Deep Dive: PyTorch 2.0 on Graviton

Hahnara Hyun

Sr Specialist SA, EC2 Graviton

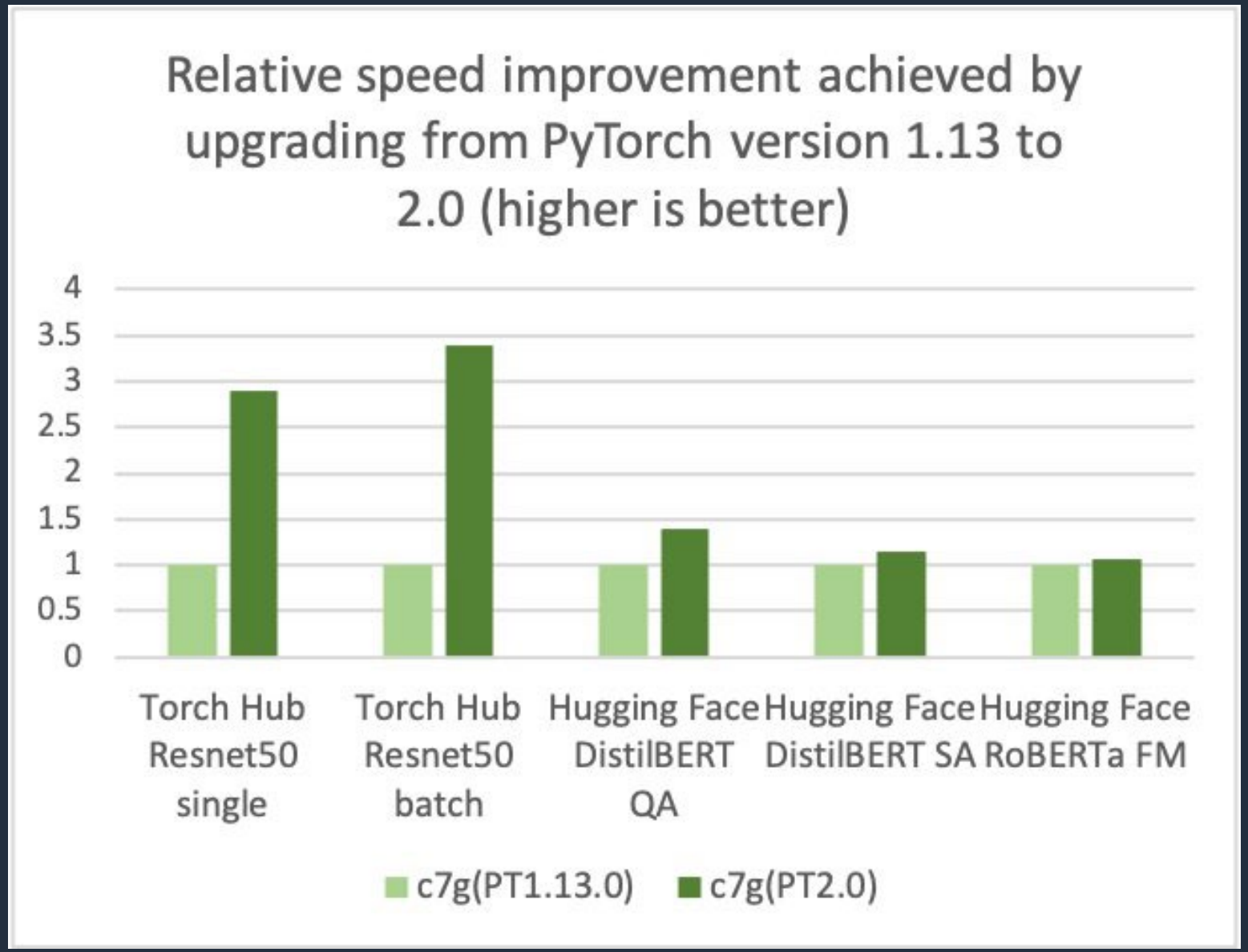
What does PyTorch 2.0 mean for Graviton?

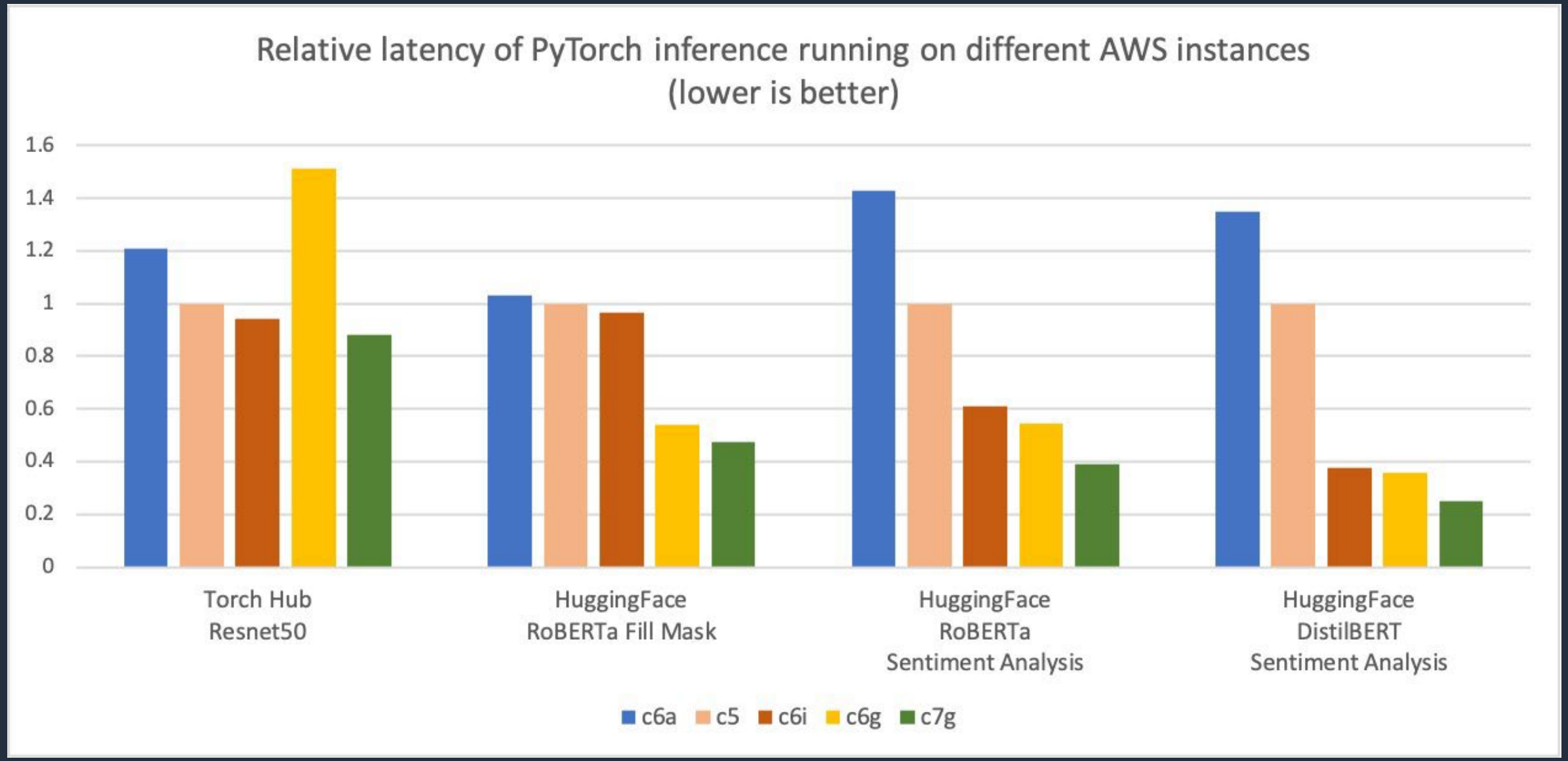
Overview

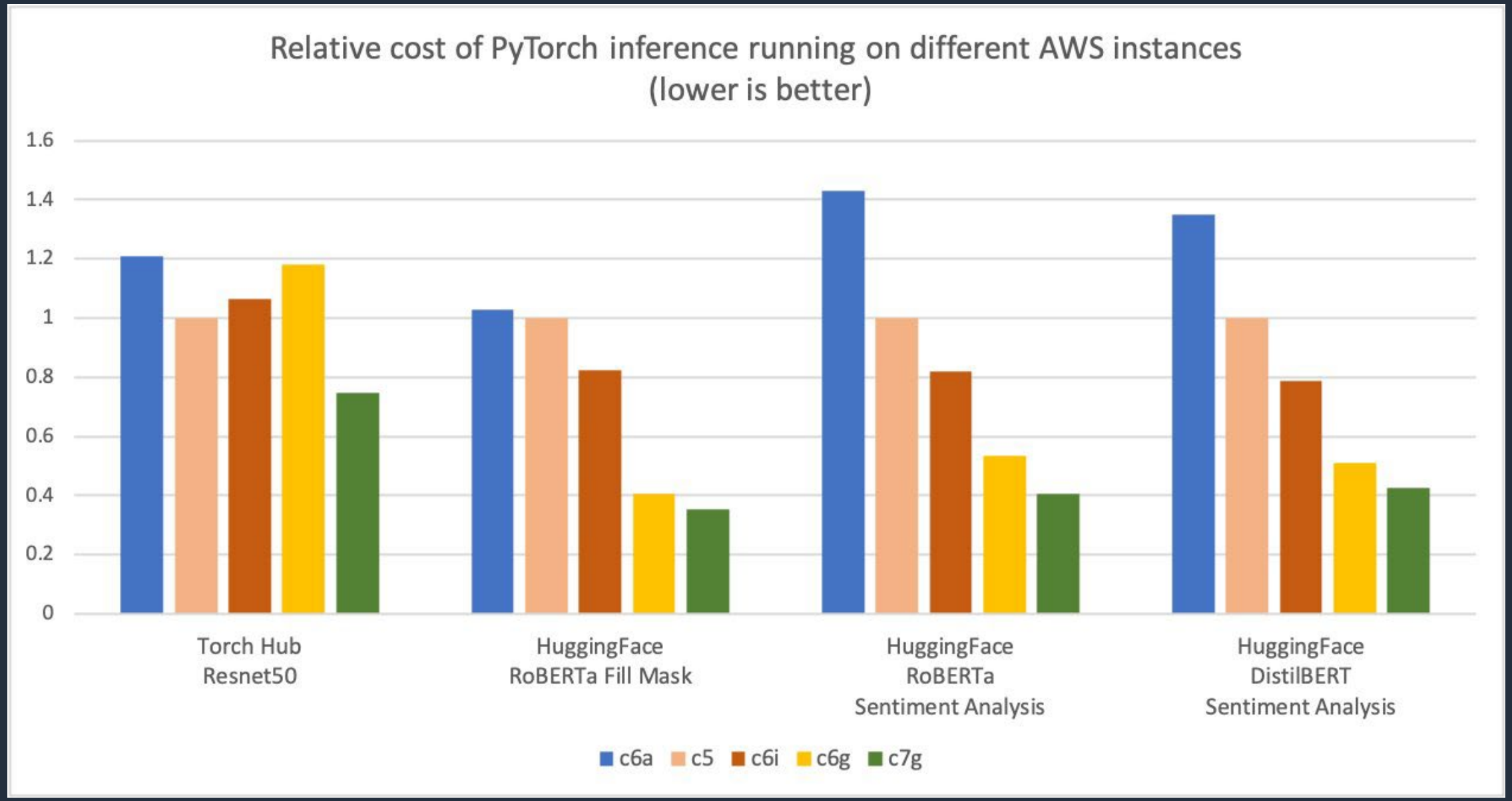
- PyTorch 2.0 on Graviton up to
  - 3.5x faster for Resnet50
  - 1.4x faster for BERT
- Graviton is now the fastest compute optimized instance on AWS Resnet50 and BERT
- 50% cost savings for PyTorch inference with Graviton3

Optimizations on Graviton

- PyTorch supports Arm Compute Library (ACL)
  - Provides Neon and SVE optimized GEMM kernels for bfloat16
- Graviton3 bfloat16 support
  - Enables deployment of models using bfloat16, fp32, AMP
  - fp32 models use bfloat16 kernels w/o model quantization
- Primitive caching
  - Reduce redundant GEMM kernel initialization







What’s new with PyTorch 2.0?

torch.compile()

Compiler Breakdown

TorchDynamo
- New technique for graph acquisition

AOTAutograd

- Allows acceleration of both forwards and backwards traces

PrimTorch

- Reduced and stabilized operator sets

TorchInductor

- New PyTorch compiler backend

Example code

```
model =
torchvision.models.resnet18().cuda()

tcompile_model = torch.compile(model)
```

Debugging

- The Minifier
  - Produces snippet of code reproducing original issue
  - Identifying where error occurs (e.g. AOTAutograd, eager) provides more specific troubleshooting
    - Set environment variable to
      - TORCHDYNAMO_REPRO_AFTER="dynamo”
      - TORCHDYNAMO_REPRO_AFTER="aot”

Backwards Incompatible Changes

Upgrade

- Linux Kernel >= 5.10
  - For best PyTorch inference performance on Graviton3
- Python version >= 3.8
- CUDA version >= 11.0

Backwards Incompatible Changes cont...

Attribute updates

1. Gradients set to None instead of zeros by default

PyTorch 1.13

```
>>> module.zero_grad()
>>> module.weight.grad == None
False
>>> module.weight.grad.data
 tensor([[0., 0.], [0., 0.]])
```

PyTorch 2.0

```
>>> module.zero_grad()
>>> module.weight.grad == None
True
>>> module.weight.grad.data
AttributeError: 'NoneType' object has
no attribute 'data'
```

2. Store attributes using `torch.utils.weak.WeakTensorKeyDictionary`

How to get started on Graviton

Python 2.0 on Graviton w SageMaker or EC2

AWS Deep Learning Container

```
sudo apt-get update

sudo apt-get -y install awscli docker

# Login to ECR

aws ecr get-login-password --region us-east-1 \
| docker login --username AWS \ --password-
stdin 763104351884.dkr.ecr.us-east-
1.amazonaws.com

# Pull the AWS DLC for pytorch

docker pull 763104351884.dkr.ecr.us-east-
1.amazonaws.com/pytorch-inference-
graviton:2.0.0-cpu-py310-ubuntu20.04-ec2
```

Python Wheel

```
sudo apt-get update

sudo apt-get install -y python3 python3-pip

 # Upgrade pip3 to the latest version

python3 -m pip install --upgrade pip

# Install PyTorch and extensions

python3 -m pip install torch

python3 -m pip install torchvision torchaudio
torchtext

# Turn on Graviton3 optimization

export DNNL_DEFAULT_FPMATH_MODE=BF16

export LRU_CACHE_CAPACITY=1024
```

Optimal Configurations

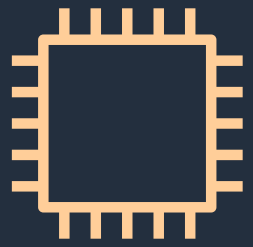
```
grep -q bf16 /proc/cpuinfo && export DNNL_DEFAULT_FPMATH_MODE=BF16
export LRU_CACHE_CAPACITY=1024


export THP_MEM_ALLOC_ENABLE=1


num_vcpus=$(getconf _NPROCESSORS_ONLN)
num_processes=<number of processes>
export OMP_NUM_THREADS=$((1 > ($num_vcpus/$num_processes) ? 1 :
($num_vcpus/$num_processes)))
export OMP_PROC_BIND=false
export OMP_PLACES=cores
```

Further Performance and Cost Optimization

Amazon SageMaker Inference Recommender



Instance Recommendation

Load tests

Endpoint recommendations

Looking forward

Check for updates on

- AWS Graviton Technical Guide
  - https://github.com/aws/aws-graviton-getting-started/blob/main/machinelearning/pytorch.md
- Ask the engineers: 2.0 Live Q&A series
  - https://pytorch.org/get-started/pytorch-2.0/#ask-the-engineers-20-live-qa-series

Thank you!

Hahnara Hyun

hahnarh@amazon.com

@HahnaraHyun