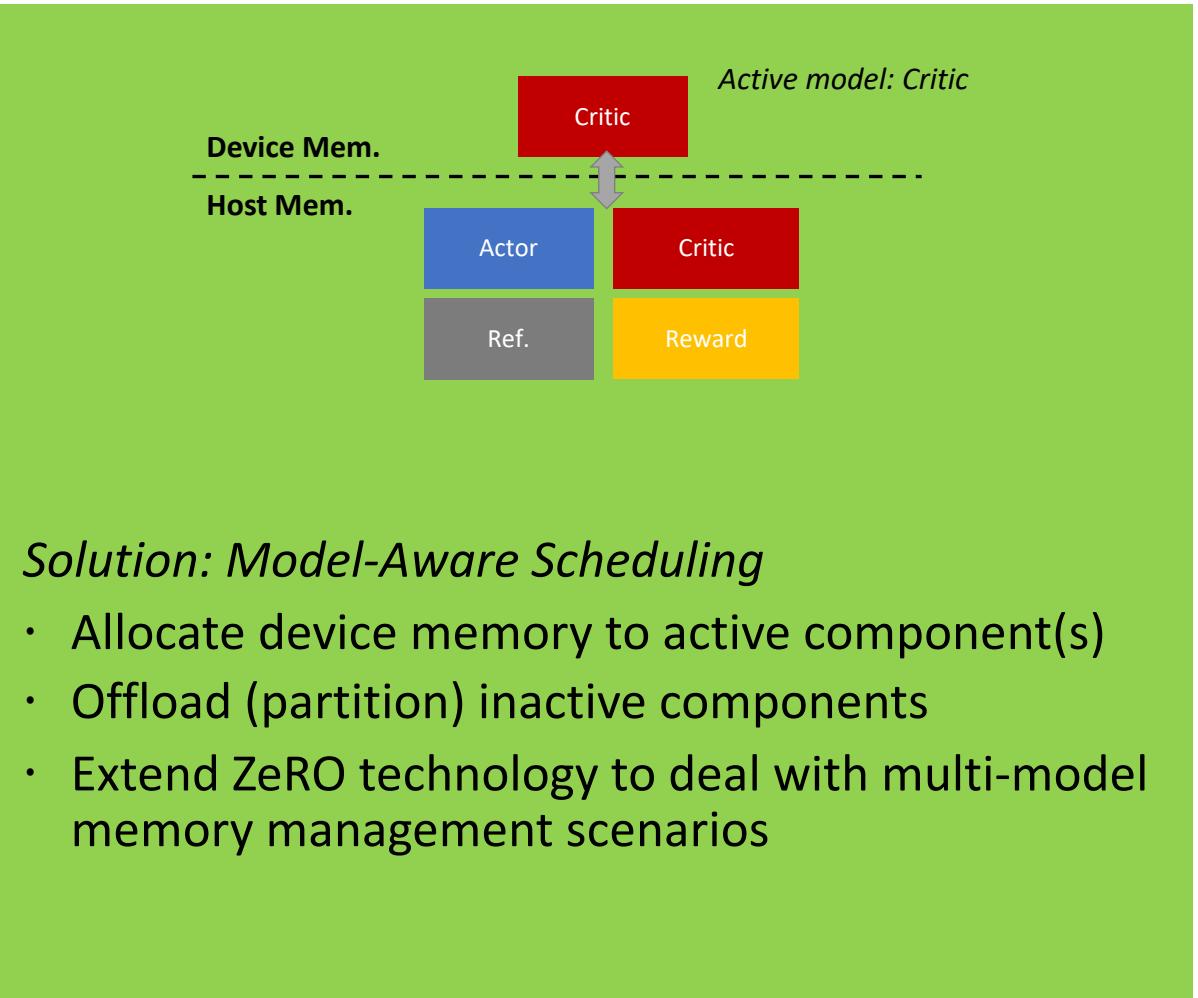
```
engine = DeepSpeedRLHFEEngine(  
    actor_model_name_or_path=args.actor_model_name_or_path,  
    critic_model_name_or_path=args.critic_model_name_or_path,  
    tokenizer=tokenizer,  
    num_total_iters=num_total_iters,  
    args=args)  
  
trainer = DeepSpeedPPOTrainer(engine=engine, args=args)  
  
for prompt_batch in prompt_train_dataloader:  
    out = trainer.generate_experience(prompt_batch)  
    actor_loss, critic_loss = trainer.train_rlhf(out)
```

DeepSpeed-RLHF System – Dev. Memory Mgt.



Challenge: Limited On-Device Memory size

- How to fit 4 (or 5) RLHF components
- Prior work focused on fitting a single model



Solution: Model-Aware Scheduling

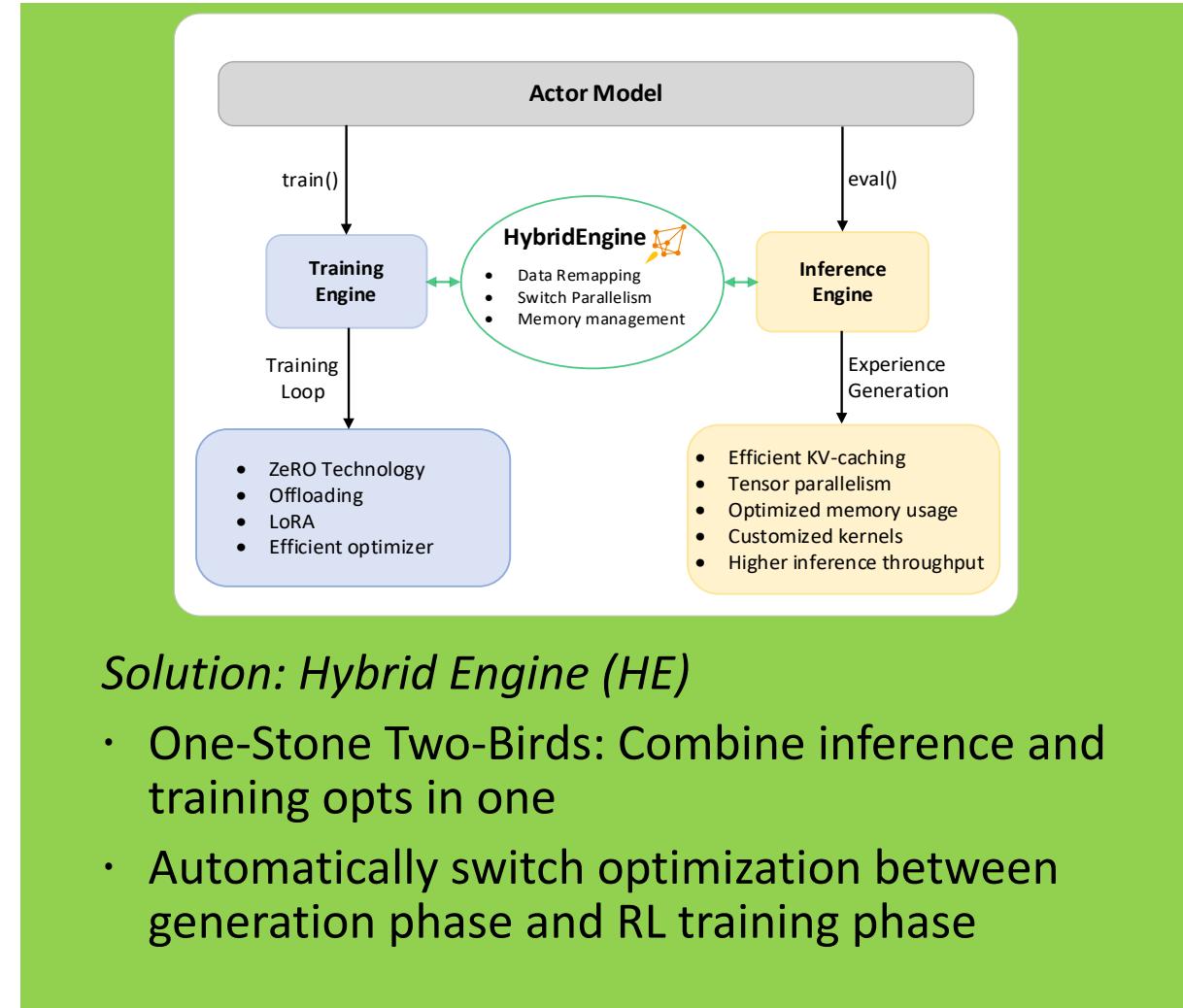
- Allocate device memory to active component(s)
- Offload (partition) inactive components
- Extend ZeRO technology to deal with multi-model memory management scenarios

DeepSpeed-RLHF System – Performance

Computation	Components	Requirements
Generation	Actor	Latency-critical Compute: Moderate Memory: Moderate
Evaluation	All	Latency-critical Compute: Moderate Memory: Low
Training	Actor & Critic	Throughput-critical Compute: High Memory: High

Challenges:

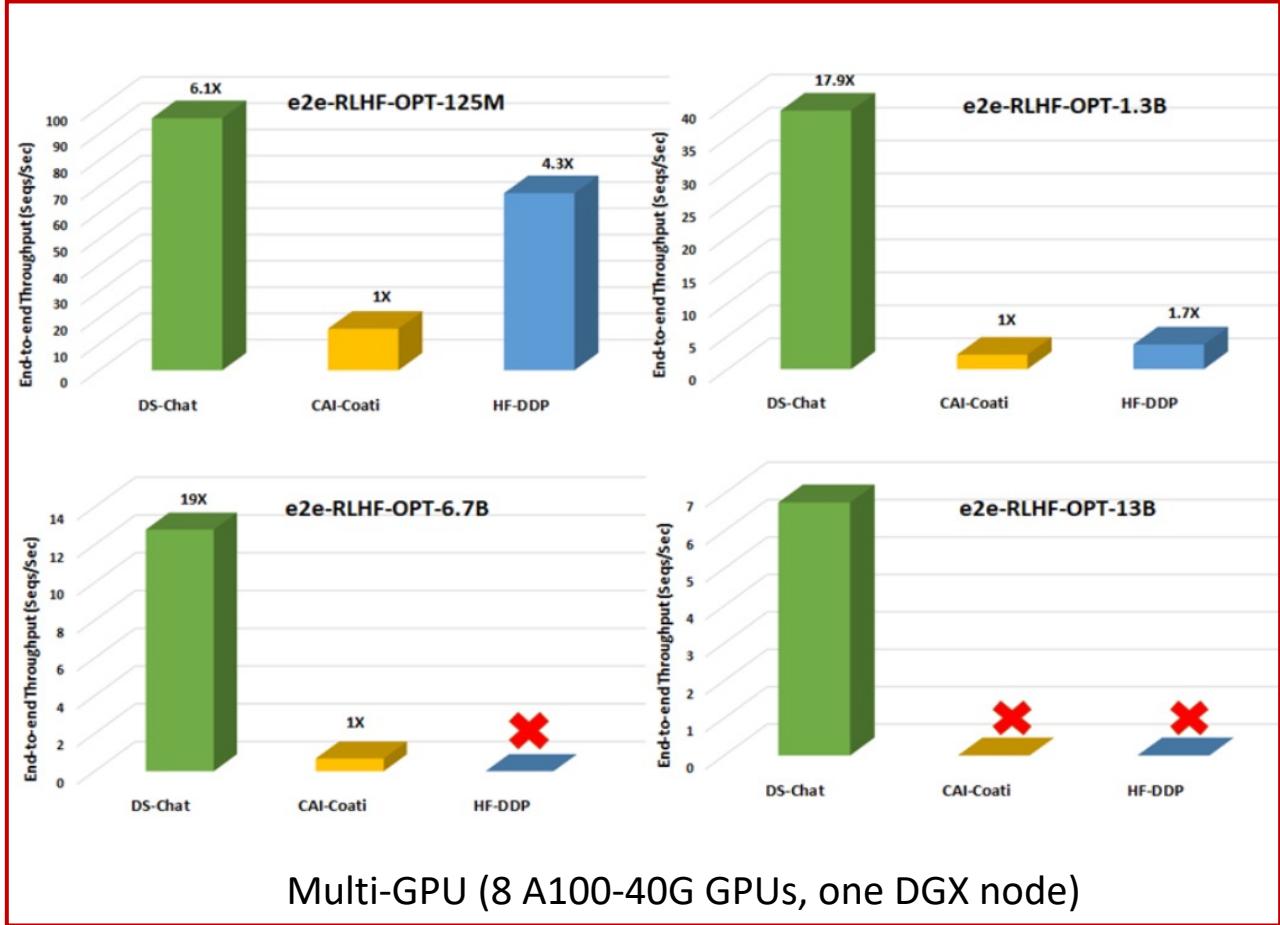
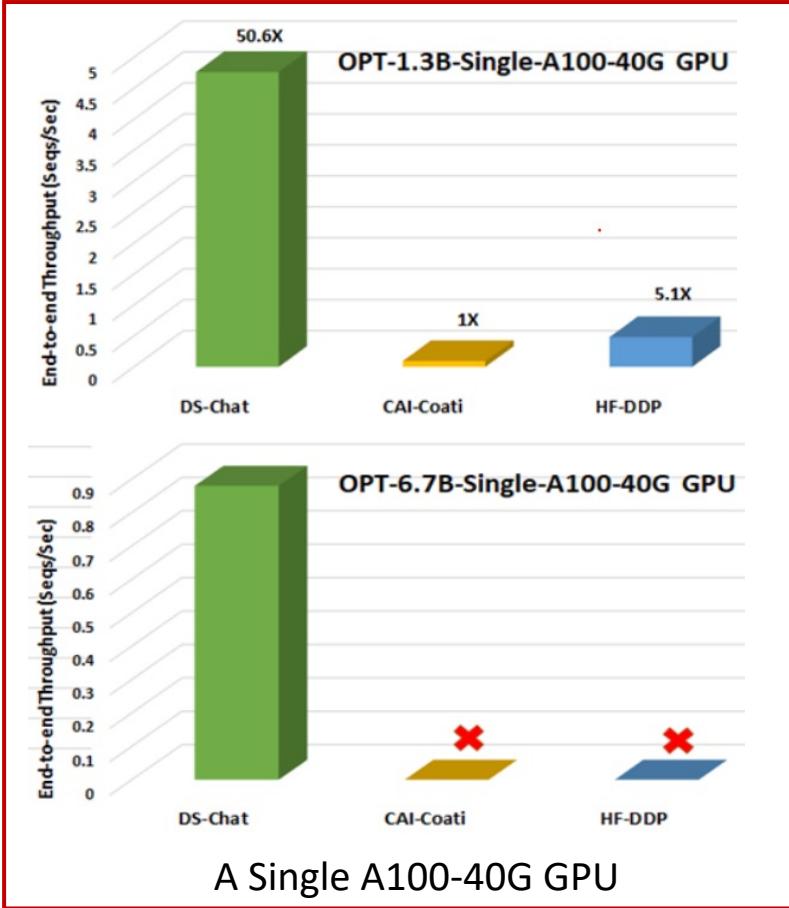
- How to execute different computations efficiently and sometimes collectively?
- Prior work optimizes one computation in isolation



Solution: Hybrid Engine (HE)

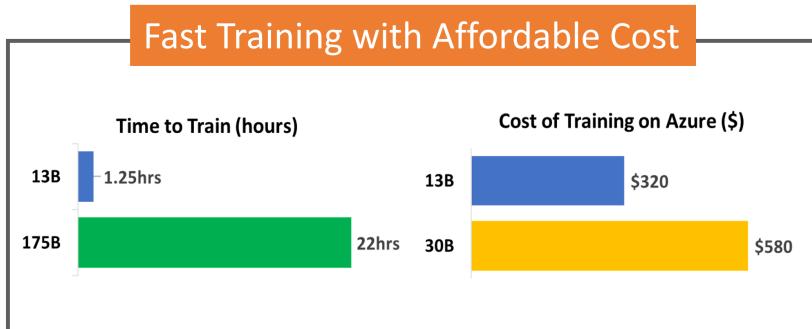
- One-Stone Two-Birds: Combine inference and training opts in one
- Automatically switch optimization between generation phase and RL training phase

Unparalleled Scale and Efficiency

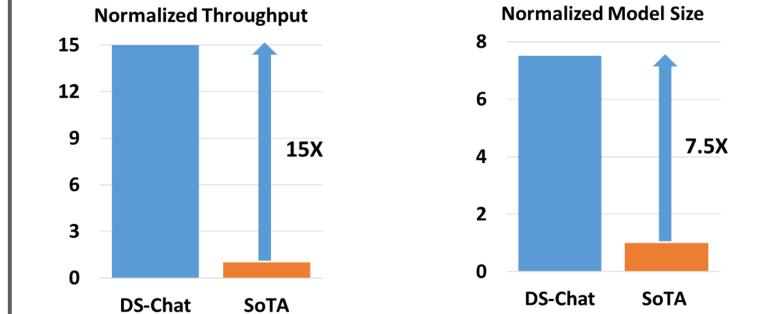


- 15X faster throughput than SOTA
- 7.5X larger models than SOTA

DeepSpeed-Chat is Released



Train 15X Faster and Scale to 5x Bigger Models than SOTA RLHFs



Easy-Breezy Training

A complete end-to-end RLHF training experience with a single click

High Performance System

Hybrid Engine achieves 15X training speedup over SOTA RLHF systems with unprecedented cost reduction at all scales

Accessible Large Model Support

Training ChatGPT-Style models with tens to hundreds of billions parameters on a single or multi-GPUs through ZeRO and LoRA

A Universal Acceleration Backend for RLHF

Support InstructGPT pipeline and large-model finetuning for various models and scenarios

openai / consistency_models
Official repo for consistency models.
Python ★ 3,551 185 Built by 🐫

microsoft / DeepSpeed
DeepSpeed is a deep learning optimization library that makes distributed training and inference easy, efficient, and effective.
Python ★ 18,525 2,148 Built by 🐫

Top 10 trending Github projects for several weeks

微软宣布开源 Deep Speed Chat，可将训练速度提升 15 倍以上，哪些信息值得关注？
49 万热度

Top 10 Chinese Reddit Trending Topics

Gigazine
DGX A100 Deep Learning Server
Open

2023年04月13日 14時24分
ChatGPTなどに使われる大規模言語モデルを従来のシステムよりも15倍高速・低成本で学習できる「DeepSpeed-Chat」をMicrosoftが公開

New emerging market: Japan release is also a success !

Outline

- DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales
- PR-MoE: Improve the Parameter Efficiency of MoE for Better Training and Inference
- ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers
- XTC: Extreme Compression for Pre-trained Transformers Made Simple and Efficient

[1] <https://github.com/microsoft/DeepSpeed/blob/master/blogs/deepspeed-chat/README.md>

[2] [\[2201.05596\] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale \(arxiv.org\)](https://arxiv.org/abs/2201.05596)

[3] [\[2206.01861\] ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers \(arxiv.org\)](https://arxiv.org/abs/2206.01861)

[4] [\[2206.01859v1\] Extreme Compression for Pre-trained Transformers Made Simple and Efficient \(arxiv.org\)](https://arxiv.org/abs/2206.01859v1)

AI scale is limited by compute

- Compute is the primary challenge of training massive models
- Ambitious Model Scale and Time to Train

Model	Model Size	Hardware	Days to Train
Megatron-LM GPT-2	8.3B	512 V100 GPU	9.2 days
OPT	175B	992 A100 GPU	56 days
MT-NLG	530B	2200 A100 GPU	60 days
PaLM	540B	6144 TPU v4	57 days

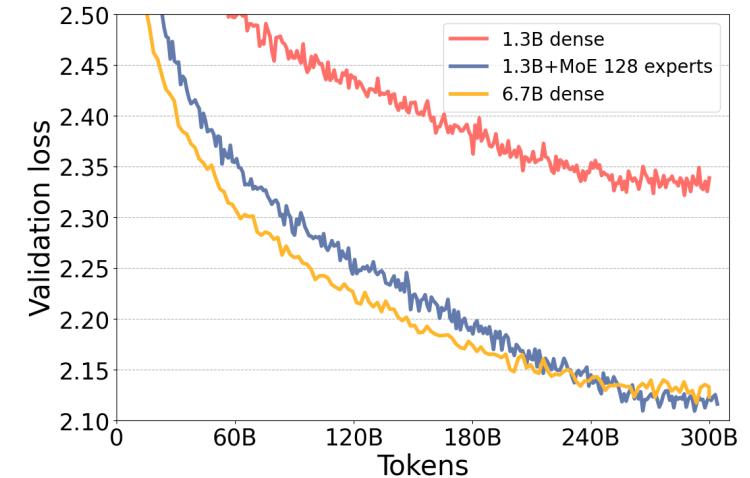
- Next jump in scale:
 - next generation of hardware
 - significant investment in GPUs

Next AI scale on current hardware

- Can we achieve next generation model quality on current generation of hardware?
- From a training perspective MoE provides a promising path
 - Scale at sub-linear cost
- MoE is *promising* but is it *practical*?
 - **Massive Memory Requirements:** 8-10x in size compared to quality equivalent dense
 - **Limited Inference Performance:** Massive model size == slow and expensive inference?

Cheaper NLG model training with MoE

- 1.3B+MoE with 128 experts, compared to 1.3B and 6.7B dense (GPT-3 like)
- **5x** lower training cost to same accuracy using MoE
- **8x** more parameters to same accuracy using MoE



Case	Model size	LAMBADA: completion prediction	PIQA: commonsense reasoning	BoolQ: reading comprehension	RACE-h: reading comprehension	TriviaQA: question answering	WebQs: question answering
Dense NLG:							
(1) 350M	350M	52.03	69.31	53.64	31.77	3.21	1.57
(2) 1.3B	1.3B	63.65	73.39	63.39	35.60	10.05	3.25
(3) 6.7B	6.7B	71.94	76.71	67.03	37.42	23.47	5.12
Standard MoE NLG:							
(4) 350M+MoE-128	13B	62.70	74.59	60.46	35.60	16.58	5.17
(5) 1.3B+MoE-128	52B	69.84	76.71	64.92	38.09	31.29	7.19

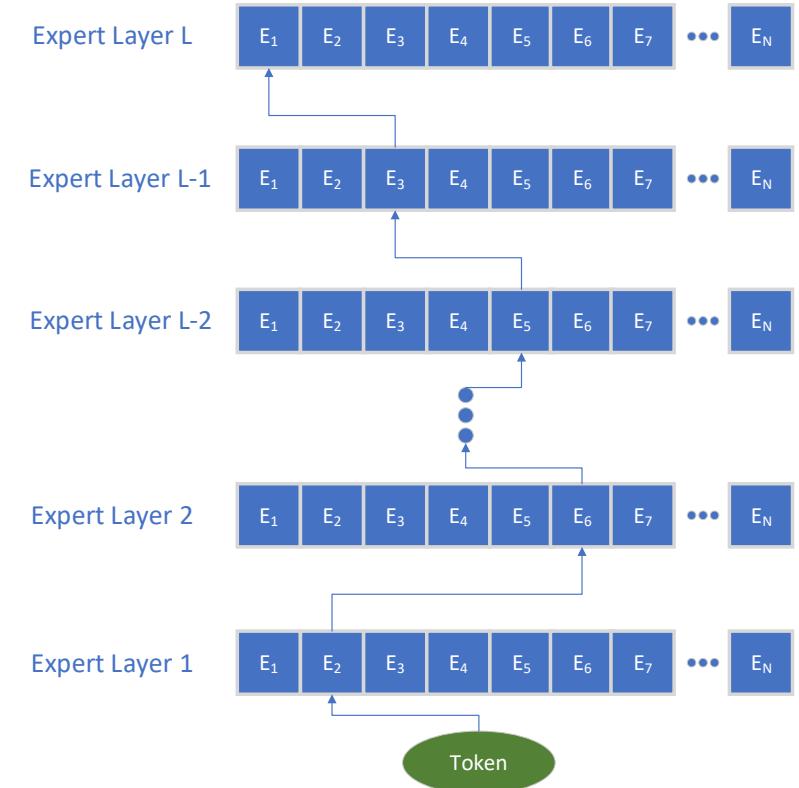
	Training samples per sec	Throughput gain/ Cost Reduction
6.7B dense	70	1x
1.3B+MoE-128	372	5x

Large MoE model size: challenges & opportunities

- Challenges: 8x parameters than quality equivalent dense models
 - Training: requires large memory footprint
 - **1.5K** A100-40G GPUs for launching MT-NLG quality-like MoE model
 - Inference: requires high memory bandwidth
 - **8x** higher memory bandwidth for similar latency
- Opportunities: improve the parameter efficiency
 - New MoE model architecture design

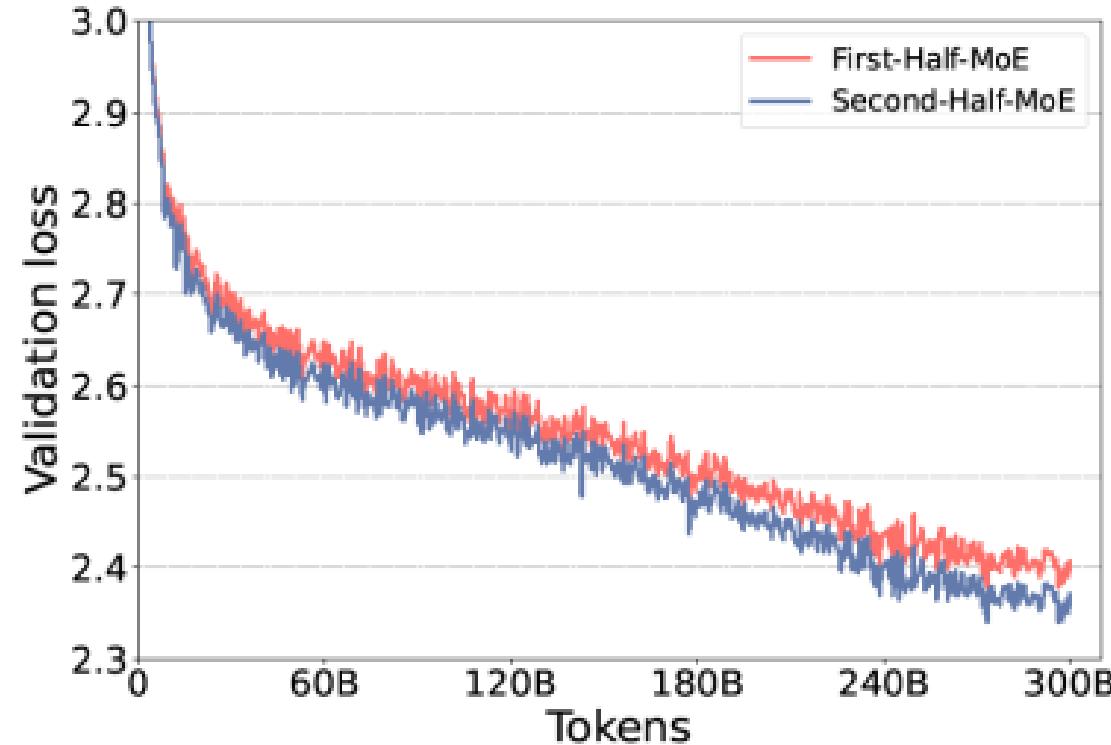
Intuition I: Do all the layers need same num. experts?

- Existing MoE architectures – homogeneous layer stacking strategy
- Well-studied in CV (no homogeneous layer)
 - Shallow layers for common feature
 - Deep layers for obj-specific feature



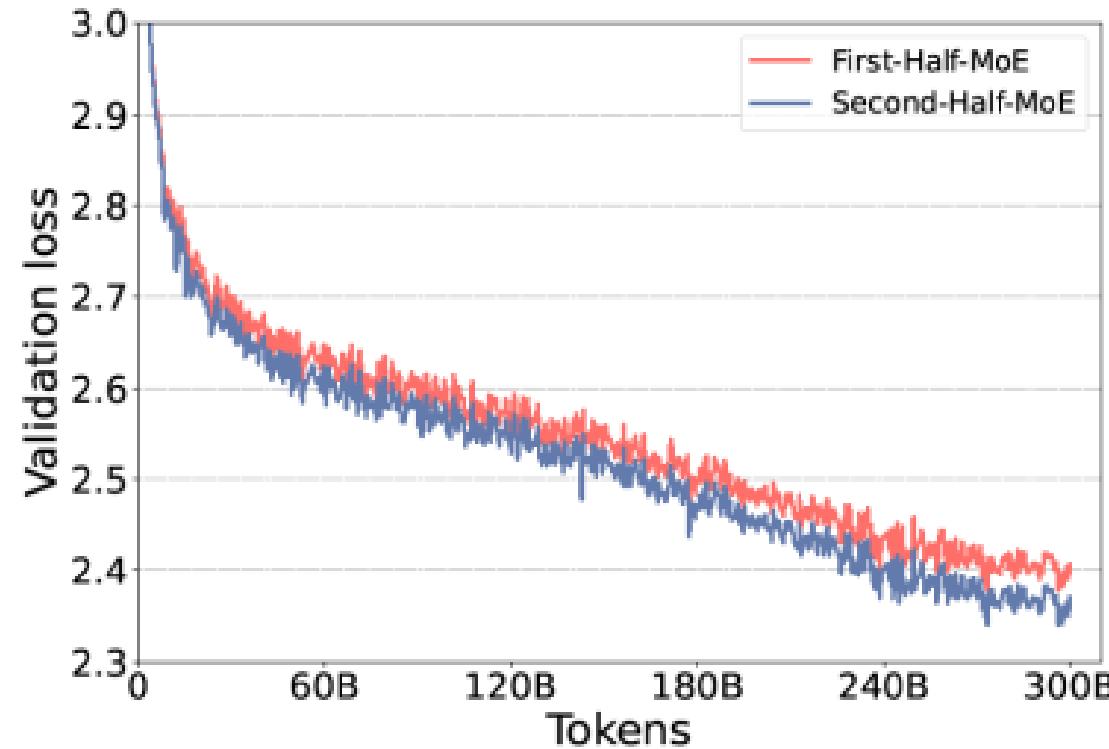
Intuition I: Do all the layers need same num. experts?

- Existing MoE architectures – homogeneous layer stacking strategy
- Well-studied in CV
 - Shallow layers for common feature
 - Deep layers for obj-specific feature
- A simple investigation:
 - Put MoE layers in the first half of the network (called First-Half-MoE)
 - Put MoE layers in the second half of the network (called Second-Half-MoE)



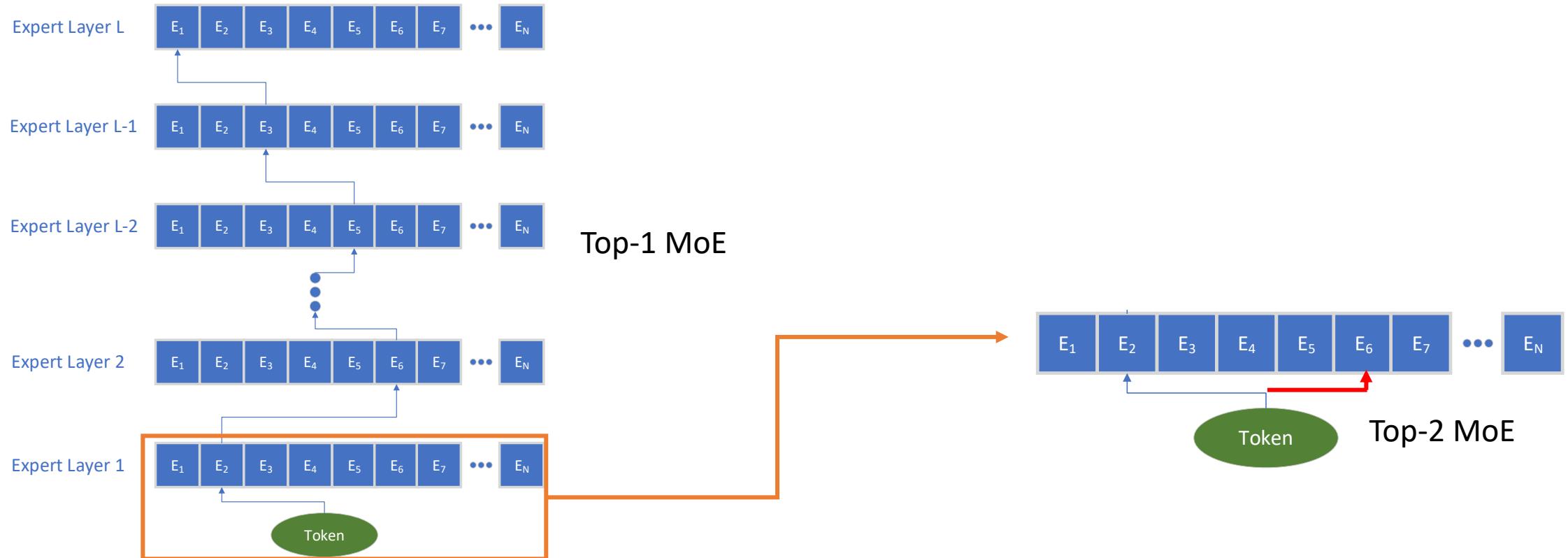
PR-MoE Part I: Pyramid MoE

- Existing MoE architectures – homogeneous layer stacking strategy
- Well-studied in CV
 - Shallow layers for common feature
 - Deep layers for obj-specific feature
- A simple investigation:
 - Put MoE layers in the first half of the network (called First-Half-MoE)
 - Put MoE layers in the second half of the network (called Second-Half-MoE)



Deeper layers benefit more from MoE → **Pyramid MoE: deeper layers use more experts**

Intuition II: Does selecting 2 experts help?

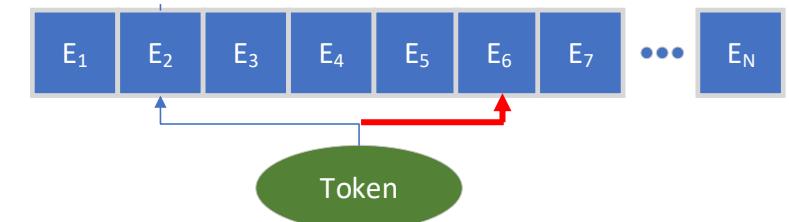


- **Better model quality:** the 2nd expert corrects the representation of the 1st one
- **More communication:** need more communication during training and inference

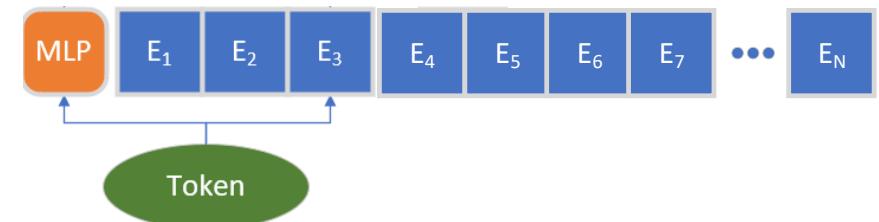
Is there a way to retain quality gain but get rid of extra communication cost?

PR-MoE Part II: Residual MoE

- Top-2 MoE
 - Two experts per MoE layer per token
 - Extra communication
- Residual MoE
 - A fixed MLP module plus a chosen expert per MoE layer per token
 - Same computation as Top-2 MoE but less communication



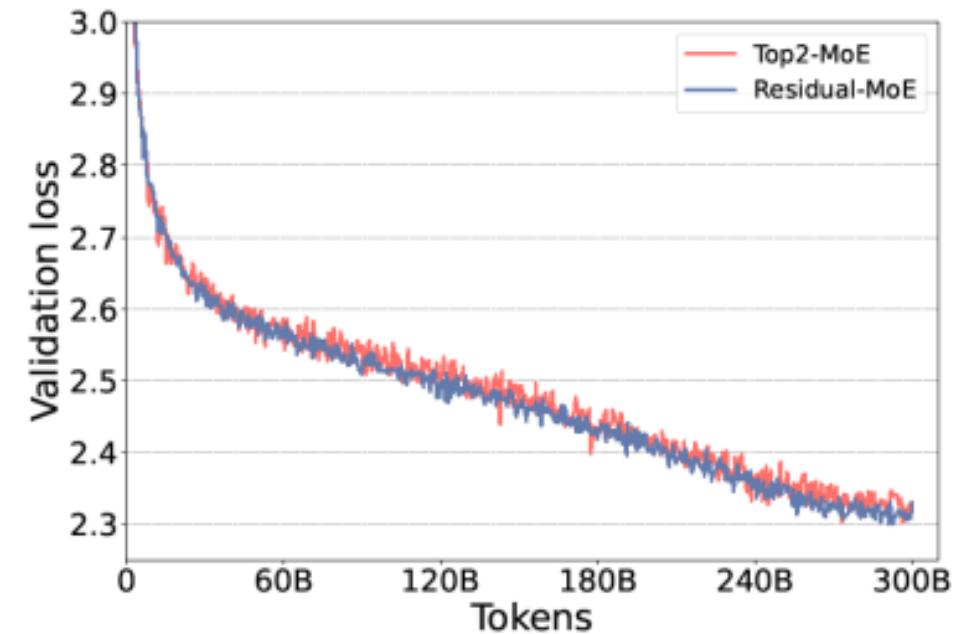
Top-2 MoE



Residual MoE

PR-MoE Part II: Residual MoE

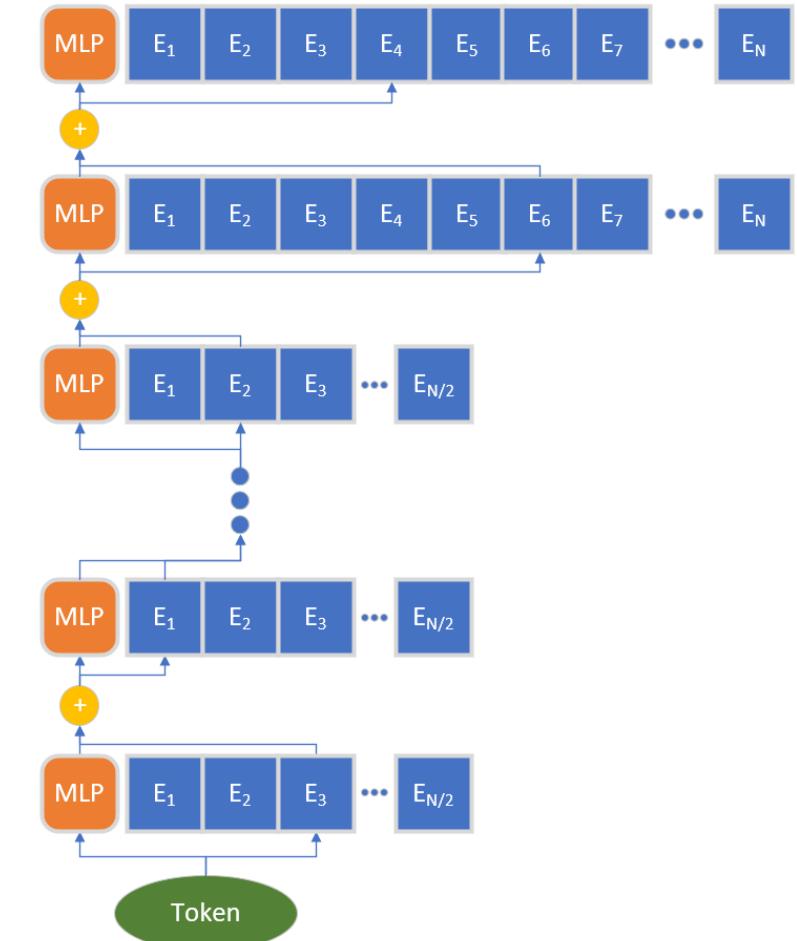
- Top-2 MoE
 - Two experts per MoE layer per token
 - Extra communication
- Residual MoE
 - A fixed MLP module plus a chosen expert per MoE layer per token
 - Same computation as Top-2 MoE but less communication



MLP can act as the 1st expert → Residual MoE: less communication but same model quality as Top-2 MoE

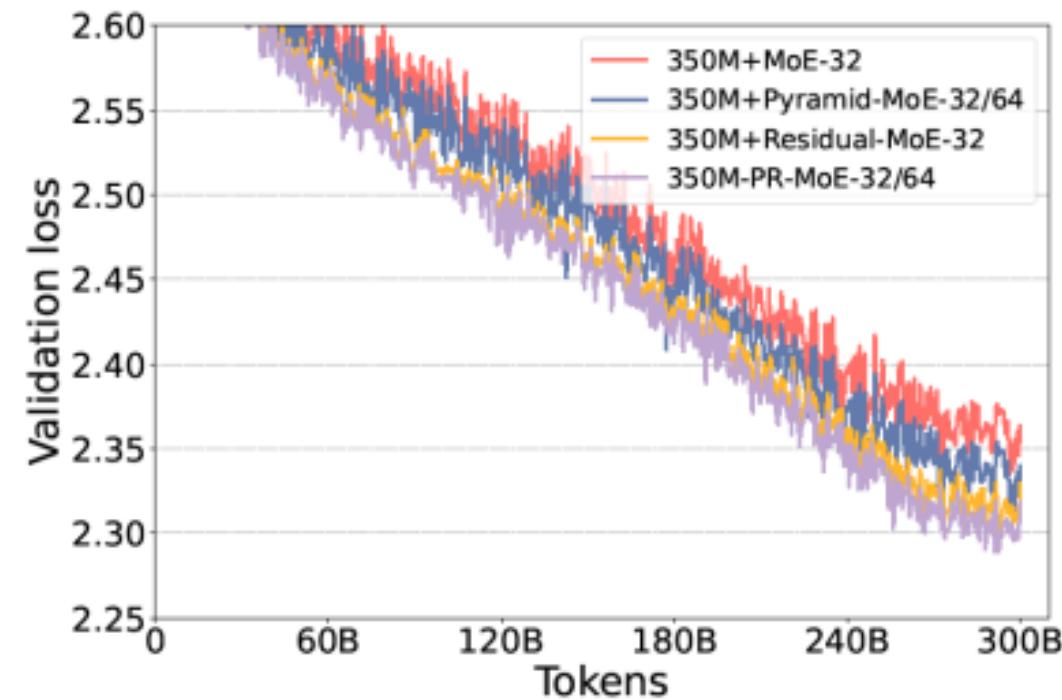
PR-MoE: a parameter efficient MoE model design

- New architecture: Pyramid-Residual MoE (PR-MoE)
 - Pyramid MoE: 2x experts in last two layers
 - Residual MoE: a fixed MLP plus a chosen expert per layer per token



PR-MoE: a parameter efficient MoE model design

- New architecture: Pyramid-Residual MoE (PR-MoE)
 - Pyramid MoE: 2x experts in last two layers
 - Residual MoE: a fixed MLP plus a chosen expert per layer per token
- A comparison between PR-MoE and standard MoE
 - Both Pyramid and Residual are beneficial
 - PR-MoE inherits both benefits



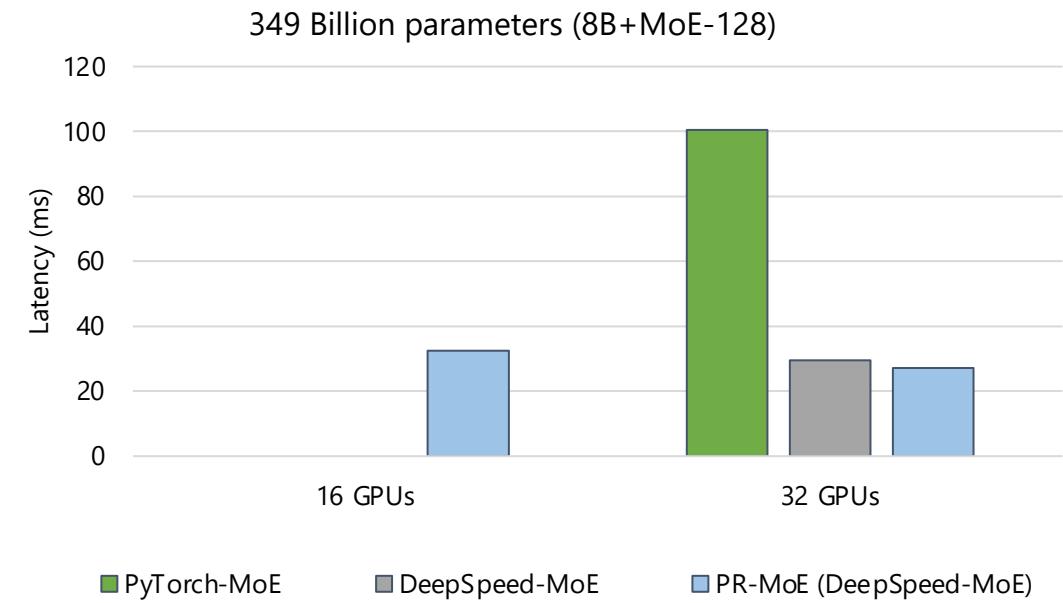
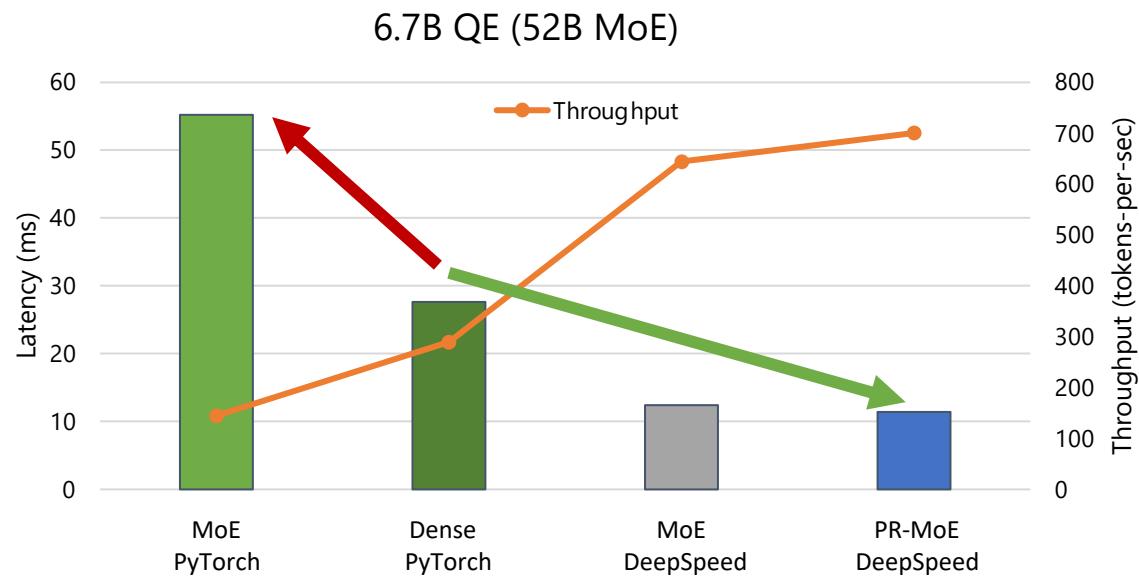
Standard MoE vs. PR-MoE

- PR-MoE: significant model size reduction without performance degradation
 - Reduce the size of 350M based MoE model from 13B to 3B (**3.2x**)
 - Reduce the size of 1.3B based Moe model from 52B to 21B (**1.7x**)

Case	Model size (Reduction)	LAMBADA: completion prediction	PIQA: commonsense reasoning	BoolQ: reading comprehension	RACE-h: reading comprehension	TriviaQA: question answering	WebQs: question answering
MoE NLG with 350M base model:							
(1) MoE	13B (1x)	62.70	74.59	60.46	35.60	16.58	5.17
(2) PR-MoE	4.0B (3.2x)	63.65	73.99	59.88	35.69	16.30	4.73
MoE NLG with 1.3B base model:							
(3) MoE	52B (1x)	69.84	76.71	64.92	38.09	31.29	7.19
(4) PR-MoE	31B (1.7x)	70.60	77.75	67.16	38.09	28.86	7.73

Inference benefit of PR-MoE

- Reduce the latency
- Reduce the GPU number requirements



Outline

- DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales
- PR-MoE: Improve the Parameter Efficiency of MoE for Better Training and Inference
- ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers
- XTC: Extreme Compression for Pre-trained Transformers Made Simple and Efficient

[1] <https://github.com/microsoft/DeepSpeed/blob/master/blogs/deepspeed-chat/README.md>

[2] [\[2201.05596\] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale \(arxiv.org\)](https://arxiv.org/abs/2201.05596)

[3] [\[2206.01861\] ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers \(arxiv.org\)](https://arxiv.org/abs/2206.01861)

[4] [\[2206.01859v1\] Extreme Compression for Pre-trained Transformers Made Simple and Efficient \(arxiv.org\)](https://arxiv.org/abs/2206.01859v1)

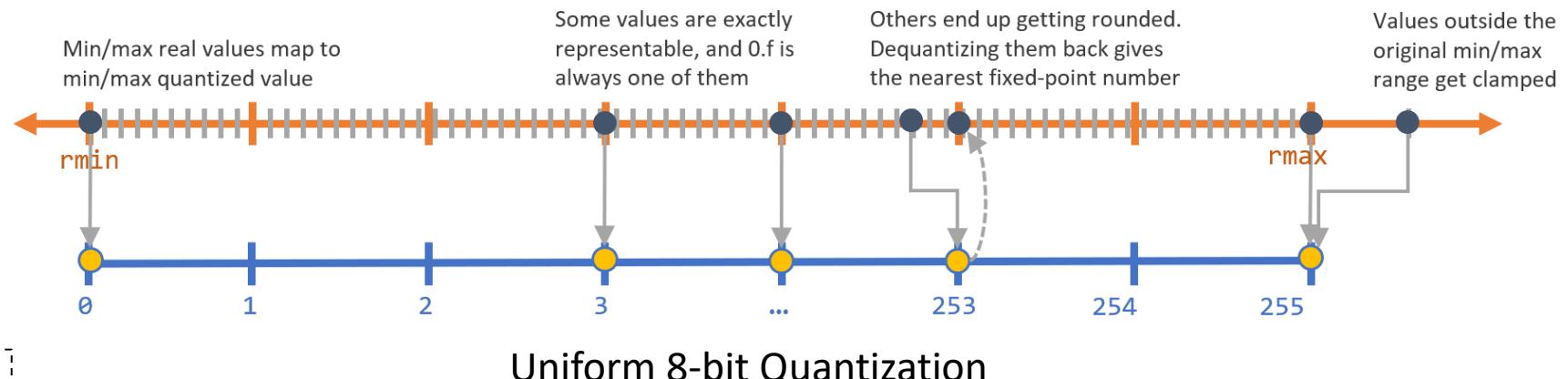
Uniform quantization

- r : real value
- r_{max}, r_{min} : max/min range of values
- B : Quantization Bits
- S (FP32/16), z (int): Scale and bias
- Q : Fixed point quantized values

$$S = \frac{r_{max} - r_{min}}{2^B - 1}$$

$$Q(r) = \text{Int} \left(\frac{r - Z}{S} \right)$$

$$\tilde{r} = S \times Q(r) + Z$$



Challenges of inferencing large scale models

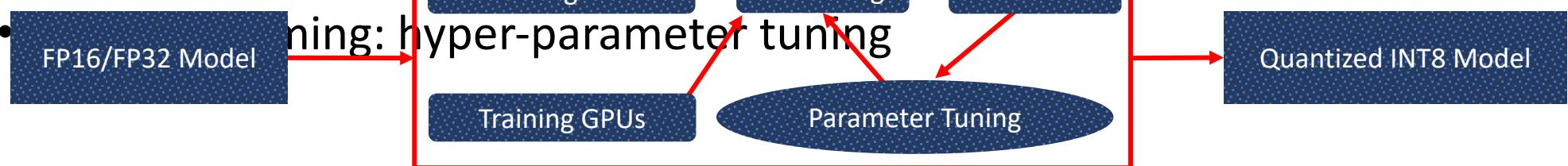
- Two main challenges of inferencing large scale models
 - High memory consumption: 40G A-100 for a ~20B model (FP16)
 - Slow speed: ~30ms for a token using GPT-J (6B)

- Quantization is one promising approach, but QAT ...

- Data unavailable: private or confidential issues

- A large number of GPUs: retrain large model

- Training: hyper-parameter tuning



Adv. and Disadv. of post-training quantization

- PTQ has better compression efficiency
 - Portion of training data
 - A small amount of GPUs
 - Little to no retraining
- Directly applying PTQ leads to accuracy loss

BERT-Base GLUE Performance with various precisions

Precision	CoLA	MNLI-m	MNLI-mm	MRPC	QNLI	QQP	RTE	SST-2	STS-B	Ave.
W16A16	59.72	84.94	85.06	86.27/90.57	92.15	91.51/88.56	72.20	93.23	90.06/89.59	83.95
W8A16	60.77	84.65	84.92	85.29/89.86	91.84	91.52/88.56	71.84	93.46	89.89/89.50	83.87
W16A8	56.85	80.55	81.48	84.07/89.33	91.34	91.30/88.07	68.59	93.46	88.74/88.74	81.93
W8A8	58.74	79.99	81.06	84.31/89.51	91.18	91.24/88.03	70.76	92.66	88.33/88.73	82.16
W4/8A16	0.00	16.74	16.95	31.62/0.00	50.74	63.18/0.00	47.29	70.64	16.48/15.91	33.11

Adv. and Disadv. of post-training quantization

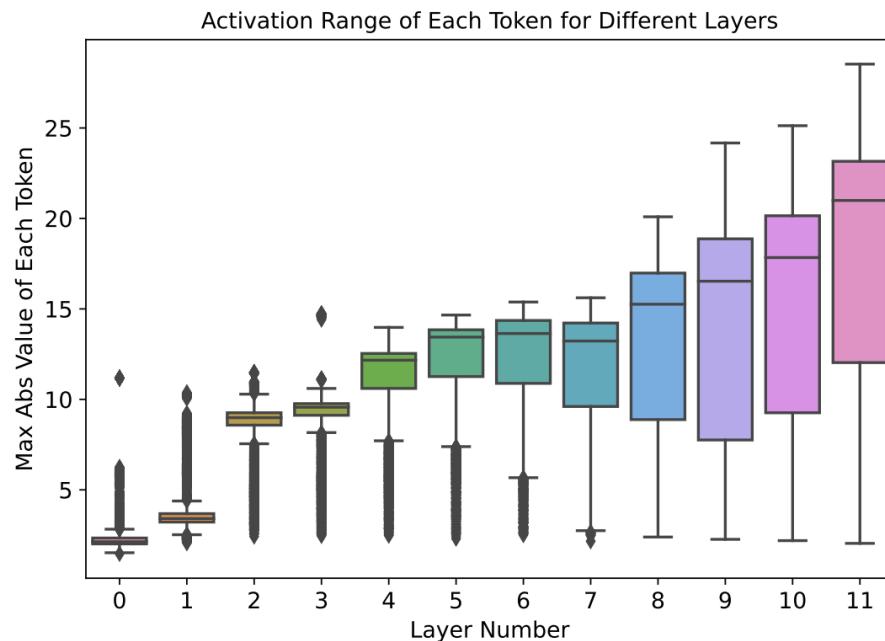
- PTQ has better compression efficiency
 - Portion of training data
 - A small amount of GPUs
 - Little to no retraining
- Directly applying PTQ leads to accuracy loss

Zero-shot evaluation of GPT-3-350M with various precisions

Precision	Lambada (↑)	PIQA (↑)	OpenBookQA (↑)	RTE (↑)	ReCoRd (↑)	Ave. 19 Tasks (↑)	Wikitext-2 (↓)
W16A16	49.3	66.3	29.4	53.8	75.1	38.9	21.5
W8A16	49.3	66.1	29.6	54.2	74.8	38.5	22.1
W16A8	44.7	64.8	28.2	52.7	69.2	37.8	24.6
W8A8	42.6	64.1	28.0	53.1	67.5	37.8	26.2
W4/8A16	0.00	51.4	30.2	52.7	16.1	28.9	1.76e5

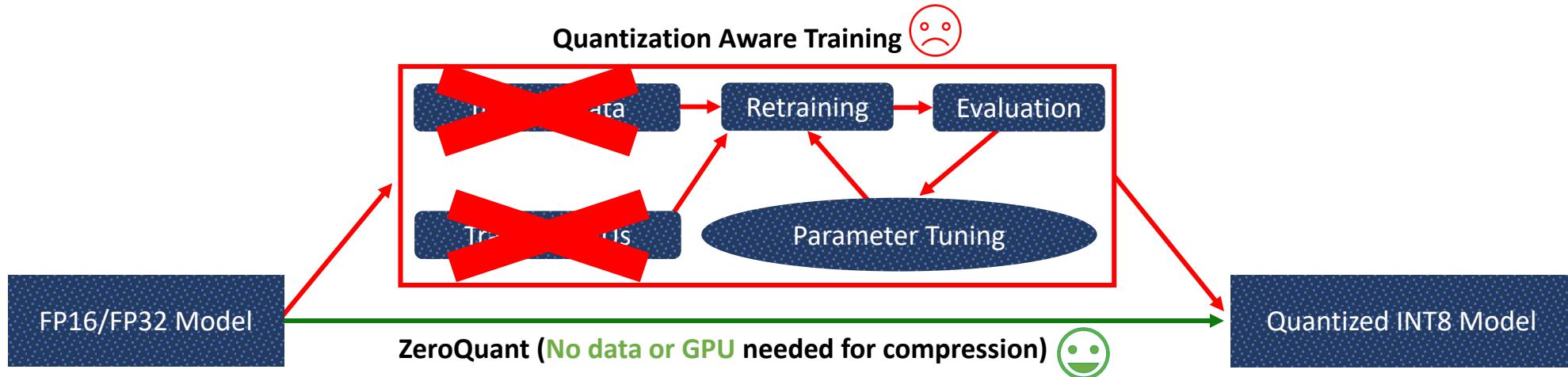
Why PTQ does not work

- Dynamic activation range
 - Different tokens have dramatically different activation ranges
- Different ranges of neurons in weight
 - No enough precision left for small-range neurons



ZeroQuant: INT8 without Compression Cost

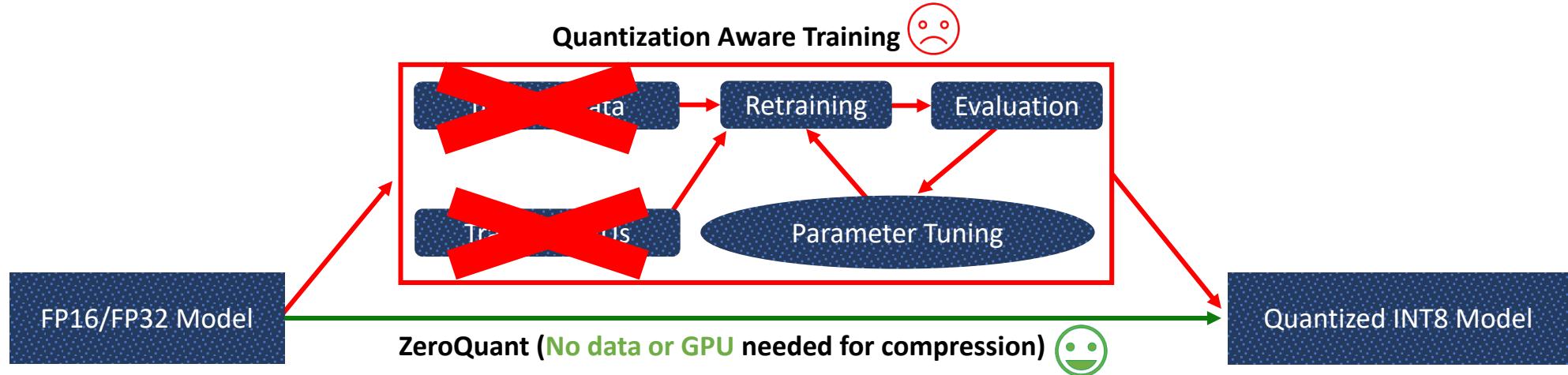
- Quantize large-scale models within limited time/resource budget



- Fine-grained quantization schemes to reduce quantization error
- Specified INT8 kernels to get real latency reduction

ZeroQuant: INT8 without Compression Cost

- Quantize large-scale models within limited time/resource budget



FP16 Weight Matrix

1.1	2.2	3.3	4.4	5.5	6.6
...					
...					
...					
...					
1.2	2.4	3.6	4.8	6.0	7.2

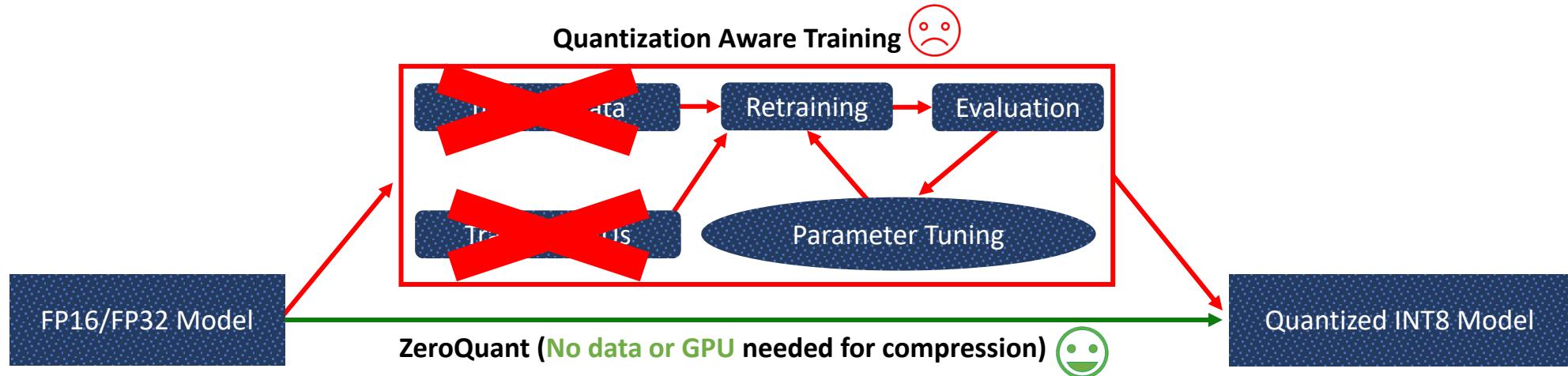


Scaling Factor: 1.1 *

1	2	3	4	5	6
...					
...					
...					
...					
1	2	3	4	5	6

ZeroQuant: INT8 without Compression Cost

- Quantize large-scale models within limited time/resource budget



FP16 Weight Matrix

1.1	2.2	3.3	4.4	5.5	6.6
...					
...					
...					
...					
1.2	2.4	3.6	4.8	6.0	7.2

≈

ZeroQuant INT8

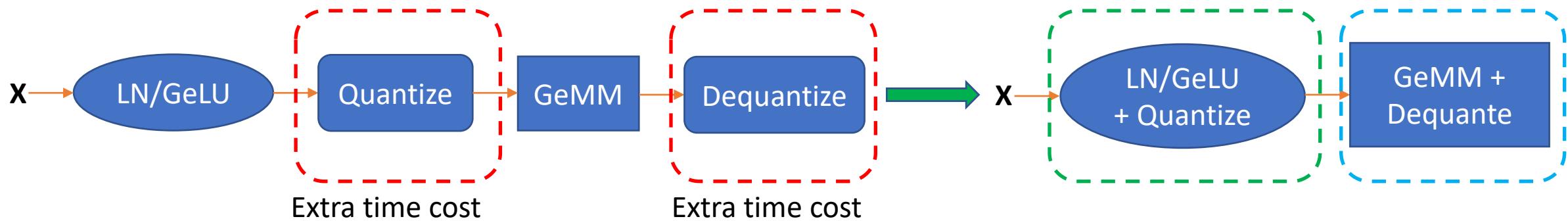
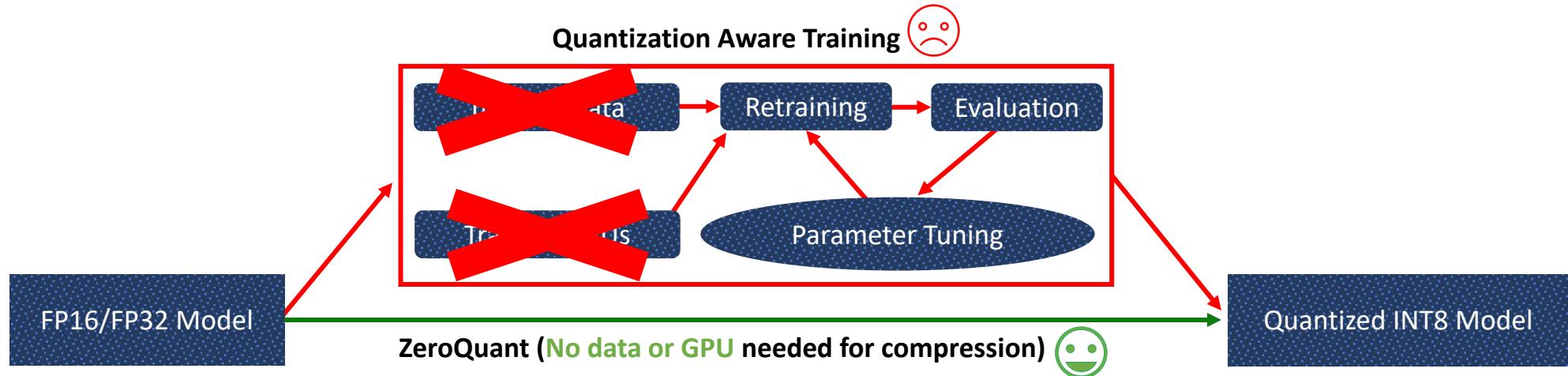
Scaling Factor: 1.1 *

1	2	3	4	5	6
...					
...					
...					
...					
1	2	3	4	5	6

Scaling Factor: 1.2 *

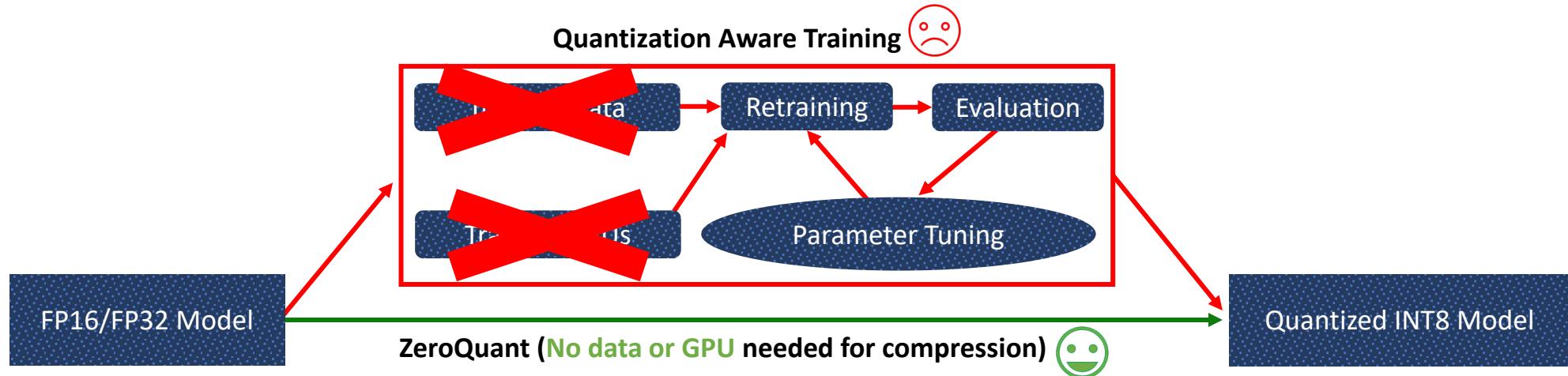
ZeroQuant: INT8 without Compression Cost

- Quantize large-scale models within limited time/resource budget



ZeroQuant: INT8 without Compression Cost

- Quantize large-scale models within limited time/resource budget

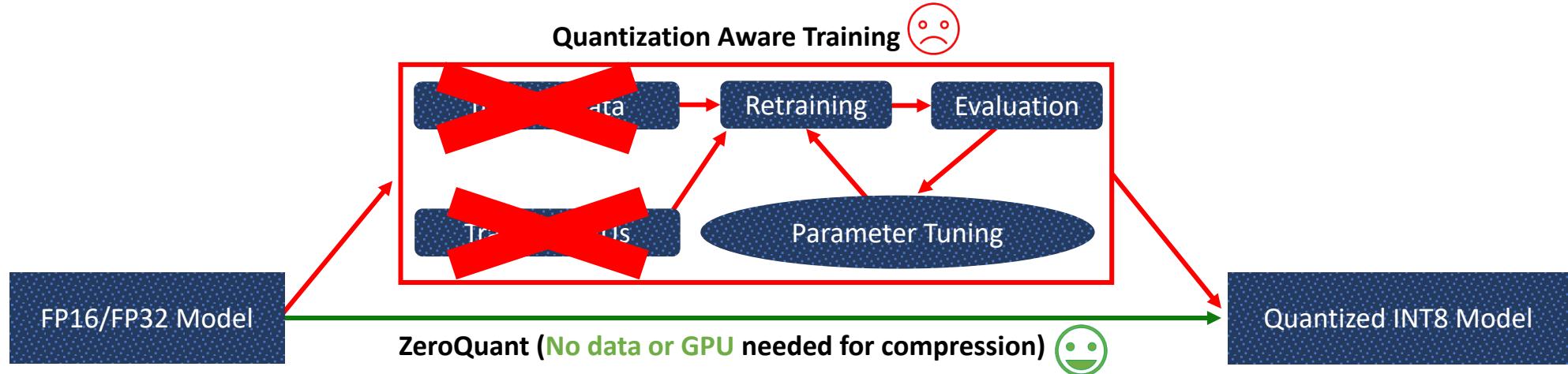


BERT-Base GLUE Performance with QAT and PTQ

Precision (Method)	CoLA	MNLI-m	MNLI-mm	MRPC	QNLI	QQP	RTE	SST-2	STS-B	Ave.	Ave. Time (s)
W16A16 (Baseline)	59.72	84.94	85.06	86.27/90.57	92.15	91.51/88.56	72.20	93.23	90.06/89.59	83.95	N/A
W8A8 [56] (QAT) ⁺	—	83.91	83.83	—	—	—	—	92.83	—	—	—
W8A8 [76] (QAT)	58.48	—	—	—/89.56	90.62	—/87.96	68.78	92.24	89.04/—	—	—
W8A8 (QAT)	61.21	84.80	84.64	83.82/88.85	91.29	91.29/88.28	71.12	92.89	88.39/88.18	83.37	2900
W8A8 (PTQ)	56.06	79.99	81.06	75.49/79.67	87.35	89.92/86.82	48.38	91.40	86.58/86.44	77.41	6
W8A8/16 [6] (PTQ)*	58.63	82.67	82.67	88.74	90.41	89.40	68.95	92.66	88.00	82.46	Unknown
W8A8 (ZeroQuant)	59.59	84.83	85.13	86.03/90.39	91.98	91.45/88.46	71.12	93.12	90.09/89.62	83.75	0

ZeroQuant: INT8 without Compression Cost

- Quantize large-scale models within limited time/resource budget

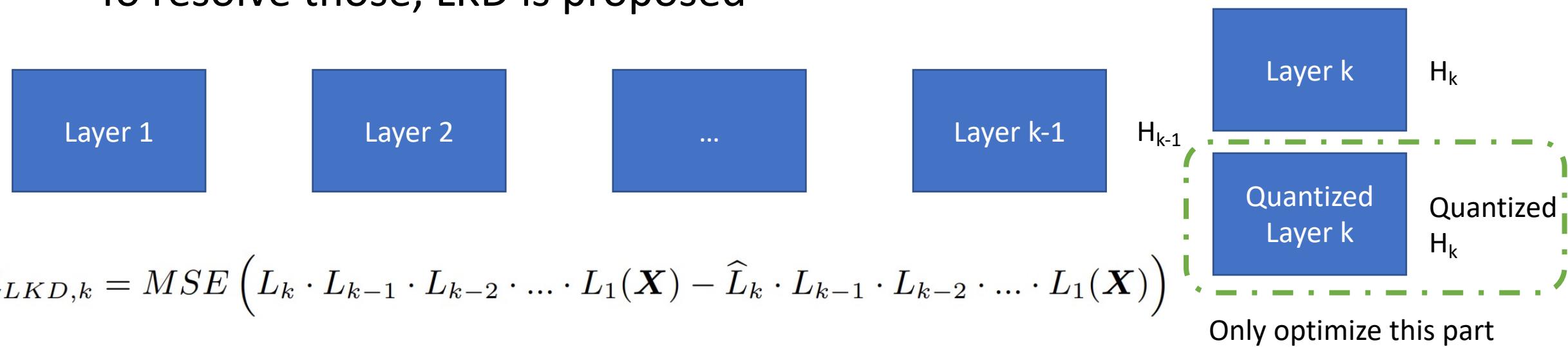


GPT-3-Style 125M	Ave. over 19 Tasks	Wikitext (lower better)	# GPUs for Compression	Time for Compression	Data Requirement
Baseline, FP16	36.31	29.4	N/A	N/A	N/A
QAT, INT8	35.99	33.24	32	20 hours	Yes
ZeroQuant, INT8	36.32	29.5	None	None	No

GPT-Neox 20B	LAMBADA	PIQA	Ave. over 19 Tasks	# GPUs for Compression	Time for Compression	Data Requirement	GPU x Latency (ms)	Inf Cost Reduction
Baseline, FP16	71.7	77.7	50.44				2x65	1x
QAT, INT8	--	--	--	96	20 days	Yes	--	--
ZeroQuant, INT8	71.9	78.3	50.38	None	None	No	1x25	5.2x

Layer-by-layer knowledge distillation

- Knowledge distillation for even lower-bit quantization
 - Hold a teacher and a student model together
 - Several copies (gradient, first/second order momentum) of the weight
 - Original training data
- To resolve those, LKD is proposed



Layer-by-layer knowledge distillation

- Knowledge distillation for even lower-bit quantization
 - Hold a teacher and a student model together
 - Several copies (gradient, first/second order momentum) of the weight
 - Original training data
- To resolve those, LKD is proposed
 - No need a separate teacher
 - Reduced memory overhead
 - Work even without original training data

Mixed-precision results with LKD

- Without tuning, LKD
 - ~1 point gain with 31s on BERT-base

BERT-Base GLUE Performance with QAT and PTQ

Precision (Method)	CoLA	MNLI-m	MNLI-mm	MRPC	QNLI	QQP	RTE	SST-2	STS-B	Ave.	Ave. Time (s)
W16A16 (Baseline)	59.72	84.94	85.06	86.27/90.57	92.15	91.51/88.56	72.20	93.23	90.06/89.59	83.95	N/A
W8A8 [56] (QAT) ⁺	—	83.91	83.83	—	—	—	—	92.83	—	—	—
W8A8 [76] (QAT)	58.48	—	—	—/89.56	90.62	—/87.96	68.78	92.24	89.04/—	—	—
W8A8 (QAT)	61.21	84.80	84.64	83.82/88.85	91.29	91.29/88.28	71.12	92.89	88.39/88.18	83.37	2900
W8A8 (PTQ)	56.06	79.99	81.06	75.49/79.67	87.35	89.92/86.82	48.38	91.40	86.58/86.44	77.41	6
W8A8/16 [6] (PTQ)*	58.63	82.67	82.67	88.74	90.41	89.40	68.95	92.66	88.00	82.46	Unknown
W8A8 (ZeroQuant)	59.59	84.83	85.13	86.03/90.39	91.98	91.45/88.46	71.12	93.12	90.09/89.62	83.75	0
W4/8A16 (PTQ)	0.00	16.74	16.95	31.62/0.00	50.74	63.18/0.00	47.29	70.64	16.48/15.91	33.11	6
W4/8A16 (ZeroQuant)	57.29	82.69	83.27	84.56/88.40	90.04	86.52/79.49	70.76	92.78	88.46/88.61	81.65	0
W4/8A16 (ZeroQuant-LKD)	58.50	83.16	83.69	84.80/89.31	90.83	88.94/84.12	70.04	92.78	88.49/88.67	82.35	31
W4/8A8 (ZeroQuant)	56.69	82.46	83.06	84.07/88.03	90.13	87.04/80.50	70.76	92.78	88.07/88.44	81.55	0
W4/8A8 (ZeroQuant-LKD)	58.80	83.09	83.65	85.78/89.90	90.76	89.16/84.85	71.84	93.00	88.16/88.55	82.71	31

Mixed-precision results with LKD

- Without tuning, LKD
 - ~1 point gain with 31s on BERT-base
 - >3% acc and 50 PPL gain on GPT-3-350M

Zero-shot Eval Performance of GPT-3-350M

Precision (Method)	Lambada (↑)	PIQA (↑)	OpenBookQA (↑)	RTE (↑)	ReCoRd (↑)	Ave. 19 Tasks (↑)	Wikitext-2 (↓)	Time Cost
W16A16	49.3	66.3	29.4	53.8	75.1	38.9	21.5	N/A
W8A8 (PTQ)	42.6	64.1	28.0	53.1	67.5	37.8	26.2	7 mins
W8A8 (ZeroQuant)	51.0	66.5	29.2	53.4	74.9	38.7	21.7	0
W4/8A16 (PTQ)	0.00	51.4	30.2	52.7	16.1	28.9	1.76e5	7 mins
W4/8A16 (ZeroQuant)	10.1	58.5	27.2	52.0	56.5	33.5	88.6	0
W4/8A16 (ZeroQuant-LKD)	39.8	63.8	29.4	53.1	70.1	37.0	30.6	1.1 hours
W4/8A8 (ZeroQuant)	10.5	57.7	28.0	52.7	55.3	33.4	92.1	0
W4/8A8 (ZeroQuant-LKD)	37.4	61.8	28.2	53.1	68.5	36.6	31.1	1.1 hours

Mixed-precision results with LKD

- Without tuning, LKD
 - ~1 point gain with 31s on BERT-base
 - >3% acc and 50 PPL gain on GPT-3-350M
- With tuning (LR and Iter)
 - Extra ~0.5 gain with in total 36 GPU hours for all tasks on BERT-Base

Precision (Method)	CoLA	MNLI-m	MNLI-mm	MRPC	QNLI	QQP	RTE	SST-2	STS-B	Ave.
W16A16 (Baseline)	59.72	84.94	85.06	86.27/90.57	92.15	91.51/88.56	72.20	93.23	90.06/89.59	83.95
W8A8 (ZeroQuant-LKD No Tuning)	59.59	84.83	85.13	86.03/90.39	91.98	91.45/88.46	71.12	93.12	90.09/89.62	83.75
W8A8 (ZeroQuant-LKD Tuned)	60.90	84.95	85.10	86.27/90.60	92.07	91.47/88.47	71.84	93.46	90.09/89.62	84.07
W4/8A16 (ZeroQuant-LKD No Tuning)	58.50	83.16	83.69	84.80/89.31	90.83	88.94/84.12	70.04	92.78	88.49/88.67	82.35
W4/8A16 (ZeroQuant-LKD Tuned)	60.04	83.64	84.31	85.78/89.53	91.01	90.66/87.26	71.84	93.12	88.68/88.79	83.26
W4/8A8 (ZeroQuant-LKD No Tuning)	58.80	83.09	83.65	85.78/89.90	90.76	89.32/84.85	71.84	93.00	88.16/88.55	82.71
W4/8A8 (ZeroQuant-LKD Tuned)	60.30	83.47	84.03	85.78/89.90	90.87	90.77/87.38	71.84	93.00	88.38/88.70	83.22

LKD without original training data

- A good data resource can provide similar model accuracy
- Random data gives accuracy boost compared to ZeroQuant (no LKD)

Zero-shot Eval Performance of GPT-3-350M with W4/8A8

Method	Data Resource	Lambada (↑)	PIQA (↑)	OpenBookQA (↑)	RTE (↑)	ReCoRd (↑)	Ave. 19 Tasks (↑)	Wikitext-2 (↓)
ZeroQuant	—	10.5	57.7	28.0	52.7	55.3	33.4	92.1
ZeroQuant-LKD	Random data	26.1	59.3	29.2	50.5	64.9	34.5	40.6
ZeroQuant-LKD	Wikipedia	33.9	62.4	28.0	52.7	69.5	36.2	30.4
ZeroQuant-LKD	Original data	37.4	61.8	28.2	53.1	68.5	36.6	31.1

Outline

- DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales
- PR-MoE: Improve the Parameter Efficiency of MoE for Better Training and Inference
- ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers
- XTC: Extreme Compression for Pre-trained Transformers Made Simple and Efficient

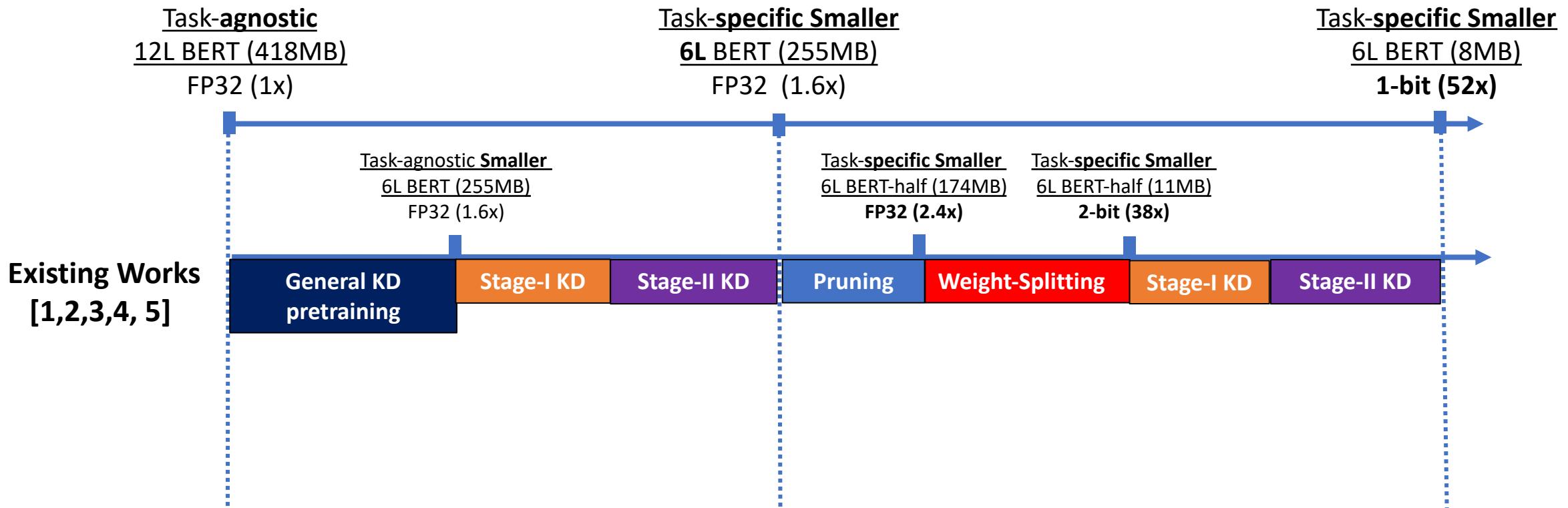
[1] <https://github.com/microsoft/DeepSpeed/blob/master/blogs/deepspeed-chat/README.md>

[2] [\[2201.05596\] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale \(arxiv.org\)](https://arxiv.org/abs/2201.05596)

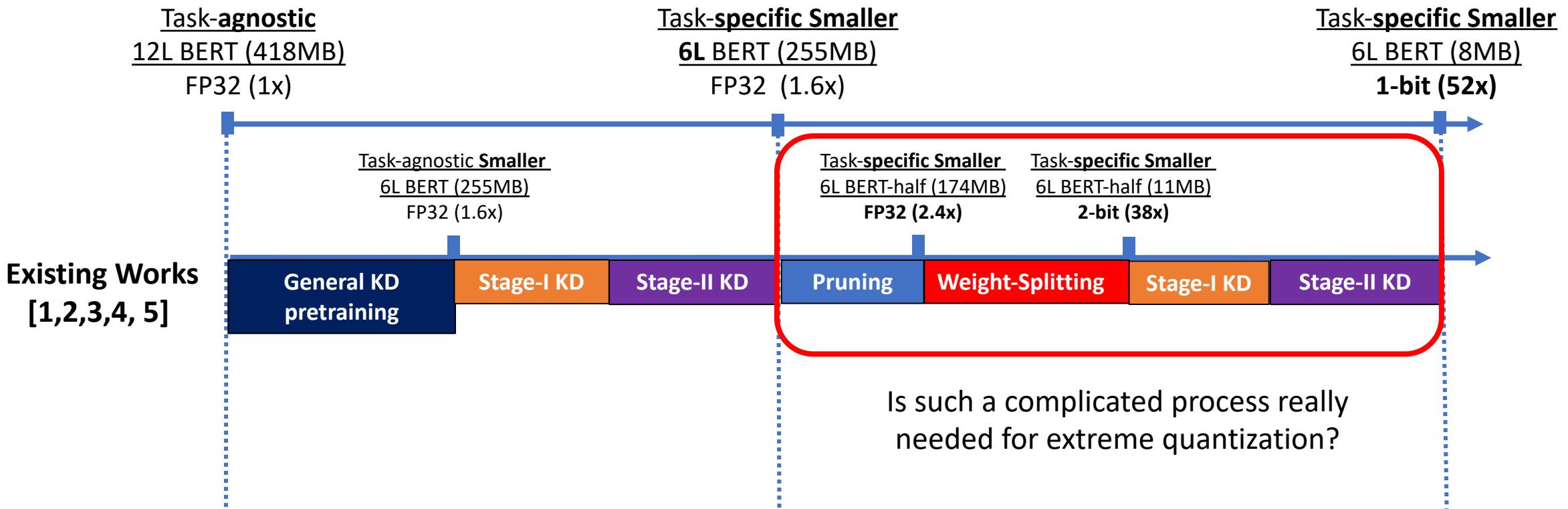
[3] [\[2206.01861\] ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers \(arxiv.org\)](https://arxiv.org/abs/2206.01861)

[4] [\[2206.01859v1\] Extreme Compression for Pre-trained Transformers Made Simple and Efficient \(arxiv.org\)](https://arxiv.org/abs/2206.01859v1)

The complexity of extreme compression



Complicated process for extreme quantization



Longer training can simplicity close the gap

- Short training → Large acc loss (shown in BinaryBERT paper)
- Longer training → Better acc
- 10x longer training → ~0.5 acc drop

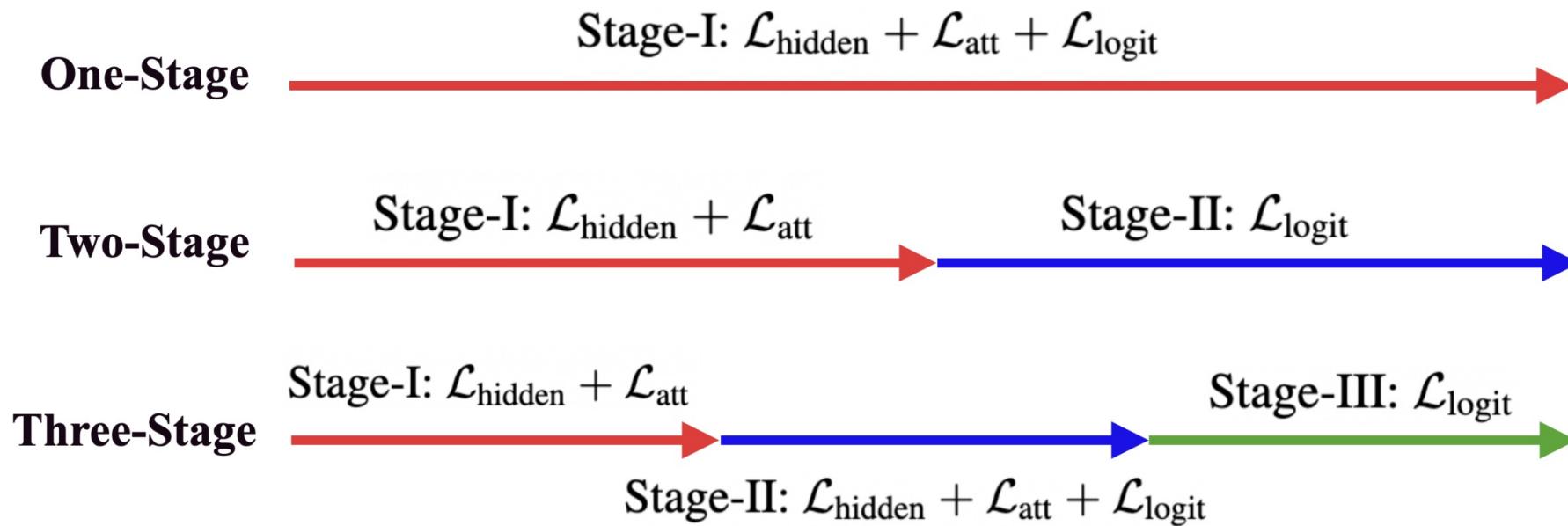
Dataset	Training Budget	
	Data Aug.	Training epochs: Budget-A
QQP/MNLI	✗	3
QNLI	✓	1
SST-2/STS-B/RTE	✓	1
CoLA/MRPC	✓	1

BinaryBERT-Base GLUE Performance

#	Cost	CoLA Mcc	MNLI-m/-mm Acc/Acc	MRPC F1/Acc	QNLI Acc	QQP F1/Acc	RTE Acc	SST-2 Acc	STS-B Pear/Spea	Avg.	Acc Drop
0	Teacher	59.7	84.9/85.6	90.6/86.3	92.1	88.6/91.5	72.2	93.2	90.1/89.6	83.95	-
1	Budget-A	50.4	83.7/84.6	90.0/85.8	91.3	88.0/91.1	72.9	92.8	88.8/88.4	82.38	1.57

Single stage KD is better

- Complicated multi-stage knowledge distillation
 - Exhibit better accuracy in many literatures
 - More hyperparameter tuning, e.g., different LR for different stages



Single stage KD is better

- Complicated multi-stage knowledge distillation
 - Exhibit better accuracy in many literatures
 - More hyperparameter tuning, e.g., different LR for different stages

BinaryBERT-Base GLUE Performance

Training Cost	Stages	learning rate	CoLA Mcc	MNLI-m/-mm Acc/Acc	MRPC F1/Acc	QNLI Acc	QQP F1/Acc	RTE Acc	SST-2 Acc	STS-B Pear/Spea	Avg.	Avg. w/o CoLA
One-Stage		2e-5	44.6	83.1/83.7	88.8/83.1	91.1	87.4/90.7	66.1	92.8	87.8/87.5	80.33	84.80
		1e-4	50.4	83.7/84.6	90.1/85.3	91.3	88.0/91.1	71.5	92.7	88.8/88.4	82.16	86.12
		5e-4	42.3	83.3/84.1	90.0/85.8	89.5	87.8/90.9	72.9	92.5	88.1/87.8	81.04	85.89
		Best (above)	50.4	83.7/84.6	90.0/85.8	91.3	88.0/91.1	72.9	92.8	88.8/88.4	82.38	86.37
Budget-A	Two-Stage	2e-5	48.2	83.3/83.8	89.3/84.6	90.7	87.7/90.9	70.4	92.5	88.7/88.4	81.46	85.61
		1e-4	48.5	83.3/83.4	90.0/85.5	90.4	87.4/90.7	70.4	92.4	88.6/88.2	81.47	85.59
		5e-4	16.2	74.9/76.4	89.7/84.8	87.8	85.0/89.0	68.2	91.6	86.2/86.5	75.01	82.36
		Best (above)	48.5	83.3/83.8	90.0/85.5	90.7	87.7/90.9	70.4	92.5	88.7/88.4	81.59	85.72
Three-Stage		2e-5	49.3	83.1/83.6	89.5/84.3	91.0	87.6/90.9	70.8	92.4	88.7/88.3	81.57	85.60
		1e-4	49.0	83.2/83.4	89.7/84.8	90.9	87.6/90.7	73.6	92.2	88.8/88.5	81.84	85.95
		5e-4	29.5	81.3/81.9	89.1/83.6	88.4	85.0/89.3	66.1	91.9	84.1/83.9	77.34	83.32
		Best (above)	49.3	83.2/83.4	89.7/84.8	91.0	87.6/90.9	73.6	92.4	88.8/88.5	81.93	86.01

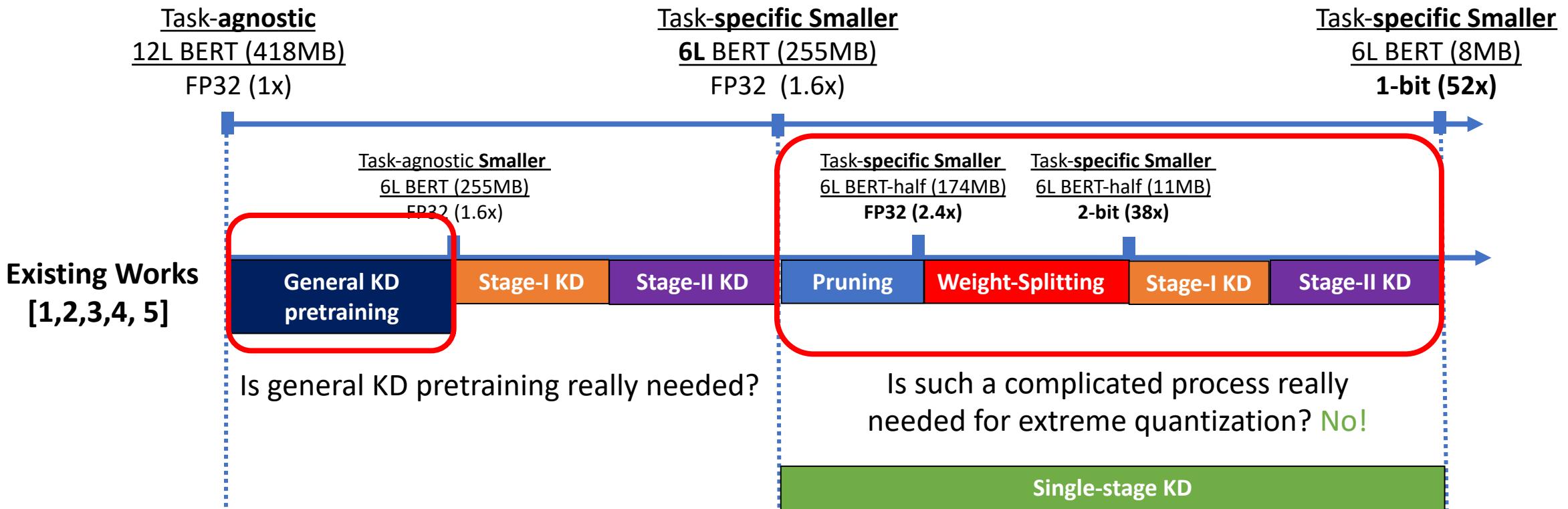
Single stage KD is better

- Complicated multi-stage knowledge distillation
 - Exhibit better accuracy in many literatures
 - More hyperparameter tuning, e.g., different LR for different stages

BinaryBERT-Base GLUE Performance

Training Cost	Stages	learning rate	CoLA Mcc	MNLI-m/-mm Acc/Acc	MRPC F1/Acc	QNLI Acc	QQP F1/Acc	RTE Acc	SST-2 Acc	STS-B Pear/Spea	Avg.	Avg. w/o CoLA
One-Stage		2e-5	57.3	84.1/84.0	90.7/86.5	91.0	88.2/91.2	74.0	93.1	89.2/88.8	83.38	86.64
		1e-4	55.6	84.2/84.4	90.4/86.0	90.4	88.3/91.3	73.6	93.0	88.8/88.4	83.03	86.46
		5e-4	46.1	82.8/82.9	89.9/86.0	89.5	87.7/90.8	71.1	92.4	88.2/88.0	81.09	85.46
		Best (above)	57.3	84.2/84.4	90.7/86.5	91.0	88.3/91.3	74.0	93.1	89.2/88.8	83.44	86.71
Budget-C	Two-Stage	2e-5	55.9	84.2/83.9	90.6/86.5	90.7	88.3/91.3	73.3	93.1	89.2/88.9	83.12	86.52
		1e-4	52.0	83.2/83.8	90.9/86.5	90.1	88.0/91.0	74.0	93.1	87.8/87.8	82.39	86.19
		5e-4	32.2	79.3/80.3	90.6/86.5	89.6	85.3/89.1	70.0	91.2	87.0/86.8	78.36	84.12
		Best (above)	55.9	84.2/83.9	90.6/86.5	90.7	88.3/91.3	74.0	93.1	89.2/88.9	83.20	86.61
Three-Stage		2e-5	57.1	84.0/84.4	90.6/86.5	90.9	88.1/91.1	72.6	92.9	89.5/89.2	83.22	86.49
		1e-4	53.8	83.5/84.1	90.8/86.8	90.1	88.3/91.3	72.6	92.8	88.2/87.9	82.58	86.18
		5e-4	31.3	80.0/81.3	89.8/85.3	89.2	86.7/90.0	70.8	91.3	86.4/86.8	78.40	84.29
		Best (above)	57.1	84.0/84.4	90.8/86.8	90.9	88.3/91.3	72.6	92.9	89.5/89.2	83.28	86.55

Complicated process for extreme quantization



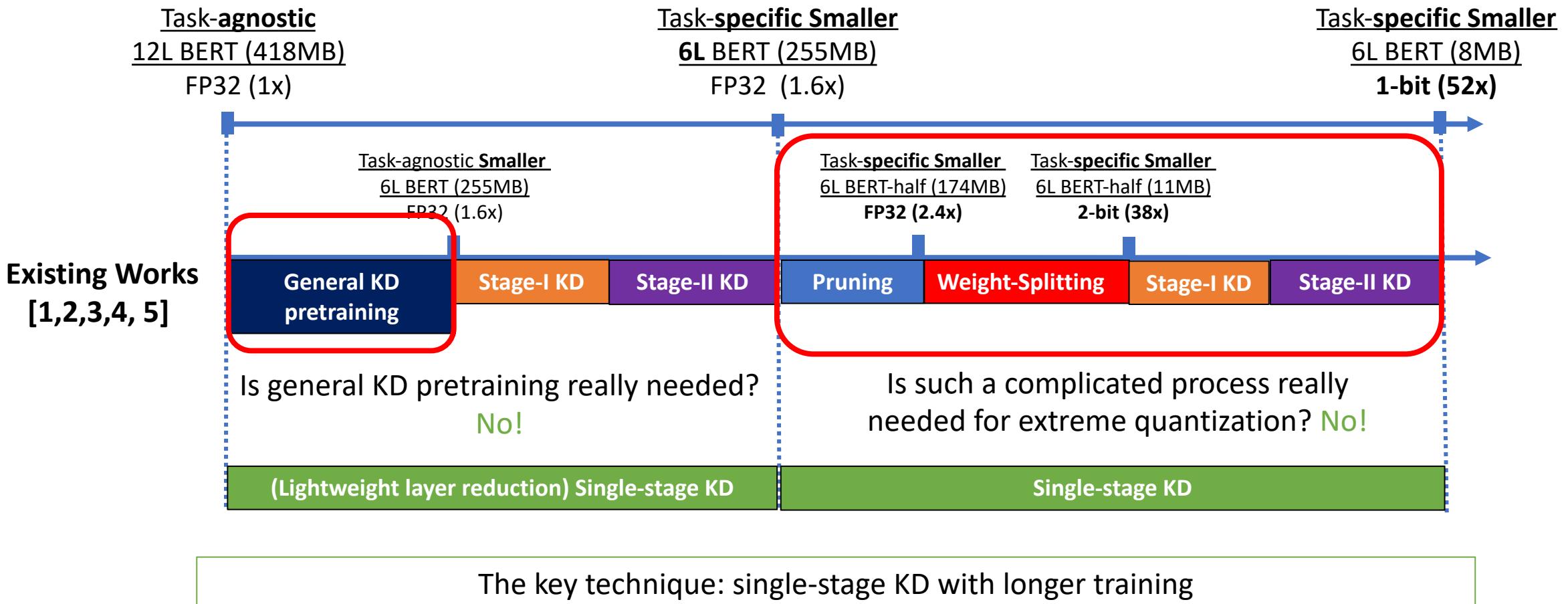
Pretraining KD is not necessary

- Pretraining KD can be removed when applying
 - Longer training
 - Single stage KD

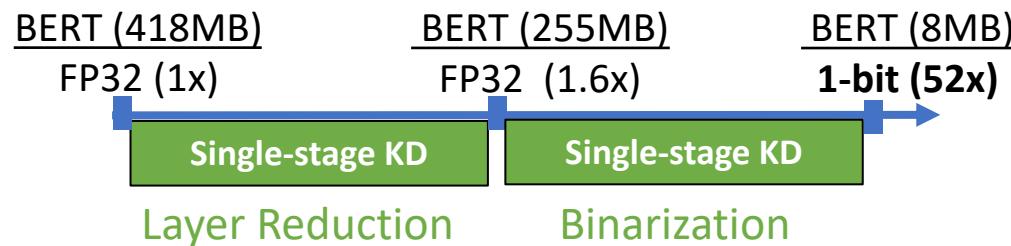
Results of different methods with various number of layers

#	Model	size	CoLA Mcc	MNLI-m/-mm Acc/Acc	MRPC F1/Acc	QNLI Acc	QQP F1/Acc	RTE Acc	SST-2 Acc	STS-B Pear/Spea	Avg. all	Acc. Drop
1	BERT-base fp32	417.2	59.7	84.9/85.6	90.6/86.3	92.1	88.6/91.5	72.2	93.2	90.1/89.6	83.95	-
Training cost: greater than Budget-C (see [20] or § B)												
2	Pretrained TinyBERT ₆ ([20])	255.2 ($\times 1.6$)	54.0	84.5/84.5	90.6/86.3	91.1	88.0/91.1	73.4	93.0	90.1/89.6	83.11	-0.84
3	Pretrained TinyBERT ₆ (rep.*)	255.2 ($\times 1.6$)	56.9	84.4/84.8	90.1/85.5	91.3	88.4/91.4	72.2	93.2	90.3/90.0	83.33	-0.62
Training cost: Budget-C												
4	Pretrained TinyBERT ₆	255.2 ($\times 1.6$)	54.4	84.6/84.3	90.4/86.3	91.5	88.5/91.5	69.7	93.3	89.2/89.0	82.76	-1.19
5	Pretrained MiniLM ₆ -v2	255.2 ($\times 1.6$)	55.4	84.4/84.5	90.7/86.5	91.4	88.5/91.5	71.8	93.3	89.4/89.0	83.13	-0.82
6	Skip-BERT ₆ (ours)	255.2 ($\times 1.6$)	56.9	84.6/84.9	90.4/85.8	91.8	88.6/91.6	72.6	93.5	89.8/89.4	83.50	-0.45
7	Skip-BERT ₅ (ours)	228.2 ($\times 1.8$)	57.9	84.3/85.1	90.1/85.5	91.4	88.5/91.5	72.2	93.3	89.2/88.9	83.38	-0.57
8	Skip-BERT ₄ (ours)	201.2 ($\times 2.1$)	53.3	83.2/83.4	90.0/85.3	90.8	88.2/91.3	70.0	93.5	88.8/88.4	82.18	-1.77

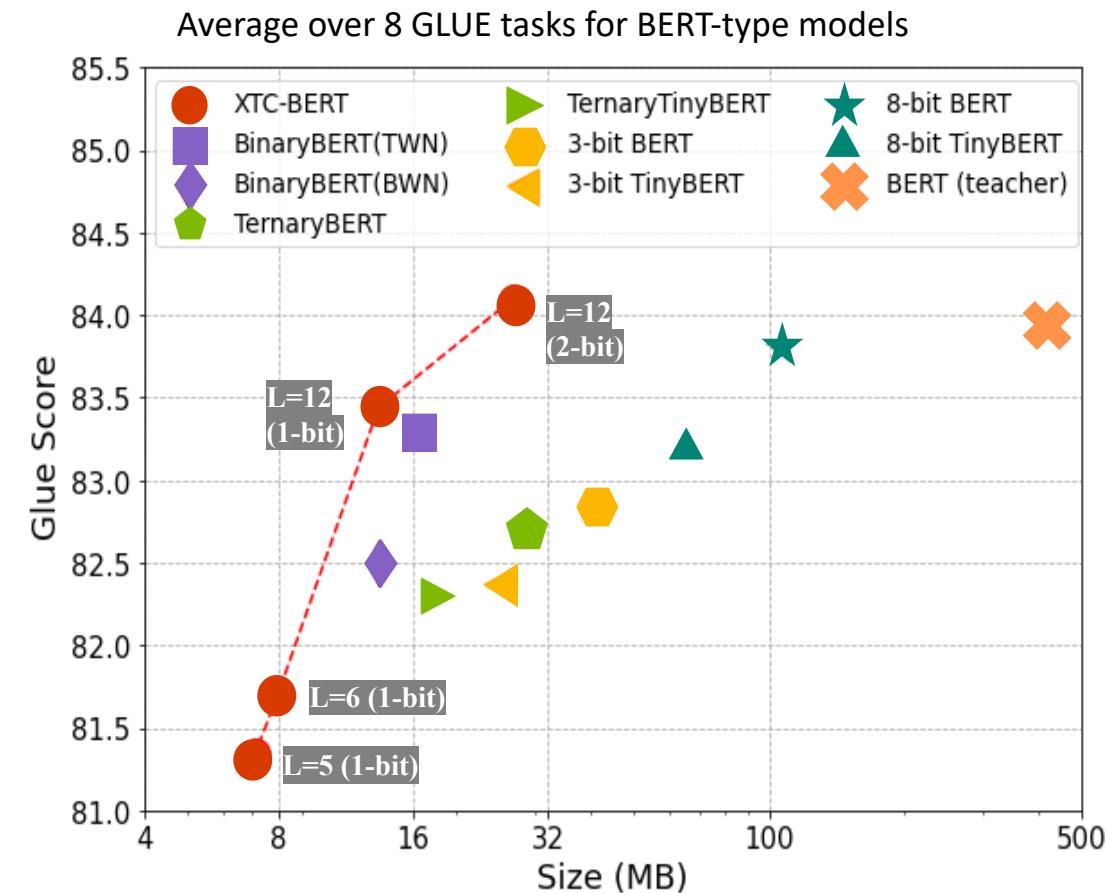
Complicated process for extreme quantization



XTC: Simple yet effective extreme compression



- Great model size reduction (**>50x**)
 - Layer Reduction (**~2x**)
Flexible to choose any layers of teacher model
 - Weight Binarization (**32x**)
Simple and easy to implement
- Setting a new Pareto Frontier



System design, implementation, and OSS

- DeepSpeed PR-MoE (released)
 - The entire distributed training system for flexible # experts assignment
- DeepSpeed Compression Library (released)
 - The entire system for compression
 - The easy to use API design
 - ...

Thank You for Listening!

zheweiyao@gmail.com