Deep Learning in GPUs using Limited Precision

PREPARED BY:

CHITRARTH SINGH(A20387080)

PRADYOT MAYANK(A20405826)

NAVNEET GOEL(A20405197)

Contents

Introduction

- Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi supervised or unsupervised.

- Deep learning architectures such as deep neural networks, deep belief networks and recurrent neural networks have been applied to fields including computer vision, speech recognition, NLP, audio recognition, social network filtering, where they have produced results comparable to and in some cases superior to human experts

Why GPU!!

- Numerical operations on CPUs are challenging in terms of accuracy, precision, and performance making it essential to choose the best representation of data.

- A simple way to understand the difference between a GPU and a CPU is to compare how they process tasks.

- A CPU consists of a few cores optimized for sequential serial processing while a GPU has a massively parallel architecture consisting of thousands of smaller, more efficient cores designed for handling multiple tasks simultaneously.

Literature Review

- High Precision Data formats consume relatively higher amount of memory and yield mediocre performance as opposed to limited precision data format.

- The half precision (FP16) Format is not new to GPUs. In fact, FP16 has been supported as a storage format for many years on NVIDIA GPUs, mostly used for reduced precision floating point texture storage and filtering and other special-purpose operations.

- The Pascal GPU architecture implements general-purpose, IEEE 754 FP16 arithmetic. High performance FP16 is supported at full speed on TeslaP100 (GP100), and at lower throughput (similar to double precision) on other Pascal GPUs (GP102,GP104,andGP106).

Problem Statement

- Applications today being data-intensive, generating huge datasets which requires faster computing and greater accuracy. CPUs and GPUs are exhausting in their potential to be at par in processing this kind of data.

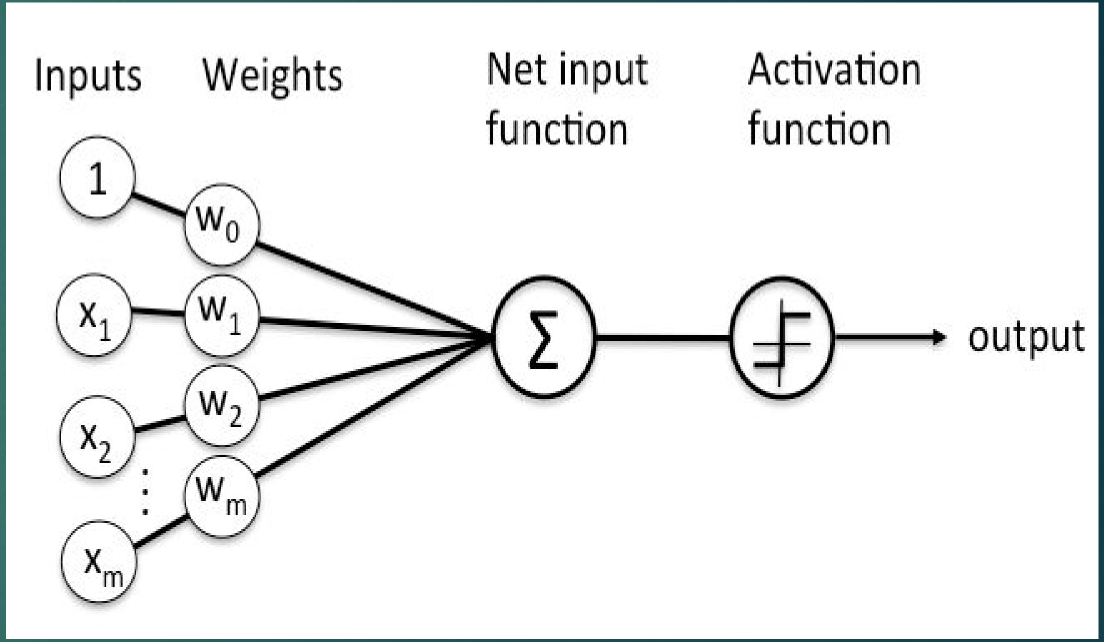
- High Precision Data Formats, as we already know, consumes significantly huge amount of memory space to achieve higher accuracy with great precision which reduces the performance of GPUs and system throughput overall .e.g. Neural network with supervised learning are rather meant to be more efficient than power consuming.

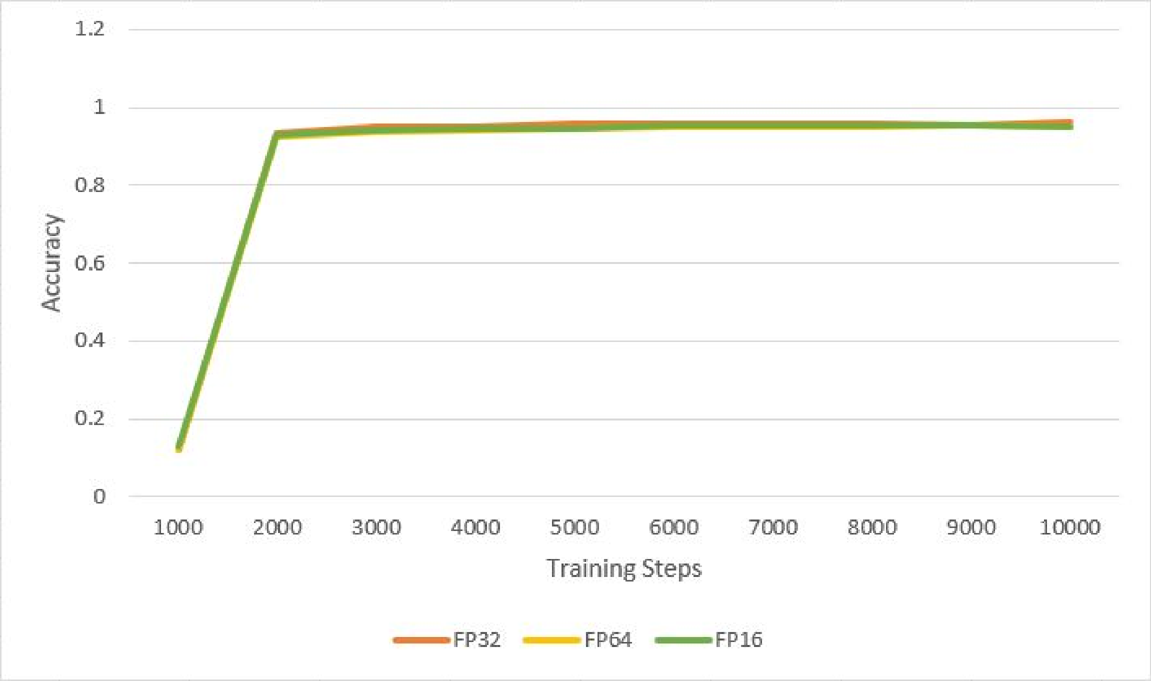
Proposed Solution

- We have implemented a deep neural network model using TensorFlow libraries on GPUs and cuDNN GPU-accelerated library of primitives for deep neural networks
- GPUs through their multi-core architecture provides enhanced throughput to train models/neural nets in deep neural network when compared to CPUs.
- TensorFlow libraries automatically discovers and uses GPUs and multiple cores. Also,it supports seamless quantization of limited precision datatypes.
- We have used dataset for our model: MNIST
- MNIST database of handwritten digits of 28x28 pixels which has a training set of 60,000 examples, and a test set of 10,000 examples.

Model Functionality

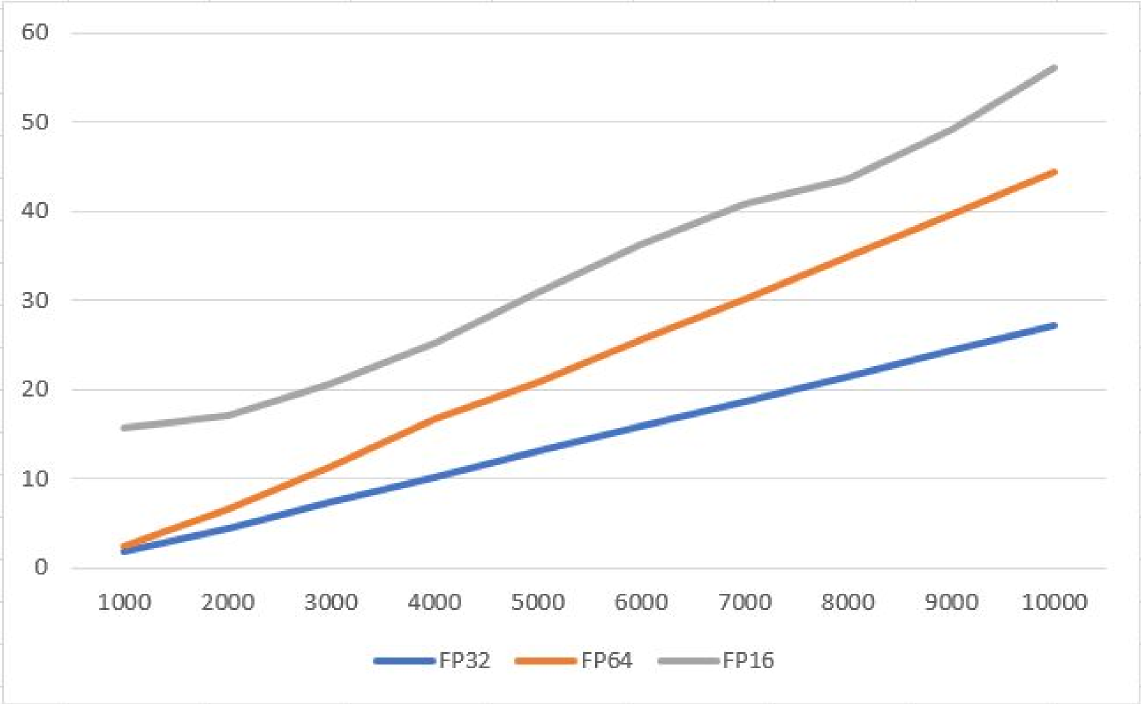
- Activation Function:
  - Sigmoid
  - Softmax
-
- Three hidden layers
- Adam Optimizer for optimization of variables





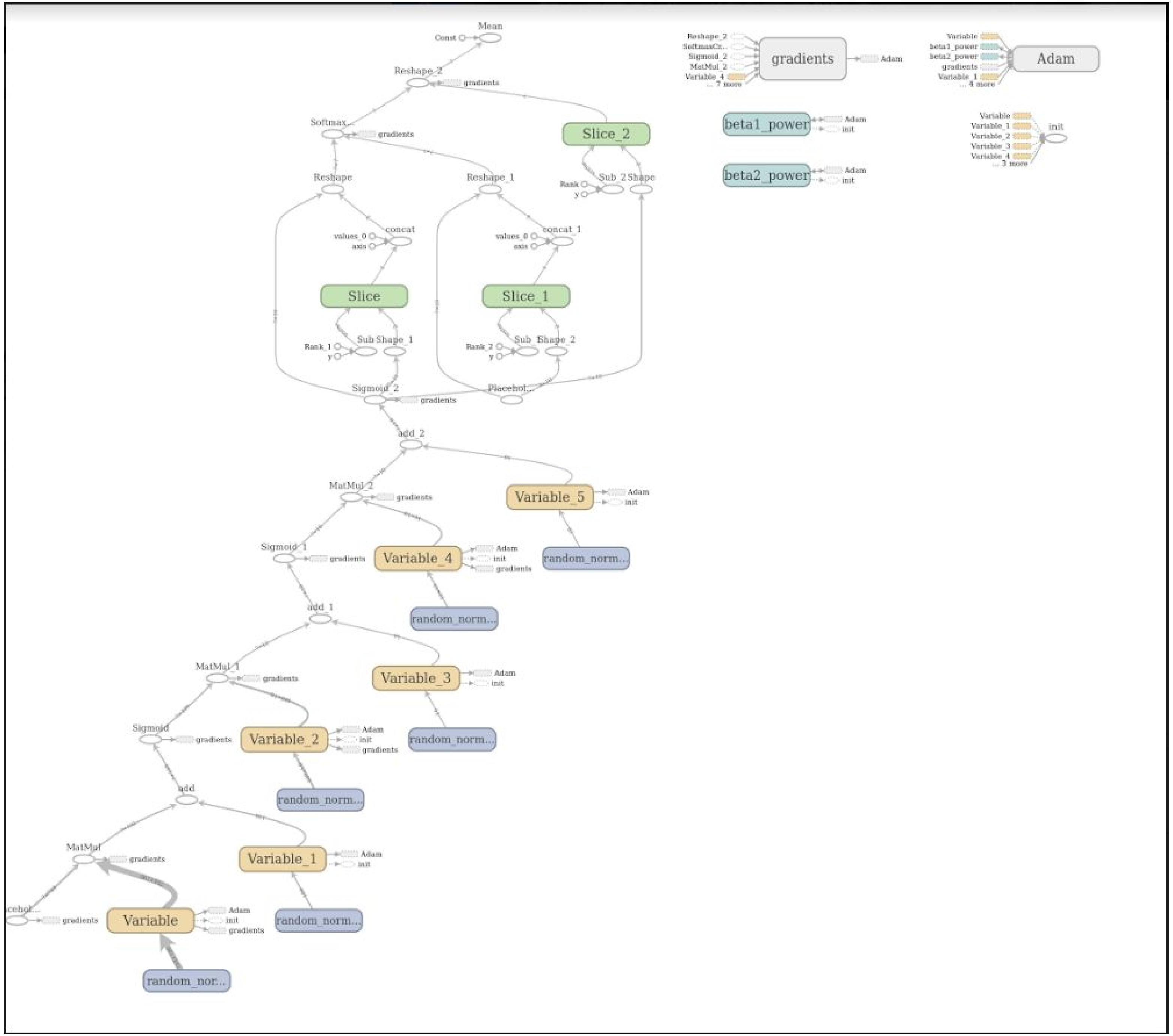
	Accuracy		
Training Steps	**FP32**	**FP64**	**FP16**
1000	0.1226	0.1206	0.1306
2000	0.9328	0.9239	0.9273
3000	0.9504	0.9363	0.9422
4000	0.9522	0.9432	0.9463
5000	0.958	0.945	0.9447
6000	0.9566	0.9511	0.9538
7000	0.9601	0.9492	0.953
8000	0.9605	0.9513	0.9528
9000	0.9558	0.9529	0.9527
10000	0.961	0.9515	0.9497

Training steps vs. Accuracy



	Time elapsed		
Training Steps	FP32	FP64	FP16
1000	1.776305	2.39399	15.67
2000	4.475359	6.62688	16.98
3000	7.361821	11.34327	20.58
4000	10.22937	16.66568	25.18
5000	13.10573	20.81766	30.95
6000	15.96127	25.58936	36.2
7000	18.64432	30.1295	40.81
8000	21.51693	34.84176	43.66
9000	24.38518	39.56943	49.26
10000	27.24746	44.45809	56.2

Training steps vs Time Elapsed



Graph generated on Tensor board

Conclusion/What We Learnt

- Learnt how to implement deep neural networks efficiently with low precision.
- F16 is sufficient for training deep neural networks to classify data with accuracy comparable to F32.
- Using a higher precision for the parameters during the implementation helps for understanding of the model.
- Achieve faster computationon GPUs using these limited precision datatypes.
- Our goal is to train MNIST using INT8 and compute the accuracy of the model. We will also try to build a different model for a different dataset.

Related Work

- Research paper by Yoshua et al.(2015) talk about training deep neural networks using low precision multiplication and achieving optimized memory usage on general purpose hardware.

- Jordan et al.adapted limited precision calculations to train deep neural network for back propagation algorithms.

- Guptaet al used 16- bit fixed point format to train neural networks on FPGA gaining significant amount of efficiency and computation throughput.

References

- Presley, R.K. and Haggard, R.L. (1994). A fixed point implementation of the backpropagation learning algorithm. In Southeastcon’94. Creative Technology Transfer-A Global Affair.,ProceedingsIEEE, pages136–138.
- IEEE.Holt,J.L. andBaker,T.E.(1991). Backpropagation simulations using limited precision calculations. In Neural Networks, 1991., IJCNN-91-Seattle International Joint Conferenceon, volume2, IEEE Courbariaux, M., Bengio, Y.,David, J.P.: Training deep neural networks with low precision multiplications. arXiv preprint.
- Gupta, Suyog, Agrawal, Ankur,Gopalakrishnan, Kailash, and Narayanan plearning with limited, numericalprecision.arXiv preprint arXiv:1502.02551,2015.
Devblogs nvidia: Mixed-Precision Programming with CUDA8
Tensorflow: Quantizing Neural Networks with Tensorflow.

THANK YOU

Q & A