CSYE 7105

Mixed Precision fine-tuning and Parallel Training for LLM

Team 2:
Yang Qu, Rui Wu

Advisor:
Professor Handan Liu

Northeastern University
School of Engineering

Contents

- ❖ Model and Purpose
- ❖ Model structure Fine-tune
- ❖ Data Process
- ❖ Optimization
- ❖ Future work

Northeastern University
School of Engineering

Model and Purpose

Northeastern University
School of Engineering

Model and Purpose-----LLM Introduction

BERT

Northeastern University
School of Engineering

Model and Purpose-----Llama3 Model

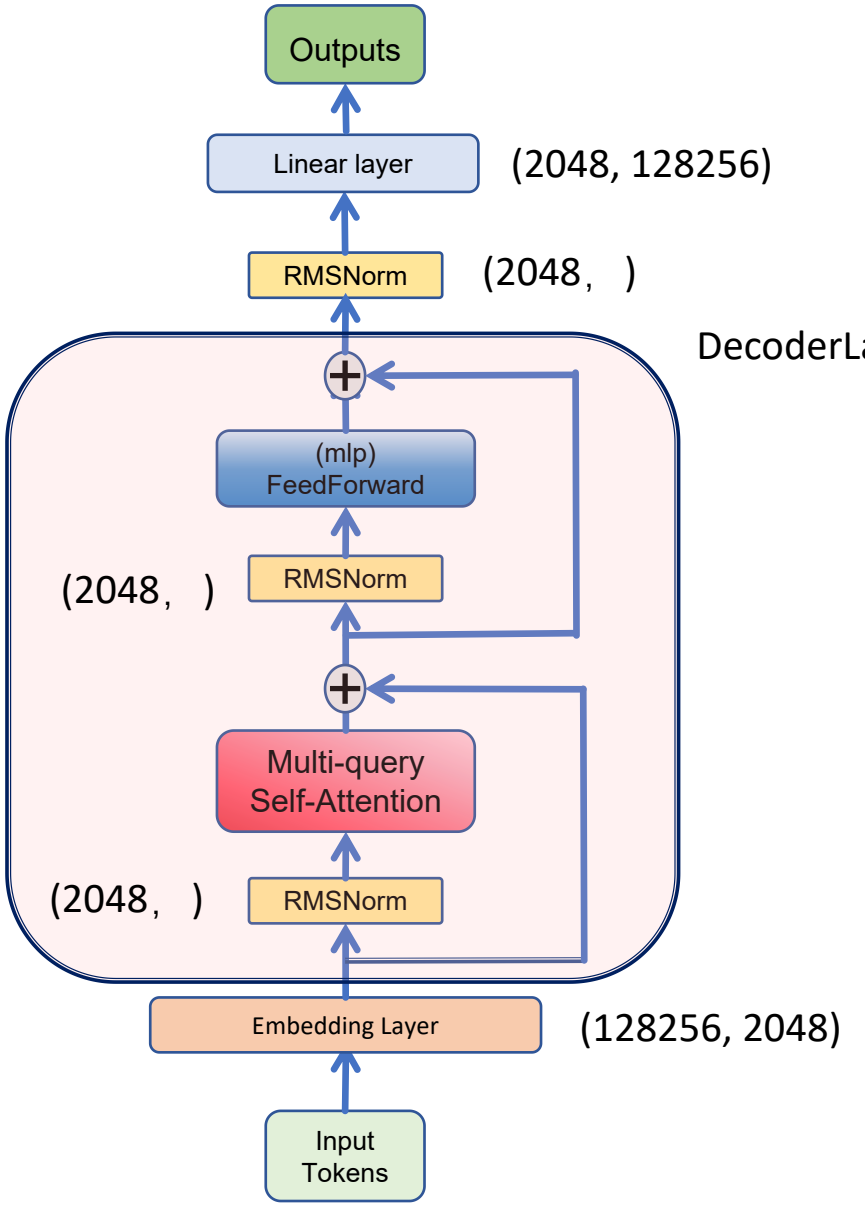
Llama is tansformer-based decoder model relased by Meta starting in February 2023.

Model	Version	Reference
Llama	7B, 13B, 32B, 65B	No commercial viability
Llama2	7B, 13B, 70B	https://www.Llama.com/Llama-downloads/
Llama3.1	8B, 70B, 405B	
Llama3.2	1B, 3B, 11B, 90B	
Llama3.3	70B	
Llama4	13B	

The latest version of Llama is Llama 4, released on 5 April 2025.

Northeastern University
School of Engineering

Model and Purpose-----Llama 3.2 1B

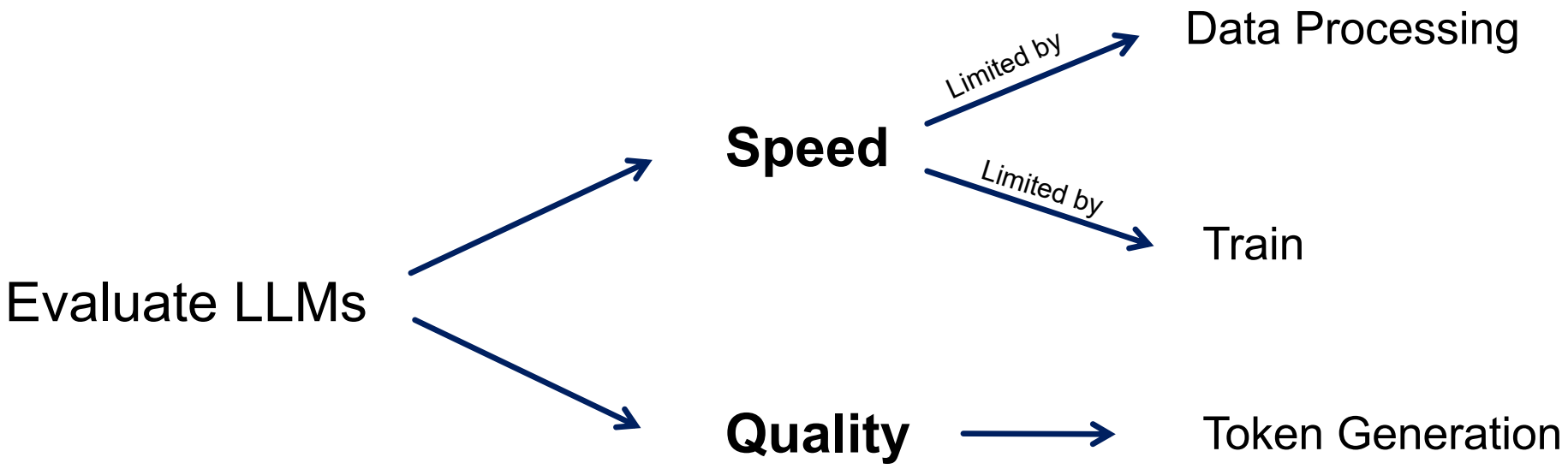


Structure of Llama 3.2 1B

[arxiv:2204.05109]
[arxiv: 2405.16406]
(was published at ICLR in feb 2025)

Northeastern University
School of Engineering

Model and Purpose----- Optimization Approach



Northeastern University
School of Engineering

Model and Purpose----- Optimization Approach

Data Processing

Train

Parallelization $\xrightarrow{\text{Boost}}$ Speed

Token Generation $\longrightarrow$ **Fine-tune** $\xrightarrow{\text{Enhance}}$ Quality

Northeastern University
School of Engineering

Model Structure Fine-tune

Northeastern University
School of Engineering

Model structure Fine-tune-----Introduction of Low_rank

Low-rank [arxiv: 2106.09685]

Rank represents the independent information in deep learning.

Low Rank Approximation:

For weight matirx W

$$W \in \mathbb{R}^{in \times out},$$

we use

$$W \approx A \times B,$$

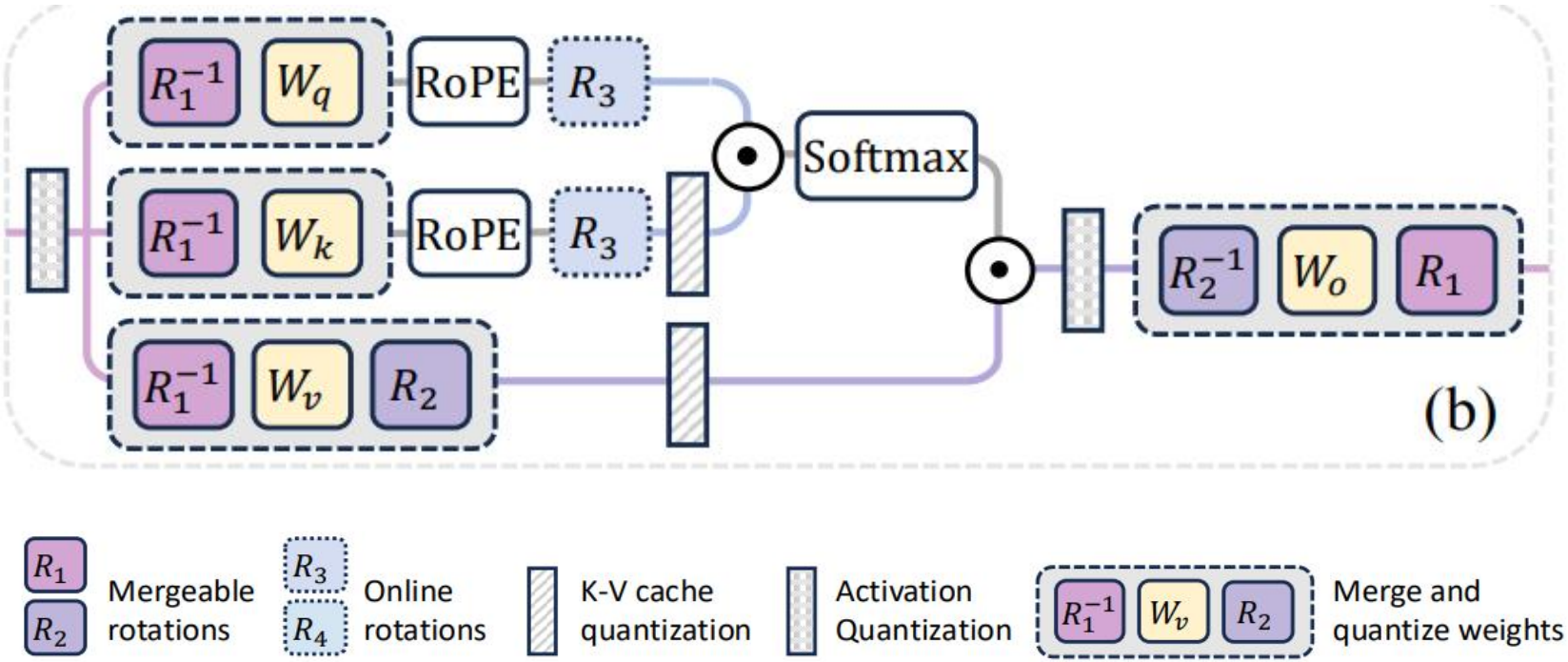
where $A \in \mathbb{R}^{in \times r}$, $B \in \mathbb{R}^{r \times out}$, $r \leq min(in, out)$.

We replace big weights matrix W with two small matrices A and B.

Northeastern University
School of Engineering

Model structure Fine-tune-----LowRank Modify

We modified q_proj, v_porj and o_proj in Attention



Multi-Queried self-attention structure from [arxiv: 2405.16406]

$$\left\{ \begin{array}{l} W_q \approx A \times B \\ W_v \approx A \times B \\ W_o \approx A \times B \end{array} \right\}$$

We set rank=64, alpha=128:

```
Total trainable parameters (LoRA only): 11010048
```

Northeastern University
School of Engineering

Model structure Fine-tune-----LowRank Modify

We modified q_proj, v_porj and o_proj in Attention

```
Replacing model.layers.0.self_attn.q_proj with LoRA
Replacing model.layers.0.self_attn.v_proj with LoRA
Replacing model.layers.0.self_attn.o_proj with LoRA
Replacing model.layers.1.self_attn.q_proj with LoRA
Replacing model.layers.1.self_attn.v_proj with LoRA
Replacing model.layers.1.self_attn.o_proj with LoRA
Replacing model.layers.2.self_attn.q_proj with LoRA
Replacing model.layers.2.self_attn.v_proj with LoRA
Replacing model.layers.2.self_attn.o_proj with LoRA
Replacing model.layers.3.self_attn.q_proj with LoRA
Replacing model.layers.3.self_attn.v_proj with LoRA
Replacing model.layers.3.self_attn.o_proj with LoRA
Replacing model.layers.4.self_attn.q_proj with LoRA
Replacing model.layers.4.self_attn.v_proj with LoRA
Replacing model.layers.4.self_attn.o_proj with LoRA
Replacing model.layers.5.self_attn.q_proj with LoRA
Replacing model.layers.5.self_attn.v_proj with LoRA
Replacing model.layers.5.self_attn.o_proj with LoRA
Replacing model.layers.6.self_attn.q_proj with LoRA
Replacing model.layers.6.self_attn.v_proj with LoRA
Replacing model.layers.6.self_attn.o_proj with LoRA
Replacing model.layers.7.self_attn.q_proj with LoRA
Replacing model.layers.7.self_attn.v_proj with LoRA
Replacing model.layers.7.self_attn.o_proj with LoRA
Replacing model.layers.8.self_attn.q_proj with LoRA
Replacing model.layers.8.self_attn.v_proj with LoRA
Replacing model.layers.8.self_attn.o_proj with LoRA
Replacing model.layers.9.self_attn.q_proj with LoRA
Replacing model.layers.9.self_attn.v_proj with LoRA
Replacing model.layers.9.self_attn.o_proj with LoRA
Replacing model.layers.10.self_attn.q_proj with LoRA
Replacing model.layers.10.self_attn.v_proj with LoRA
Replacing model.layers.10.self_attn.o_proj with LoRA
Replacing model.layers.11.self_attn.q_proj with LoRA
Replacing model.layers.11.self_attn.v_proj with LoRA
Replacing model.layers.11.self_attn.o_proj with LoRA
Replacing model.layers.12.self_attn.q_proj with LoRA
Replacing model.layers.12.self_attn.v_proj with LoRA
Replacing model.layers.12.self_attn.o_proj with LoRA
Replacing model.layers.13.self_attn.q_proj with LoRA
Replacing model.layers.13.self_attn.v_proj with LoRA
Replacing model.layers.13.self_attn.o_proj with LoRA
Replacing model.layers.14.self_attn.q_proj with LoRA
Replacing model.layers.14.self_attn.v_proj with LoRA
Replacing model.layers.14.self_attn.o_proj with LoRA
Replacing model.layers.15.self_attn.q_proj with LoRA
Replacing model.layers.15.self_attn.v_proj with LoRA
Replacing model.layers.15.self_attn.o_proj with LoRA
```

```
LoRA is placed at model.layers.0.self_attn.q_proj
LoRA is placed at model.layers.0.self_attn.v_proj
LoRA is placed at model.layers.0.self_attn.o_proj
LoRA is placed at model.layers.1.self_attn.q_proj
LoRA is placed at model.layers.1.self_attn.v_proj
LoRA is placed at model.layers.1.self_attn.o_proj
LoRA is placed at model.layers.2.self_attn.q_proj
LoRA is placed at model.layers.2.self_attn.v_proj
LoRA is placed at model.layers.2.self_attn.o_proj
LoRA is placed at model.layers.3.self_attn.q_proj
LoRA is placed at model.layers.3.self_attn.v_proj
LoRA is placed at model.layers.3.self_attn.o_proj
LoRA is placed at model.layers.4.self_attn.q_proj
LoRA is placed at model.layers.4.self_attn.v_proj
LoRA is placed at model.layers.4.self_attn.o_proj
LoRA is placed at model.layers.5.self_attn.q_proj
LoRA is placed at model.layers.5.self_attn.v_proj
LoRA is placed at model.layers.5.self_attn.o_proj
LoRA is placed at model.layers.6.self_attn.q_proj
LoRA is placed at model.layers.6.self_attn.v_proj
LoRA is placed at model.layers.6.self_attn.o_proj
LoRA is placed at model.layers.7.self_attn.q_proj
LoRA is placed at model.layers.7.self_attn.v_proj
LoRA is placed at model.layers.7.self_attn.o_proj
LoRA is placed at model.layers.8.self_attn.q_proj
LoRA is placed at model.layers.8.self_attn.v_proj
LoRA is placed at model.layers.8.self_attn.o_proj
LoRA is placed at model.layers.9.self_attn.q_proj
LoRA is placed at model.layers.9.self_attn.v_proj
LoRA is placed at model.layers.9.self_attn.o_proj
LoRA is placed at model.layers.10.self_attn.q_proj
LoRA is placed at model.layers.10.self_attn.v_proj
LoRA is placed at model.layers.10.self_attn.o_proj
LoRA is placed at model.layers.11.self_attn.q_proj
LoRA is placed at model.layers.11.self_attn.v_proj
LoRA is placed at model.layers.11.self_attn.o_proj
LoRA is placed at model.layers.12.self_attn.q_proj
LoRA is placed at model.layers.12.self_attn.v_proj
LoRA is placed at model.layers.12.self_attn.o_proj
LoRA is placed at model.layers.13.self_attn.q_proj
LoRA is placed at model.layers.13.self_attn.v_proj
LoRA is placed at model.layers.13.self_attn.o_proj
LoRA is placed at model.layers.14.self_attn.q_proj
LoRA is placed at model.layers.14.self_attn.v_proj
LoRA is placed at model.layers.14.self_attn.o_proj
LoRA is placed at model.layers.15.self_attn.q_proj
LoRA is placed at model.layers.15.self_attn.v_proj
LoRA is placed at model.layers.15.self_attn.o_proj
Using device cuda
Dataset Loaded: 38105 training, 4234 validation samples.
```

Low-rank can reduce the number of learnable parameters!

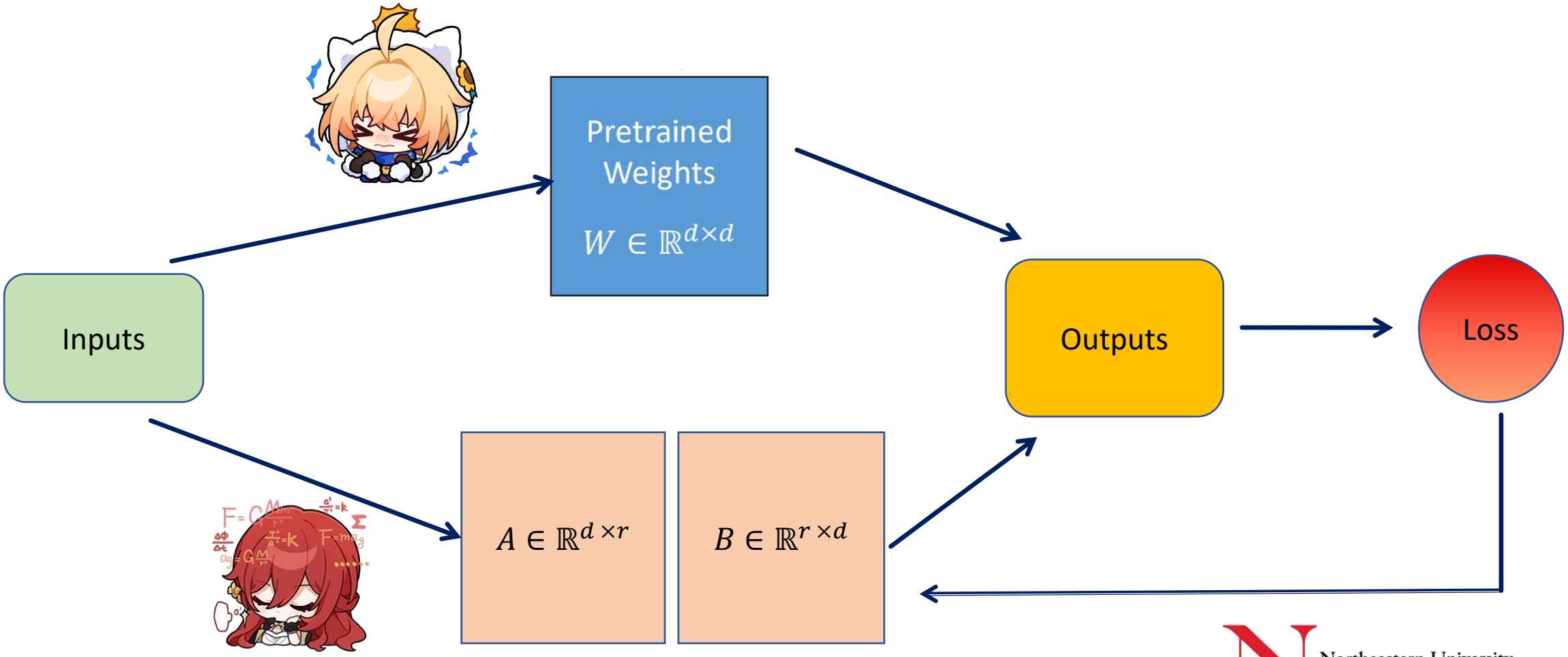
	pretrained	LorA+pretrained
Total learnable parameters	141557760	11010048

Only 8% parameters are learnable!

Northeastern University
School of Engineering

Model structure Fine-tune-----Work Flow

Total weights = Pretrained model weights + Low Rank weights



Northeastern University
School of Engineering

Data Process

Northeastern University
School of Engineering

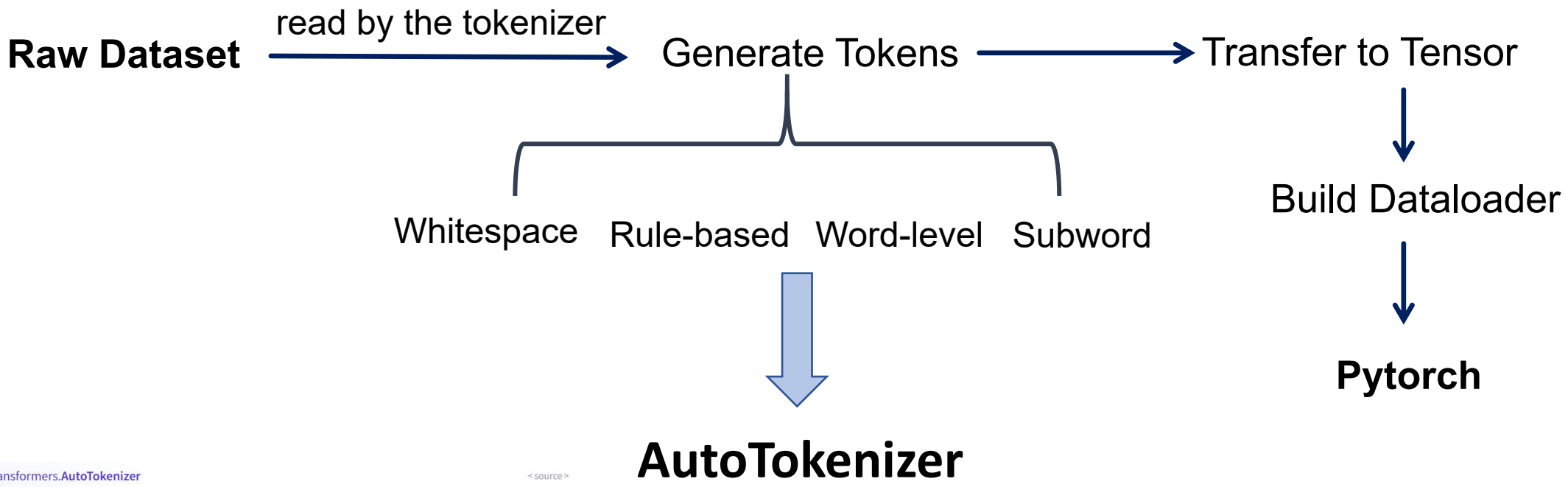
Data Process-----Datasets

Open-Orca/OpenOrca	
Size of Datasets	2.87GB
Number of rows	About 3M

```
OpenOrca Dataset
└── Sample (dict)
    ├── id: str
    ├── system_prompt: str
    ├── question: str
    └── response: str
```

Northeastern University
School of Engineering

Data Process-----Tokenizer



class transformers.AutoTokenizer <source>

( )

This is a generic tokenizer class that will be instantiated as one of the tokenizer classes of the library when created with the AutoTokenizer.from_pretrained() class method.

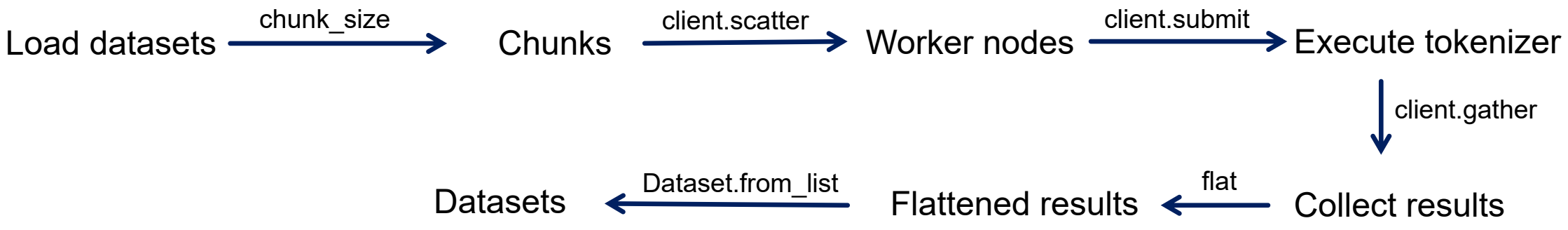
This class cannot be instantiated directly using `__init__()` (throws an error).

from_pretrained <source>

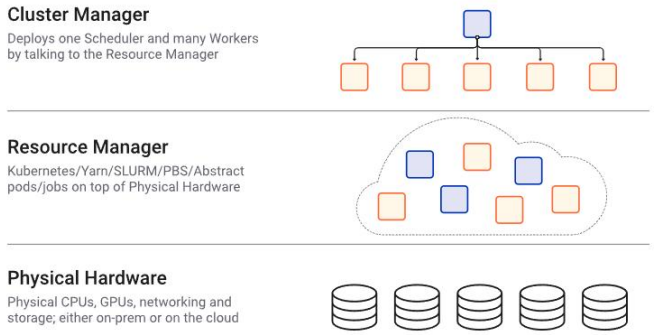
`( pretrained_model_name_or_path, *inputs, **kwargs )`

Northeastern University
School of Engineering

Data Process-----Dask



	Pre-Dask	Post-Dask (job=6)
Throughput (item/sec)	238.11	873.26



Northeastern University
School of Engineering

Optimization

Northeastern University
School of Engineering

Optimization

Speed Up

Quality

Northeastern University
School of Engineering

Speed Up

- Mixed precision
- DDP
- FSDP
- TP
- Pipeline
- ....

Northeastern University
School of Engineering

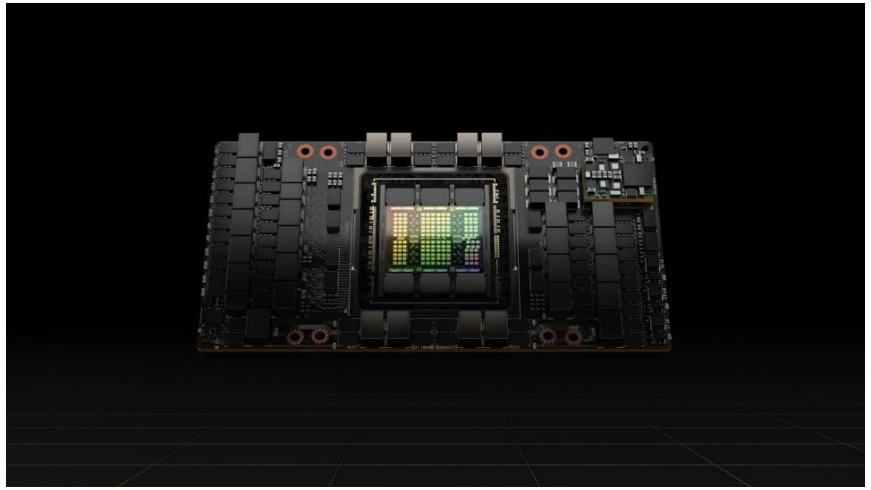
Optimization-----Mixed Precision

Format	Bits	precision	Memory	Use case
FP32	32	High	HIgh	pytorch default
FP16	16	Medium	Medium (almost half of FP32)	mixed precision, training and inference
BF16	16	Lower mantissa than FP16, but higher training stability	almost equal to FP16	mixed precision trainig and inference
FP8	8	lowest	lowest	quantized training and inference

Northeastern University
School of Engineering

Optimization-----Mixed Precision

Experiment setting	
num of epoch	5
batch_size	20
data_size	1% (38105 training samples, 4234 validation samples)
GPU	NVIDIA H100 80GB HBM3 ×15



```
+-----------------------------------------+
|   0  NVIDIA H100 80GB HBM3          On  |
| N/A   54C    P0             599W /  700W |
|                                         |
+-----------------------------------------+
```

Northeastern University
School of Engineering

Optimization-----Mixed Precision

Results：

FP32:

Blue (last epoch)	0.9968
Rouge_L (last epoch)	0.9995
Validation loss (last epoch)	1.5306
Epoch Time (average)	2573.025s
Total Training Time	12865.124s~3.57h
Throughput	16.2301 samples/sec
Max Memory Usage	57.6588 GB

BF16:

Blue (last epoch)	0.9972
Rouge_L (last epoch)	0.9996
Validation loss (last epoch)	1.5322
Epoch Time (average)	590.352s
Total Training Time	2,951.7622s~49.196min
Throughput	81.4855 samples/sec
Max Memory Usage	45.6404 GB

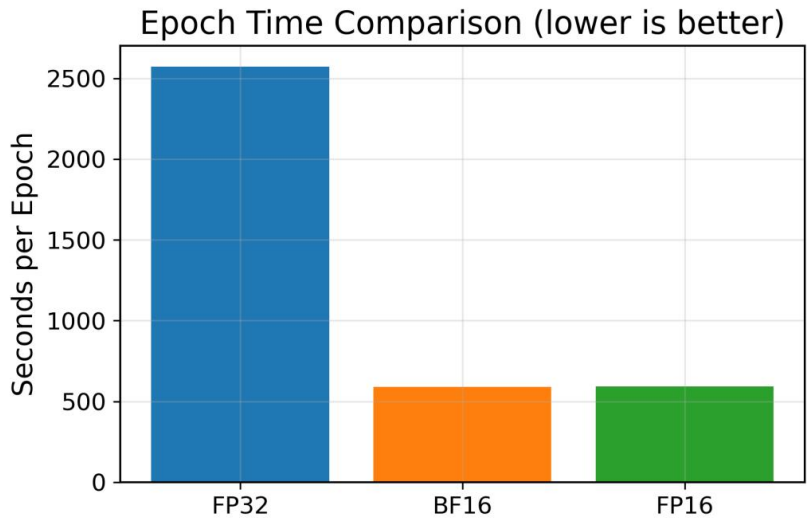
FP16:

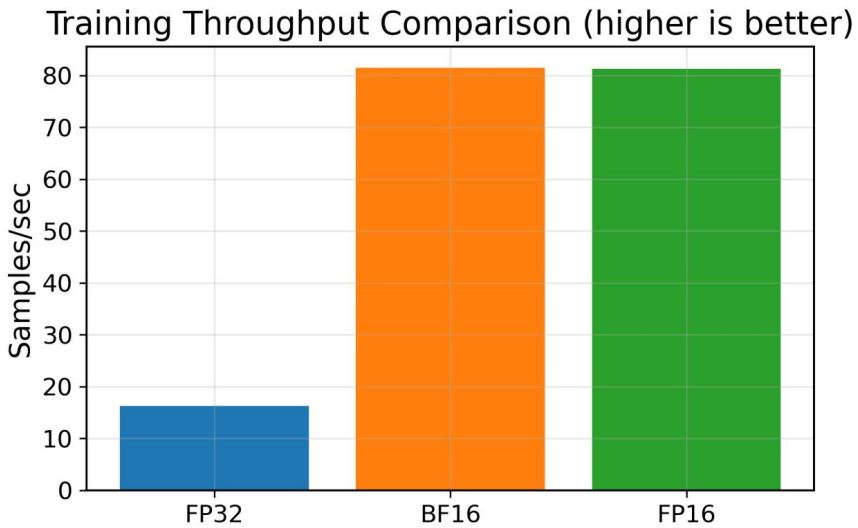
Blue (last epoch)	0.9951
Rouge_L	0.9986
Validation loss (last epoch)	1.5363
Epoch Time (average)	593.573s
Total Training Time	2,967.8669s = 49.464min
Throughput	81.2979 samples/sec
Max Memory Usage	45.6404GB

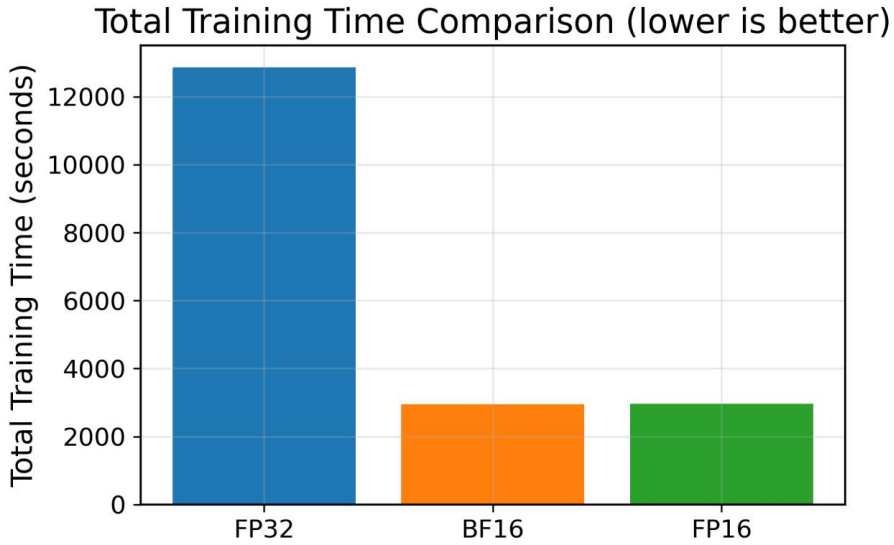
Northeastern University
School of Engineering

Optimization-----Mixed Precision

Results：Speed Comparision







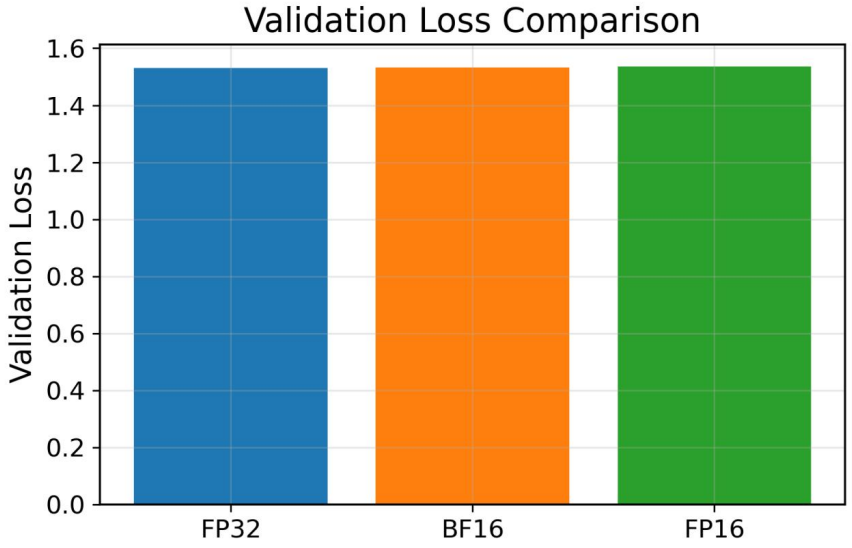
Both BF16 and FP16 increase training throughput by about 5× compared to FP32!

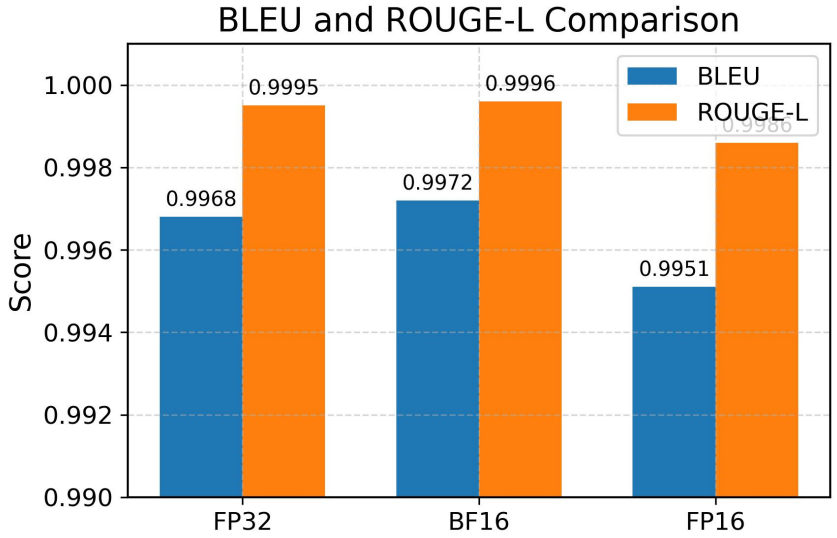
Both BF16 and FP16 reduce total training time by more than 4× compared to FP32.

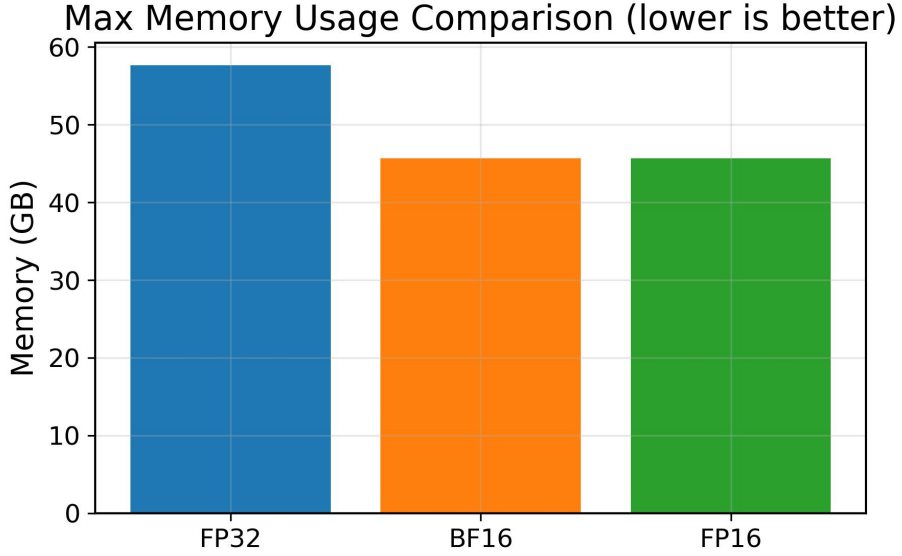
Northeastern University
School of Engineering

Optimization-----Mixed Precision

Results：Quality and Memory Usage comparision







Both BF16 and FP16 save ~20% GPU memory compared to FP32.

BLEU and ROUGE-L are very close among FP32, BF16, and FP16

Validation loss is also very close.

Northeastern University
School of Engineering

Optimization-----Mixed Precision

We can use BF16 or FP16 to speed up model running!

Northeastern University
School of Engineering

Optimization-----DDP

Experiment setting	
num of epoch	5
batch_size	24
data_size	3% (114316 training samples, 12702 validation samples)
GPU	NVIDIA H100 80GB HBM3 ×15
Mixed Precision	FP16

```
|=========================================+
|   0  NVIDIA H100 80GB HBM3          On  |
| N/A   53C    P0            596W /  700W |
|                                         |
+-----------------------------------------+
|   1  NVIDIA H100 80GB HBM3          On  |
| N/A   58C    P0            606W /  700W |
|                                         |
+-----------------------------------------+
|   2  NVIDIA H100 80GB HBM3          On  |
| N/A   59C    P0            609W /  700W |
|                                         |
+-----------------------------------------+
|   3  NVIDIA H100 80GB HBM3          On  |
| N/A   55C    P0            613W /  700W |
|                                         |
+-----------------------------------------+
|   4  NVIDIA H100 80GB HBM3          On  |
| N/A   53C    P0            613W /  700W |
|                                         |
+-----------------------------------------+
|   5  NVIDIA H100 80GB HBM3          On  |
| N/A   59C    P0            598W /  700W |
|                                         |
+-----------------------------------------+
|   6  NVIDIA H100 80GB HBM3          On  |
| N/A   61C    P0            618W /  700W |
|                                         |
+-----------------------------------------+
|   7  NVIDIA H100 80GB HBM3          On  |
| N/A   54C    P0            594W /  700W |
|                                         |
+-----------------------------------------+
```

Northeastern University
School of Engineering

Optimization-----DDP

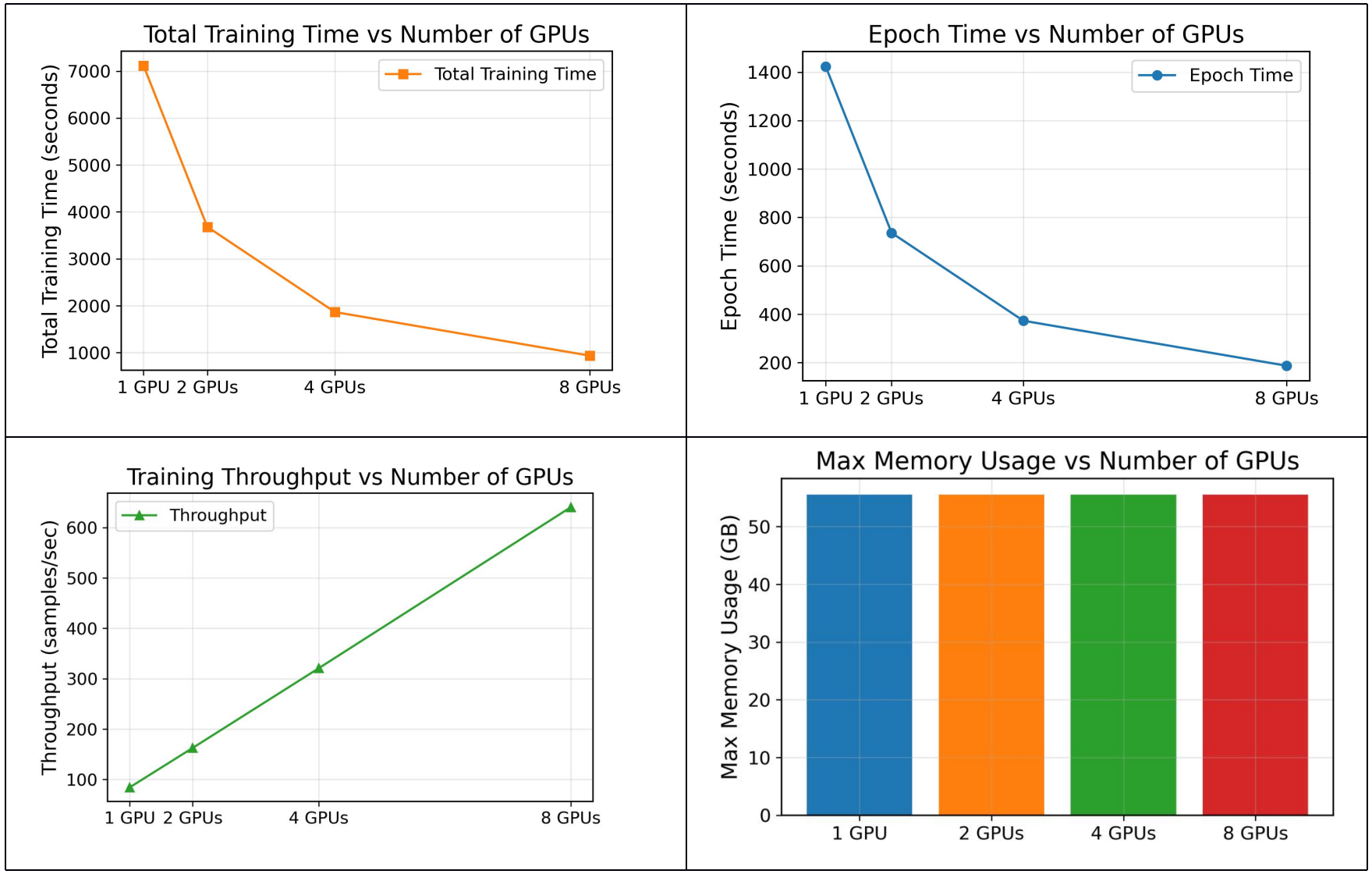
Results:

GPU number	Max memory Usage	Epoch Time (average)	Total Training Time	Throughput
1	55.5422 GB	1423.2714s	7,116.3569s ~ 1.98h	84.232 samples/second
2	55.548 GB	736.2461s	3,681.2036s ~ 1.02h	162.649 samples/second
4	55.5443GB	373.2647s	1,866.3235s ~ 31.105min	320.843 samples/second
8	55.541GB	187.1795s	935.8976s ~ 15.60min	640.639 samples/second

Northeastern University
School of Engineering

Optimization-----DDP

Results:



GPU ↑

Speed ↑

Northeastern University
School of Engineering

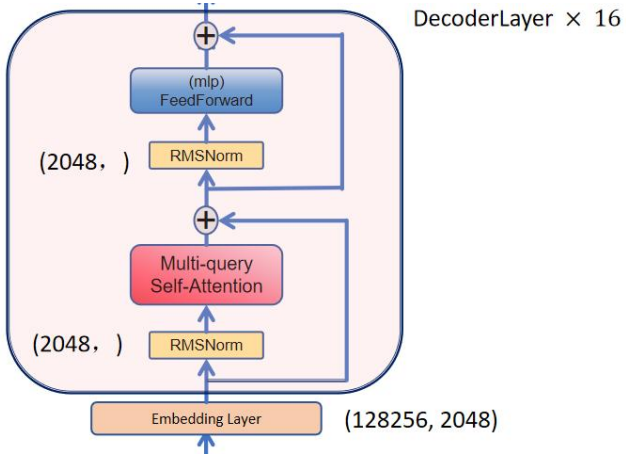
Optimization-----FSDP

Warp Embedding Layer and each DecoderLayer not wrapping separately:

RMSNorm

Attention module

MLP module individually



Experiment setting	
num of epoch	5
batch_size	20
data_size	20000
GPU	NVIDIA H100 80GB HBM3 ×15
Precision	FP32

```
|=========================================+
|   0  NVIDIA H100 80GB HBM3          On  |
| N/A   53C    P0            596W /  700W |
|                                         |
+-----------------------------------------+
|   1  NVIDIA H100 80GB HBM3          On  |
| N/A   58C    P0            606W /  700W |
|                                         |
+-----------------------------------------+
|   2  NVIDIA H100 80GB HBM3          On  |
| N/A   59C    P0            609W /  700W |
|                                         |
+-----------------------------------------+
|   3  NVIDIA H100 80GB HBM3          On  |
| N/A   55C    P0            613W /  700W |
|                                         |
+-----------------------------------------+
|   4  NVIDIA H100 80GB HBM3          On  |
| N/A   53C    P0            613W /  700W |
|                                         |
+-----------------------------------------+
|   5  NVIDIA H100 80GB HBM3          On  |
| N/A   59C    P0            598W /  700W |
|                                         |
+-----------------------------------------+
|   6  NVIDIA H100 80GB HBM3          On  |
| N/A   61C    P0            618W /  700W |
|                                         |
+-----------------------------------------+
|   7  NVIDIA H100 80GB HBM3          On  |
| N/A   54C    P0            594W /  700W |
|                                         |
+-----------------------------------------+
```

Northeastern University
School of Engineering

Optimization-----FSDP

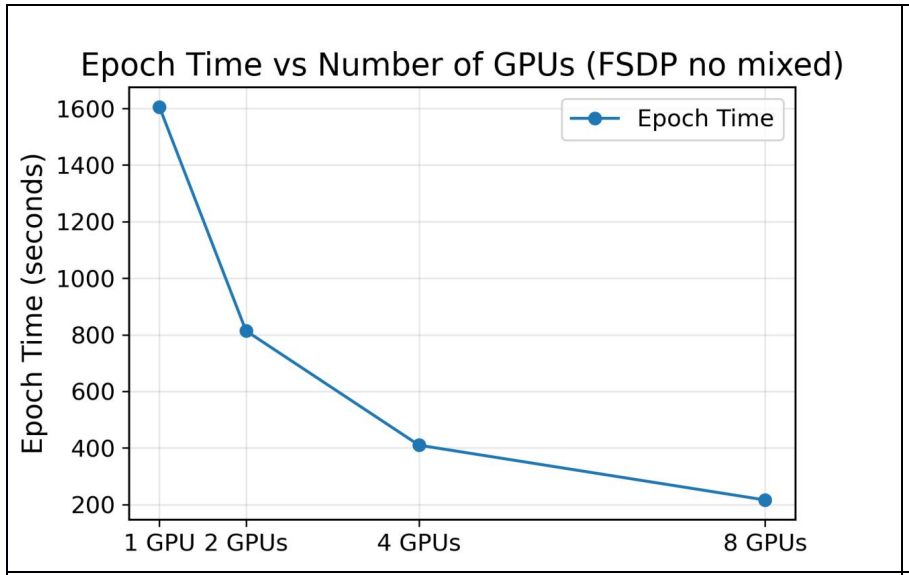
Results:

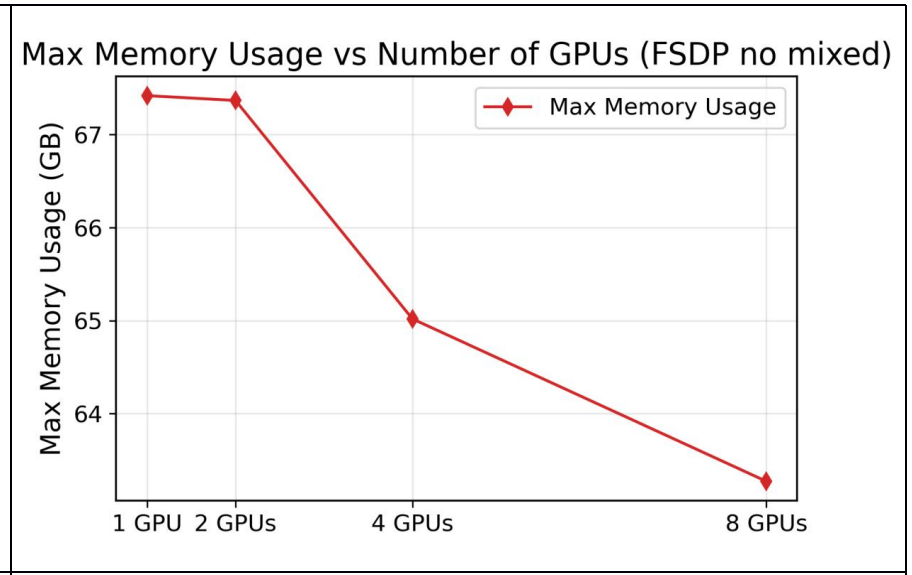
GPU number	Max memory Usage	Epoch Time (average)	Total Training Time	Throughput
1	67.4191GB	1605.5199s	8,027.5996s ~ 2.23h	11.638 samples/second
2	67.367GB	814.1584s	4,070.792s ~ 1.13h	22.942 samples/second
4	65.0189GB	409.7302s	2,048.6511s ~ 34.14min	45.609 samples/second
8	63.2731GB	216.0439s	1,080.2196s ~ 18.00min	86.510 samples/second

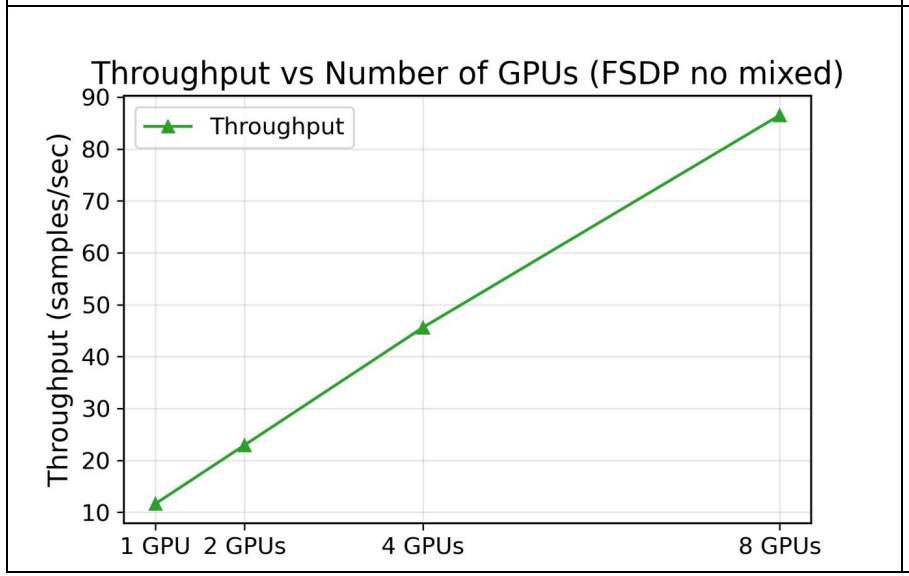
Northeastern University
School of Engineering

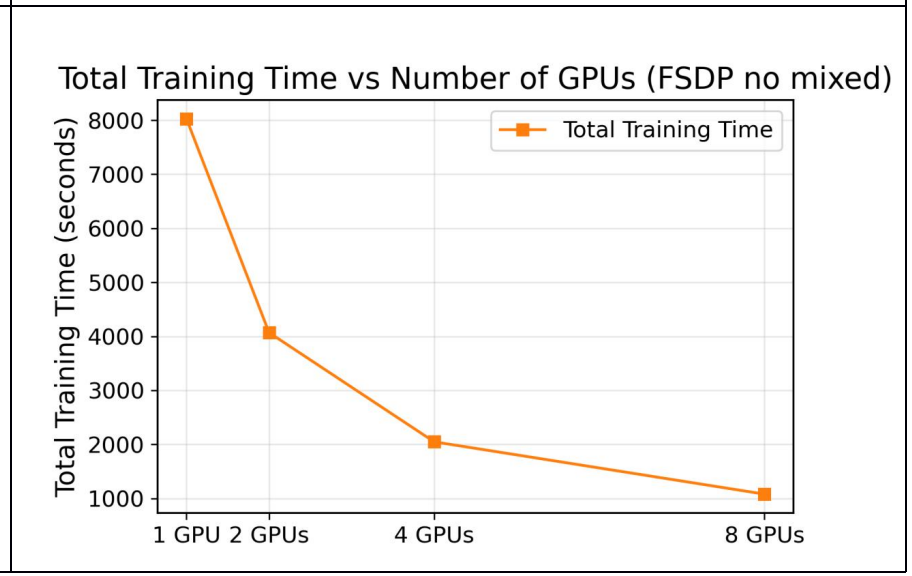
Optimization-----FSDP

Results:







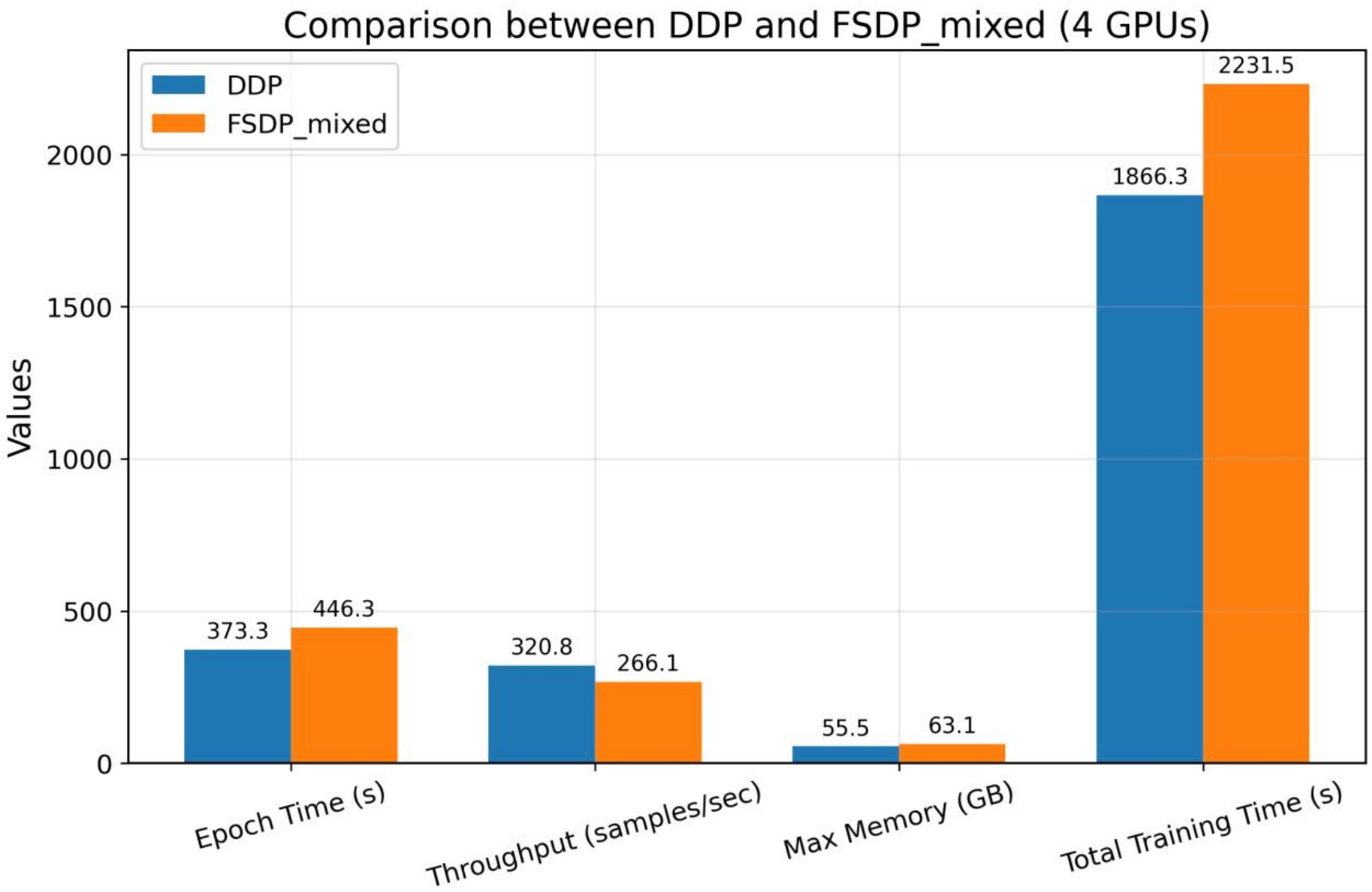


GPU ↑

Speed ↑

Northeastern University
School of Engineering

Optimization-----FSDP and DDP comparasion



DDP may be better for our model accelerate!

Northeastern University
School of Engineering

Optimization-----Quality Improvment

Experiment

Experiment setting	
num of epoch	4
batch_size	24
data_size	30% (1143159 training samples, 127018 validation samples)
GPU	NVIDIA H100 80GB HBM3 ×15
Precision	FP16
Parallelism	DDP

Results

Metric	Pretrained Model	LoRA Fine-tuned Model	Improvement
BLEU	9.17	9.24	+0.76%
ROUGE-L	18.81	19.06	+1.33%

Northeastern University
School of Engineering

Future Work

Future work

- Larger dataset: 30% →Full Dataset (~3M)
- Fine-tune more parts: q_proj, k_proj, o_proj → attention+MLP+...
- Larger LoRA parameters: adjustment
- More num of epochs: 5 →6~8

May be better performance and obvious improvment!

Northeastern University
School of Engineering

Thanks!

Northeastern University
School of Engineering