Towards High-Performance AI4NP Applications on Modern GPU Platforms

Xinxin Mei [1] Nathan Brei [1] Thomas Britton [1] David Lawrence [1]

[1]Thomas Jefferson National Accelerator Facility, Newport News, VA, USA

Introduction

We study the performance of AI4NP applications on 2 NVIDIA GPU platforms: Tesla T4 and A100 80GB PCIe, with their specifications listed in Table 1, where **Tensor Core** (TC) is low-precision matrix-multiplication hardware independent of common floating-point processing units.

Table 1: Specifications of NVIDIA T4 and A100 80GB PCIe GPUs

GPU	#SM	Mem	Peak BW	FP32	FP16 Tensor Core
T4	40	15110 MB	245.2 GB/s	8.1 TFLOPS	65 TFLOPS
A100 PCIe	108	81251 MB	1607.3 GB/s	19.4 TFLOPS	312 TFLOPS
Speedup	2.5x	5.5x	6.5x	2.5x	5x

Roofline Performance Model

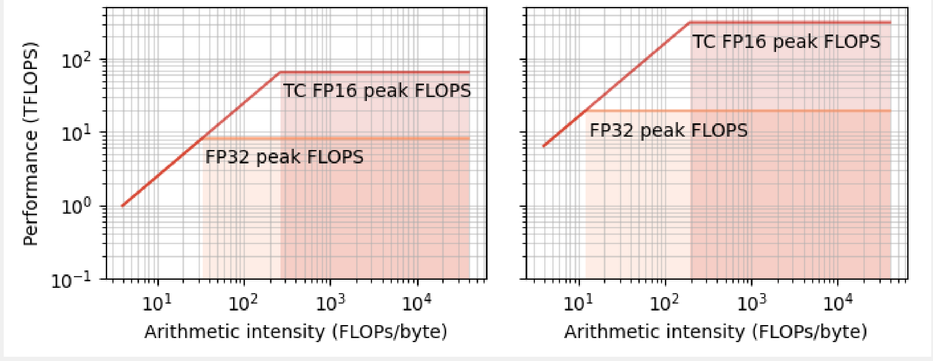


Figure 1: Roofline model for NVIDIA T4 and A100 PCIe GPUs. Both x-axis and y-axis are log-scale.

- X-axis: **arithmetic intensity**, defined as #operations/BW. FLOPs: floating-point operations.
- Y-axis: achieved throughput in TFLOPS. FLOPS: floating-point operations per second.

The maximum achievable performance shapes like a 2D roofline, where the diagonal roof denotes the processor's peak bandwidth (BW, in GB/s) and the horizontal roof denotes the peak floating-point performance (in FLOPS) [4]. We define the **ridge point** as where the diagonal and horizontal roofs meet. T4 has FP32&FP16 ridge points of larger arithmetic intensity, which means that it is more likely to be bounded by bandwidth.

Hyper-Parameterized Fully-Connected Neural Network (FCNN)

Model Definition

Table 2: Hyper-Parameters of FCNN Models

Variable	Layers (l)	Nodes (D_H)	Input (D_I)	Output (D_O)	Batch size (bs)
Min	4	32	2000	200	64
Max	128	8192	8000	1000	16384
Inc	×2	×2	+2000	+200	×2
#Samples	6	9	4	5	9

Hyper-parameterized FCNN models with homogeneous hidden layers [3]:

- Dimension of the hidden layer: D_H. Number of hidden layers: l.
- **Number of model parameters**: $\Phi = {D_H}^2 \times (l-1) + D_H \times (D_I + D_O)$.
- **Operations per step**: $O = 6 \times \Phi \times bs$. bs is the batch size.
  Only the forward and backward propagation operations are counted.
- **Estimated performance in FLOPS**: $O\times$ training steps / training time.
  Estimated FLOPS is smaller than the actual value.
- Environment: gcc 9.1.0, Python 3.10, PyTorch 1.13.0, CUDA 11.6, cuDNN 8.3

Achieved Peak Performance

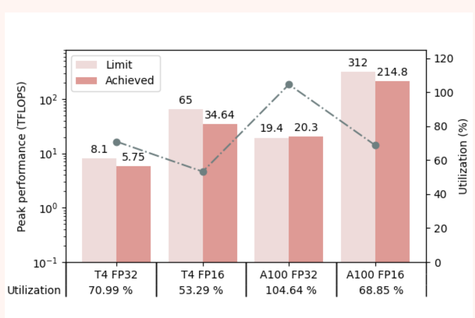


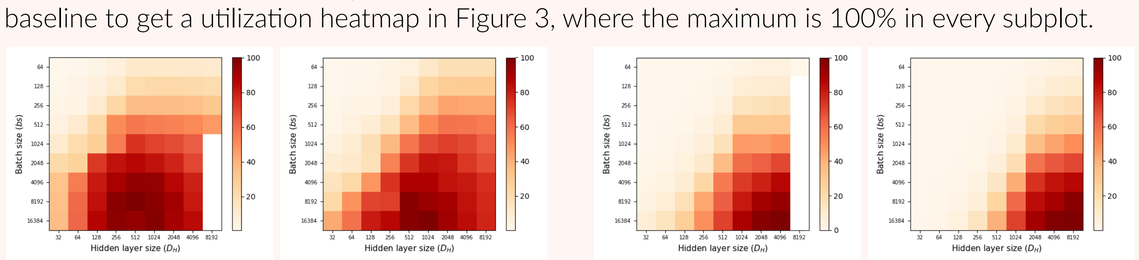
Figure 2: Achieved peak FCNN performance and the utilization. Left y-axis is log-scale.

A100 has higher peak utilization than T4.

- Utilization is compared to the theoretical FP32/FP16 performance limit. Compared to T4, peak A100 FP32 FLOPS shows $3.5\times$ speedup while FP16 shows $6.2\times$, both higher than listed in Table 1.
- Peak A100 FP32 utilization is higher than 100%, indicating FP16 Tensor Cores are automatically activated, which is not the case for T4.
- Peak utilization of T4 & A100 $> 50\%$, better than NVIDIA V100 (data given in [3]).

FLOPS Utilization

For each of the 4 experiment groups, we set the peak FLOPS as baseline, and divide real FLOPS to baseline to get a utilization heatmap in Figure 3, where the maximum is 100% in every subplot.



(a) T4 FP32, l=4, D_I=8000, D_O=200. Peak is at D_H=512, bs=8192.
(b) A100 FP32, l=4, D_I=8000, D_O=200. Peak is at D_H=1024, bs=16384.
(c) T4 FP16, l=32, D_I=8000, D_O=1000. Peak is at D_H=4096, bs=16384.
(d) A100 FP16,l=64,D_I=8000,D_O=600. Peak is at D_H=8192, bs=16384.

Figure 3: FCNN FLOPS relative utilization. Blank cells indicate out-of-memory.

- Both T4 and A100 **prefer large model parameters** to be saturated. More so for A100 or FP16.
- $O \propto {D_H}^2 \Longrightarrow D_H$ **has optimum**. When D_H is larger than optimum, performance slightly decreases. This is for both FP16 and FP32 but more obvious for FP32.

FCNN Roofline Performance Analysis

Performance v.s. Batch Size

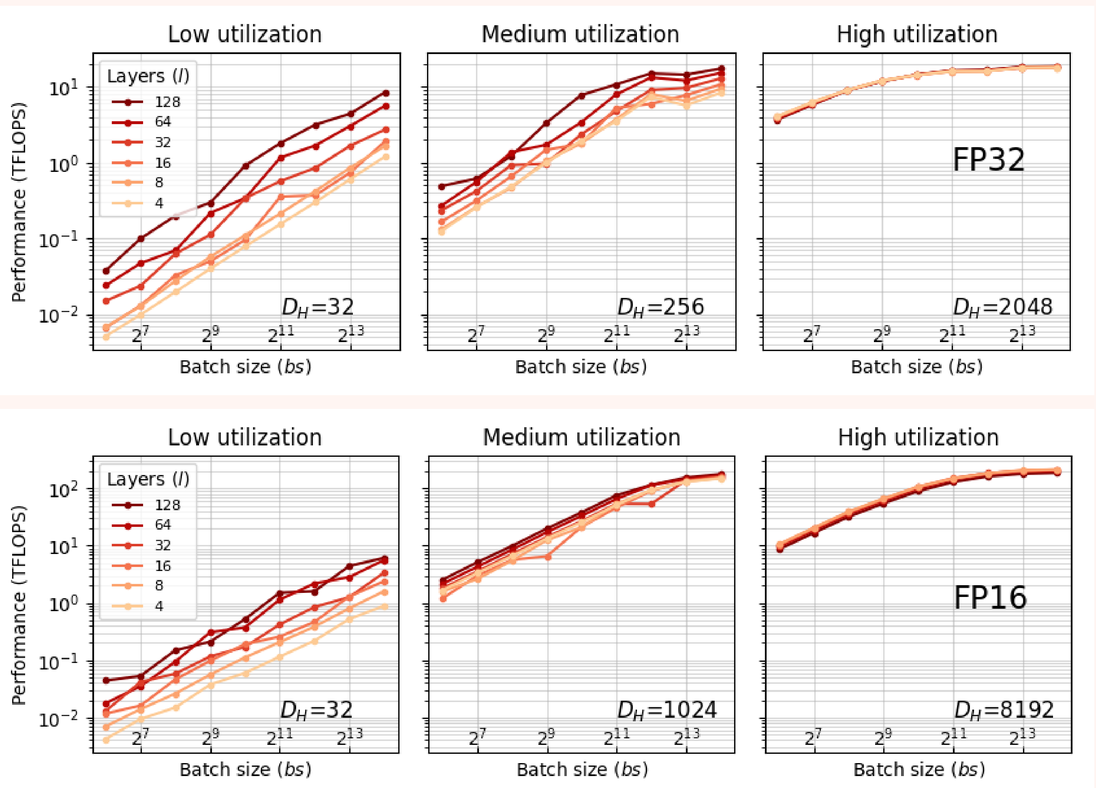


Figure 4: Achieved FLOPS with D_H of low, medium and high utilization levels on A100.

We choose D_H of 3 utilization levels, and plot the achieved FLOPS to batch size with both axes as log-scale in Figure 4. Though $O \propto bs$, **performance-batch size relationship shapes like a roofline.**

- **Low utilization**: diagonal roof only. Performance is linear to l and batch size.
- **Medium utilization**: typical roofline. Performance with larger l reaches horizontal roof faster.
- **High utilization**: flatter roofline. Performance is insensitive to l and achieves peak fast.

Summary

- Tensor Cores are automatically activated on A100, which leads to higher FLOPS utilization.
- GPUs prefer large models, but there are optimal hidden layer dimensions. With high utilization, performance is insensitive to number of layers.
- Performance is most sensitive to batch size. The performance-batch size relationship is in a roofline manner. FLOPS increases drastically to batch size with low utilization.

Convolution Neural Network (CNN) Application: Hydra

Hydra [1] is a successful AI-driven monitoring system in production at GlueX. It is based on Google's **Inception v3** image classification model [2]. We train the model with real GlueX data, where the training set is of 28.9k samples and the validation set is of 1.5K samples, and each sample is an 800x600x3 RGB image. The model fits perfectly that validation error is within 1.5%.

Environment: Python 3.8.6, Tensorflow 2.11, Keras 2.11.0, cuDNN 8.2. Multiple GPUs are connected to a single CPU to minimize the inter-GPU memory traffic.

Mixed Precision

Mixed precision (MP) is the use of both FP16 (run on Tensor Cores) and FP32 (run on floating-point units) in a model during training. It can improve the compute throughput while at the same time reduce the memory utilization. We trigger MP manually by adding some lines into the source code.

Table 3: Performance of Hydra Training

Configuration	2 T4s	2 T4s MP	2 A100s	2 A100s MP
Batch size (samples)	24	24	192	192
Speed (samples/sec)	22.08	45.89	83.53	84.27
Speedup	1	2.1×	3.8×	3.8×
Validation accuracy (%)	99.23	99.47	98.95	99.41

We set the training speed of 2 T4 GPUs as baseline. As listed in Table 3, MP significantly benefits T4 that it receives 2.1x speedup. Though MP almost has no influence on A100, it still shows 3.8x speedup. We infer that MP is automatically activated, similar to the FP32 FCNN model. For all experiments, MP does not affect the final validation accuracy.

Conclusions

- Performance-batch size has a roofline relationship. Use large batch size to achieve high utilization.
- Leverage mixed precision to speed up training, especially on GPUs of compute capacity 7.x (i.e. T4).

References

[1] Thomas Britton, David Lawrence, and Kishansingh Rajput.
AI enabled data quality monitoring with hydra.
In *EPJ Web of Conferences*, volume 251. EDP Sciences, 2021.

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna.
Rethinking the inception architecture for computer vision.
In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[3] Yu Emma Wang, Gu-Yeon Wei, and David Brooks.
A Systematic Methodology for Analysis of Deep Learning Hardware and Software Platforms.
In *The 3rd Conference on Machine Learning and Systems (MLSys)*, 2020.

[4] Samuel Williams, Andrew Waterman, and David Patterson.
Roofline: an insightful visual performance model for multicore architectures.
Communications of the ACM, 52(4):65–76, 2009.

Presenter: **Xinxin (Cissie) Mei**
Email: **xmei@jlab.org**

EPSCI

Jefferson Lab
Thomas Jefferson National Accelerator Facility

U.S. DEPARTMENT OF ENERGY