

# Accelerating a CNN on Vortex

Anmol Gupta

Akshara Kuduvalli

Shilpa Rao

University of California, Los Angeles | CS 259 | Fall 2025

**Abstract**—We accelerate a CNN using the Vortex GPGPU.

**Keywords**—CNN, Vortex GPGPU, FashionMNIST, GPU, Tensor Core Unit

## I. INTRODUCTION & MOTIVATION

Convolutional Neural Networks (CNNs) consist mostly of convolution operations, which comprise 90% of overall runtime in typical inference workloads. Per-kernel analysis via standalone regression tests shows that convolution is the dominant bottleneck.

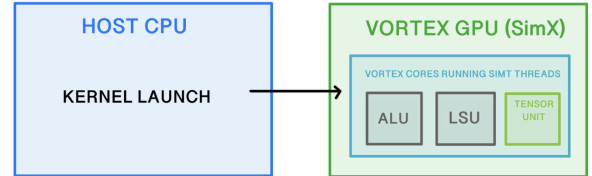
- Convolution IPC  $\approx 2.8$
- Pooling IPC  $\approx 1.26$
- ReLU IPC  $\approx 0.81$

(These IPC values are measured from isolated regression kernels and represent peak per-kernel behavior rather than end-to-end pipeline IPC.)

The regression tests alongside the data-parallel nature of convolutional operations demonstrated to us that CNNs were optimal for parallelization on GPU architectures. Our motivation was to explore how a CNN inference pipeline can be mapped onto the Vortex GPGPU architecture, and to evaluate the potential performance benefits of accelerating convolution using Vortex’s Tensor Core Unit (TCU). We trained a CNN on the FashionMNIST dataset using TensorFlow, implemented an end-to-end CNN inference pipeline on Vortex as a baseline using conventional floating-point operations. The tensor-core-accelerated path is evaluated at the convolution kernel level using `im2col` + GEMM and is not fully integrated into the end-to-end multi-image CNN pipeline. We trained a simplified CNN architecture on the FashionMNIST dataset using TensorFlow, and ported the resulting weights to the Vortex execution environment. Our primary goal was to leverage the Vortex Tensor Core Unit (TCU) to accelerate the convolution.

## II. IMPLEMENTATION

### A. High Level Architecture



We use a Host CPU (running C++) to launch kernels on the Vortex GPU (SimX). The Vortex GPU contains Vortex cores that run SIMT (Single Instruction, Multiple Thread) threads, which can leverage the dedicated Tensor Unit for accelerated operations. The CNN consists of the following layers:

- Conv2D:  $3 \times 3$  kernel, 8 output channels
- MaxPool2D:  $2 \times 2$  pooling
- ReLU activation
- Flatten
- Fully Connected layer:  $1352 \rightarrow 10$

The model was trained in Python using Keras on the Fashion-MNIST dataset and achieves approximately 90% validation accuracy. The trained weights were exported and reused without modification in the Vortex inference pipeline.

### B. Software stack & execution paths

On the software side, CNN layers were implemented in C++ targeting the Vortex ISA. The full inference pipeline—including convolution, ReLU activation, and pooling—was executed on the GPU. Two distinct convolution paths were developed: a baseline implementation using standard floating-point operations, and an accelerated implementation that transforms the convolution into a matrix multiplication using `im2col`, enabling execution on the tensor core.

We used the FashionMNIST dataset, consisting of 60,000 grayscale  $28 \times 28$  images labeled into 10 clothing categories. A simple CNN architecture (convolution, activation, pooling, followed by a fully connected layer) was trained in TensorFlow. The trained weights were exported and ported into

the Vortex environment. While convolution, ReLU, and pooling were implemented on the GPU, the fully connected layer remained on the host CPU due to time and stability constraints.

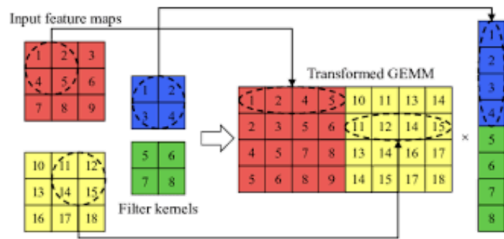
To ensure functional correctness, the pipeline was validated layer by layer against a Python reference implementation. Intermediate tensors from the Vortex execution (convolution output, pooling output, ReLU output, and logits) were printed and numerically compared against Python outputs for identical inputs and weights. All layers matched the Python reference within floating-point tolerance. Final predictions and probability distributions were also consistent with the reference model.

The two execution paths used to implement the convolution layer:

**1. Baseline Path (FPU-only execution):** Uses the regular CNN operation, relying on the Floating-Point Unit (FPU) for computation.

| INPUT IMAGE |     |     |     |     |  | WEIGHT |   |   |     |
|-------------|-----|-----|-----|-----|--|--------|---|---|-----|
| 18          | 54  | 51  | 239 | 244 |  | 1      | 0 | 1 | 429 |
| 55          | 121 | 75  | 78  | 95  |  | 0      | 1 | 0 |     |
| 35          | 24  | 204 | 113 | 109 |  | 1      | 0 | 1 |     |
| 3           | 154 | 104 | 235 | 25  |  |        |   |   |     |
| 15          | 253 | 225 | 159 | 78  |  |        |   |   |     |

**2. Accelerated Path (Tensor core execution):** Uses the **im2col + GEMM** conversion method to map the convolution operation to matrix multiplication, which is then accelerated by the Tensor Core Unit (TCU) in Vortex.



### C. GPU Pipeline Architecture

The CNN model used consists of convolutional, pooling, and ReLU layers.

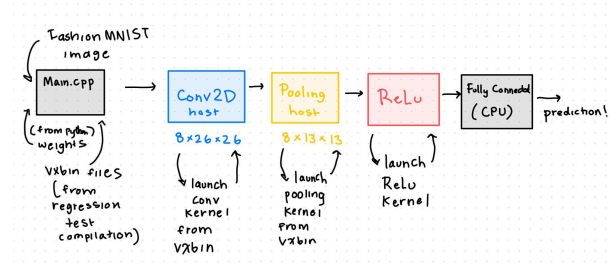


Figure 1: GPU Pipeline Architecture

## III. EVALUATION AND RESULTS

### A. Overall evaluation of results

The Vortex CNN achieves 89% accuracy on 100 Fashion-MNIST images, closely matching the Python reference model. Misclassifications primarily occur between visually similar classes such as Shirt and Coat, consistent with expected dataset behavior.

Functionally, the baseline CNN pipeline executed end-to-end on Vortex, producing consistent intermediate outputs (convolution, ReLU, pooling) when compared against reference CPU results for single-image inference. The accelerated tensor-core version was also compiled and executed successfully, confirming integration of the **im2col + GEMM** path. Performance was evaluated using SimX metrics, including instruction count, execution cycles, and instructions per cycle (IPC), across multiple warp and thread configurations. Results showed that the accelerated CNN achieved modest but measurable IPC improvements in certain configurations, particularly at higher warp and thread counts.

For example, with 4 warps and 8 threads, IPC increased from approximately 0.576 (baseline) to 0.614 (accelerated). In other configurations, performance was comparable, highlighting the sensitivity of results to kernel configuration and workload size. While single-image correctness was validated, multi-image accuracy evaluation was not fully completed due to a persistent convolution output mismatch leading to constant predictions. Our evaluation nonetheless shows the feasibility of tensor-core acceleration on Vortex.

## B. Baseline convolutional layer vs. im2col + TCU accelerated GEMM performance results

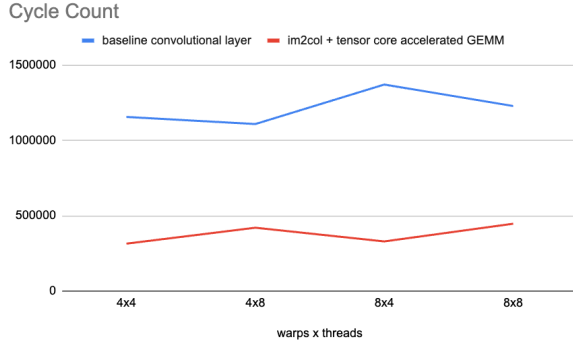


Figure 2: Cycle count comparison between 2 execution paths

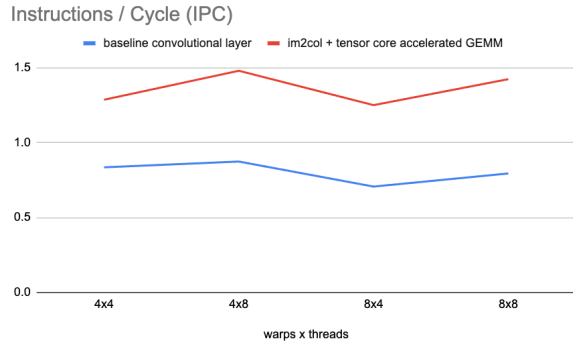


Figure 3: IPC comparison between 2 execution paths

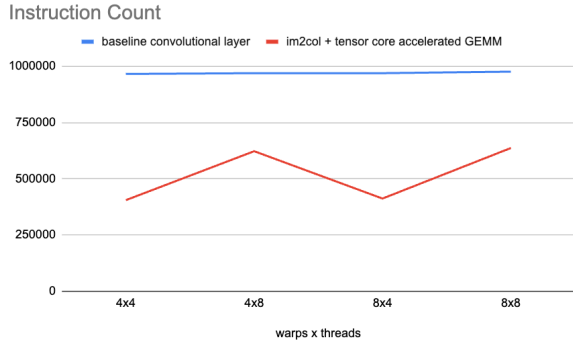


Figure 4: Instruction count comparison between 2 execution paths

These results show that the im2col + TCU path reduces cycle count and increases IPC in higher-parallelism configurations, while instruction count remains comparable due to data reorganization overhead.

| Instructions    |                              |                                       |
|-----------------|------------------------------|---------------------------------------|
| warps x threads | baseline convolutional layer | im2col + tensor core accelerated GEMM |
| 4x4             | 965937                       | 404965                                |
| 4x8             | 969329                       | 622506                                |
| 8x4             | 969345                       | 412065                                |
| 8x8             | 976113                       | 636646                                |
| Cycle count     |                              |                                       |
| warps x threads | baseline convolutional layer | im2col + tensor core accelerated GEMM |
| 4x4             | 1156420                      | 314735                                |
| 4x8             | 1109267                      | 420558                                |
| 8x4             | 1371561                      | 329484                                |
| 8x8             | 1228958                      | 447052                                |
| IPC             |                              |                                       |
| warps x threads | baseline convolutional layer | im2col + tensor core accelerated GEMM |
| 4x4             | 0.8352821639                 | 1.286685624                           |
| 4x8             | 0.8738464229                 | 1.480190604                           |
| 8x4             | 0.7067458174                 | 1.25063736                            |
| 8x8             | 0.7942606664                 | 1.424098315                           |

Table 1: Data from Figures 2,3,4 demonstrating baseline performance vs accelerated performance.

Future work includes fully integrating WMMA-based convolution into the multi-image inference pipeline.

## IV. ARTIFACTS EVALUATION

To reproduce our results, please follow these steps:

GitHub:  
<https://github.com/gupann/vortex-accelerating-cnn>

### A. BUILD VORTEX

```
mkdir build && cd build
./configure
make
```

### B. RUN CNN INFERENCE

```
make -C demo/CNN
make -C demo/CNN run-simx
```

### C. SWEEP PERFORMANCE

```
./ci/blackbox.sh --driver=simx --app=CNN \
--cores=4 --warps=4 --threads=4 --args="-n64"
```

### D. RUN REGRESSION KERNELS

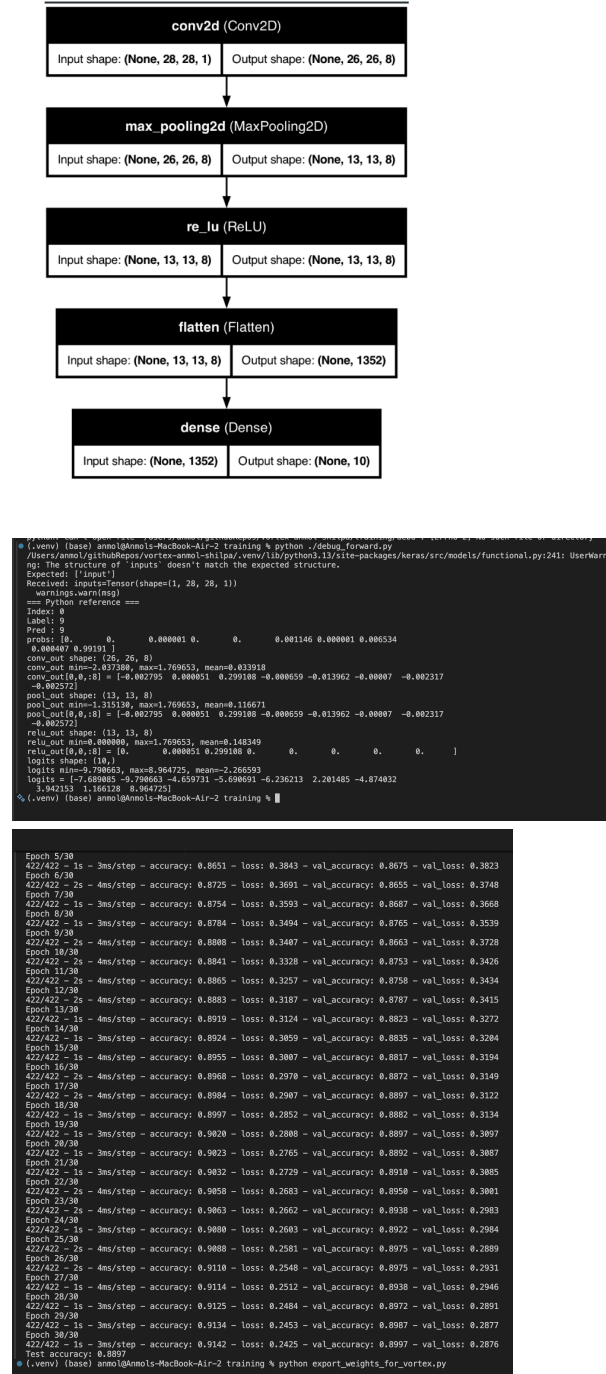
```
./ci/blackbox.sh --driver=simx --app=cnn_conv
./ci/blackbox.sh --driver=simx --app=cnn_pool
./ci/blackbox.sh --driver=simx --app=cnn_relu
```

### E. PYTHON REFERENCE

```
cd training
python cnn.py
python debug_forward.py
```

## V. APPENDIX

### A. TRAINING MODEL WITH PYTHON & KERAS



TERMINAL

(.venv) (base) annol@Annols-MacBook-Air-2 training % python cnn.py

/Users/annol/github/repos/vortex-annol-shilpa/.venv/lib/python3.11/site-packages/keras/src/layers/convolutional/base\_conv.py:113: UserWarning: Do not pass an 'input\_shape' / 'input\_dim' argument to a layer. Go using Sequential models, prefer using an 'Input(shape)' object as the first layer in the model instead.

super().\_\_init\_\_(activity\_regularizer=activity\_regularizer, \*\*kwargs)

Model: "sequential"

| Layer (type)                 | Output Shape      | Param # |
|------------------------------|-------------------|---------|
| conv2d (Conv2D)              | (None, 26, 26, 8) | 64      |
| max_pooling2d (MaxPooling2D) | (None, 13, 13, 8) | 0       |
| re_lu (ReLU)                 | (None, 13, 13, 8) | 0       |
| flatten (Flatten)            | (None, 1352)      | 0       |
| dense (Dense)                | (None, 10)        | 13,530  |

Total params: 13,618 (53.16 KB)  
Trainable params: 13,618 (53.16 KB)  
Non-trainable params: 0 (0.00 B)

Epoch 1/30  
422/422 - 2s - 4ms/step - accuracy: 0.7666 - loss: 0.6823 - val\_accuracy: 0.8332 - val\_loss: 0.4776  
Epoch 2/30  
422/422 - 2s - 5ms/step - accuracy: 0.8388 - loss: 0.4631 - val\_accuracy: 0.8410 - val\_loss: 0.4419  
Epoch 3/30  
422/422 - 2s - 4ms/step - accuracy: 0.8524 - loss: 0.4253 - val\_accuracy: 0.8553 - val\_loss: 0.4132  
Epoch 4/30  
422/422 - 1s - 3ms/step - accuracy: 0.8617 - loss: 0.3991 - val\_accuracy: 0.8613 - val\_loss: 0.3927  
Epoch 5/30  
422/422 - 1s - 3ms/step - accuracy: 0.8651 - loss: 0.3843 - val\_accuracy: 0.8675 - val\_loss: 0.3823  
Epoch 6/30  
422/422 - 2s - 4ms/step - accuracy: 0.8725 - loss: 0.3691 - val\_accuracy: 0.8655 - val\_loss: 0.3748  
Epoch 7/30  
422/422 - 1s - 3ms/step - accuracy: 0.8754 - loss: 0.3593 - val\_accuracy: 0.8687 - val\_loss: 0.3668  
Epoch 8/30  
422/422 - 1s - 3ms/step - accuracy: 0.8784 - loss: 0.3494 - val\_accuracy: 0.8765 - val\_loss: 0.3539  
Epoch 9/30  
422/422 - 2s - 4ms/step - accuracy: 0.8808 - loss: 0.3487 - val\_accuracy: 0.8663 - val\_loss: 0.3728  
Epoch 10/30  
422/422 - 2s - 4ms/step - accuracy: 0.8865 - loss: 0.3257 - val\_accuracy: 0.8758 - val\_loss: 0.3426  
Epoch 11/30  
422/422 - 2s - 4ms/step - accuracy: 0.8865 - loss: 0.3257 - val\_accuracy: 0.8758 - val\_loss: 0.3426  
Epoch 12/30  
422/422 - 2s - 4ms/step - accuracy: 0.8883 - loss: 0.3187 - val\_accuracy: 0.8787 - val\_loss: 0.3415  
Epoch 13/30  
422/422 - 1s - 4ms/step - accuracy: 0.8919 - loss: 0.3124 - val\_accuracy: 0.8823 - val\_loss: 0.3272  
Epoch 14/30  
422/422 - 1s - 3ms/step - accuracy: 0.8924 - loss: 0.3059 - val\_accuracy: 0.8835 - val\_loss: 0.3284  
Epoch 15/30  
422/422 - 1s - 4ms/step - accuracy: 0.8955 - loss: 0.3007 - val\_accuracy: 0.8817 - val\_loss: 0.3194  
Epoch 16/30  
422/422 - 2s - 4ms/step - accuracy: 0.8968 - loss: 0.2970 - val\_accuracy: 0.8872 - val\_loss: 0.3149

### B. VORTEX REGRESSION PERFORMANCE

```
root@30145d770925:/vortex/build# make -s
root@30145d770925:/vortex/build# ./ci/blackbox.sh --driver=simx --cores=4 --warps=4 --threads=4 --app=cnn_conv
Running: CONFIGS="--DNUM_CORES=4 --DNUM_WARPS=4 --DNUM_THREADS=4" make -C ./ci../runtime/simx > /dev/null
Running: make -C "/ci../tests/regression/cnn_conv" run-simx
make: Entering directory '/vortex/build/tests/regression/cnn_conv'
LD_LIBRARY_PATH=/vortex/build/runtime: VORTEX_DRIVER=simx ./cnn_conv
PERF: core0: instrs=245002, cycles=331073, IPC=0.740024
PERF: core1: instrs=248386, cycles=340011, IPC=0.730523
PERF: core2: instrs=248386, cycles=343882, IPC=0.722300
PERF: core3: instrs=244630, cycles=350084, IPC=0.697162
PERF: instrs=986404, cycles=350884, IPC=2.81197
PASSED
make: Leaving directory '/vortex/build/tests/regression/cnn_conv'
root@30145d770925:/vortex/build# ./ci/blackbox.sh --driver=simx --cores=4 --warps=4 --threads=4 --app=cnn_pool
Running: CONFIGS="--DNUM_CORES=4 --DNUM_WARPS=4 --DNUM_THREADS=4" make -C ./ci../runtime/simx > /dev/null
Running: make -C "/ci../tests/regression/cnn_pool" run-simx
make: Entering directory '/vortex/build/tests/regression/cnn_pool'
LD_LIBRARY_PATH=/vortex/build/runtime: VORTEX_DRIVER=simx ./cnn_pool
Size of output: 1024
PERF: core0: instrs=15828, cycles=49921, IPC=0.317061
PERF: core1: instrs=15824, cycles=50156, IPC=0.315496
PERF: core2: instrs=15829, cycles=49798, IPC=0.317864
PERF: core3: instrs=15817, cycles=49947, IPC=0.316676
PERF: instrs=63298, cycles=50156, IPC=1.262022
POOL PASSED
make: Leaving directory '/vortex/build/tests/regression/cnn_pool'
root@30145d770925:/vortex/build# ./ci/blackbox.sh --driver=simx --cores=4 --warps=4 --threads=4 --app=cnn_relu
Running: CONFIGS="--DNUM_CORES=4 --DNUM_WARPS=4 --DNUM_THREADS=4" make -C ./ci../runtime/simx > /dev/null
Running: make -C "/ci../tests/regression/cnn_relu" run-simx
make: Entering directory '/vortex/build/tests/regression/cnn_relu'
LD_LIBRARY_PATH=/vortex/build/runtime: VORTEX_DRIVER=simx ./cnn_relu
PERF: core0: instrs=13031, cycles=64007, IPC=0.203587
PERF: core1: instrs=13010, cycles=62252, IPC=0.208989
PERF: core2: instrs=13004, cycles=62187, IPC=0.209111
PERF: core3: instrs=13020, cycles=62348, IPC=0.208828
PERF: instrs=52065, cycles=64007, IPC=0.813427
ReLU PASSED
make: Leaving directory '/vortex/build/tests/regression/cnn_relu'
root@30145d770925:/vortex/build#
```

### C. RELU IPC PERFORMANCE IMPROVEMENT

```
root@30145d770925:/vortex/build# ./ci/blackbox.sh --driver=simx --app=CN --cores=4 --warps=4 --threads=4 --args="n04"
Summary -----
Tested 180 images
Correct: 89
Accuracy: 89%
-----
PERF: core0: instrs=16167, cycles=68011, IPC=0.237712
PERF: core1: instrs=16853, cycles=67531, IPC=0.237712
PERF: core2: instrs=16880, cycles=70013, IPC=0.226672
PERF: core3: instrs=16130, cycles=67028, IPC=0.237457
PERF: instrs=66430, cycles=70013, IPC=1.020258
make: Leaving directory '/vortex/build/demo/CN'
root@30145d770925:/vortex/build#

root@30145d770925:/vortex/build# ./ci/blackbox.sh --driver=simx --app=CN --cores=4 --warps=4 --threads=4 --args="n04"
Summary -----
Tested 180 images
Correct: 73
Accuracy: 73%
-----
PERF: core0: instrs=17307, cycles=51283, IPC=0.339040
PERF: core1: instrs=17866, cycles=50064, IPC=0.334864
PERF: core2: instrs=17864, cycles=49870, IPC=0.337453
PERF: core3: instrs=17846, cycles=50007, IPC=0.340872
PERF: instrs=68859, cycles=51283, IPC=1.336054
make: Leaving directory '/vortex/build/demo/CN'
root@30145d770925:/vortex/build#
```

## D. VORTEX ACCURACY: TERMINAL OUTPUT

```
model leaving directory /vortex/build/demo/CNN
root@30145d770925:/vortex/build demo-CNN run-simx
make: Entering directory '/vortex/build/demo/CNN'
LD_LIBRARY_PATH=/vortex/build/runtime: VORTEX_DRIVER=simx ./CNN -n64
----- Vortex Fashion-MNIST CNN -----
Opened Vortex device.
Loading weights...
img 0 gt=9 (Ankle boot) | pred=9 (Ankle boot) | CORRECT
----- DEBUG (img 0) -----
Conv output: 8x26x26
conv_out shape: 8x26x26
conv_out min=-0.83739, max=1.76965, mean=0.8393184
conv_out [y=0, x=0, c=0..7] = -0.00279455 5.89787e-05 0.299188 -0.000659352 -0.0139618 -6.963e-05 -0.00231739 -0.00257161
Pool output: 8x13x13
pool_out shape: 8x13x13
pool_out min=-1.3513, max=1.76965, mean=0.116671
pool_out [y=0, x=0, c=0..7] = -0.00279455 5.89787e-05 0.299188 -0.000659352 -0.0139618 -6.963e-05 -0.00231739 -0.00257161
relu_out shape: 8x13x13
relu_out min=0, max=1.76965, mean=0.148340
relu_out [y=0, x=0, c=0..7] = 0 5.89787e-05 0.299188 0 0 0 0 0
logits = -7.68988 -9.79867 -4.65973 -5.69069 -6.23622 2.20149 -4.87483 3.94215 1.16613 0.96472
Ground Truth: 9 (Ankle boot)
Prediction: 9 (Ankle boot)
Result: CORRECT
Prediction: 9
Probabilities: 5.00518e-08 7.89754e-09 1.20073e-06 4.2826e-07 2.48193e-07 0.00114614 9.6912e-07 0.00653429 0.000406993 0.99191
----- END DEBUG -----
img 1 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 2 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 3 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 4 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 5 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 6 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 7 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 8 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 9 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 10 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 11 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 12 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 13 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 14 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 15 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 16 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 17 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 18 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 19 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 20 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 21 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 22 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 23 gt=9 (Ankle boot) | pred=9 (Ankle boot) | CORRECT
img 24 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 25 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 26 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 27 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 28 gt=9 (Ankle boot) | pred=9 (Ankle boot) | CORRECT
img 29 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 30 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 31 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 32 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 33 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 34 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 35 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 36 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 37 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 38 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 39 gt=9 (Ankle boot) | pred=9 (Ankle boot) | CORRECT
img 40 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 41 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 42 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 43 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 44 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 45 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
```

```
img 45 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 46 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 47 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 48 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 49 gt=2 (Pullover) | pred=6 (Shirt) | WRONG
img 50 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 51 gt=4 (Coat) | pred=6 (Shirt) | WRONG
img 52 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 53 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 54 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 55 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 56 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 57 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 58 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 59 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 60 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 61 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 62 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 63 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 64 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 65 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 66 gt=2 (Pullover) | pred=3 (Dress) | WRONG
img 67 gt=3 (Dress) | pred=4 (Coat) | WRONG
img 68 gt=9 (Ankle boot) | pred=7 (Sneaker) | WRONG
img 69 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 70 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 71 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 72 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 73 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 74 gt=2 (Pullover) | pred=4 (Coat) | WRONG
img 75 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 76 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 77 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 78 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 79 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 80 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 81 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 82 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 83 gt=9 (Ankle boot) | pred=9 (Ankle boot) | CORRECT
img 84 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 85 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 86 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 87 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
img 88 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 89 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 90 gt=5 (Sandal) | pred=5 (Sandal) | CORRECT
img 91 gt=3 (Dress) | pred=3 (Dress) | CORRECT
img 92 gt=6 (Shirt) | pred=6 (Shirt) | CORRECT
img 93 gt=7 (Sneaker) | pred=7 (Sneaker) | CORRECT
img 94 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 95 gt=8 (Bag) | pred=8 (Bag) | CORRECT
img 96 gt=0 (T-shirt/top) | pred=0 (T-shirt/top) | CORRECT
img 97 gt=1 (Trouser) | pred=1 (Trouser) | CORRECT
img 98 gt=4 (Coat) | pred=4 (Coat) | CORRECT
img 99 gt=2 (Pullover) | pred=2 (Pullover) | CORRECT
```

----- Summary -----

Tested 100 images

Correct: 89

Accuracy: 89%

----- Done -----

PERF: instrs=53308, cycles=50814, IPC=1.049081

make: Leaving directory '/vortex/build/demo/CNN'

root@30145d770925:/vortex/build#