Profiling the Performance of Binarized Neural Networks

Daniel Lerner, Jared Pierce, Blake Wetherton, Jialiang Zhang

Outline

- Project Significance
- Prior Work
- Research Objectives
- Hypotheses
- Testing Framework
- Results
- Conclusions and (Hypothetical) Future Work

Project Significance

Applications [edit]

Applications for machine learning include:

- Adaptive websites
- Affective computing
- Bioinformatics
- Brain-machine interfaces
- Cheminformatics
- Classifying DNA sequences
- Computational anatomy
- Computer vision, including object recognition
- Detecting credit card fraud
- Game playing
- Information retrieval
- Internet fraud detection
- Marketing
- Machine learning control
- Machine perception
- Medical diagnosis
- Economics
- Natural language processing
- Natural language understanding
- Optimization and metaheuristic
- Online advertising
- Recommender systems
- Robot locomotion
- Search engines
- Sentiment analysis (or opinion mining)
- Sequence mining
- Software engineering
- Speech and handwriting recognition
- Financial market analysis
- Structural health monitoring
- Syntactic pattern recognition
- User behavior analytics
- Translation[36]

Binarization

- Assigns weights of +1, -1
  - Turns multiply-accumulates into accumulates
  - Better when done stochastically
- More energy efficient
  - Datacenters power and cooling limited
- Faster inference phase
  - Lower latency in commercial applications
- Could be important moving to low-power/mobile platforms

Prior Work

Intel Paper: Hardware Acceleration of BNNs

- Compared standard neural networks with their binarized counterparts on a variety of hardware platforms
  - FPGA, CPU, GPU, ASIC
- Applies two optimizations for CPU and GPU: Binarizing and Batching
  - Batches of size 10
- Focuses on the classification or inference stage
- Batching stays small to avoid latency (commercial applications)
  - GPU limited by this

- Binarize
  - CPU 5x faster
  - GPU 11x faster
- Batching (size 10)
  - CPU 80% faster
  - GPU 5.8x faster
- Binarize + Batch
  - GPU, ASIC fastest
  - ASIC, FPGA most efficient
  - High throughput, latency
- CPU, GPU Utilization
  - Poor without batching

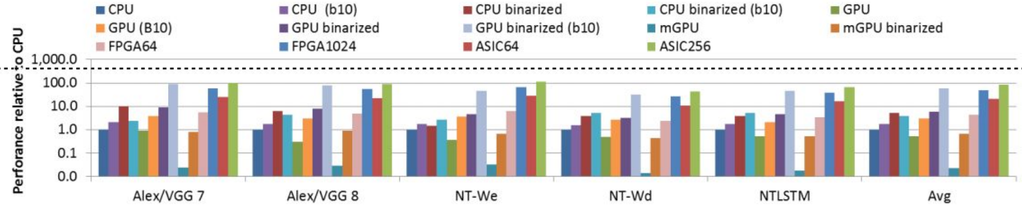


Fig. 6. Performance relative to baseline software on CPU. I.e., above 1 means speedup, while less than 1 means slowdown.

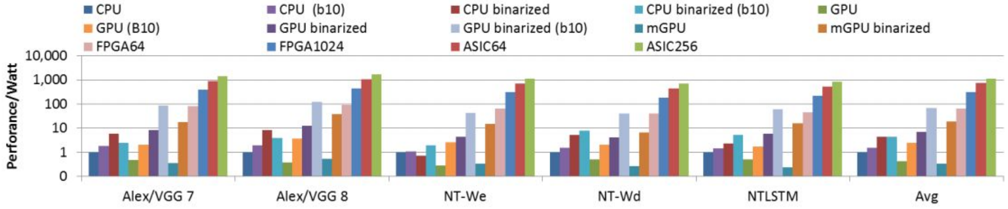


Fig. 7. Performance/Watt relative to baseline software on CPU.

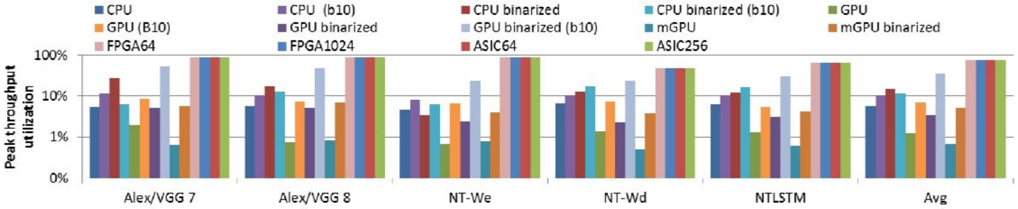
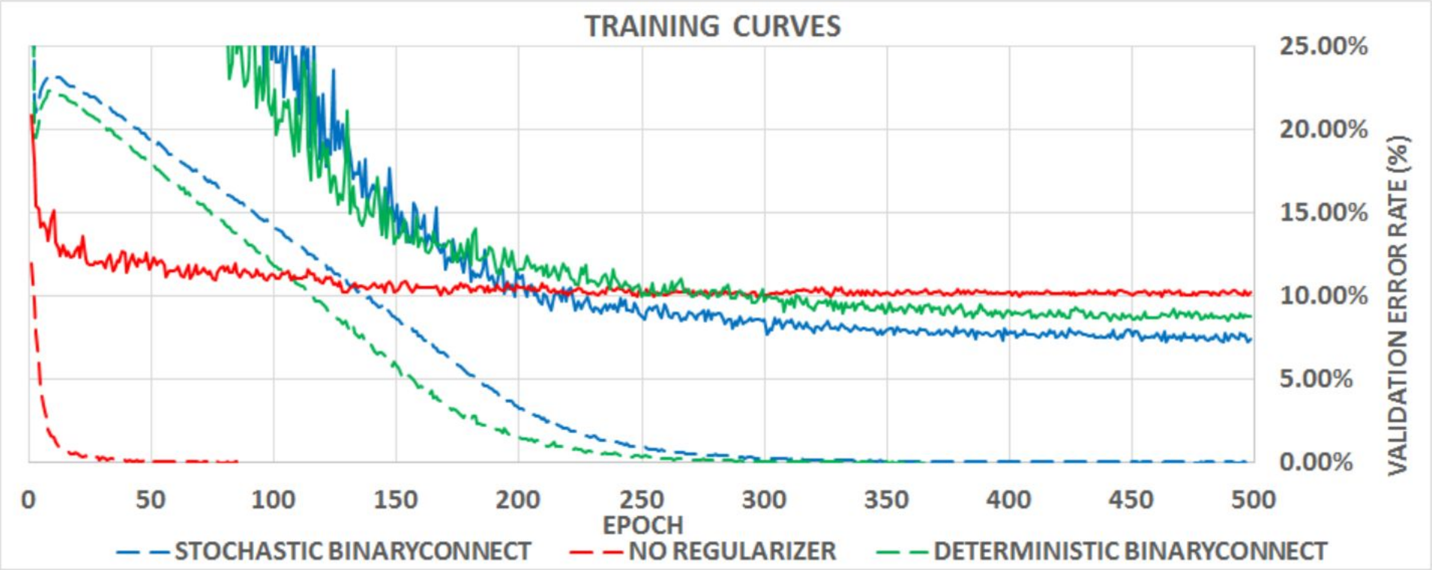


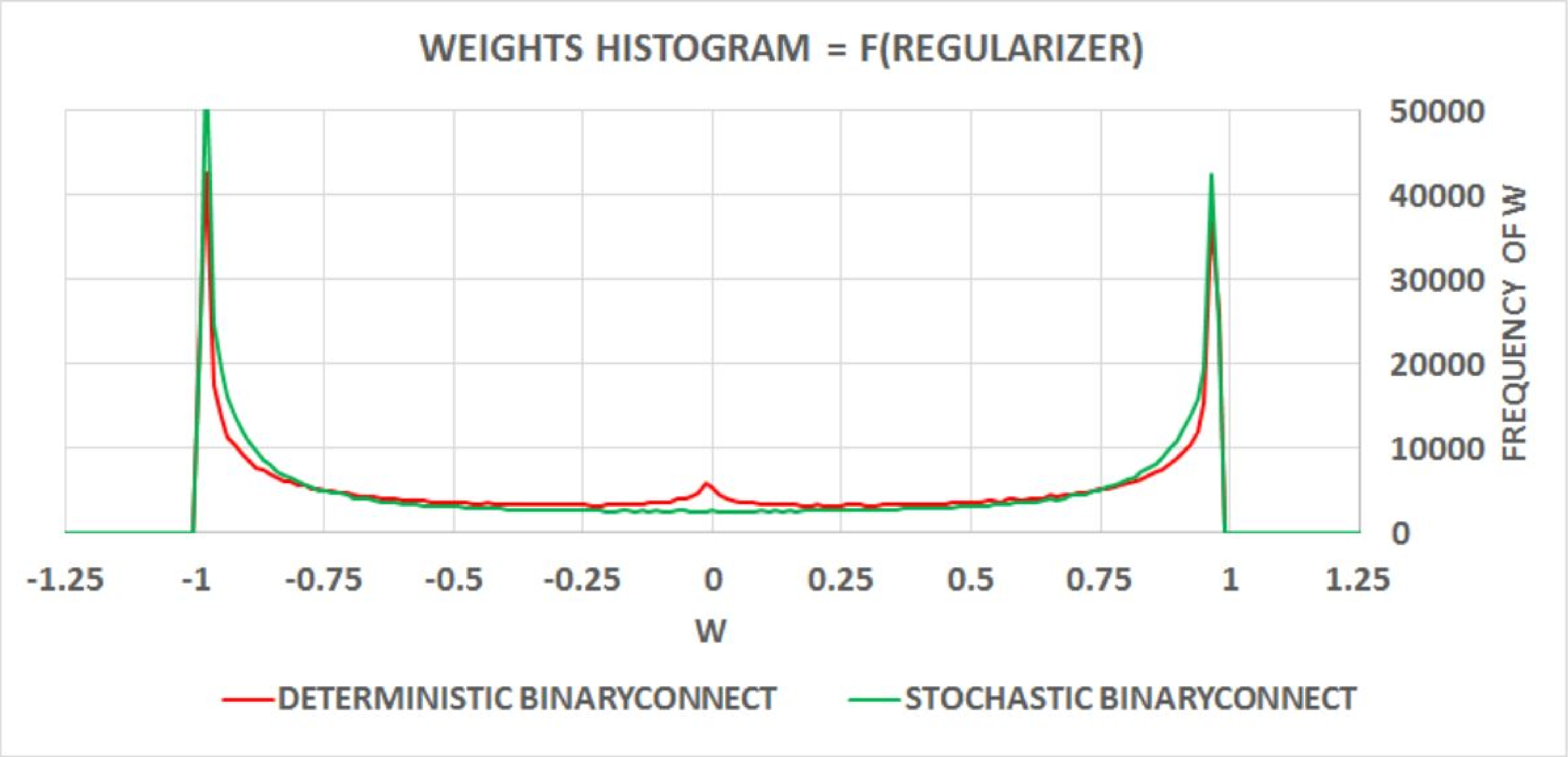
Fig. 8. Achieved performance relative to peak. E.g., 50% means only half of peak performance is realized.

Courbariaux: BinaryConnect Summary

- Introduces BinaryConnect method for training DNN with binary weights during forward and backward propagation
- Retain precisions of weights on which gradients are calculated
- Binarization acts as a regularizer
- Near state-of-the art accuracies achieved
- Both deterministic and stochastic binarization are implemented

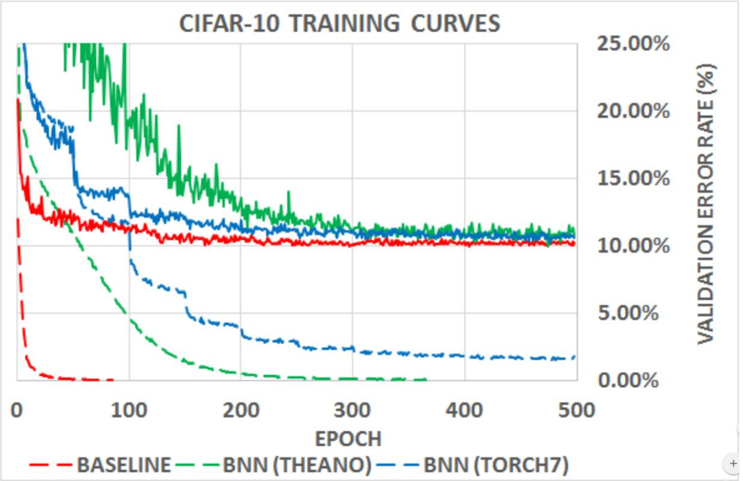
Courbariaux: BinaryConnect




WEIGHTS HISTOGRAM = F(REGULARIZER)
50000
40000
30000
20000
10000
0
FREQUENCY OF W
-1.25
-1
-0.75
-0.5
-0.25
0
0.25
0.5
0.75
1
1.25
W
DETERMINISTIC BINARYCONNECT
STOCHASTIC BINARYCONNECT

Matthieu Courbariaux: BinaryNet

- Introduces methods for training BNNs (weights and activations)
- Shows that BNNs are approximately as accurate as state-of-the-art DNNs
- Shows BNNs reduce memory usage and allow for bitwise operations, with the potential to reduce power consumption
- Implements a binary matrix multiplication GPU kernel for ~7x speedup



Objectives and Hypotheses

Objectives

- Recreate Intel’s results, test Courbariaux’s work
- Run a BNN on a CPU, GPU, and FPGA
- Test three differently sized datasets: MNIST, CIFAR-10, SVHN
- Measure performance, power consumption, and resource utilization
- Test the effects of batching
- Draw some conclusions for various applications of neural networks running on different hardware platforms

Hypotheses

- FPGA will be superior in terms of power consumption and resource utilization
- At small (~1) batch sizes, FPGA will outperform the others
- With larger batches, GPU will perform best
- Performance will scale nonlinearly with the size of dataset
- Relative performance of the three datasets will be comparable

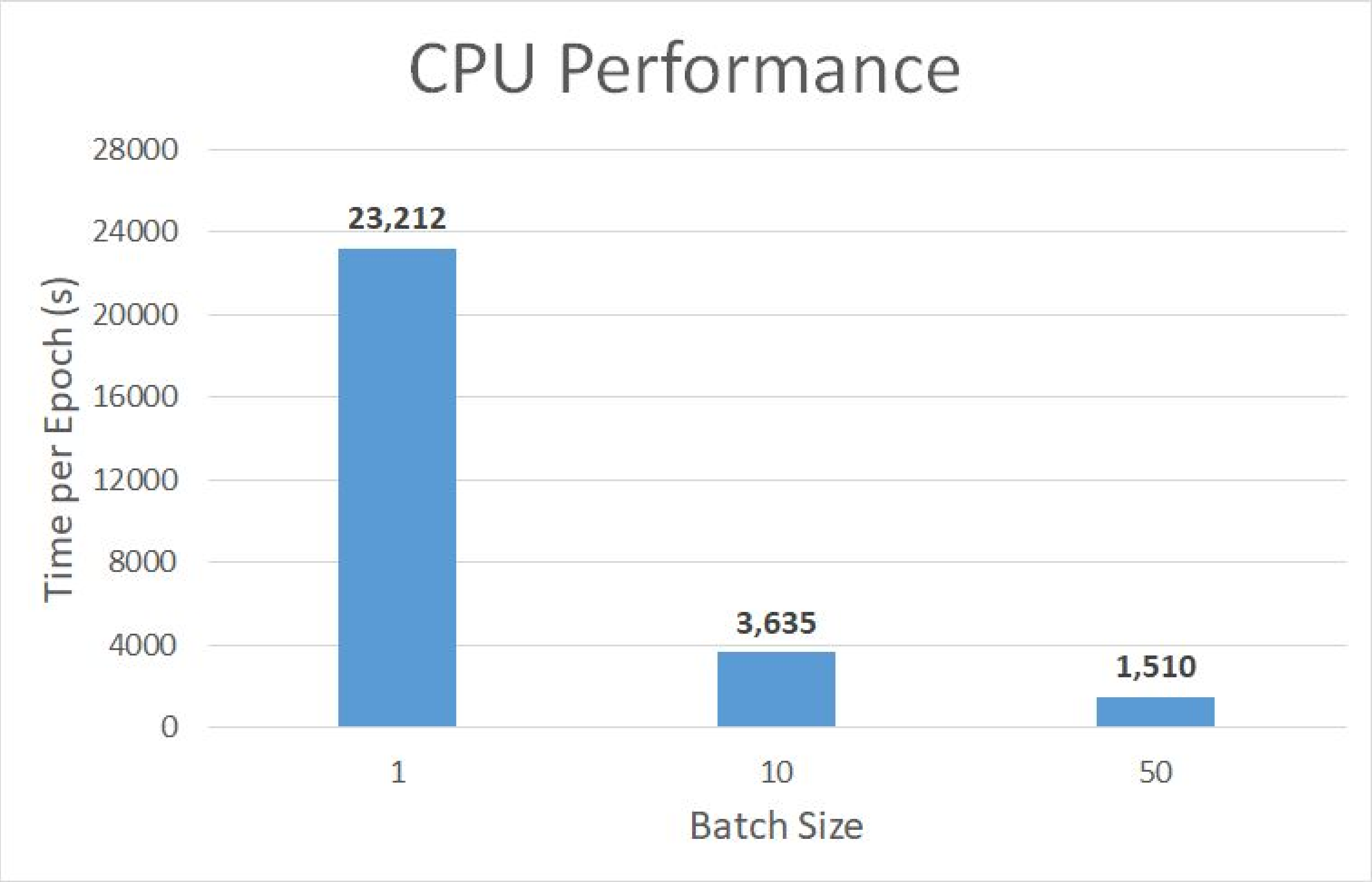
Testing Framework

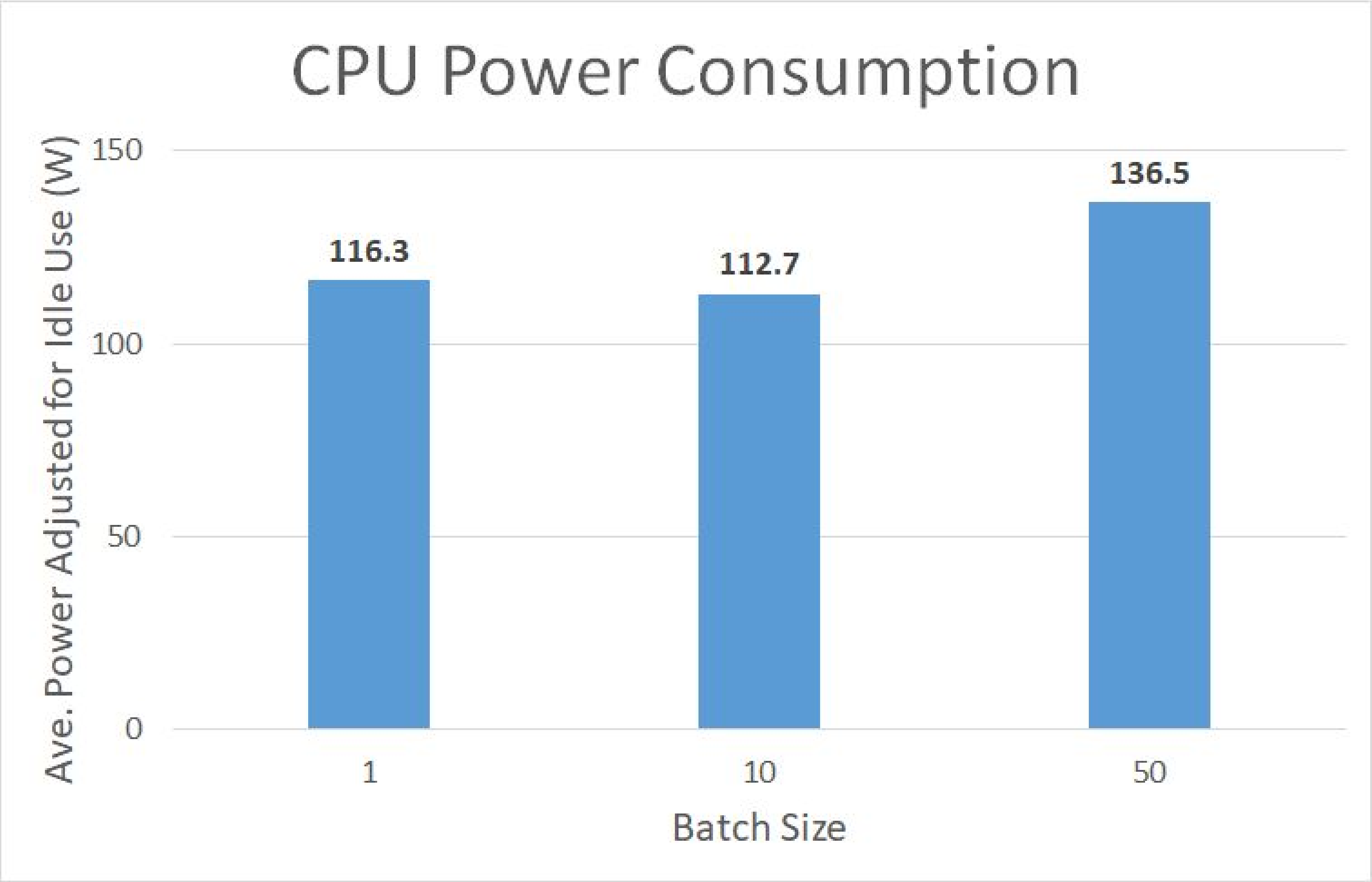
Testing Framework

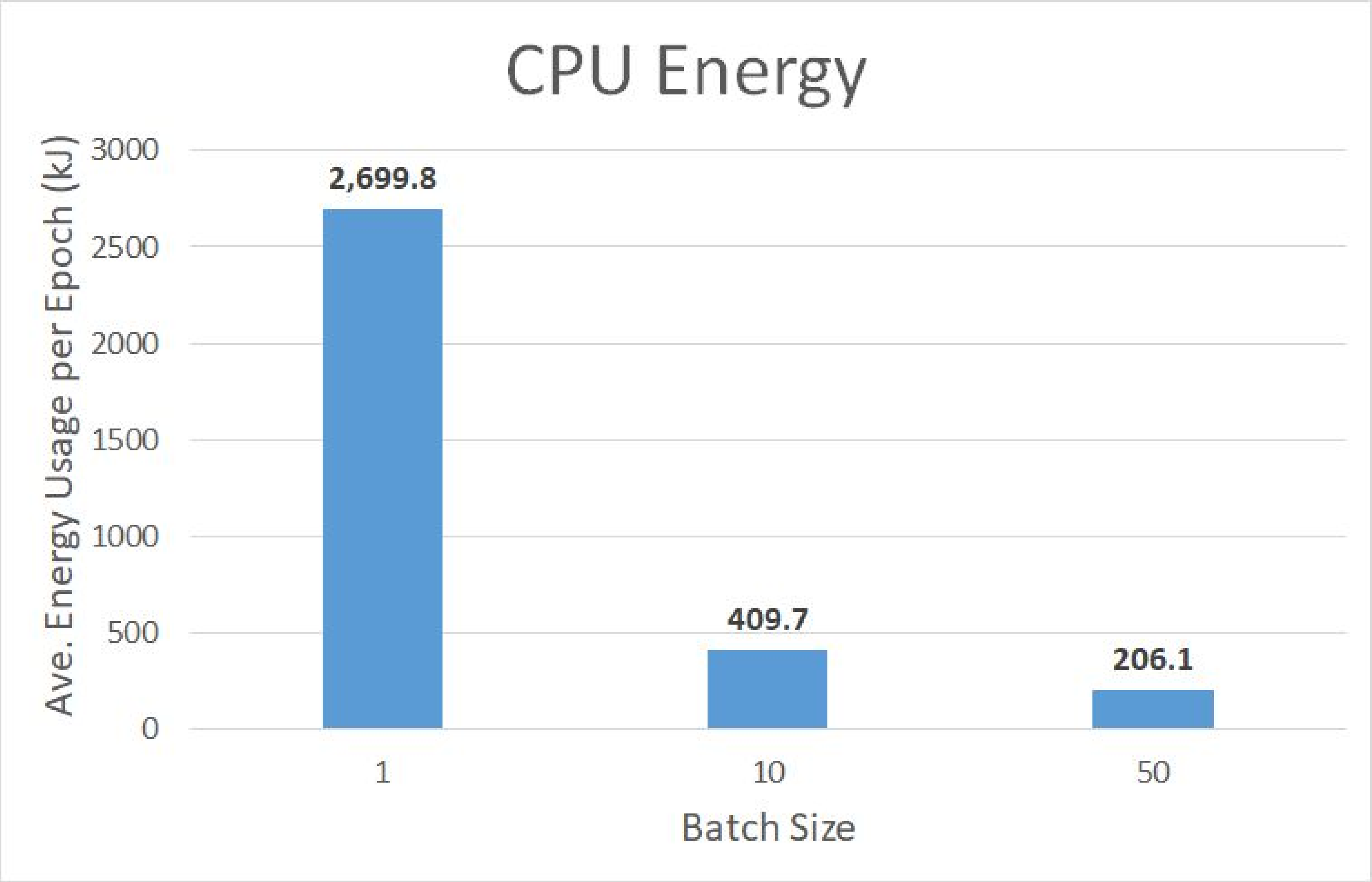
- Workload: BinaryConnect
  - Python scripts
  - Uses Theano for matrix operations
- Hardware: Intel i7 5960k, Nvidia GTX Titan X, and FPSoC (28nm XC7Z020)
- Metrics: Performance, Power, Resource Utilization
- Kill A Watt power meter

Testing Framework

- Performance
  - Measure run time for training datasets of MNIST, SVHN, and CIFAR-10 for each system
  - If dataset too large to run to completion, record epoch training times and extrapolate
- Power Consumption
  - Use a wall outlet power meter on the computer
  - Record total energy used in training session (kWh)
  - Subtract off an idle system energy (extrapolated from measured idle time consumption)
  - FPGA has its own hardware
- Resource Utilization
  - Linux command ‘top’ for CPU, ‘nvidia-smi’ for GPU
  - Whatever the FPGA does

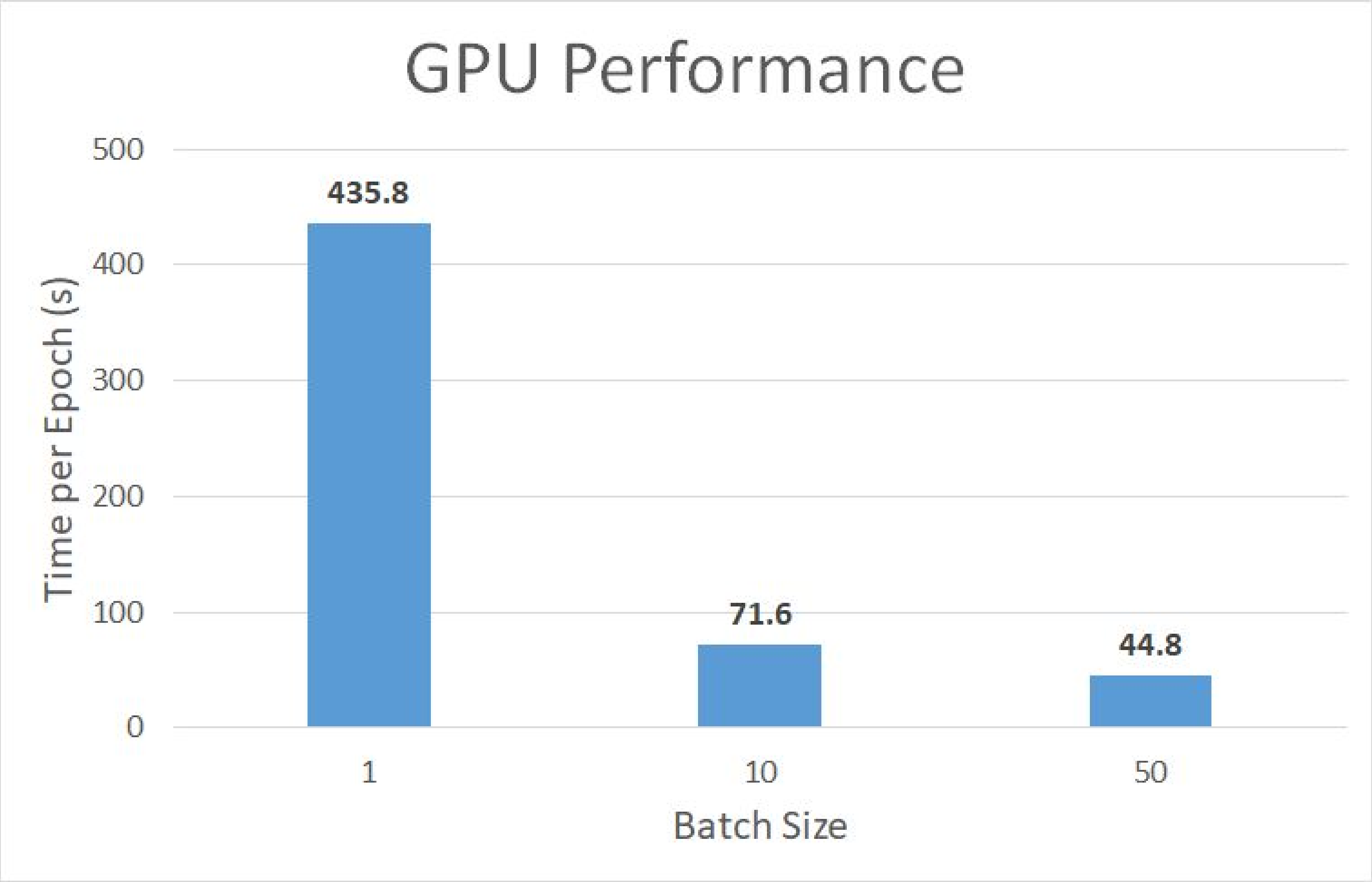

CPU Performance
28000
24000
20000
16000
12000
8000
4000
0
Time per Epoch (s)
23,212
3,635
1,510
1
10
50
Batch Size

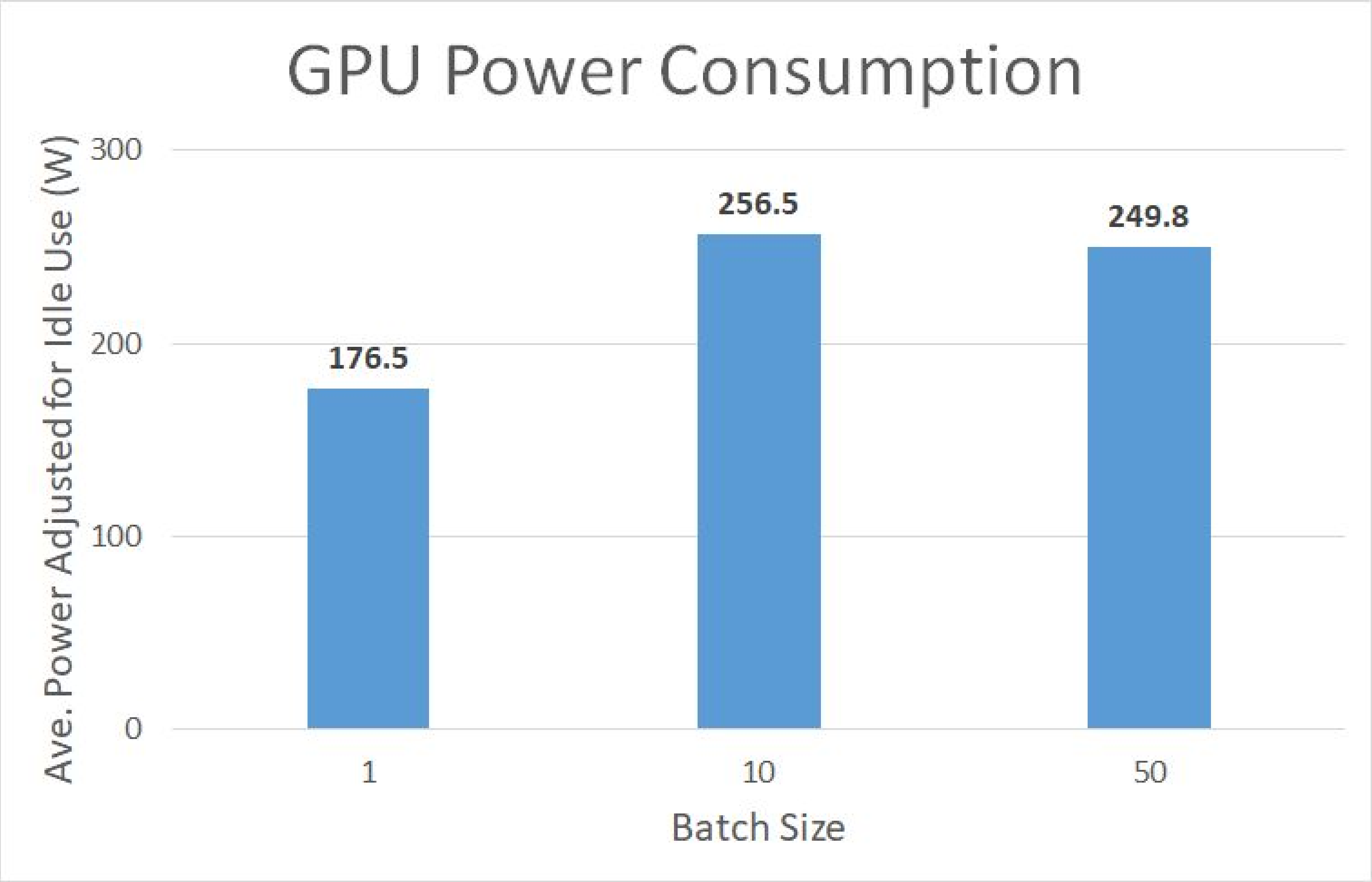

CPU Power Consumption
Ave. Power Adjusted for Idle Use (W)
150
100
50
0
116.3
112.7
136.5
1
10
50
Batch Size

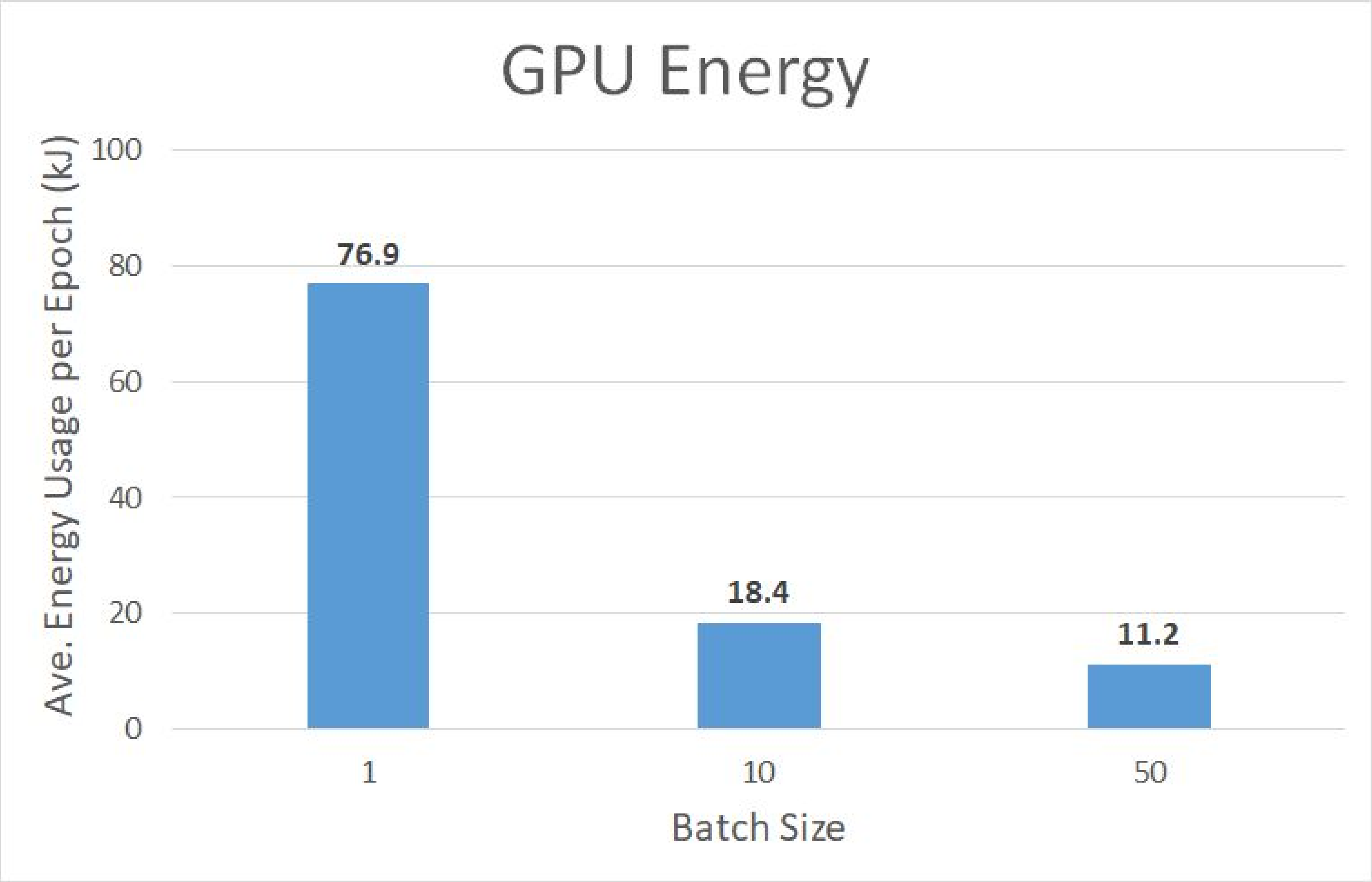

CPU Energy
Ave. Energy Usage per Epoch (kJ)
3000
2500
2000
1500
1000
500
0
2,699.8
409.7
206.1
1
10
50
Batch Size

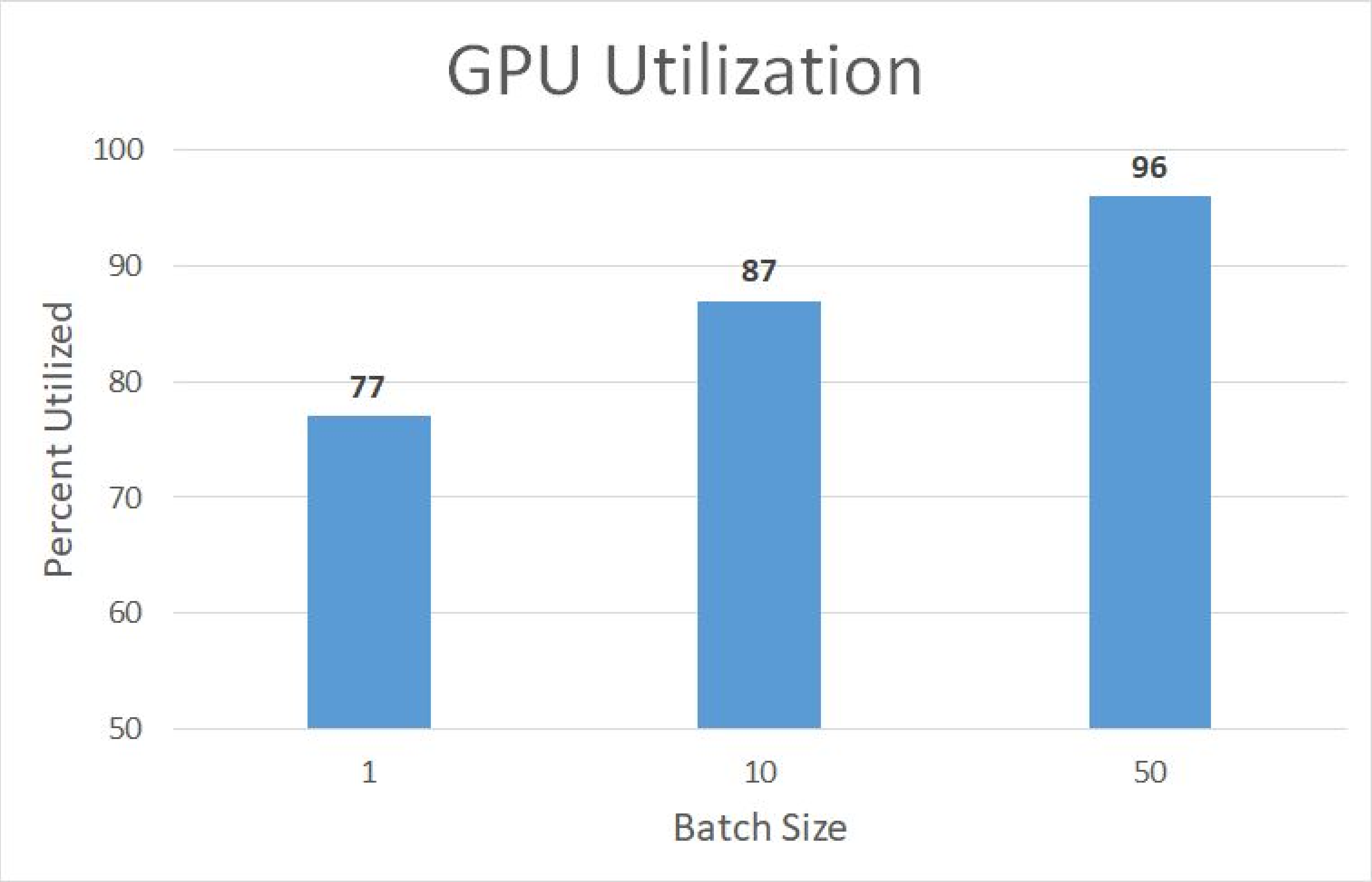
CPU Results

- The CPU is slow for training the BNN
  - The MNIST dataset takes ~18 hours
  - This makes CIFAR-10 and SVHN prohibitively large for our timeframe
  - We can take data several epochs in and extrapolate
- Power Usage
  - Higher with batching
- Utilization
  - 75% when running CIFAR-10 (6 out of 8 cores utilized)
  - i7 more appropriate for workload than Xeon


GPU Performance
Time per Epoch (s)
500
400
300
200
100
0
435.8
71.6
44.8
1
10
50
Batch Size


GPU Power Consumption
Ave. Power Adjusted for Idle Use (W)
300
200
100
0
176.5
256.5
249.8
1
10
50
Batch Size


GPU Energy
Ave. Energy Usage per Epoch (kJ)
100
80
60
40
20
0
76.9
18.4
11.2
1
10
50
Batch Size


GPU Utilization
Percent Utilized
100
90
80
70
60
50
77
87
96
1
10
50
Batch Size

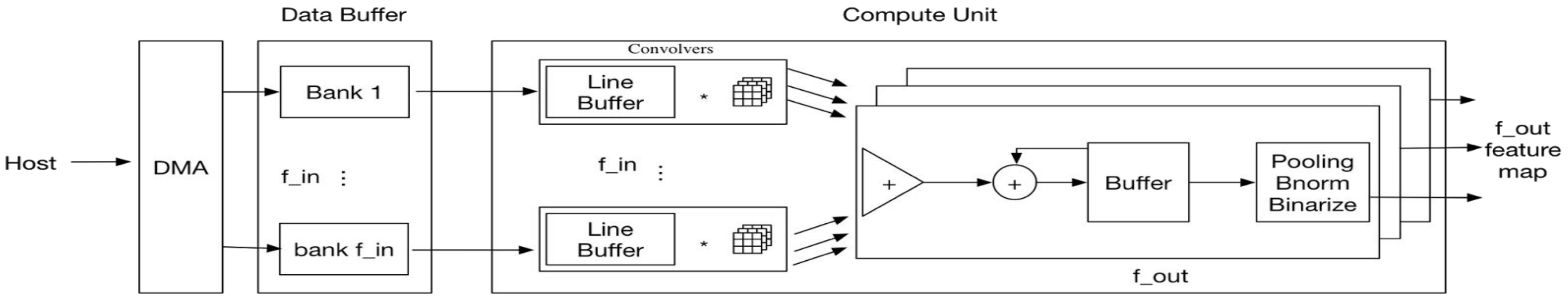
GPU Results

- GPU is faster than reported in BinaryConnect Repository
    - This could be a result of batching
- Power Consumption is heavy
    - Large batches more efficient than small batches due to performance
    - After a certain batch size, power consumption flattens out to maximum
- Resource Utilization is better than expected
    - Gets better with larger batches
    - >70% even on batch size 1

BNN Implementation on FPGA

- Target embedded FPSoC (28nm XC7Z020)
    - 53k LUTs, 106k FFs, 140 BRAMs, 220 DSPs
- Stores all feature maps on-chip
    - 4.9Mb of on-chip storage available
    - Used trained network from Theano
- Use Xilinx HLS to generate RTL from C source

Architecture of FPGA BNN Accelerator



- Two main components:
  - Data buffer and Compute Units
- f_in is the input parallelization factor
- f_out is the output parallelization factor

Experimental Results

- Fixed f_out= 1 to find the trade-off between f_in and f_out
  - LUT and FFs usage scaled with number of Convolvers
  - BRAM and DSPs generally insensitive to f_in
  - Runtime weakly scales with number of Convolvers
- Consumes 5W at f_in=8

f_in	LUT	FF	BRAM	DSP	RUNTIME
1	25859	28197	86	3	17.5ms
2	35291	37125	87	3	10.8ms
4	38906	36771	87	3	7.98ms
8	46900	46134	94	3	5.94ms

Comparing Hardware Platforms

CPU

Batch Size	Performance (s/epoch)	Energy Usage (kJ/epoch)	Average Power (W)
1	23212	2699.80	116.31
10	3635	409.74	112.72
50	1510	206.12	136.5

GPU

Batch Size	Performance (s/epoch)	Energy Usage (kJ/epoch)	Average Power (W)
1	435.8	76.92	176.5
10	71.6	18.37	256.5
50	44.8	12.11	249.81

Preliminary Results

- GPU is clearly superior to CPU in all but instantaneous power draw
- Batching helps performance and energy efficiency, but sub-linearly
- Batching increases power draw and resource utilization sub-linearly
- It is not immediately clear how GPU and CPU compare to FPGA, but we anticipate the power usage, at a minimum, will be far superior for the FPGA.

Conclusions and Hypothetical Future Work

- Hardware can improve performance of BNNs immensely
- Implement ASIC
- Better Power Measurement
- More direct comparison with standard CNNs in inference mode

References

Courbariaux, Matthieu, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." *arXiv preprint arXiv:1602.02830* (2016).

Courbariaux, Matthieu, Yoshua Bengio, and Jean-Pierre David. "BinaryConnect: Training deep neural networks with binary weights during propagations." In *Advances in Neural Information Processing Systems*, pp. 3123-3131. 2015.

Nurvitadhi, Eriko, David Sheffield, Jaewoong Sim, Asit Mishra, Ganesh Venkatesh, and Debbie Marr. "Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC." *Proc. ICFPT* (2016).

Zhang, Jialiang, and Jing Li. "Improving the Performance of OpenCL-based FPGA Accelerator for Convolutional Neural Network." In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 25-34. ACM, 2017.