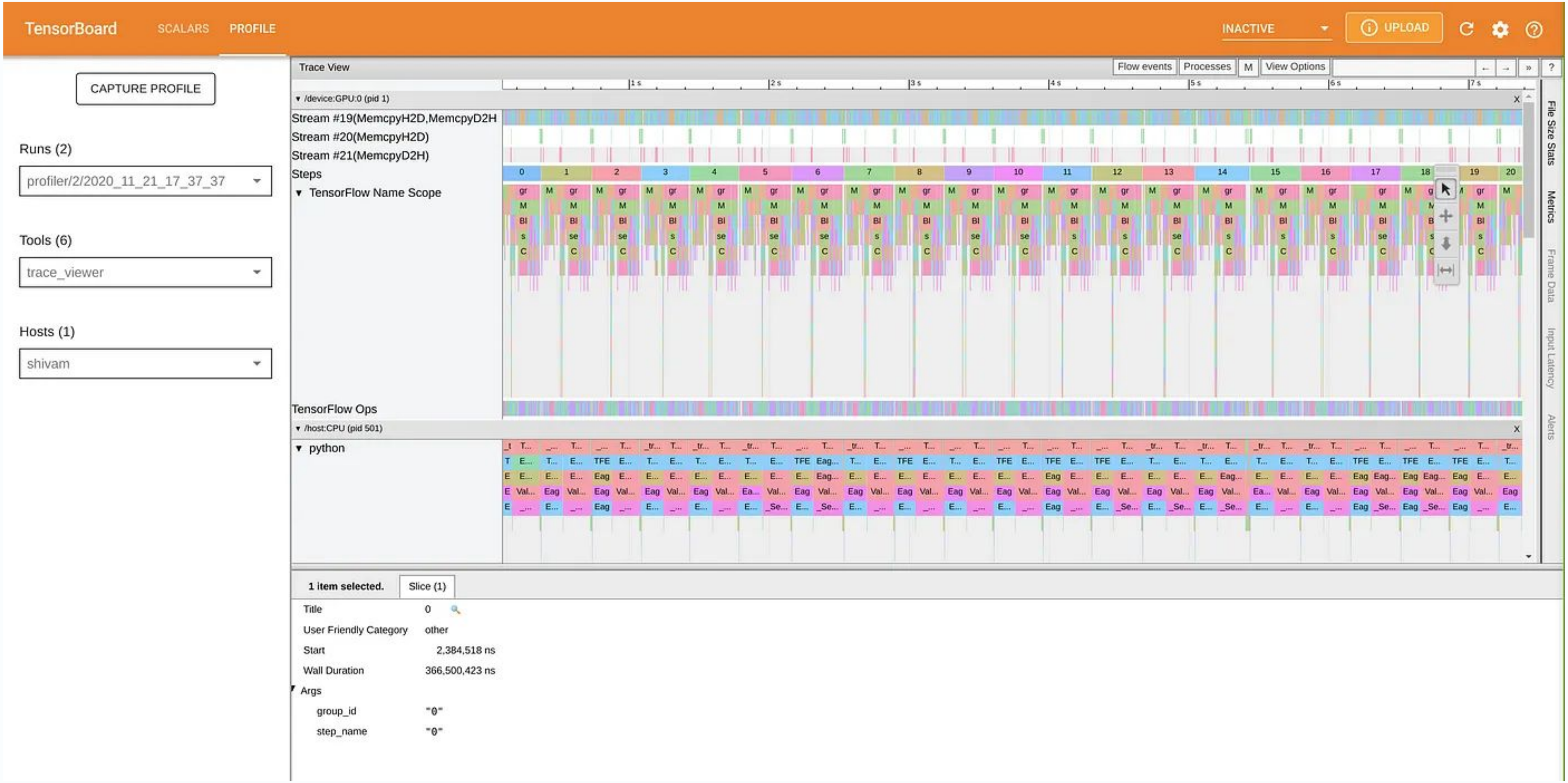
University of Sheffield

Profiling GPU-driven machine learning code

Why is my machine learning code slow?

Edwin Brown 05/07/2024

Whatuni? Student Choice Awards
UNIVERSITY OF THE YEAR 2024



Runs (2)

profiler/2/2020_11_21_17_37_37

Tools (6)

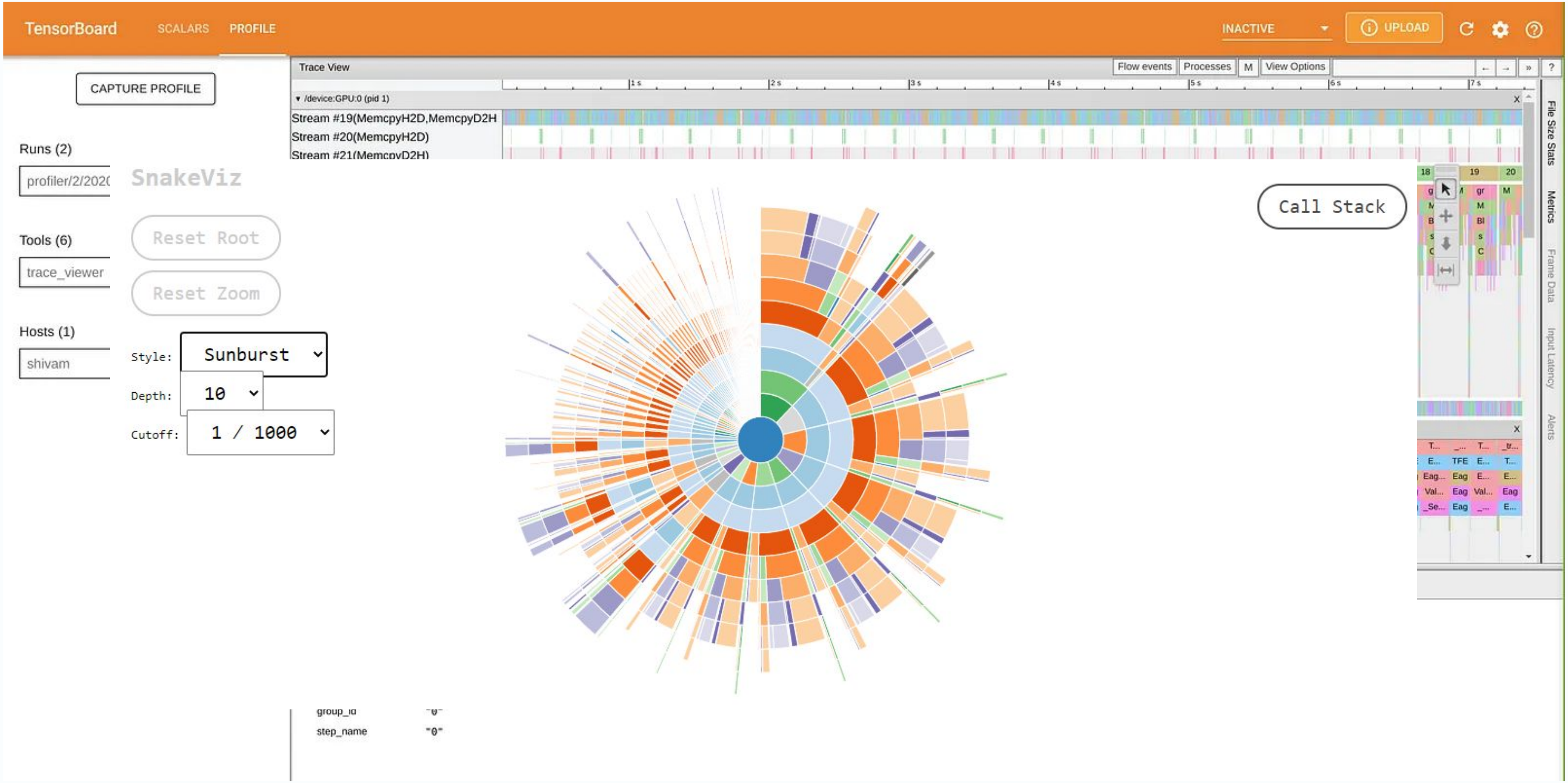
trace_viewer

Hosts (1)

shivam

1 item selected. Slice (1)

Title	0
User Friendly Category	other
Start	2,384,518 ns
Wall Duration	366,500,423 ns
Args	
group_id	"0"
step_name	"0"



Runs (2)
profiler/2/202(

Tools (6)
trace_viewer

Hosts (1)
shivam

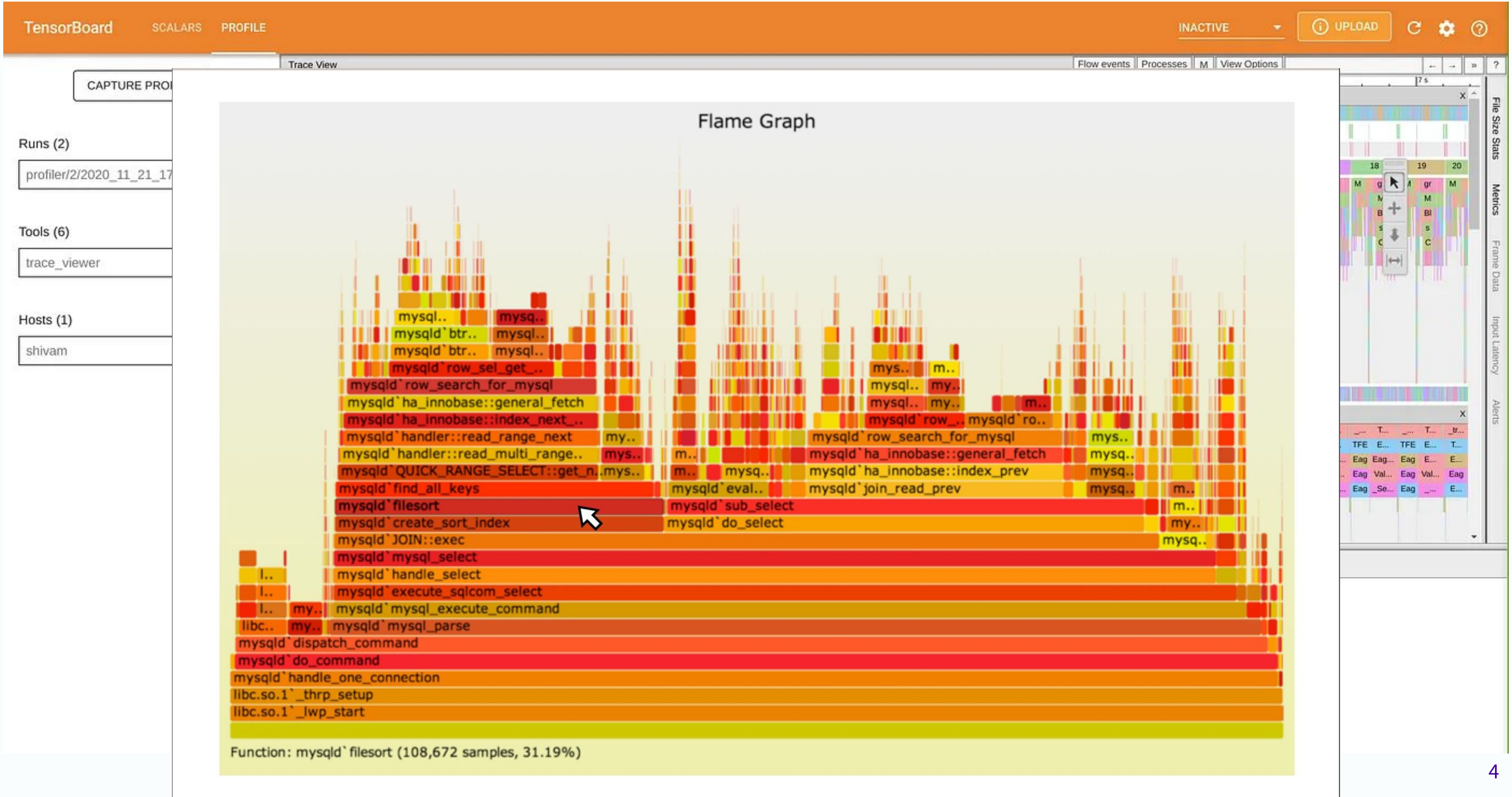
Style: Sunburst

Depth: 10

Cutoff: 1 / 1000

```
group_id      "0"
step_name     "0"
```

Call Stack



Runs (2)

profiler/2/2020_11_21_17

Tools (6)

trace_viewer

Hosts (1)

shivam

Function: mysqld`filesort (108,672 samples, 31.19%)

CAPTURE PROF

Runs (2)

profiler/2/2020_11_21_17

Tools (6)

trace_viewer

Hosts (1)

shivam

Profiling - What is it?

Profiling is a way of analysing how code is running

Statistical data is collected by a profiler when running code

Data can include what processes were run, how long they took and memory usage

Profiling - Why is it important?

Accessible—allows code to use fewer resources (less $$$)

Sustainable—reduce energy consumption (and CO2 emissions)

Scalable—efficiently utilise supercomputer resources

In particular, Machine Learning algorithms often rely on the use of GPUs which are particularly in demand and power hungry.

Profiling - How do I do it (Pytorch)?

```
my_schedule = schedule(skip_first=5, wait=2, warmup=2, active=5)
```

PyTorch Profiler — PyTorch Tutorials 2.3.0+cu121 documentation

Profiling - How do I do it (Pytorch)?

```
my_schedule = schedule(skip_first=5, wait=2, warmup=2, active=5)


profiler = torch.profiler.profile(
    schedule=my_schedule,
    on_trace_ready=torch.profiler.tensorboard_trace_handler(‘logs/test1’))
```

PyTorch Profiler — PyTorch Tutorials 2.3.0+cu121 documentation

Profiling - How do I do it (Pytorch)?

```
my_schedule = schedule(skip_first=5, wait=2, warmup=2, active=5)

profiler = torch.profiler.profile(
    schedule=my_schedule,
    on_trace_ready=torch.profiler.tensorboard_trace_handler(‘logs/test1’))

profiler.start()
for data in train_loader:
    train_step(data)
    profiler.step()
profiler.stop()
```

***.pt.trace.json file is saved to disk!**

PyTorch Profiler — PyTorch Tutorials 2.3.0+cu121 documentation

Profiling - How do I view the logs?

Perfetto - Browser based trace viewer

Tensorboard - Application for viewing training and profiling statistics developed by TensorFlow.

Holistic Trace Analysis - Open source library for interpreting logs output from pytorch profiler.

Experiment

- Train a computer vision model (Simple Unet model) to perform segmentation on 64,000 images (32,000 image and mask respectively) in Pytorch.

- Images are stored as .png files on disk. Image pipeline loads these images and masks from disk and does some pre-processing (rotation, resizing, etc.).

- **Can we use Profiling to find and remove bottlenecks to speed up our training?**

Hardware

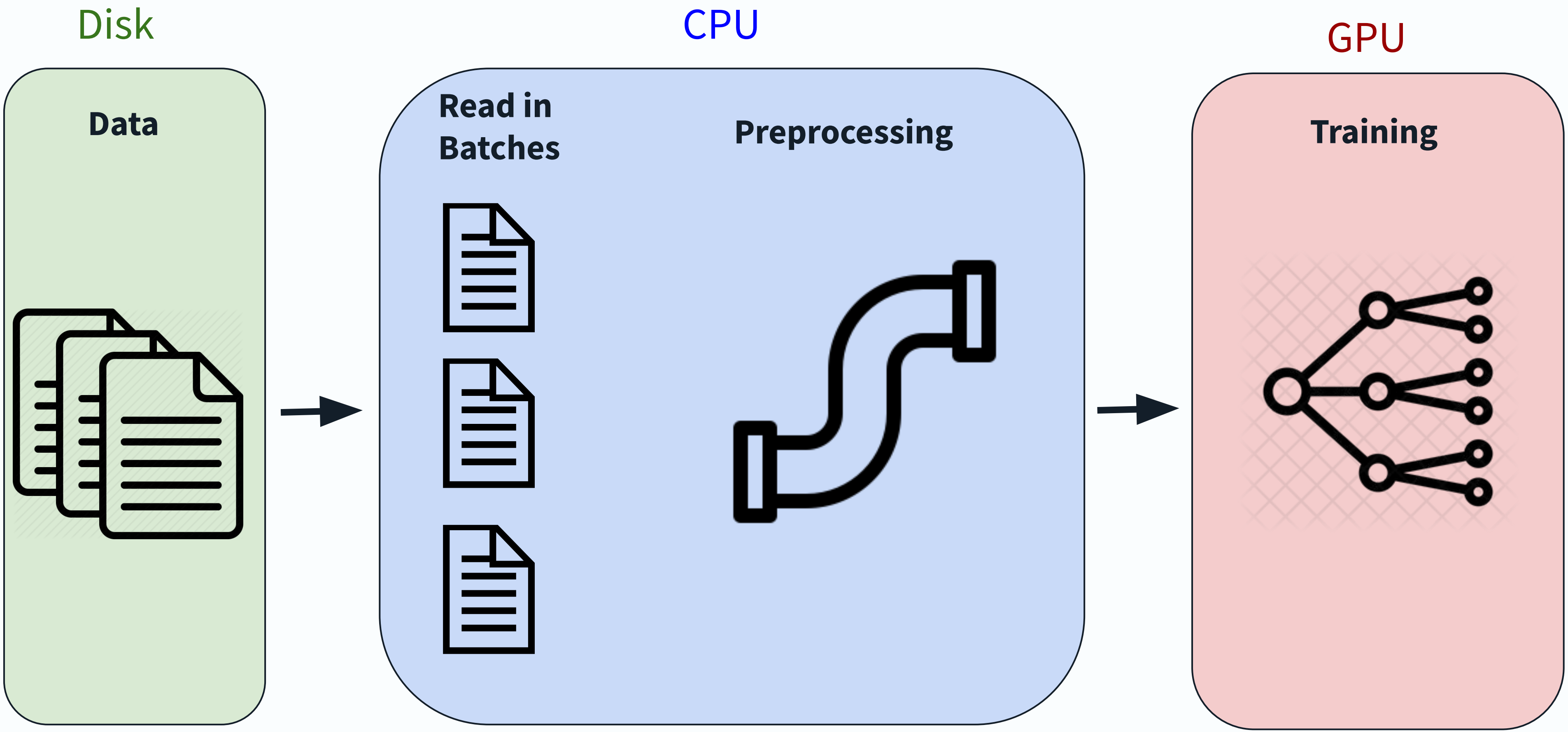
Virtual Machine

Intel i7-5930K, 6 cores, 12 threads

64GB Memory

GPUs:

- Geforce RTX 3090 (CUDA Device 0)
- Geforce RTX 3090 (CUDA Device 1)



Disk

Data

CPU

Read in Batches

Preprocessing

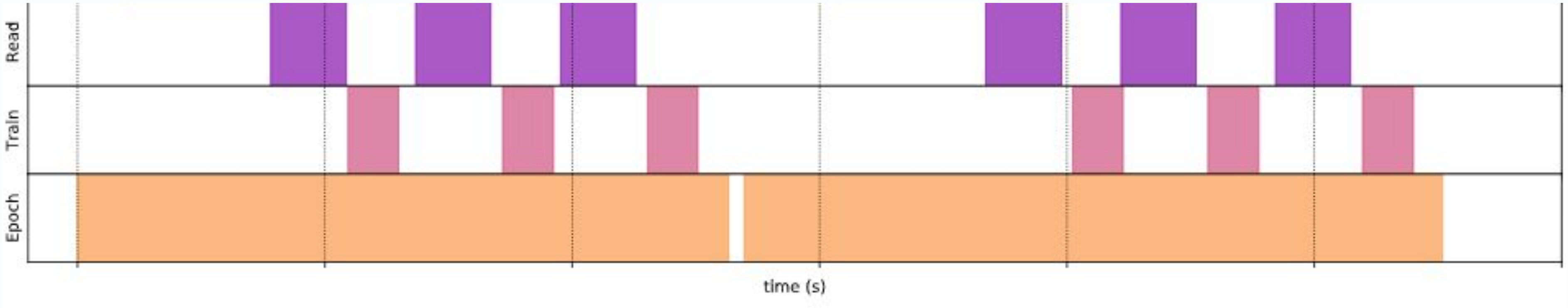
GPU

Training

Test 1 - Default

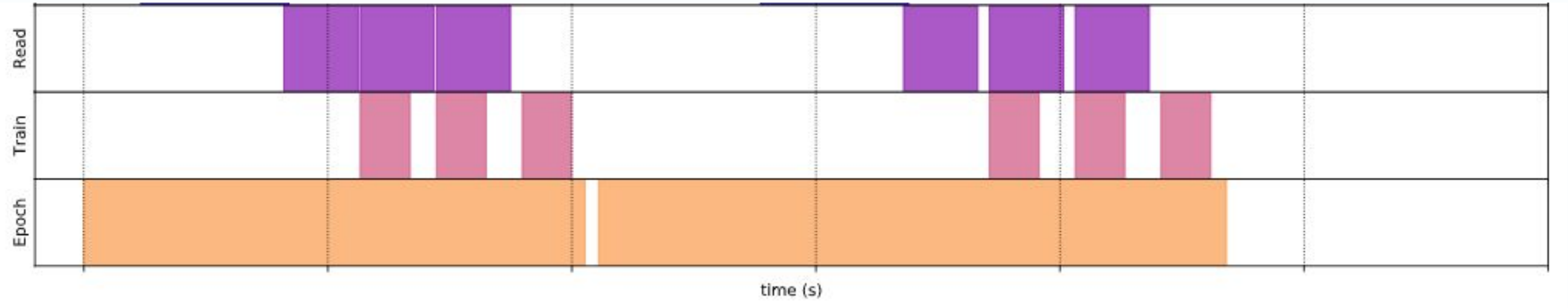
```
train_loader = DataLoader(train_dataset, batch_size=256)


for data in train_loader:
    inputs, masks = data[0].to('cuda'), data[1].to('cuda')


    outputs = model(inputs)
    loss = criterion(outputs, masks)


    loss.backward()
    optimizer.step()
```

Sequential Pipeline



https://www.tensorflow.org/guide/data_performance

Prefetching



https://www.tensorflow.org/guide/data_performance

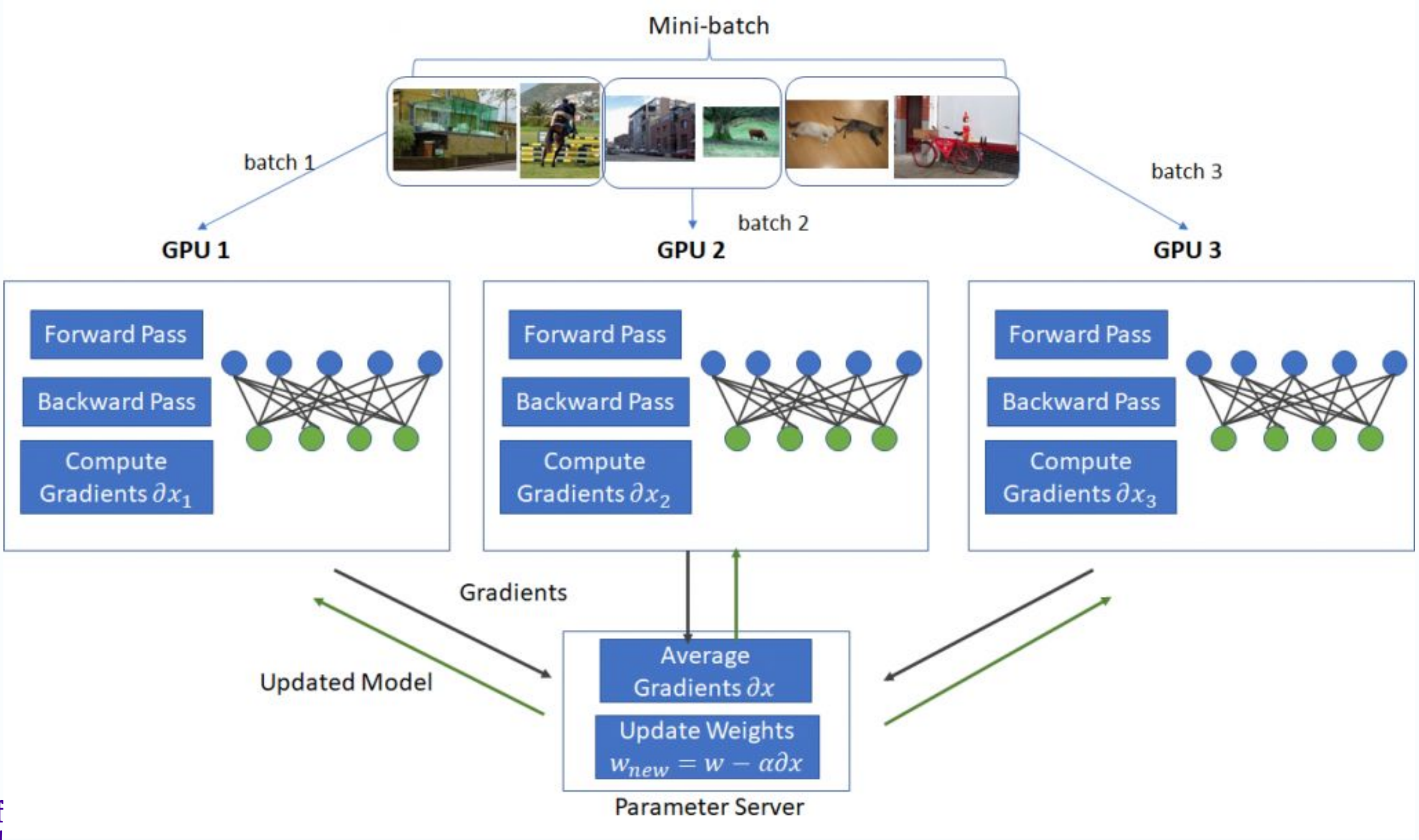
Test 2 - Multiprocessing DataLoader

```
train_loader = DataLoader(train_dataset, batch_size=256,
                      num_workers=4)
for data in train_loader:
    inputs, masks = data[0].to('cuda'), data[1].to('cuda')

    outputs = model(inputs)
    loss = criterion(outputs, masks)

    loss.backward()
    optimizer.step()
```

Test 3 - Automatic Mixed Precision

```
train_loader = DataLoader(train_dataset, batch_size=256,
                      num_workers=4)
for data in train_loader:
    inputs, masks = data[0].to('cuda'), data[1].to('cuda')
    with autocast():
        outputs = model(inputs)
        loss = criterion(outputs, masks)

    loss.backward()
    optimizer.step()
```

Test 4 - Distributed Data-Parallel


Mini-batch
batch 1
batch 2
batch 3
GPU 1
GPU 2
GPU 3
Forward Pass
Backward Pass
Compute Gradients ∂x_1
Forward Pass
Backward Pass
Compute Gradients ∂x_2
Forward Pass
Backward Pass
Compute Gradients ∂x_3
Gradients
Updated Model
Average Gradients ∂x
Update Weights $w_{new} = w - \alpha \partial x$
Parameter Server

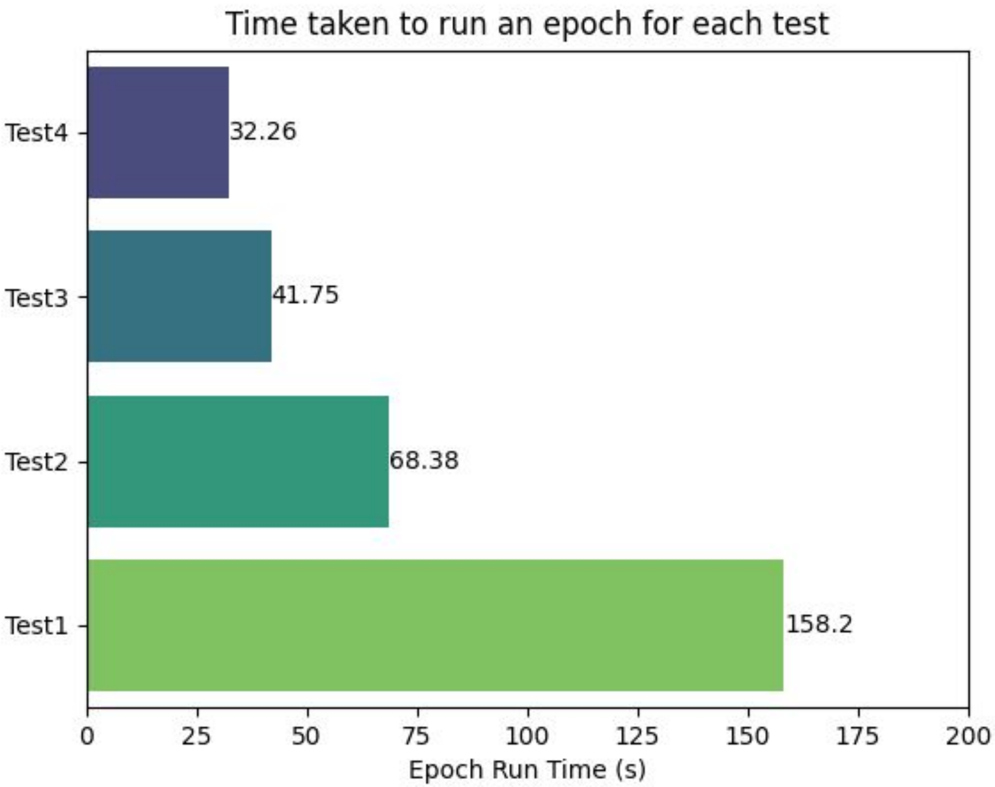
Test Comparison

Multi-Processed DataLoader
+ Mixed Precision + DDP

Multi-Processed DataLoader
+ Mixed Precision

Multi-Processed DataLoader

Default



Other test ideas

- Is there a better way of storing the data on disk other than big lists of PNGs?

- Torch compile

- Passing more of the CPU workload to the GPU?

- Performance Tuning Guide — PyTorch Tutorials 2.3.0+cu121 documentation

Conclusions

Profiling can help find bottlenecks in machine learning workflows

After bottlenecks have been identified, then simple changes can be made to speed up training

Repository here

Contact

Email: w.e.brown@sheffield.ac.uk

Sheffield RSE Team

Alan Turing Institute REG Group