**Per layer latency and bottleneck analysis TensorRT**

Model: Resnet20

Dataset: CIFAR10

Batch size = 32

Time: In Seconds

Set up: 100 training iterations on Jetson Xavier, TensorRT to do conv2d operations and all other operations done on CPU.

**Layer 1**

Input size: torch.Size([32, 3, 224, 224])

Stride, padding= (1, 1) (1, 1)

Weight size= torch.Size([32, 3, 3, 3])

Output size =torch.Size([32, 16, 224, 224])

Total: 45.030

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 3.448  1.370  0.100  0.073  0.809  1.10 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 39.099  9.106 -29.819  8.354  24.311  0.236  3.152  3.027 |

**Layer 2-6**

Input size: torch.Size([32, 16, 32, 32])

Stride, padding= (1, 1) (1, 1)

Weight size= torch.Size([16, 16, 3, 3])

Output size =torch.Size([32, 16, 32, 32])

Total = 5.569

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 0.559  0.185  0.171  0.078  0.041  0.084 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 4.630  0.618 - 3.967  0.371  2.867  0.117  0.524  0.732 |

**Layer 7**

Input size= torch.Size([32, 16, 32, 32])

stride , padding= (2, 2) (1, 1)

Weight size = torch.Size([32, 16, 3, 3])

Output size= torch.Size([32, 32, 16, 16])

Total : 2.880

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 0.480  0.244  0.079  0.041  0.000  0.058 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 2.063  0.369 - 1.684  0.289  0.503  0.135  0.673  0.480 |

**Layer 8-12**

Input size = torch.Size([32, 32, 16, 16])

stride ,, padding =(1, 1) (1, 1)

Weight size = torch.Size([32, 32, 3, 3])

Output size = torch.Size([32, 32, 16, 16])

Total: 4.837

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 0.369  0.097  0.102  0.063  0.040  0.067 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 4.032  0.397 - 3.606  0.200  2.872  0.102  0.404  0.511 |

**Layer 13**

Input size=torch.Size([32, 32, 16, 16])

stride , padding = (2, 2) (1, 1)

torch.Size([64, 32, 3, 3])

torch.Size([32, 64, 8, 8])

Total : 3.650

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 0.404  0.115  0.100  0.090  0.029  0.054 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 2.893  0.394 - 2.489  0.187  1.604  0.101  0.456  0.545 |

**Layer 14-18**

Input size= torch.Size([32, 64, 8, 8])

stride , padding = (1, 1) (1, 1)

Weight size = torch.Size([64, 64, 3, 3])

Output size = torch.Size([32, 64, 8, 8])

Total = 2.367

| Operations | Latency |
| --- | --- |
| 1. Forward    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 0.292  0.049  0.085  0.101  0.039  0.051 |
| 1. Back Prop    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 1.759  0.229 - 1.490  0.103  0.827  0.089  0.217  0.567 |

**Total Compute latency= 3.453**

**Full workflow latency and bottleneck analysis**

Model: Resnet20

Dataset: CIFAR10

Batch size = 4

Precision = Float32

Time: In Seconds

Set up: 100 training iterations on Jetson Xavier, TensorRT to do conv2d operations(both forward and back propagations) and all other operations done on CPU.

Total latency:23.727 sec

Total Forward: 4.779

Total Backward: 18.141

| Operations | Latency |
| --- | --- |
| 1. Forward(Convolutions only)    1. Load input    2. Weight refitting    3. Compute    4. stream synchronize    5. Output reshaping | 2.783  0.241  1.011  0.582  0.178  0.671 |
| 1. Back Prop(Convolution layers only)    1. Input gradient- weight gradient       1. Load input       2. Weight refitting       3. Compute       4. Stream synchronize       5. Pre-processing and output reshaping | 15.334  3.139 - 12.050  0.494  7.840  1.503  2.064  3.433 |

Model: Resnet20

Dataset: CIFAR10

Batch size = 4

Precision = float32

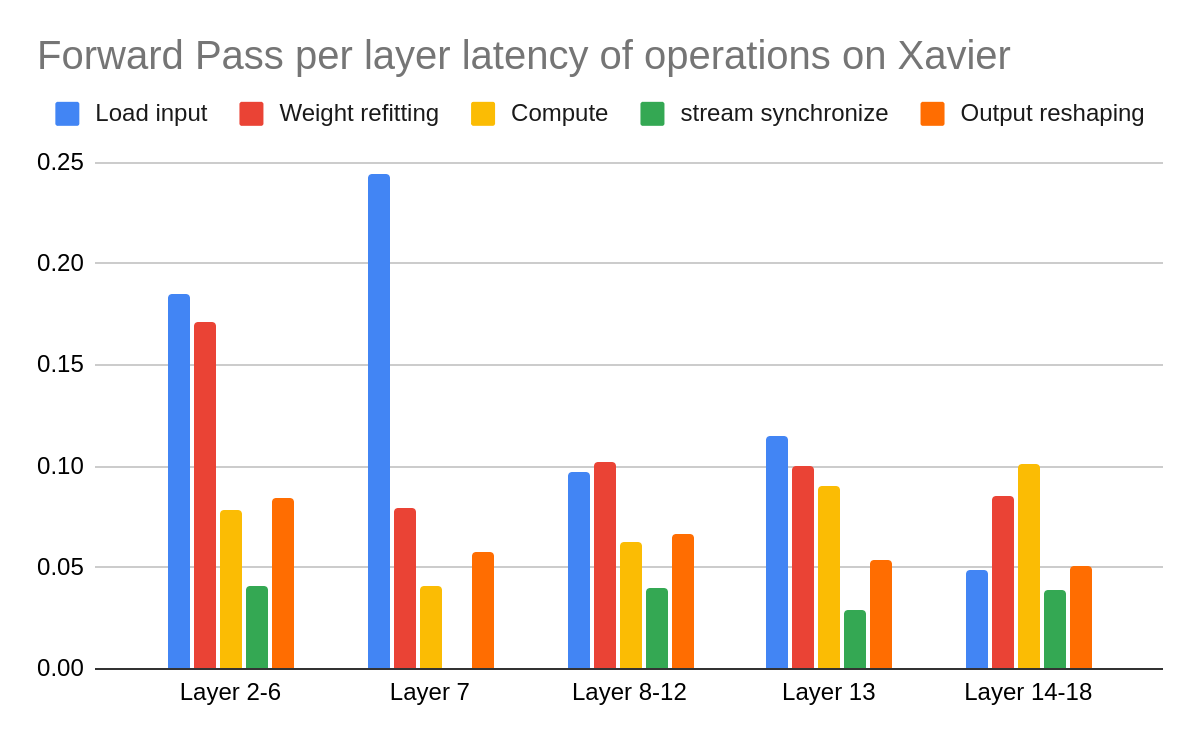
Time: In Seconds

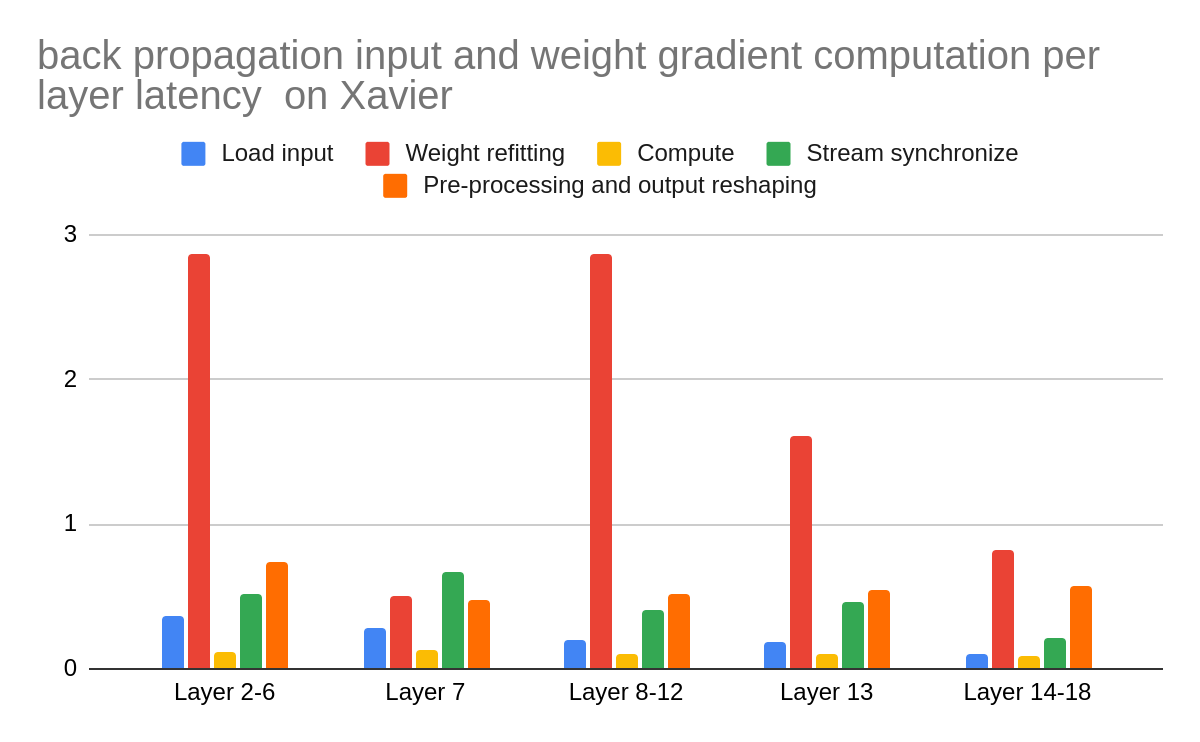
Set up: 100 training iterations on Jetson Xavier, pytorch everything executing on GPU.

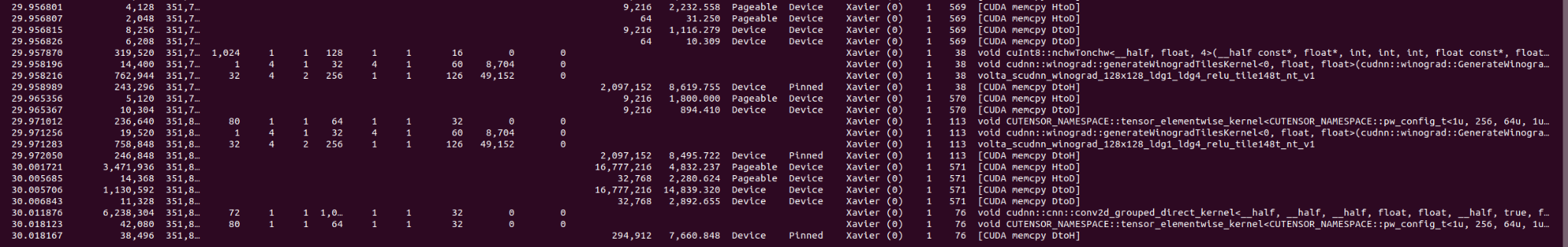
Total latency:3.732 sec

| Operations | Latency |
| --- | --- |
| 1. Forward | 1.425 |
| 1. Back Prop | 1.122 |
| 1. Forward (Conv2d built in function) | 0.431 |

**Vertical axis: time in seconds for 100 runs**

****

****

****