Parallelism Pipeline and Compression Algorithm

Introduction

Parallelism Pipeline is a method to accelerate training speed by dividing batch into minibatches and train these batches parallely. Also, we try to use compression algorithm to

1. Distributed Pipeline Parallelism Training

Gpipe training efficiency compares to data-parallelism.

1.1 Vision Tasks

Experiment	Dataset	Backend	GPUs	Batch size	Learning rate	Top-1 acc (%)	Throughput	Speed up
Dataparallel-2gpu	CIFAR10	MobilenetV2	2	64	0.005	95.83±0.04	376.47/s	1×
Pipeline-2gpu	CIFAR10	MobilenetV2	2	64(4 chunks)	0.005	95.89±0.07	228.57/s	0.607×
Pipeline-2gpu(origin code)	CIFAR10	MobilenetV2	2	64(4 chunks)	0.005	95.87	213.33/s	0.566×
Pipeline-4gpu	CIFAR10	MobilenetV2	4	256(4 chunks)	0.02	96.03±0.14	400.30/s	1.07×
Pipeline-4gpu(origin code)	CIFAR10	MobilenetV2	4	256(4 chunks)	0.005	95.89	419.67/s	1.11×
Pipeline-4gpu	CIFAR10	MobilenetV2	4	256(8 chunks)	0.02	96.07±0.05	397.30/s	1.06×
Dataparallel-4gpu	CIFAR10	MobilenetV2	4	256	0.02	95.94±0.09	627.22/s	1.66×

1.2 NLP Tasks

Experiment	Dataset	Backend	GPUs	Batch size	Learning rate	Top-1 acc (%)	Throughput	Speed up
Dataparallel-2gpu	RTE	Roberta	2	32	2e-5	79.0±0.27	76.19/s	1×
Pipeline-2gpu	RTE	Roberta	2	32(4 chunks)	2e-5	78.59±0.21	61.53/s	0.80×
Pipeline-2gpu	RTE	Roberta	2	64(4 chunks)	4e-5	77.56±0.39	68.82/s	0.90×
Pipeline-4gpu	RTE	Roberta	4	64(4 chunks)	4e-5	78.17±0.44	106.40/s	1.40×
Pipeline-4gpu	RTE	Roberta	4	64(2 chunks)	4e-5	78.15±0.22	96.40/s	1.27×
Dataparallel-4gpu	RTE	Roberta	4	64	4e-5	78.4±0.21	95.53/s	1.25

2.Sort Quantization

Here is the pseudocode

Algorithm 1 Sort Quantization

input: A tensor with size 1xl T,Split bits S, Quantization bit Q
output: A tensor with size 1xl O, mins' list M, steps' list P
$T, Index \leftarrow Sort(T)$
$T \leftarrow Chunk(T, 2^S)$
$Index \leftarrow Chunk(Index, 2^S)$
for $i \leftarrow 0, i < 2^S$ **do**
  $M[i], P[i], T[i] \leftarrow UniformQuantization(T[i], Q)$
  $O \leftarrow InsertWithIndex(O, T[i], Index[i])$
end for

Algorithm 2 Fast Quantization

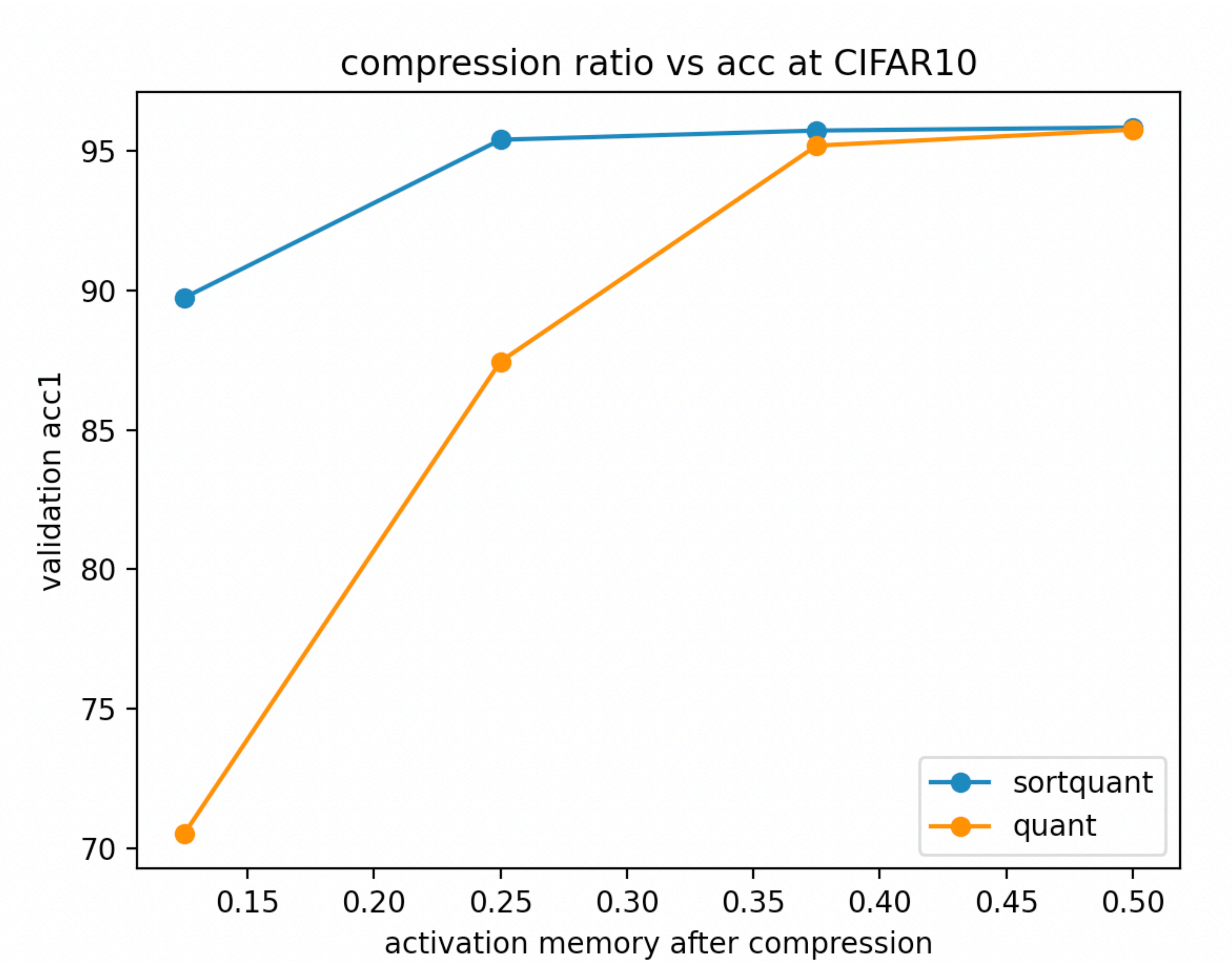
input: A tensor with size 1xl T,Split bits S, Quantization bit Q
output: A tensor with size 1xl O, mins' list M, steps' list P
for $i \leftarrow 0, i < 2^S$ **do**
  $K, Index \leftarrow Kthvalue(T)$
  $T[i] \leftarrow Findvalue(T, K)$
  $M[i], P[i], T[i] \leftarrow UniformQuantization(T[i], Q)$
  $O \leftarrow InsertWithIndex(O, T[i], Index[i])$
end for

2.1Ablation Study

Ablation Studys are performed by using pipeline parallelism.

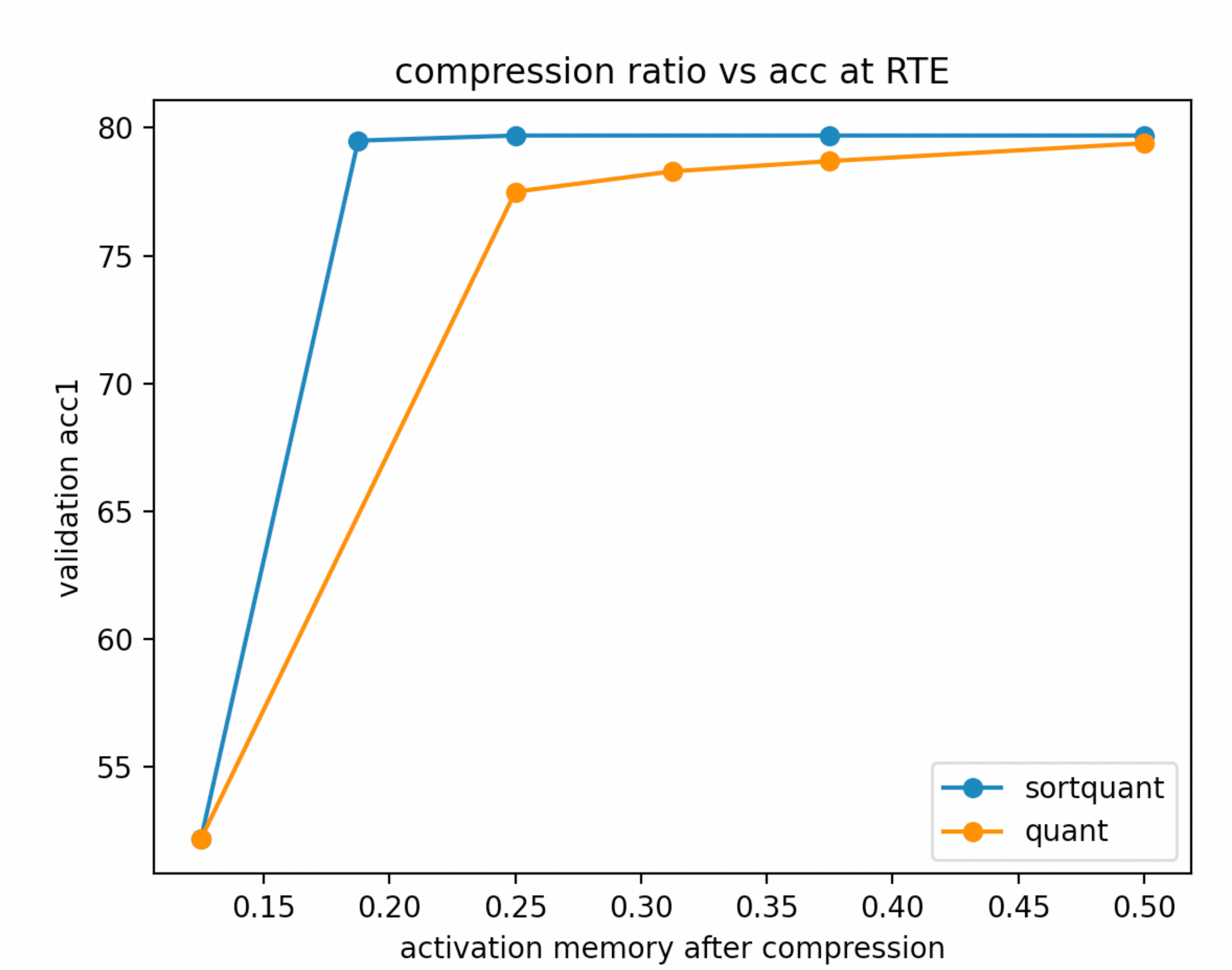
Settings For CIFAR10

Epochs	Lr	Batch Size	Scheduler	Optimizer
40	0.01	256	Cosine	SGD



Settings For RTE

Epochs	Lr	Batch Size	Scheduler	Optimizer
20	2e-5	32	Poly	AdamW



2.2 Altogether Ablation Study

These tests are all done in the environment of parallelism pipeline.

All bandwidths are detected by nccl_test.

https://github.com/NVIDIA/nccl-tests

CIFAR10

Backend:MobileNetV2

Client-Server Partition: First and last layer

Chunk: 4

Batchsize	Activation Memory Size(al together)	Compression Method(default3:1)	Compression Ratio	Validation Acc
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	None	1	95.87%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 16bits	0.5	95.84%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 12bits	0.375	95.73%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 8bits	0.25	95.68%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 4bits	0.125	87.10%

CIFAR100

Backend:MobileNetV2

Client Server Partition: First and last layer

Since the activation memory size is the same as the CIFAR10 dataset, the bandwidth is the same as the bandwidth in CIFAR10

Batchsize	Activation Memory Size(al together)	Compression Method(default3:1)	Compression Ratio	Validation Acc
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	No	1	80.92%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 16bits	0.5	80.85%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 12bits	0.375	80.61%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 8bits	0.25	78.83%

FOOD101

Backend:MobileNetV2

Client Server Partition: First and last layer

Since the activation memory size is the same as the CIFAR10 dataset, the bandwidth is the same as the bandwidth in CIFAR10

Batchsize	Activation Memory Size(al together)	Compression Method(default3:1)	Compression Ratio	Validation Acc
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	No	1	83.76%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 16bits	0.5	83.77%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 12bits	0.375	83.72%
256(8 chunks)	[256,32,112,112] [256,1280,7,7]	Sort Quantization 8bits	0.25	

RTE

Backend:Roberta-base

Client Server Partition: First two and last two layers

Batchsize	activation memory size(al together)	Compression method(default3:1)	compression ratio	Validation acc(in cola is Matthew)
32(4 chunks)	[32,128,768],[32,128,768]	None	1	78.9%
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 16bits	0.5	79.6%±0.18%
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 12bits	0.375	79.6%±0.20%
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 8bits	0.25	79.4%±0.21%
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 4bits	0.125	52.2%

COLA

Backend:Roberta-base

Client Server Partition: First two and last two layers

Since the activation memory size is the same as the RTE dataset, the bandwidth is the same as the bandwidth in RTE

Batchsize	Activation Memory Size(Al together)	Compression Method(default3:1)	Compression Ratio	Matthew's Corelation
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 16bits	0.5	64.5±0.48
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 12bits	0.375	63.93±0.22
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 8bits	0.25	63.20±0.12
32(4 chunks)	[32,128,768],[32,128,768]	Sort Quantization 4bits	0.125	0

3 Reproduce

Here is how to reproduce the sort quantization ablation study.

```
bash ./test.sh
```

Github Repo

https://github.com/timmywanttolearn/fast_pytorch_kmeans

I modify the repo to allow the fast k-means to run on multiple devices.

https://github.com/KinglittleQ/torch-batch-svd