Neural Network Acceleration on GPUs

By

Ibrahim Ahmed Lone	22i-1193
Muhammad Yaseen	22i-1221
Abdul Hadi	22i-1333

How much speedup is possible on a MNIST Neural Network using GPU—and at what accuracy cost?

Agenda

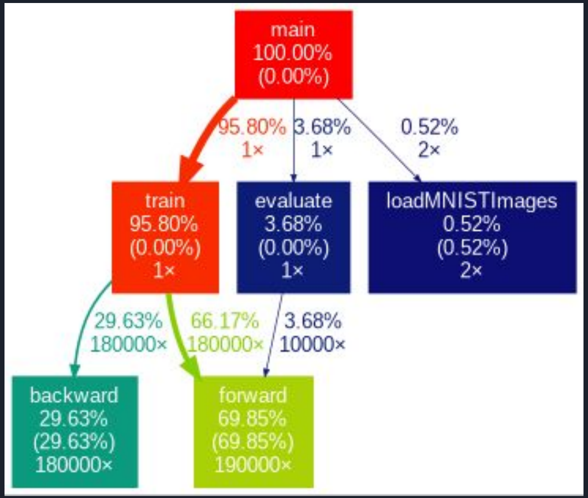
- Version 1: Sequential CPU
- Version 2: Naïve CUDA
- Version 3: Optimized CUDA
- Version 4: Tensor-Core Acceleration
- Comparative Results
- Conclusion

Version 1

Version 1: Sequential CPU

1. **Results:**
   Total training time: 22.233 s
   Test Accuracy: 96.78%.
   End-to-end execution: 23.589 s.

2. **Observations:**
   Loop-carried Dependencies (forward/backward).
   Serves as the baseline for the speedup.



Version 2

Version 2: Naïve CUDA

Key changes:

- cudaMalloc/cudaMemcpy.
- Two kernels:
  - forward/backward pass.

Results:

- Total training time: 206.037 s
- Test Accuracy: 96.58%
- Execution time: 207.567 s

Observations:

GPU overhead > compute

Low Occupancy

Memory-bound operations are the bottleneck now

Version 3

Version 3: Optimized CUDA

Key changes:

Launch & Occupancy: tuned dim3 grid/block (256–512 threads)

Communication: batched H2D/D2H, streams overlap

Memory: shared-memory, coalesced access

Results:

Total training time: 0.856s
Test Loss: 0.3440
Execution time: 1.097s

Observations:

~210 speedup in its training compared with V2.

Memory-bound operations are now well-optimized, with compute-bound kernels being the bottleneck now.

Version 4

Version 4: Tensor-Core Acceleration

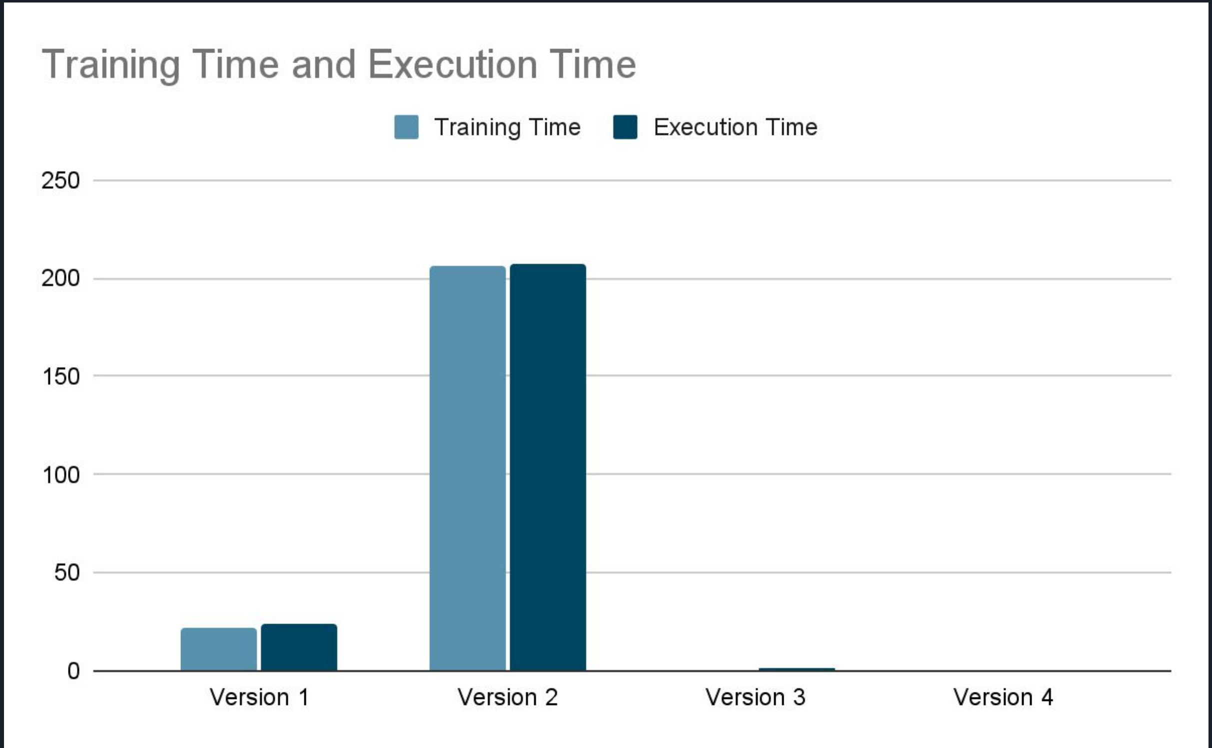
Key changes:
Precision: FP16 inputs/weights, FP32 accumulations
API: NVIDIA WMMA (16×16 warp tiles)
Alignment: data padded to 16×16 blocks

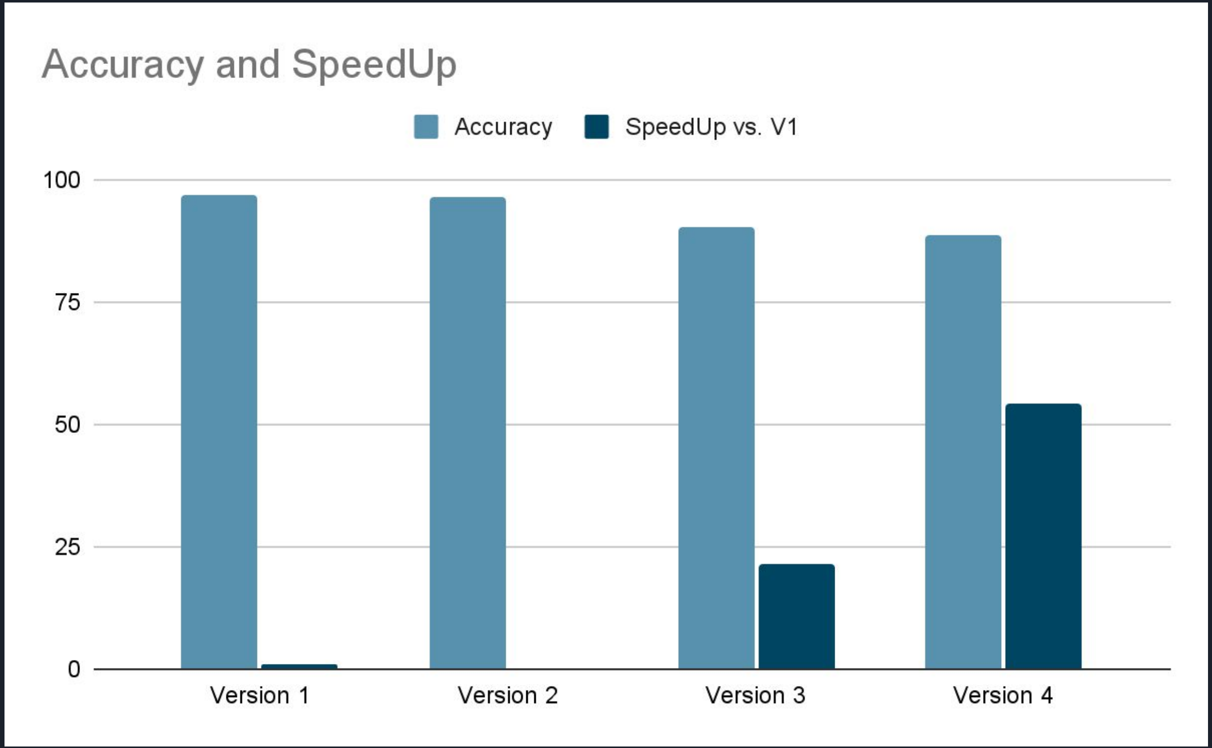
Results:
Total training time: 0.262 s
Test accuracy: 88.76%
Execution time: 0.435 s

Observations:
~3.3× faster than V3; ~54× faster than V1.
Occupancy still high; tensor cores peak compute.

Comparative Results

Version	Train Time (s)	Exec Time (s)	Test Acc. (%)	Speedup vs. V1
V1	22.233	23.589	96.78	1×
V2	206.037	207.567	96.58	0.11×
V3	0.856	1.097	90.27	21.5×
V4	0.262	0.435	88.76	54.2×


Training Time and Execution Time
Training Time
Execution Time
250
200
150
100
50
0
Version 1
Version 2
Version 3
Version 4


Accuracy and SpeedUp
Accuracy
SpeedUp vs. V1
100
75
50
25
0
Version 1
Version 2
Version 3
Version 4

Conclusion

Moving from CPU→naïve CUDA (V1→V2) without algorithmic changes delivers poor speedup due to overheads.

Advanced optimizations (shared memory, streams, WMMA) add significant code complexity and tuning effort - though yielding diminishing results.

The performance improvement trajectory showed a distinct drop from 23.6 s on CPU (V1) down to 0.44s using tensor cores (V4) at a speedup of ~54 ×. The accuracy drops slightly when one is too aggressive on optimizations - speedup.