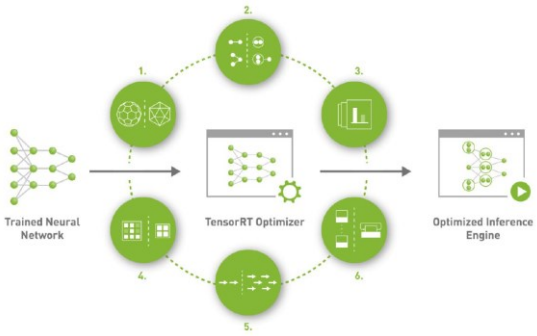
Assignment #6: Quantization, Layer Fusion

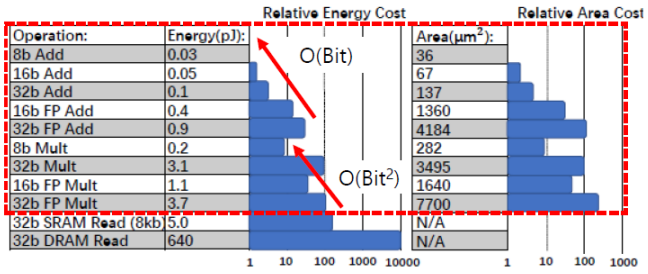
Hanyang University
Sooyoung Kim [redacted] Gaon Choi [redacted]

Introduction

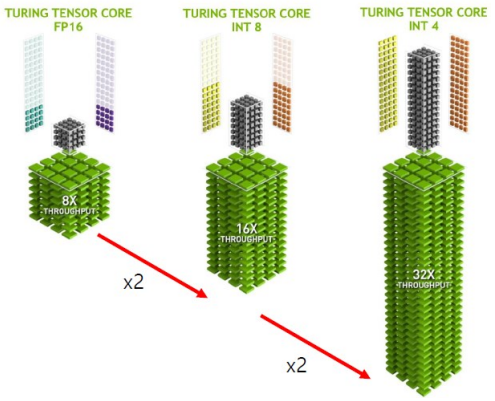
TensorRT is a high-performance neural network inference optimizer and runtime engine. It focuses on running a pre-trained network quickly and efficiently, using a GPU for the purpose of generating a result(i.e., scoring, detecting, regression, or inference).



Addition is a linear operation, requiring O(n) time complexity. However, multiplication needs quadratic time complexity, which is relatively slower and more time-consuming.



Operation:	Energy(pJ):	Area(μm²):
8b Add	0.03	36
16b Add	0.05	67
32b Add	0.1	137
16b FP Add	0.4	1360
32b FP Add	0.9	4184
8b Mult	0.2	282
32b Mult	3.1	3495
16b FP Mult	1.1	1640
32b FP Mult	3.7	7700
32b SRAM Read (8kb)	5.0	N/A
32b DRAM Read	640	N/A



To reduce the elapsed time of inference step, we use a reduced precision operation(DNN Quantization). In this experiment, we will compare the elapsed time between FP16 and FP32. Plus, in the sense of graph-level optimization, TensorRT uses both Layer Fusion and Tensor Fusion methods. Layer fusion is applied with vertical layer fusion, horizontal layer fusion. Tensor fusion is applied to simplify the model graph, greatly reducing the number of layers.

Experiment

Analysis of the TensorRT's optimization Log

1) Engine build Log

```
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_0 with Relu_1
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_3 with Relu_4
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_5 with Relu_6
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_7 with Add_9
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_7 + Add_9 with Relu_10
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_11 with Relu_12
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_13 with Relu_14
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_15 with Add_16
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_15 + Add_16 with Relu_17
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_18 with Relu_19
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_20 with Relu_21
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_22 with Add_23
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_22 + Add_23 with Relu_24
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_25 with Relu_26
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_27 with Relu_28
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_29 with Add_31
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_29 + Add_31 with Relu_32
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_33 with Relu_34
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_35 with Relu_36
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_37 with Add_38
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_37 + Add_38 with Relu_39
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_40 with Relu_41
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_42 with Relu_43
[TensorRT] VERBOSE: ConvEltwiseSumFusion: Fusing Conv_44 with Add_45
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_44 + Add_45 with Relu_46
[TensorRT] VERBOSE: ConvReluFusion: Fusing Conv_47 with Relu_48
```

2) Comparison of execution time

- Inference Time

Precision	Inference Time
FP32	0.23955941200256348 second
FP16	0.09717559814453125 second

When the experiment was conducted based on ResNet50,
FP16 showed a performance improvement of about 2.5 times in terms of inference time compared to FP32.

- Accuracy

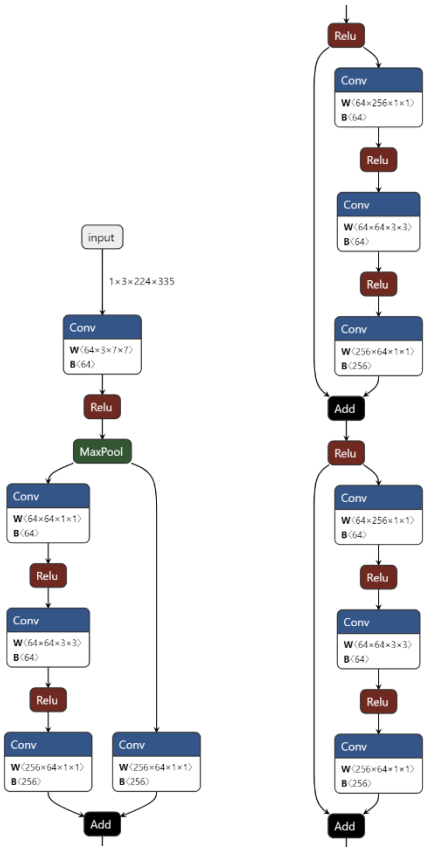
Precision	Accuracy
FP32	class: doormat, welcome mat confidence: 8.759943962097168 % index: 539
FP16	class: doormat, welcome mat confidence: 8.764363288879395 % index: 539

3) Analysis of the ResNet50 ONNX model by Netron

3-1. Fastest Tactic

```
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x32_relu_medium_nn_v1 Tactic: 1062367460111450758
[TensorRT] VERBOSE: Tactic: 1062367460111450758 Time: 2.83021
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x32_relu_large_nn_v0 Tactic: 1754984623894446479
[TensorRT] VERBOSE: Tactic: 1754984623894446479 Time: 2.88297
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x128_relu_large_nn_v0 Tactic: 3611739942397549984
[TensorRT] VERBOSE: Tactic: 3611739942397549984 Time: 4.5437
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x64_relu_large_nn_v1 Tactic: 4337000649858996379
[TensorRT] VERBOSE: Tactic: 4337000649858996379 Time: 2.2262
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x128_relu_medium_nn_v1 Tactic: 4501471010995462441
[TensorRT] VERBOSE: Tactic: 4501471010995462441 Time: 4.38521
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x64_relu_medium_nn_v1 Tactic: 6645123197870846056
[TensorRT] VERBOSE: Tactic: 6645123197870846056 Time: 2.20656
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x128_relu_large_nn_v1 Tactic: -9137461792520977713
[TensorRT] VERBOSE: Tactic: -9137461792520977713 Time: 4.46833
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x128_relu_medium_nn_v0 Tactic: -8262349710178828730
[TensorRT] VERBOSE: Tactic: -8262349710178828730 Time: 4.4788
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x64_relu_large_nn_v0 Tactic: -8133971918129952780
[TensorRT] VERBOSE: Tactic: -8133971918129952780 Time: 2.38682
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x32_relu_large_nn_v1 Tactic: -6092040395344634144
[TensorRT] VERBOSE: Tactic: -6092040395344634144 Time: 2.8824
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x32_relu_medium_nn_v0 Tactic: -4787320710726427159
[TensorRT] VERBOSE: Tactic: -4787320710726427159 Time: 2.8601
[TensorRT] VERBOSE: Conv_0 + Relu_1 Set Tactic Name: maxwell_scudnn_128x64_relu_medium_nn_v0 Tactic: -1218658103698133241
[TensorRT] VERBOSE: Tactic: -1218658103698133241 Time: 2.30609
[TensorRT] VERBOSE: Fastest Tactic: 6645123197870846056 Time: 2.20656
```

3-2. Model visualization



* Printed only the core part, removing the repeating part.

Conclusion

We learned some basic concepts of number representation: INT8, FP16, FP32, specially based on IEEE standard. FP16 yielded some relative improvement in inference time compared to FP32. Also visualizing the model with Netron helped us to understand the structure of ResNet50.