EECE-5640 **High Performance Computing** **2022-FALL**

Benchmark Deep Neural Network Models

Juncen Li, MSECE (2021)
December 4, 2022

Department of Electrical and Computer Engineering
Northeastern University

ABSTRACT

In-depth analyses of the majority of deep neural networks (DNNs) used in state-of-the-art image recognition systems are presented in this research. Numerous performance metrics, such as recognition accuracy and inference time, are seen for each DNN. Such performance metrics' behavior and their combinations are examined and debated. We test using DNNs on two distinct computer architectures—a workstation with an NVIDIA Tesla P100 with 12 nodes with 4 GPUs each and an embedded system built on an NVIDIA Tesla K80 with eight nodes with 8 GPUs each—in order to assess the index [1]. They both, however, use the same CPU, the E5-2680v4@2.40GHz. Through this investigation, DNNs operate on devices.

CONTENTS

I README

The details of how to execute the codes on different nodes with different models and inputs are in the readme.md file. Please read it carefully.

- Using P100

srun --partition=gpu --nodes=1 --pty --gres=gpu:p100:1 --ntasks=1 --mem=4GB --time=04:00:00 /bin/bash

- Using K80

srun --partition=gpu --nodes=1 --pty --gres=gpu:k80:1 --ntasks=1 --mem=4GB --time=04:00:00 /bin/bash

- Load Modules

module load anaconda3/2022.01

module load cuda/11.1

- Activate environment

source activate pytorch_env_training

- Check Pytorch

python -c'import torch; print(torch.cuda.is_available())'

The reason for torch.cuda.is_available() resulting False is the incompatibility between the versions of pytorch and cudatoolkit.

- Training GoogleNet using cifar10

python cifar10-googlenet.py

- Training GoogleNet using MNIST

python MNIST-googlenet.py

- Training GoogleNet using KMNIST

python KMNIST-googlenet.py

- Training RestNet18 using cifar10

python cifar10-resnet18.py

- Training RestNet18 using MNIST

python MNIST-resnet18.py

- Training RestNet18 using KMNIST

python KMNIST-resnet18.py

II INTRODUCTION

2.1 PROJECT PURPOSE

Deep neural networks (DNNs) have achieved remarkable results in many computer vision tasks [2]. AlexNet [3], which is the first DNN presented in the literature in 2012, drastically increased the recognition accuracy (about 10% higher) concerning traditional methods in the 1000-class ImageNet Large-Scale Visual Recognition Competition (ImageNet-1k) [4]. Since then, literature has worked both in designing more accurate networks and more efficient networks from a computational-cost point of view.

In this project, I will evaluate the performance of engineering workloads on the Discovery platform. I propose to use PyTorch, which needs GPU acceleration to evaluate executing speedup on different Discovery nodes and hardware. Three datasets will train two models on PyTorch on two discovery nodes. For each model, we will do six experiments in total.

2.2 PREREQUISITE

Two Deep Neural Network Models will be evaluated. The first one is GoogleNet, and it is a new convolutional neural network it has 22 layers. Unlike other convolutional networks, GoogleNet achieves good classification performance while controlling the amount of computation and parameters. It removes the last fully connected layer, uses a global average pooling layer, and replaces large convolutions with multiple small convolution kernels. Another one is ResNet-18. It is 18 layers deep that can learn rich feature representations for a wide range of images.

Moreover, three datasets will be used. MNIST, CIFAT-10, and KMNIST are three popular deep-learning datasets. CIFAR-10 is a 3-channel color RGB image dataset comprising 60000 32x32 color images in 10 classes, with 6000 images per class. Fifty thousand training images and 10000 test images in total. MNIST is an extensive database of handwritten digits in grayscale images commonly used for training various image processing models. The dataset contains 60,000 examples for training and 10,000 examples for testing. The numbers are size normalized and centered in the image, which is a 28x28 fixed size.

Moreover, KMNIST is a dataset adapted from Kuzushiji Dataset as a drop-in replacement for the MNIST dataset, which is the most famous dataset in the machine learning community. Just change the setting of your software from MNIST to KMNIST. We provide three types of datasets, namely Kuzushiji-MNIST, Kuzushiji-49, and Kuzushiji-Kanji, for different purposes.

III BENCHMARK

4.1 OVERVIEW

The methodologies used in each step of this project were primarily addressed in this chapter. I identified and created six deep-learning experiments that can be run on the Nvidia Tesla P100 and Nvidia Tesla K80 [5].

The experiments include ResNet-18 with MNIST, ResNet-18 with Cifar10, and ResNet-18 with KMNIST in two computer architectures. They also include GoogleNet with MNIST, Cifar10, and KMNIST.

4.2 GPU

The Nvidia Tesla P100 and Tesla K80 in the Northeastern Discovery System share the E5-2680v4@2.40GHz CPU. The experimental performance of P100 is rough twice as good as K80's, which makes it simple to assess the outcomes. Below is a list of the exact performance specifications for P100 and K80 [6].

Nvidia ® Tesla® P100

Cores	56(SM)
CUDA Cores	3584
Frequency	1126MHz
GFLOPs(double)	4670
Memory	16GB
Memory B/M	720GB/s

Nvidia® Tesla® K80

Cores	2*13(SMX)
CUDA Cores	2 * 2496
Frequency	562MHz

GFLOPs(double)	2 * 1445
Memory	24 GB
Memory B/M	480 GB/s

4.3 PROCESS

Many various sources of source code were available to me. Each experiment's epoch and batch parameters are both set to 10. All photos from MNIST, KMNIST, and Cifar10 should be converted into 224x224 images with RGB channel three since these are the only input images that GoogleNet and ResNet-18 can take. All python files should be changed to ensure that the network architectures of several experiments using the same neural network are consistent. PyTorch is then installed by adhering to the Discovery System's instructions.

Processing experiments using P100 and K80 nodes next.

IV EXPERIMENT RESULT

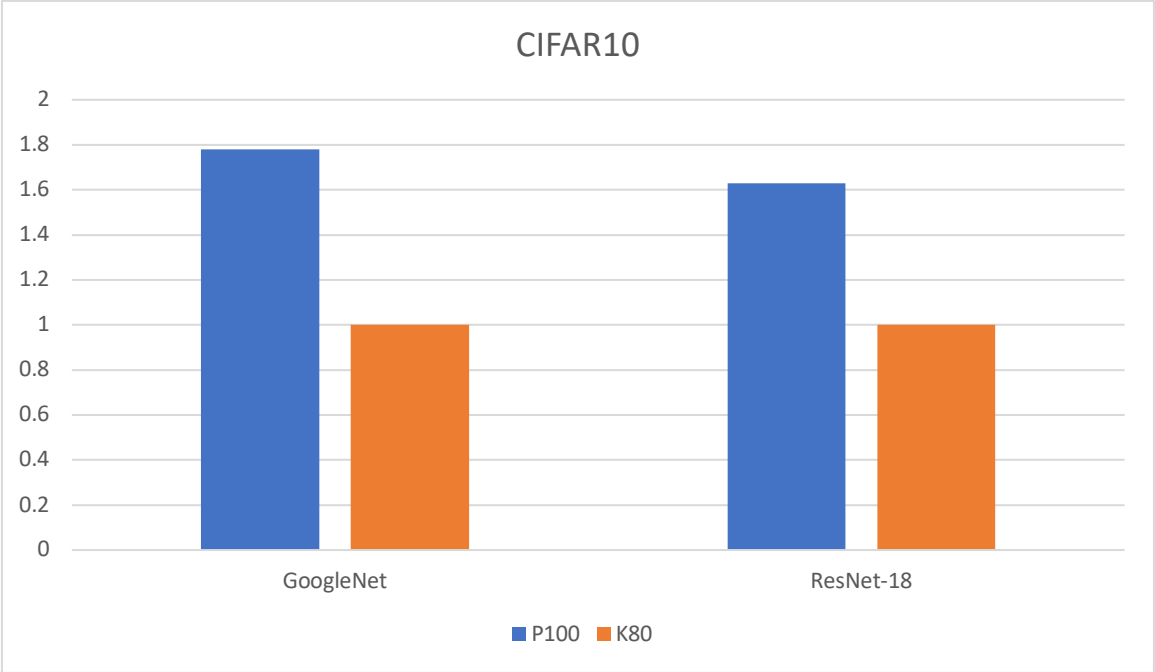
Running Time Comparison Table

		P100			K80		
		CIFAR10	MNIST	KMNIST	CIFAR10	MNIST	KMNIST
Time(s)	GoogleNet	1019.2370	914.2736	914.4191	2834.6030	2587.4574	2582.8948
	ResNet-18	693.6736	518.9218	505.2194	1823.6968	1462.0754	1449.1527

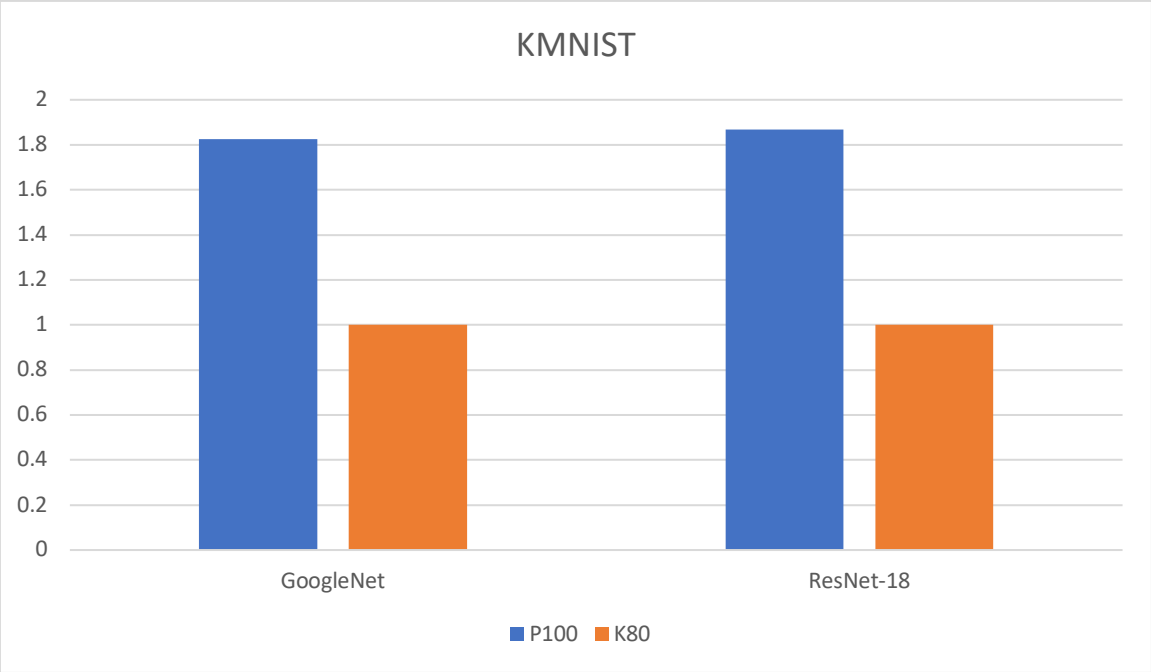
Accuracy Comparison Table

		P100			K80		
		CIFAR10	MNIST	KMNIST	CIFAR10	MNIST	KMNIST
Acc	GoogleNet	87.0600%	99.3000%	97.6100%	85.9100%	99.3200%	97.3000%
	ResNet-18	83.2600%	99.1900%	96.8000%	83.6200%	99.2800%	95.7000%

The comparison of speedup on different nodes with different datasets.


CIFAR10
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
GoogleNet
ResNet-18
P100
K80


MNIST
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
GoogleNet
ResNet-18
P100
K80


KMNIST
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
GoogleNet
ResNet-18
P100
K80

V CONCLUSION

Based on the results from chapter IV, it is straightforward to deduce that P100 outperforms K80 with two models and three datasets. I did 12 experiments, with the GoogLeNet having the highest accuracy between two different models but also requiring the most run time.

Regarding speedup, the Nvidia Tesla P100 performs far better than the Nvidia Tesla K80 regarding accelerating GoogleNet. Compared to K80, P100 can accelerate calculation by around 1.8 times regardless of the database.

P100 features more SMs, a faster Base Clock, and quicker NVLink Generation 1 than K80 despite having fewer CUDA cores and smaller memory than K80. Thanks to these characteristics and technology, it can complete deep learning workloads substantially quicker than K80. Compared to K80, it has twice as many SMs and a quicker base clock. More SMs and a faster base clock may increase calculation speed more than anticipated for workloads like GoogLeNet that demand more computation. The fastest website is GoogleNet.

It is not sure why tasks that need lower calculations will be accelerated less. Perhaps this phenomenon happens because the strong scaling of this system is not good and both P100 and K80 require some procedures that are consuming time and unavoidable. These procedures do not directly calculate any task. As a result, the task needs lower calculation and has a lower speedup. To verify this guess, further work needs to be done.

Fortunately, we still obtained intriguing findings that taught us how to implement HPC ideals in deep learning. We discovered that equipment significantly impacts performance by comparing the speedup of various nodes, and we also discovered that performance improvement is dependent on the superb coordination of hardware and software.

REFERENCE

[1] https://www.xcelerit.com/computing-benchmarks/insights/nvidia-p100-vs-k80-gpu/

[2] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning", Nature, vol. 521, pp. 436, 2015.

[3] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", Proc. Adv. Neural Inf. Process. Syst. (NIPS), pp. 1097-1105, 2012.

[4] O. Russakovsky et al., "ImageNet large scale visual recognition challenge", Int. J. Comput. Vis., vol. 115, no. 3, pp. 211-252, Dec. 2015.

[5] https://www.hpcwire.com/2017/04/20/nvidia-p100-shows-1-3-2-3x-speedup-k80-gpu-financial-apps/

[6] https://insidehpc.com/2016/11/nvidia-tesla-p100-gpu-review/