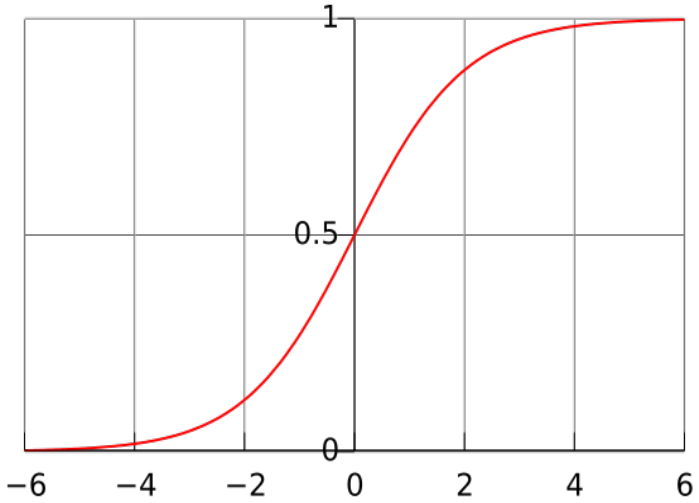
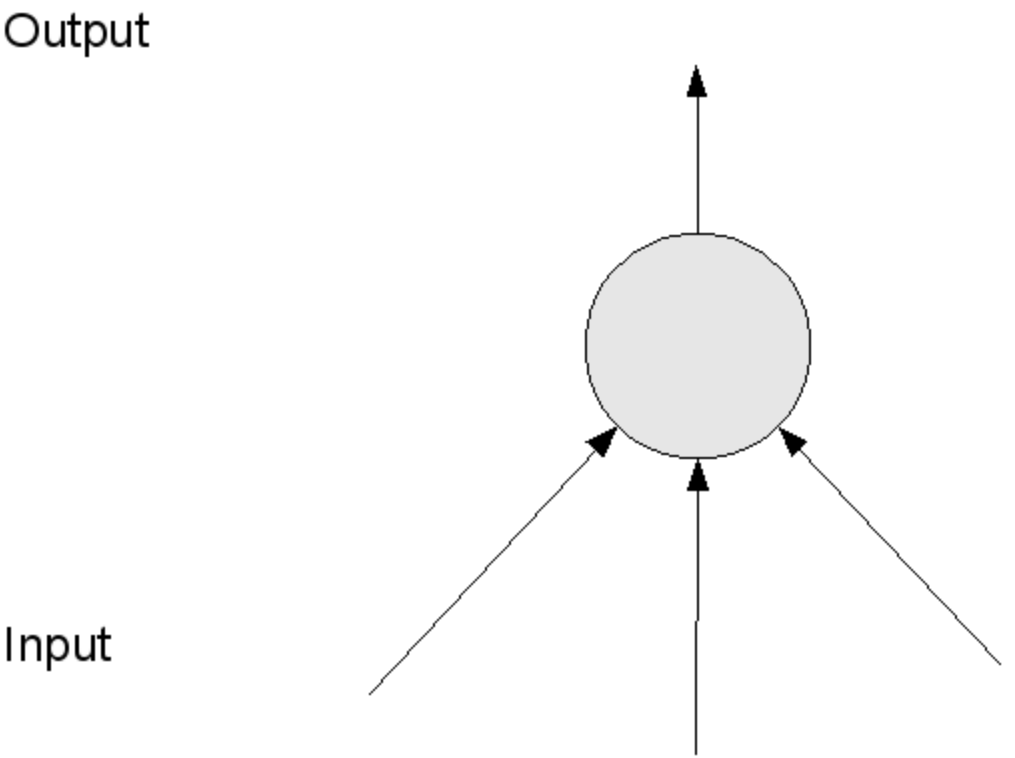
Convolutional Neural Networks in CUDA

A neuron is a function.

f(x) = 1 / (1 + e^-x)

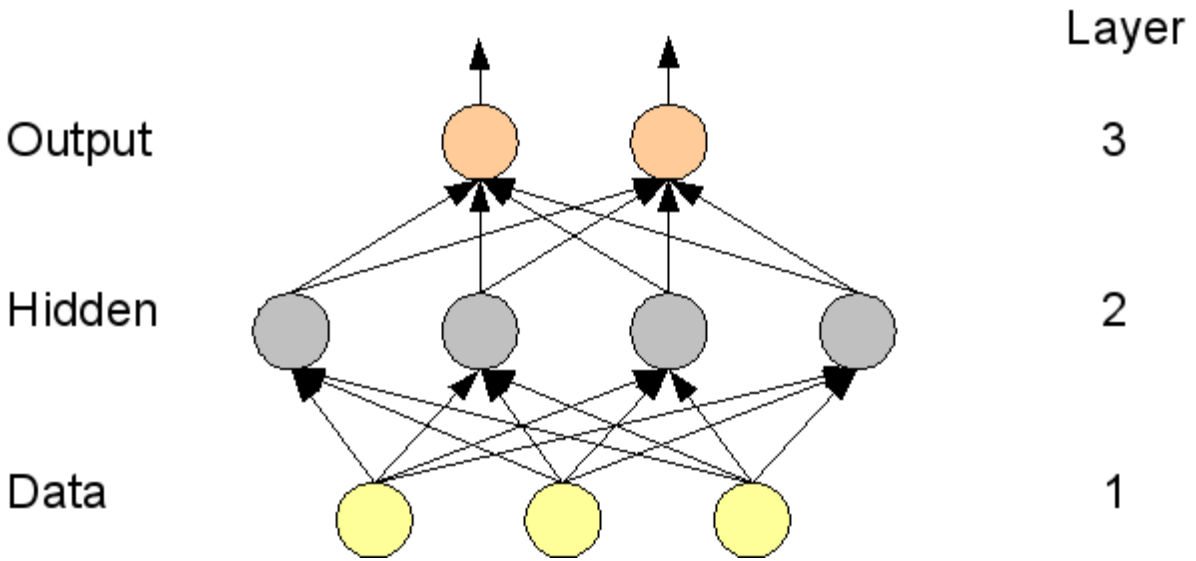


Source: Wikipedia

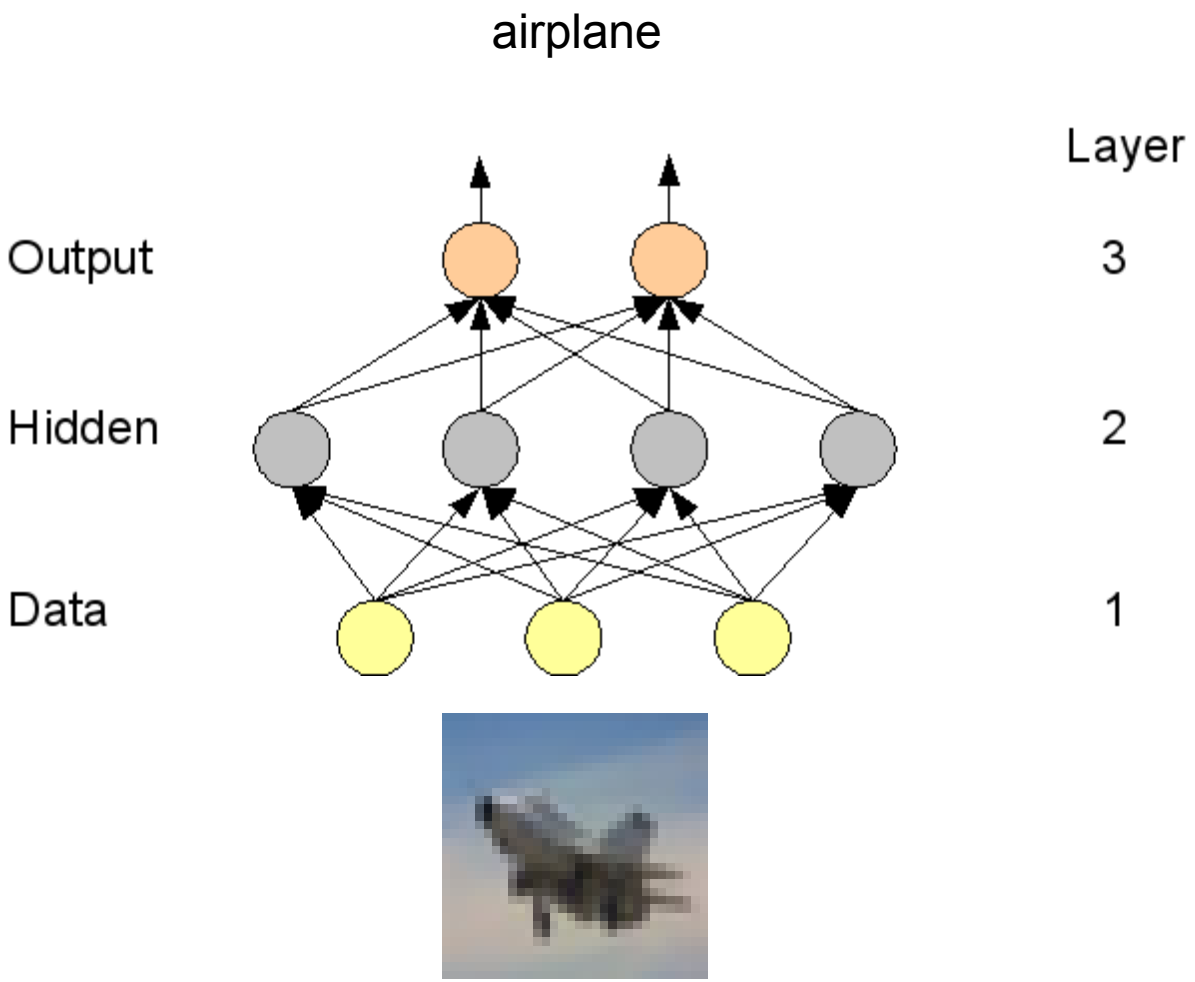


Neural networks

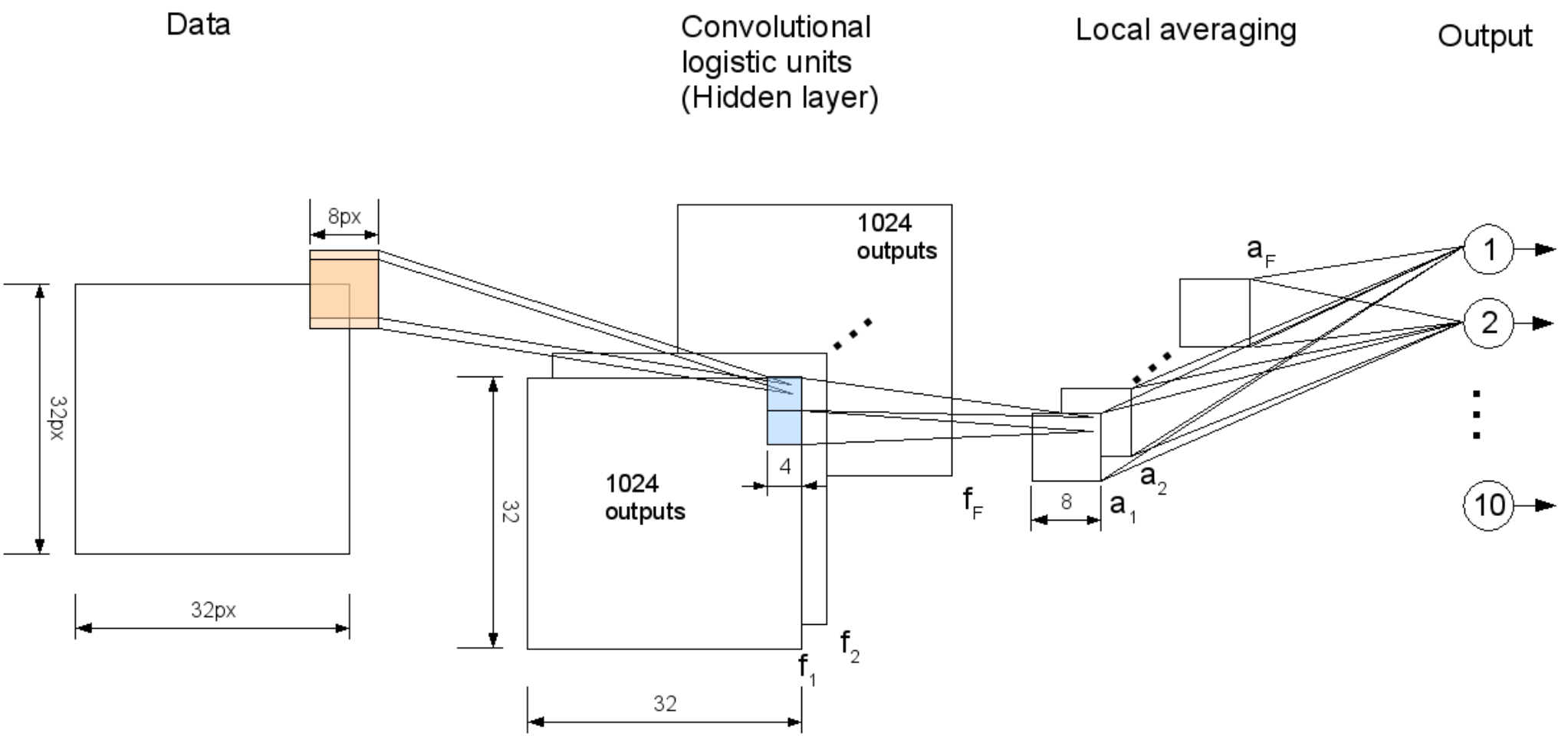
- A neural network connects layers of neurons to compute a more complicated function.



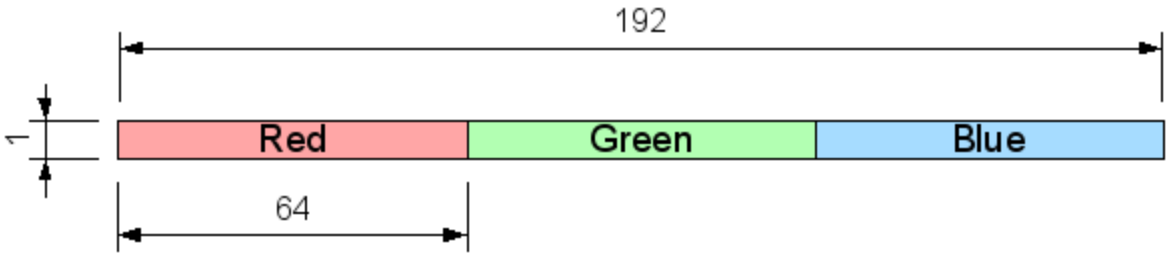
Uses of neural networks



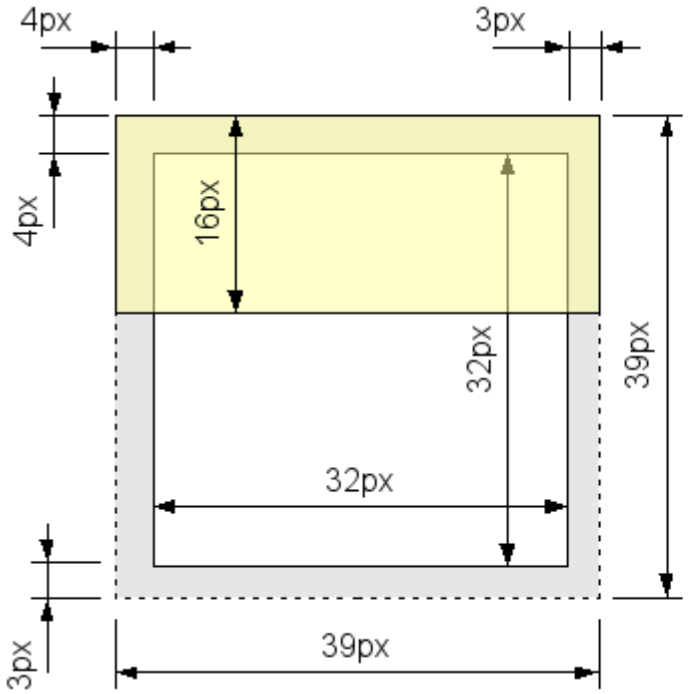
Convolutional neural networks

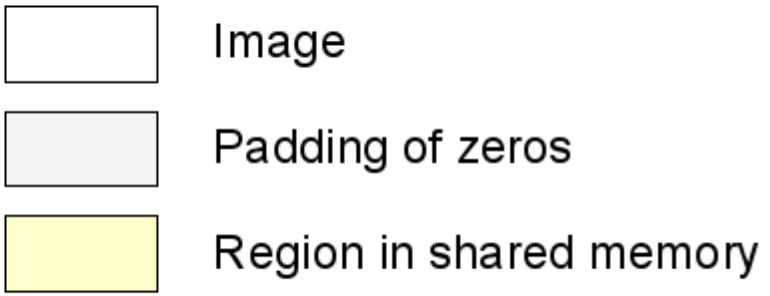


Convolution algorithm

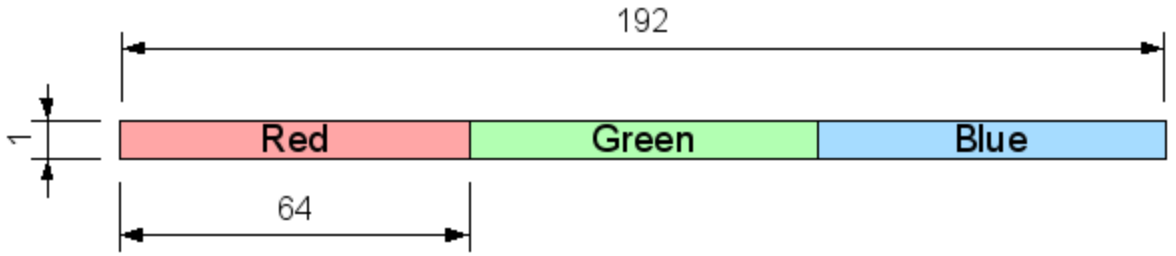


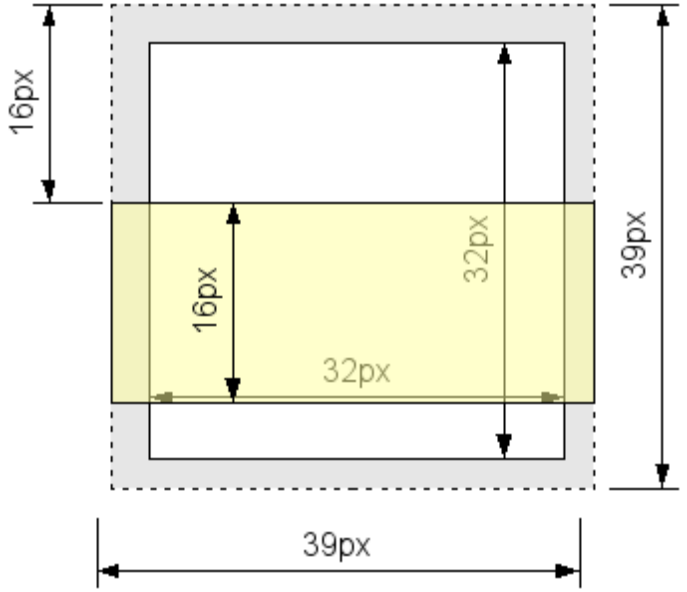
·Block size 8x32.
·Each block convolves one filter with one image.
·Each thread computes 4 outputs.

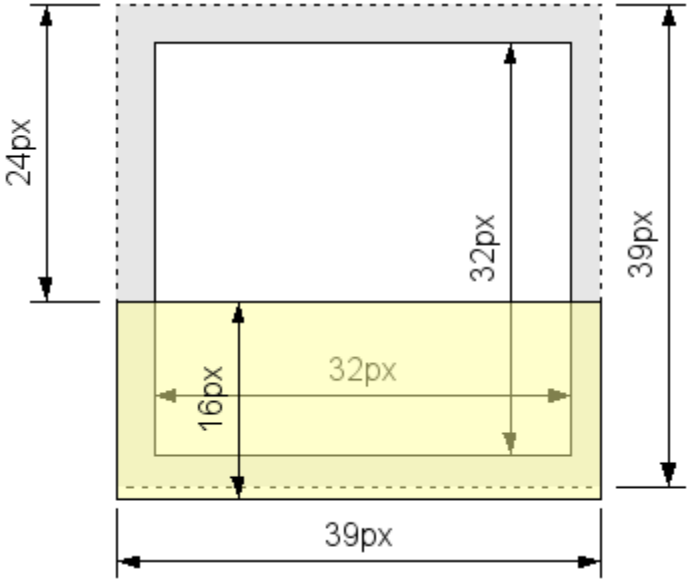




Convolution algorithm

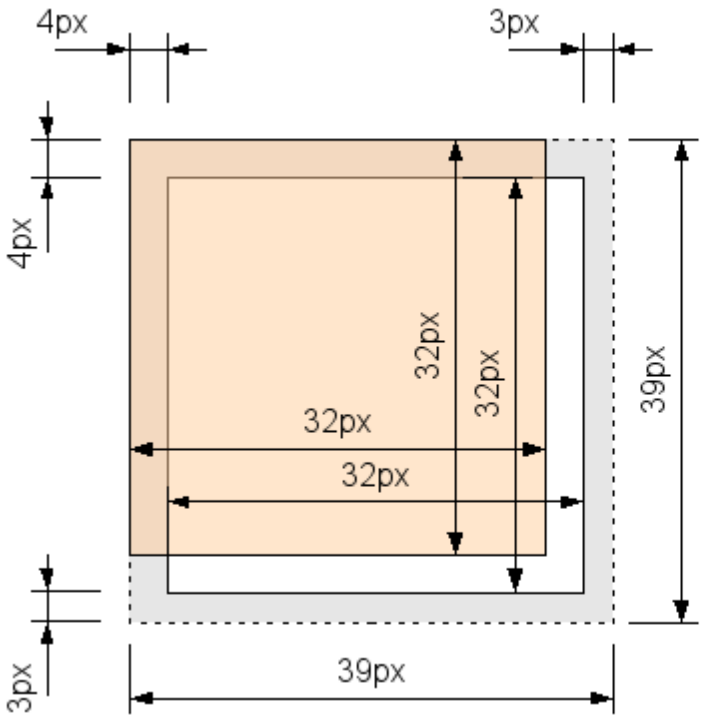


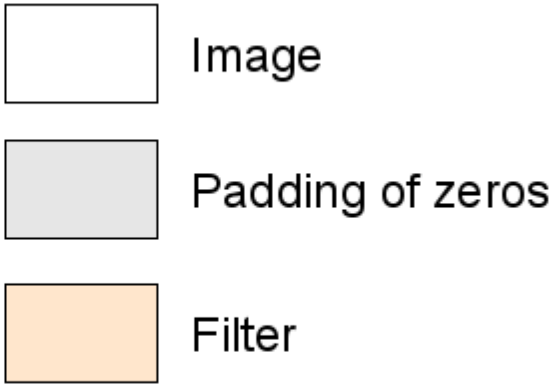




100% occupancy.
125x faster than Core 2 2.4GHz.

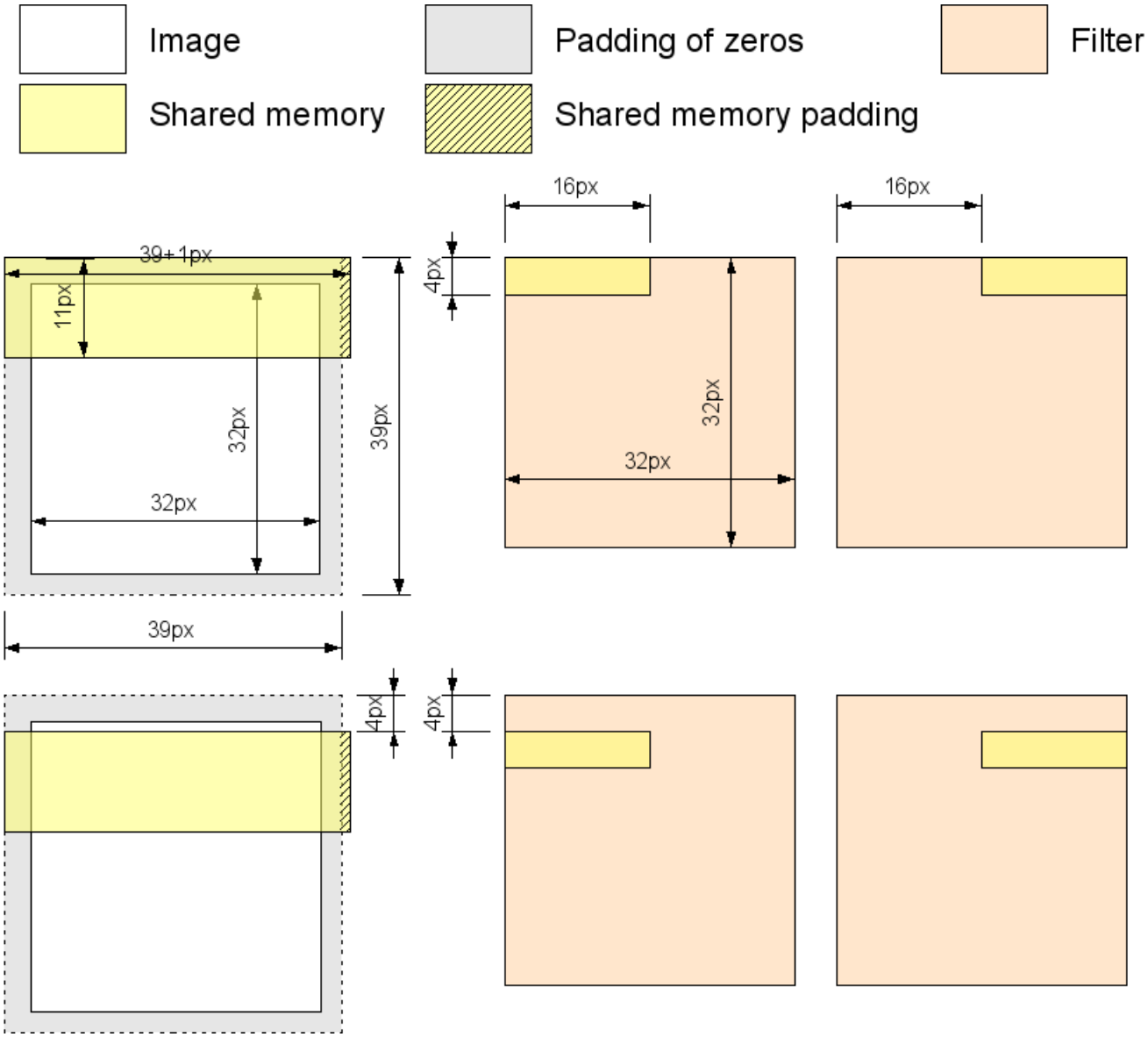
Convolution 2





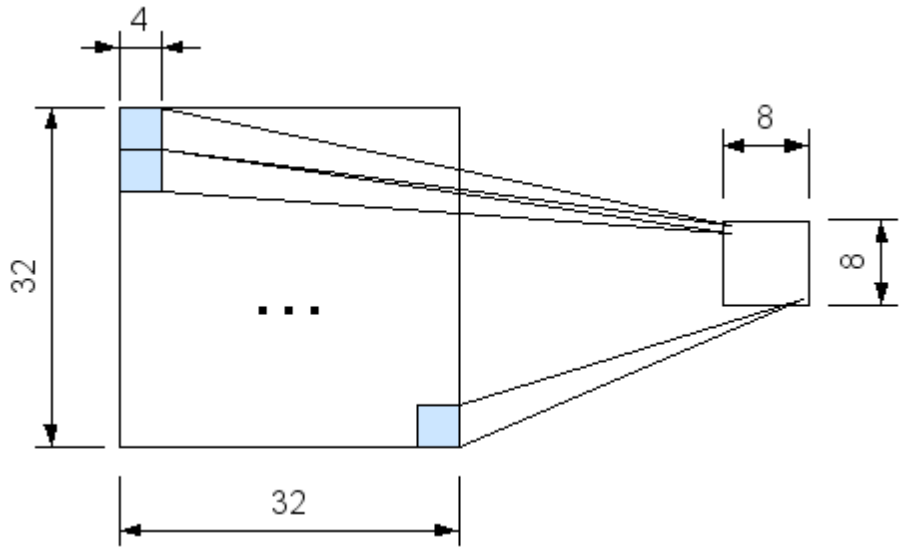
Convolution 2 algorithm

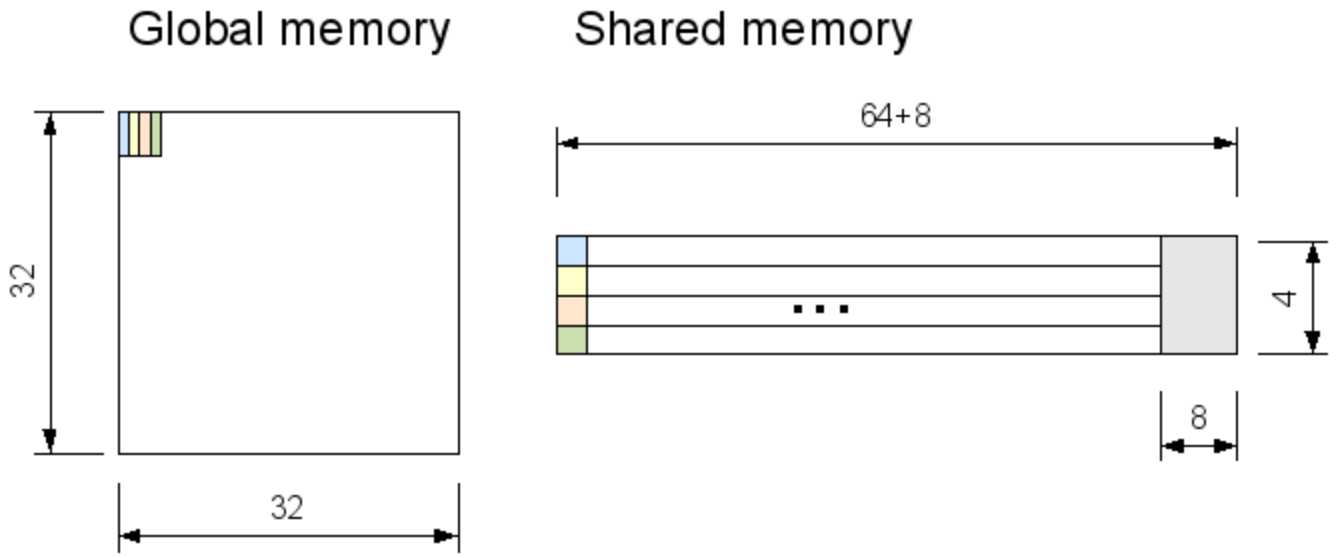
- Block size 8x8x8.
- Each block convolves one image with 16 filters (each thread doing 2 filters).
- 100% occupancy.
- 157x faster than Core 2 2.4GHz.



Subsampling

- Block size 8x32.

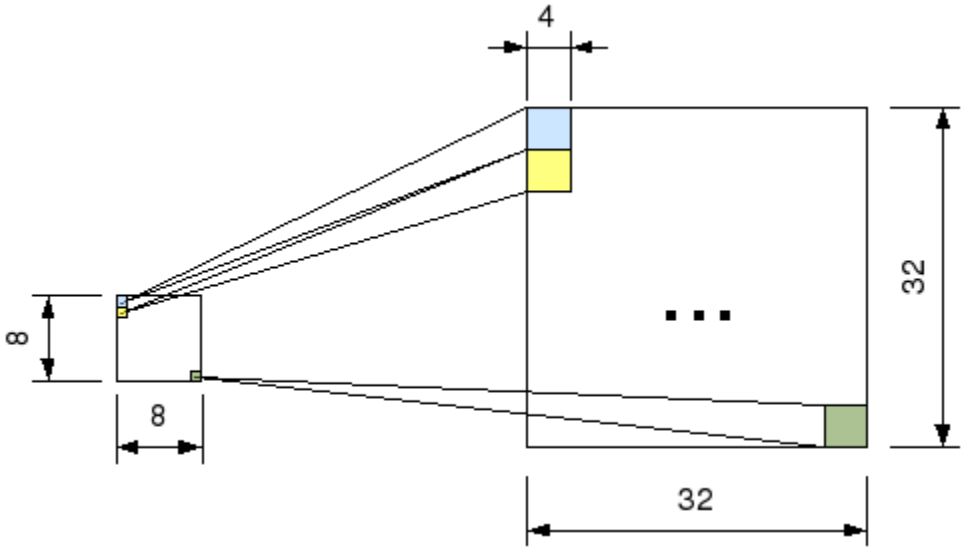




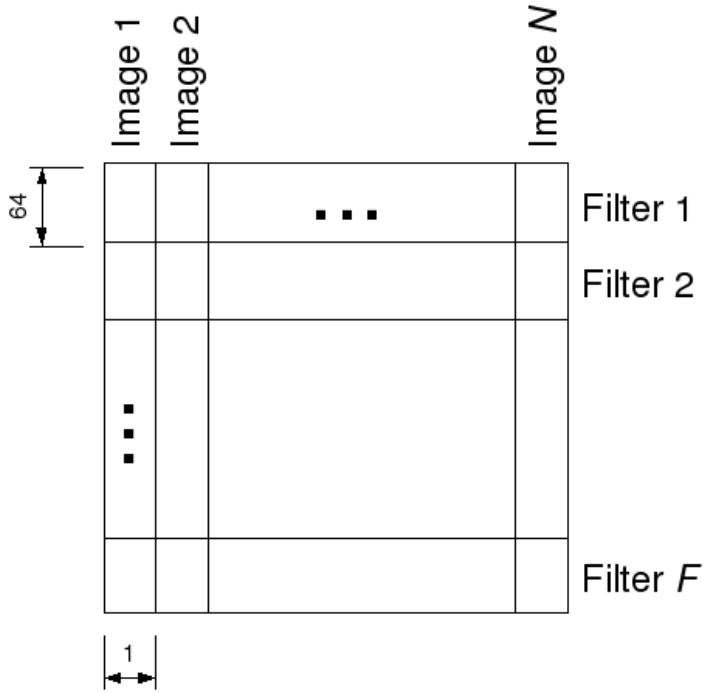
31x faster than CPU.

Supersampling

Goal:

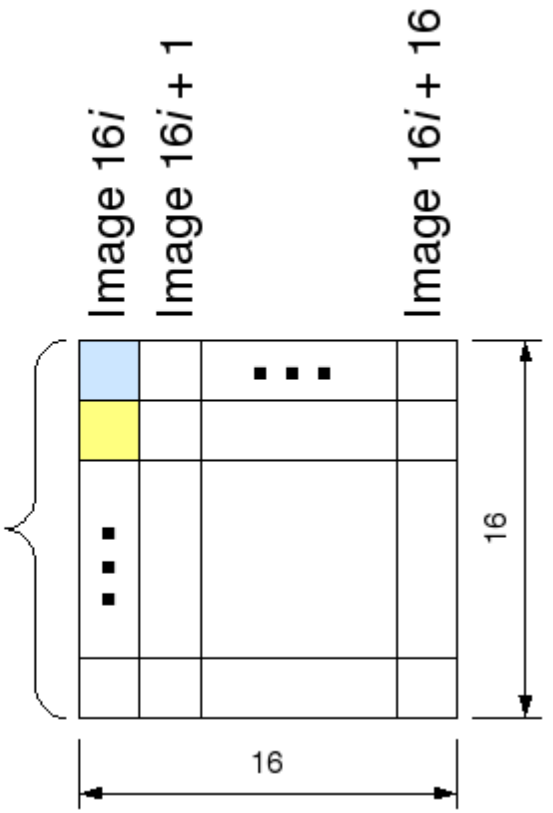


Input layout:



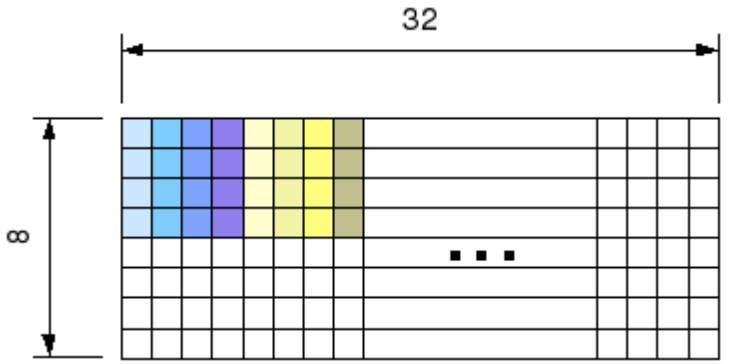
Supersampling algorithm

Shared memory:



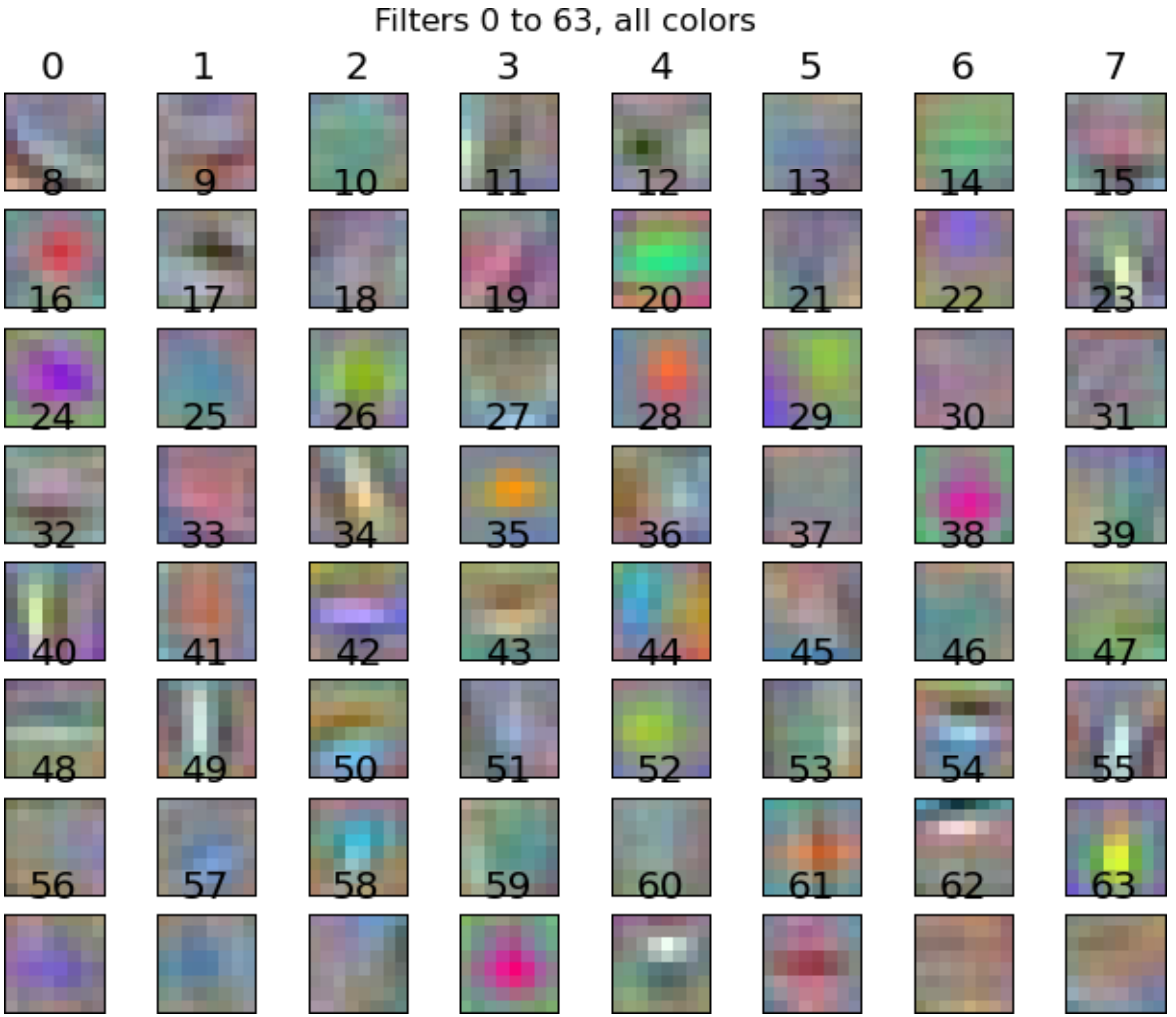
Filter j / 4.
Rows j / 4 + 16(j % 4) to j / 4 + 16(j % 4) + 16 of input matrix.

Output (global memory):



58x faster than CPU.

Results



- ~140x faster than CPU.
- Gets 58% right on data set of 60,000 images in 10 classes (my best result with other methods is 65%).