RUTGERS
WINLAB | Wireless Information Network Laboratory

Using FPGAs for Machine Learning Acceleration

Milos Seskar, Michael Yakubov
Advisors: Prasanthi Maddala, Richard Martin, Jennifer Shane

Overview

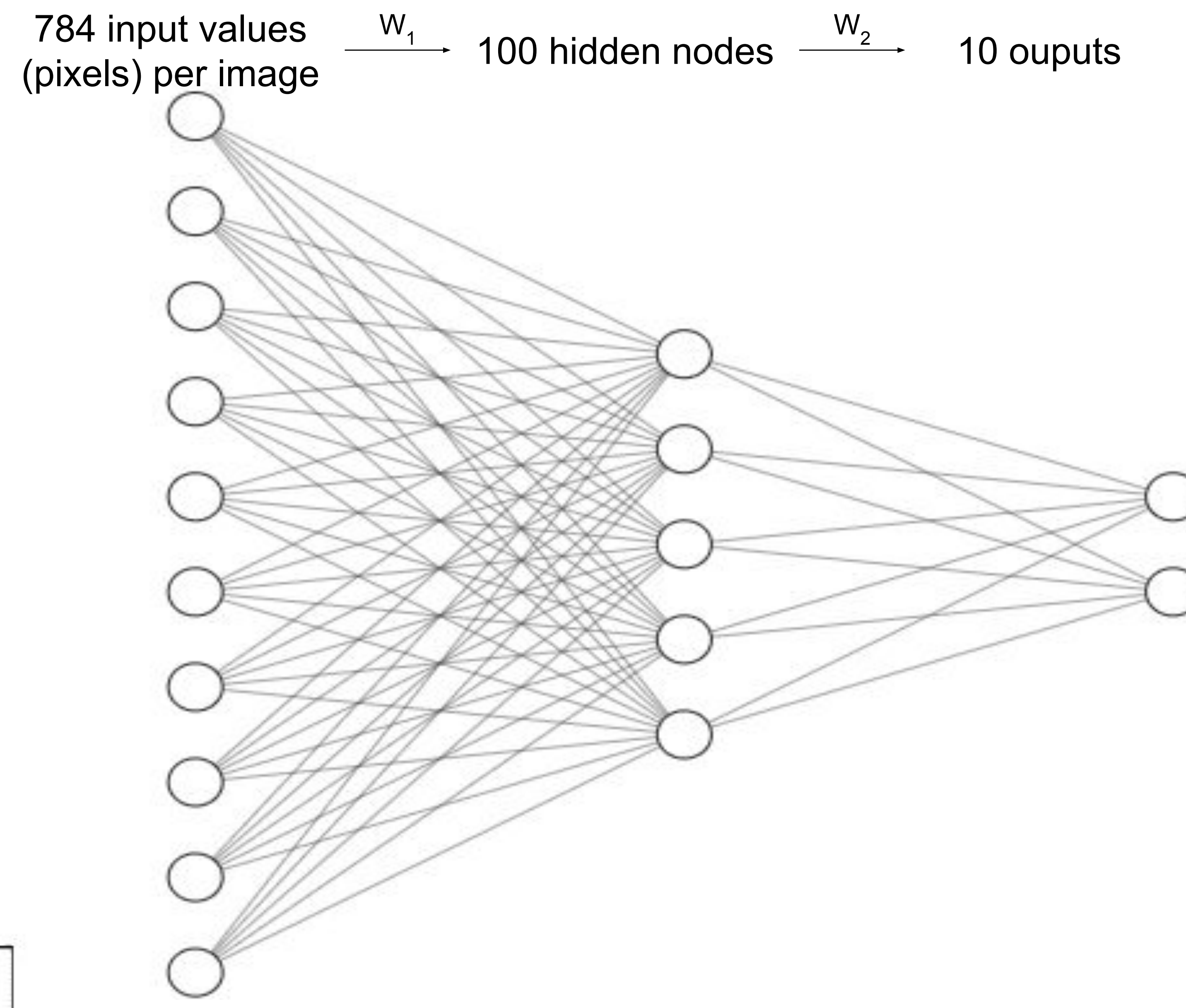
Machine learning (ML), like most software operations, is typically executed using the central processing unit (CPU) of a computer. More recently people have used graphics processing units (GPUs) as they they can perform many operations simultaneously, resulting in significantly faster performance.

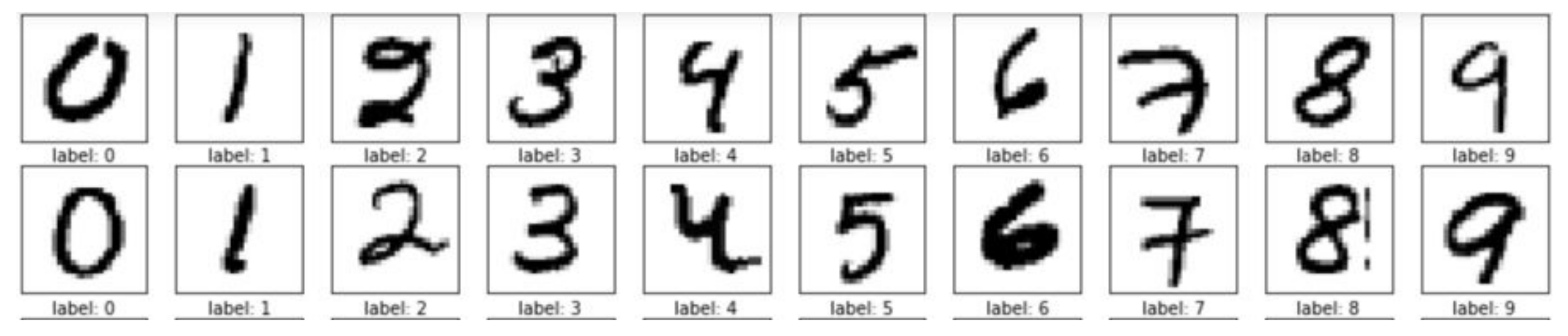
CPUs and GPUs are designed to be effective and versatile: easy to use for any number of applications, but with limits in peak performance for any one task. Alternatively, creating an Application Specific Integrated Circuit (ASIC) for every unique task is costly. Luckily, Field Programmable Gate Arrays (FPGA) are a type of integrated circuit that are designed to be reprogrammable, allowing for users to optimize their hardware for specific software applications to achieve higher peak performance. The goal of this project is to evaluate the performance of a basic ML application when deployed on an FPGA versus a traditional CPU or GPU.

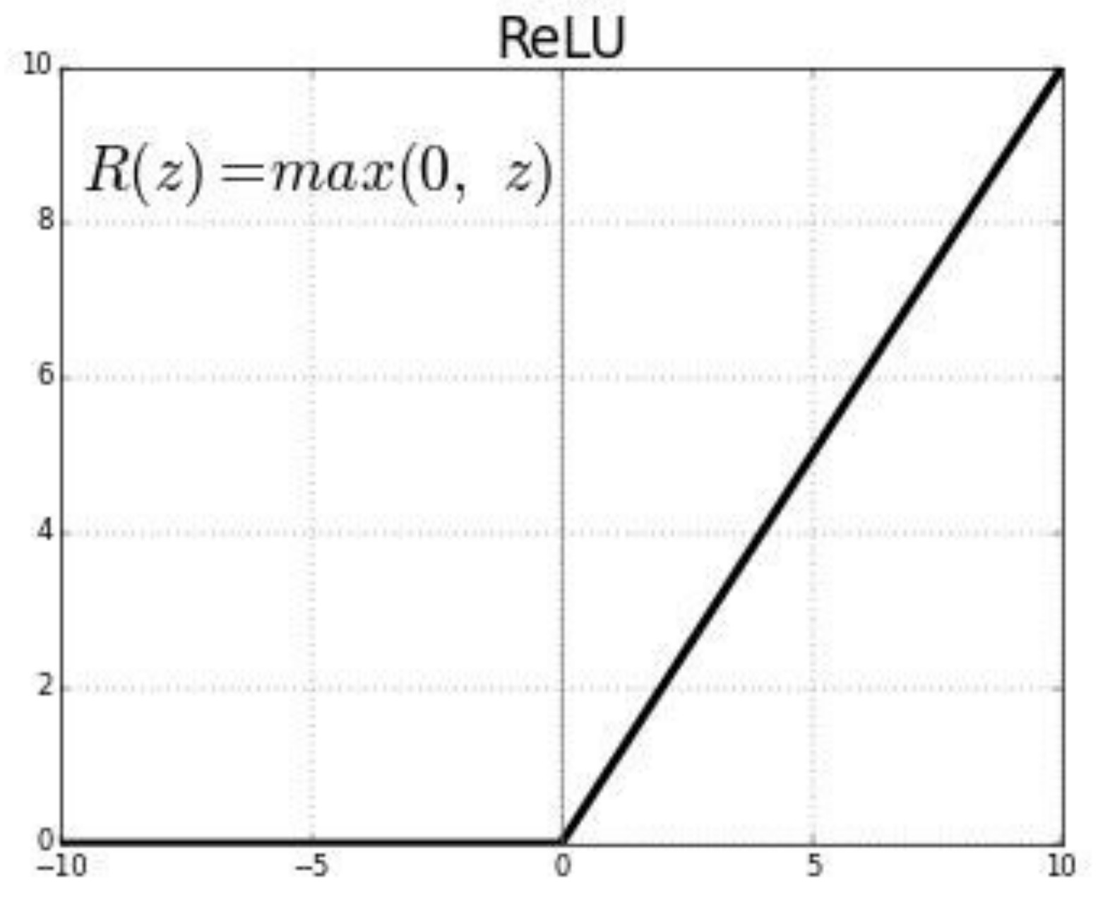
Methodology

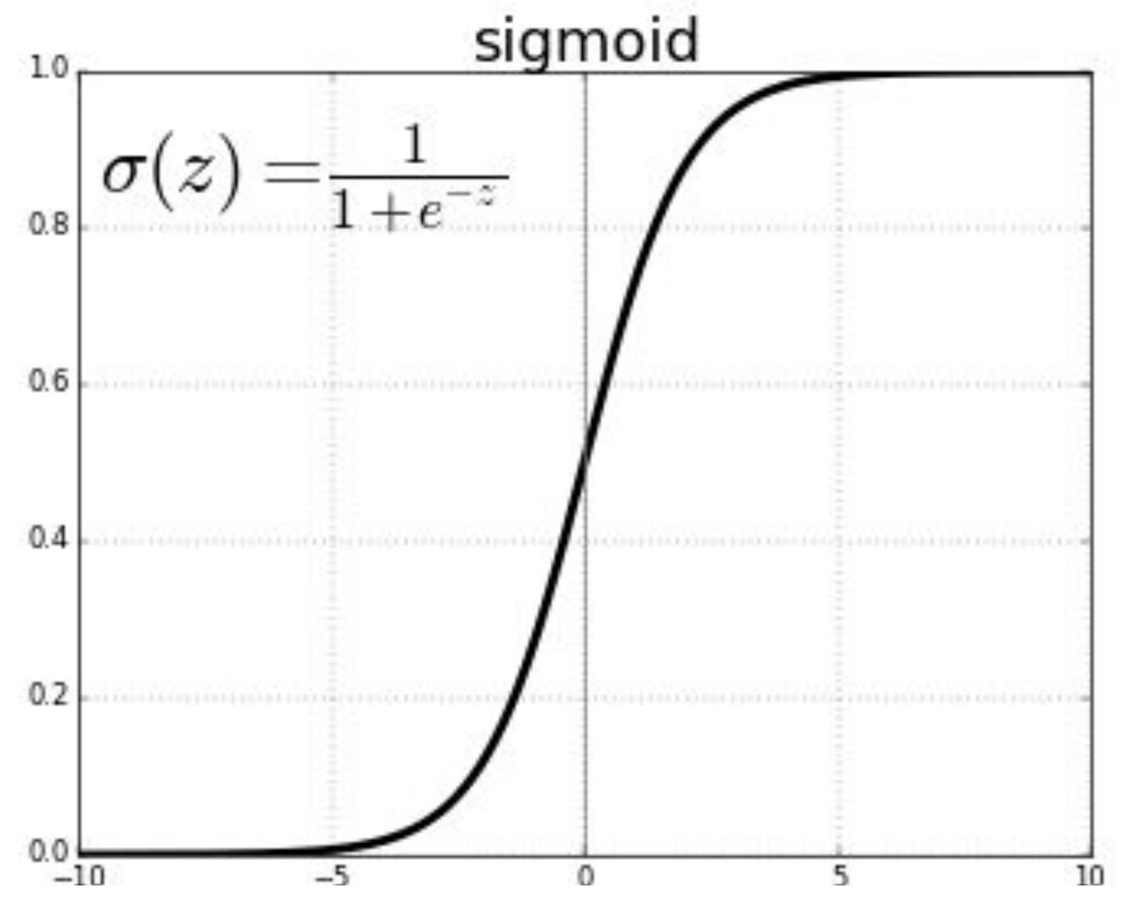
Neural Network:

- Written in Golang (Go) progamming language
- 3 layer neural network:
  - 784 inputs (28x28 grayscale image, 1 per pixel)
  - 100 hidden nodes
  - 10 outputs (1 per digit)
- Activation function: Rectified Linear Unit (ReLU)
  - Used to be Sigmoid
- Training dataset: 60000 images of handwritten numbers (MNIST dataset)
- Testing dataset: 10000 images of handwritten numbers (MNIST datsaset)







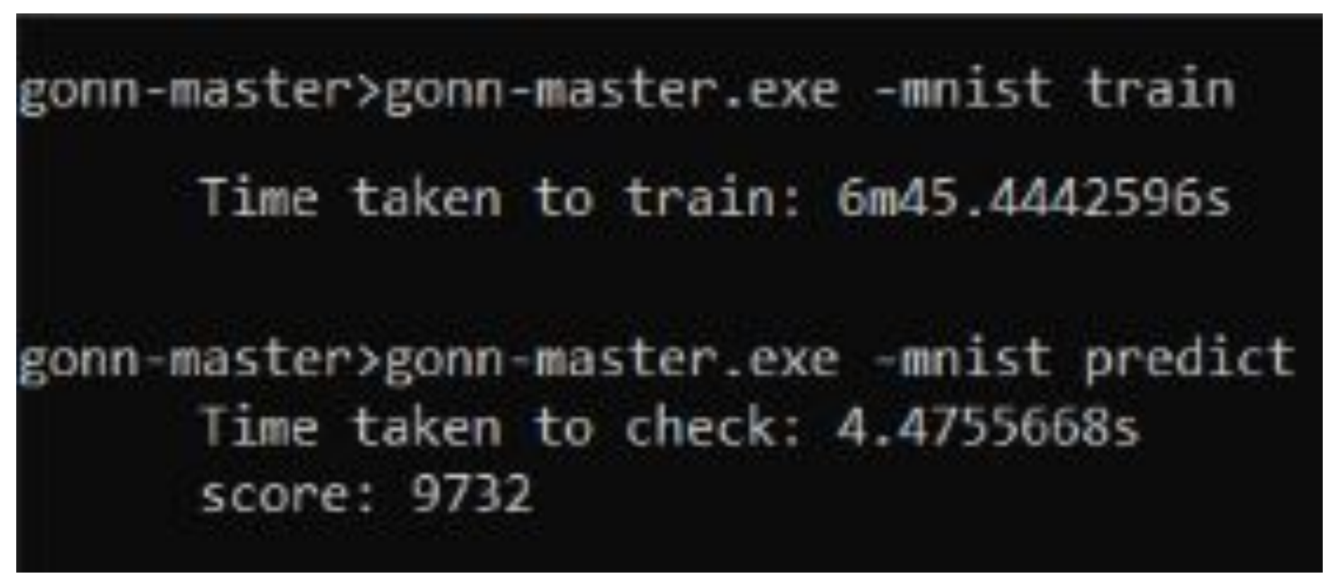


References

- Xilinx documentation: https://www.xilinx.com/support/documentation/sw_manuals/ug998-vivado-intro-fpga-design-hls.pdf
- Python machine learning model: https://machinelearningmastery.com/implement-perceptron-algorithm-scratch-python/
- Golang Neural Network: https://sausheong.github.io/posts/how-to-build-a-simple-artificial-neural-network-with-go/
- Project Repository: https://github.com/mseskar/mlaccel

Results

FPGAs work significantly better with integer arithmetic, so we needed to implement this in our neural network. Part of this was switching from a sigmoid activation function to ReLU. Sigmoid always returns a floating point value between 0 and 1, while ReLU returns either 0 or the input value, giving the potential of always returning an integer. Once this was fully implemented in our neural network, we had a prediction rate of 97.32%.



```
gonn-master>gonn-master.exe -mnist train
       Time taken to train: 6m45.4442596s

gonn-master>gonn-master.exe -mnist predict
       Time taken to check: 4.4755668s
       score: 9732
```

Future Work

- Convert the neural network from floating point to integer arithmetic
- Continue doing tests on an FPGA, as most of our tests up to this point were done on a CPU
- Implement this concept with more complex machine learning models (i.e. recognizing multicolored images)
- Evaluate how much better FPGA performs on this model compared to CPU and GPU

WINLAB