Pooling Acceleration in the DaVinci Architecture Using Im2col and Col2im Instructions

Caio S. Rohwedder,
João L. P. de Carvalho, José Nelson Amaral, Guido Araújo, Giancarlo Colmenares, Kai-Ting Amy Wang

Overview

- AI Accelerators
  - High demand for CNNs
  - DaVinci
    - Matrix Multiplier
    - Im2col & Col2im

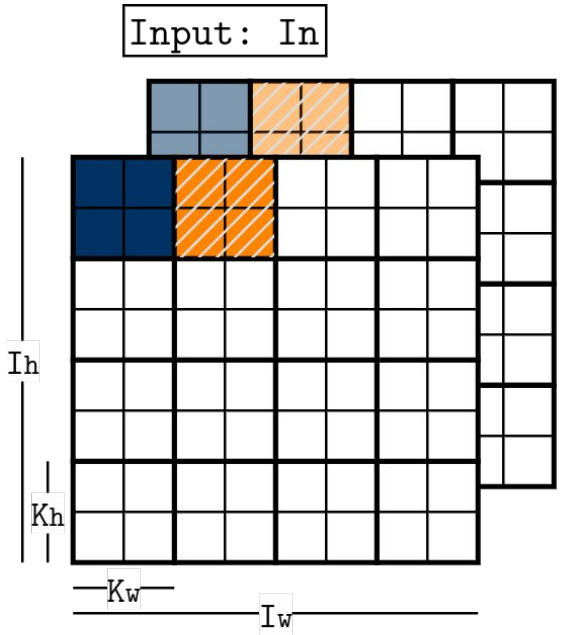
Overview

- AI Accelerators
  - High demand for CNNs
  - DaVinci
    - Matrix Multiplier
    - Im2col & Col2im
- Proposition:
  - Im2col/Col2im Based Pooling Using Specialized DaVinci Instructions

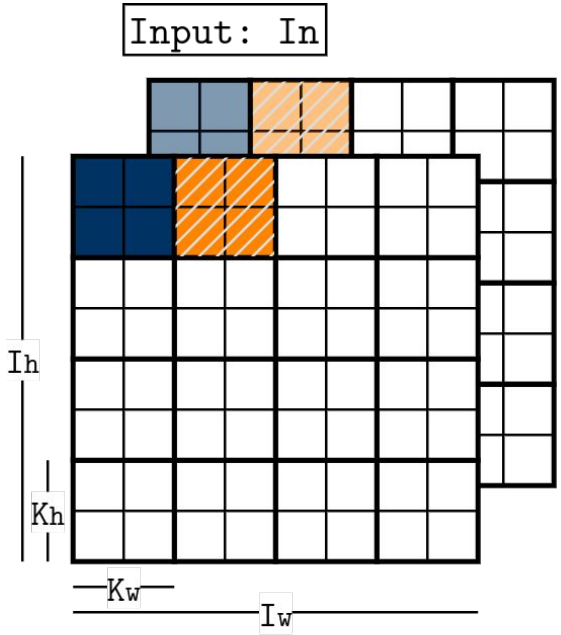
Background

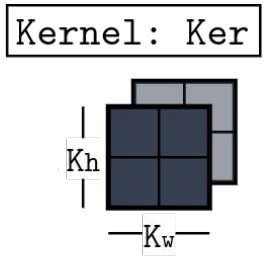
- Convolution & Im2col
- Col2im & Backward Operators
- Pooling Operators

Convolution - Input Image

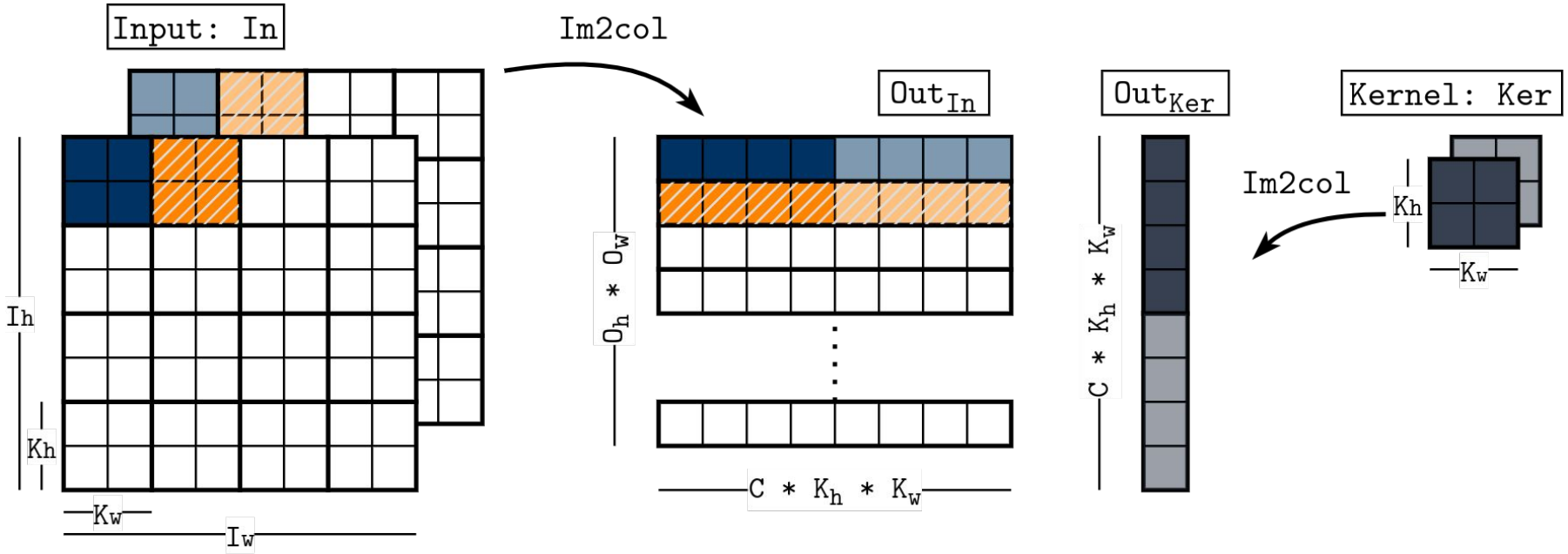


Convolution - Input Kernel



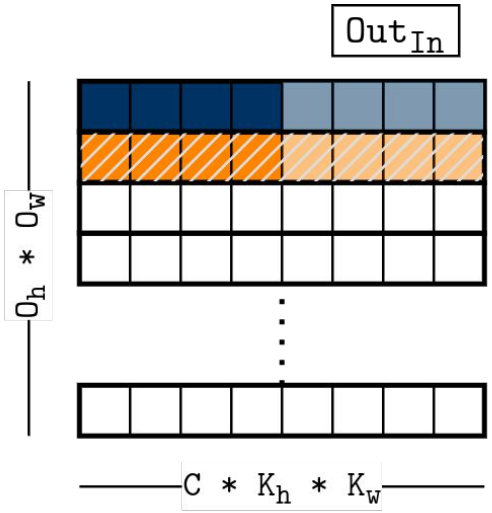


Im2col: Convolution → GEMM



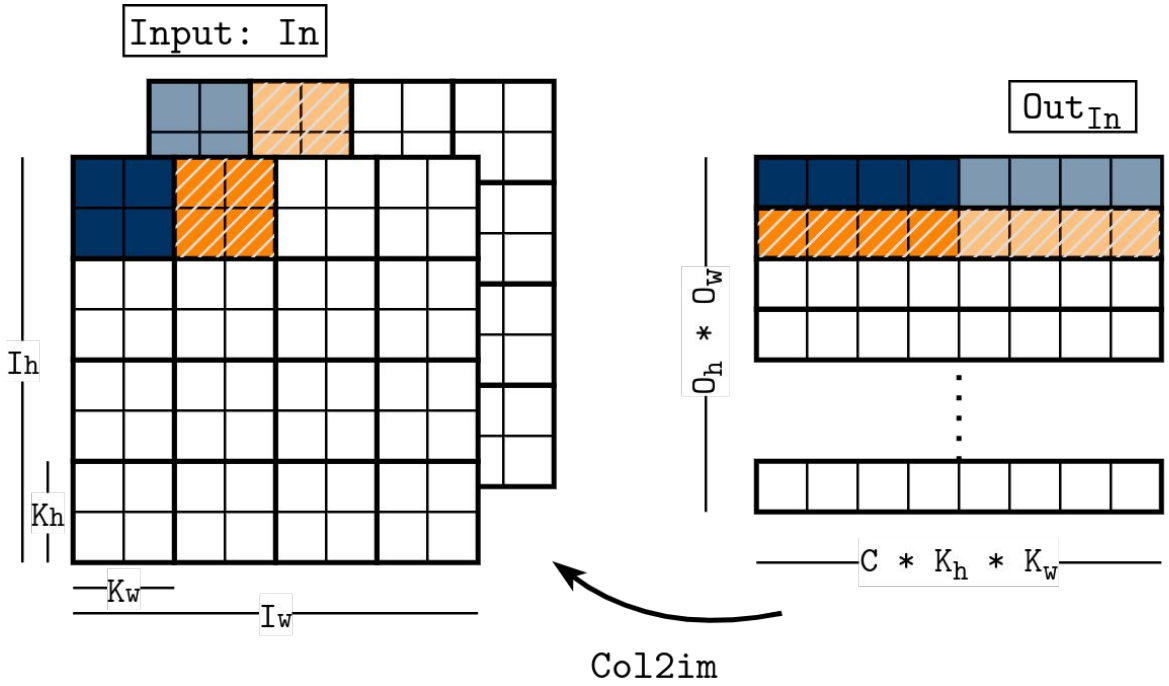
Col2im & Backward Operators

- Propagates Im2col Backward

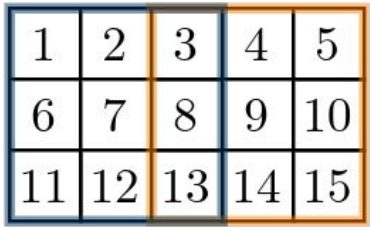


Col2im & Backward Operators

- Propagates Im2col Backward

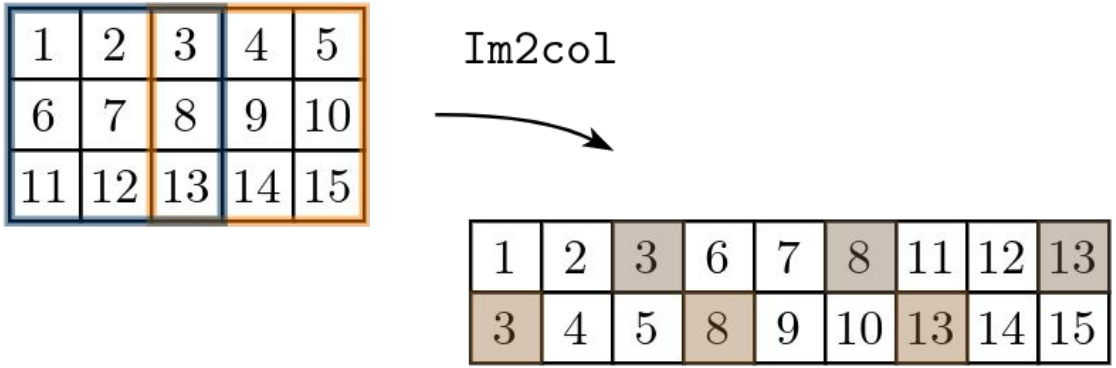


Overlapping Im2col & Col2im



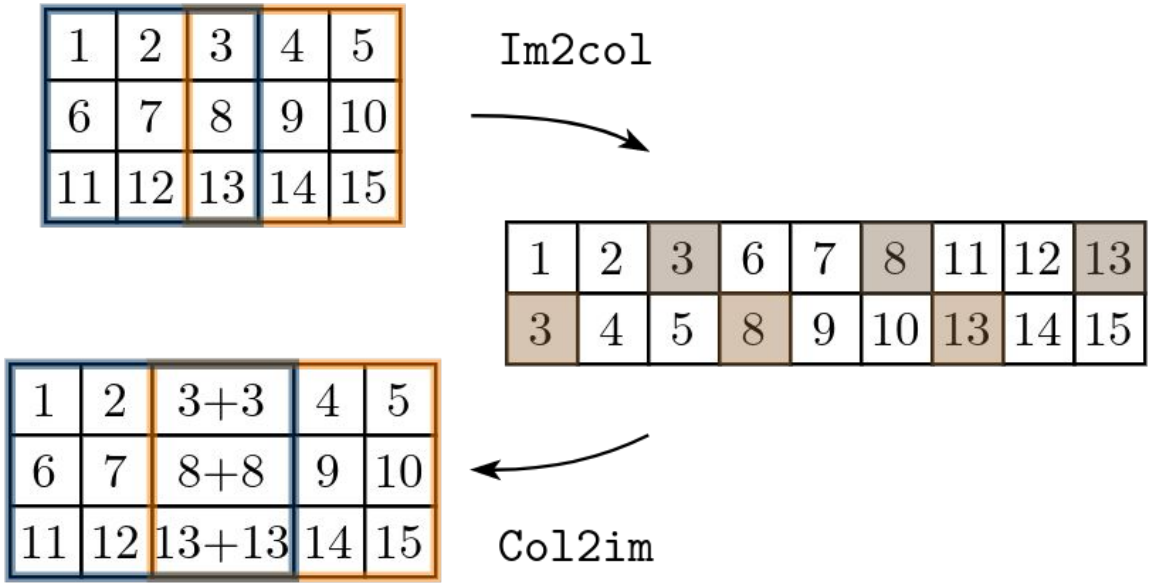
Overlapping Im2col & Col2im

- Im2col: Duplicates

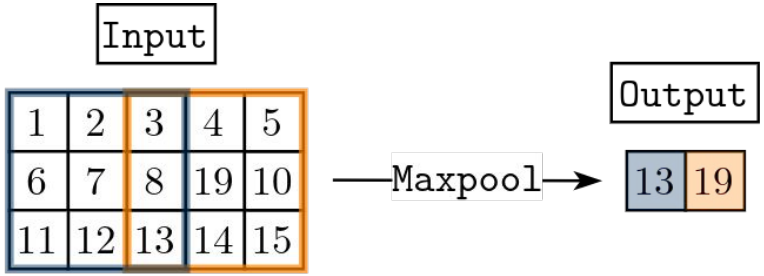


Overlapping Im2col & Col2im

- Im2col: Duplicates
- Col2im: Merges

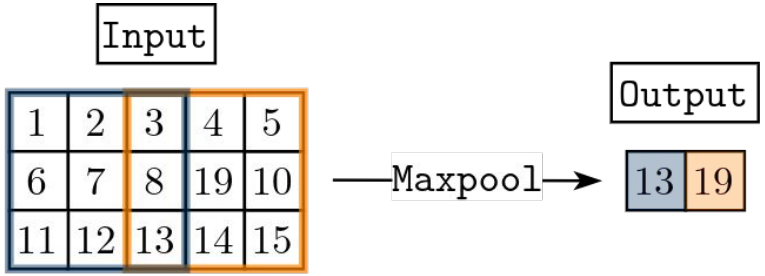


Pooling Operators

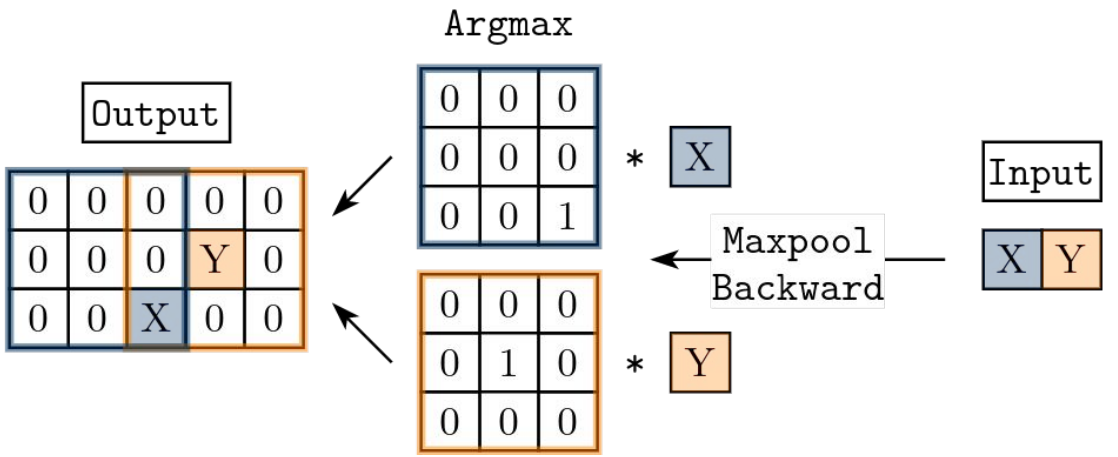


- Maxpool
  - Max Value

Pooling Operators



- Maxpool
  - Max Value



The DaVinci Architecture

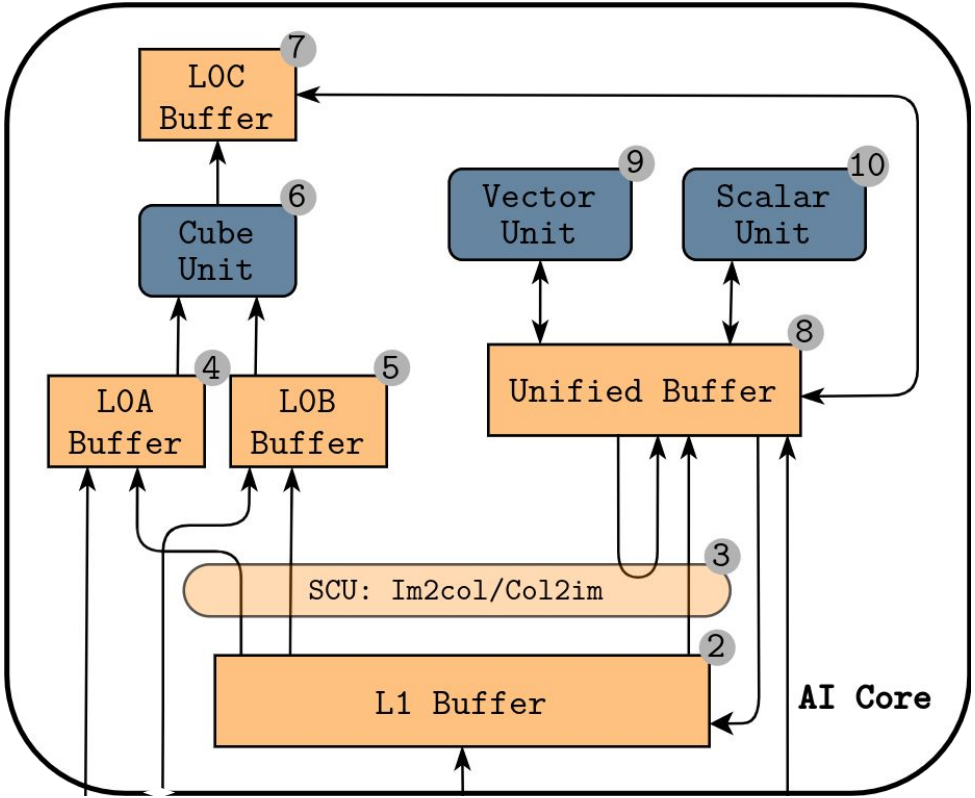
- AI Core
- Fractal Memory Layout
- Im2col & Col2im Instruction

DaVinci - AI Core

- Cube Unit
  - Matrix Multiplication
- Vector Unit
  - Pooling
- Scalar Unit

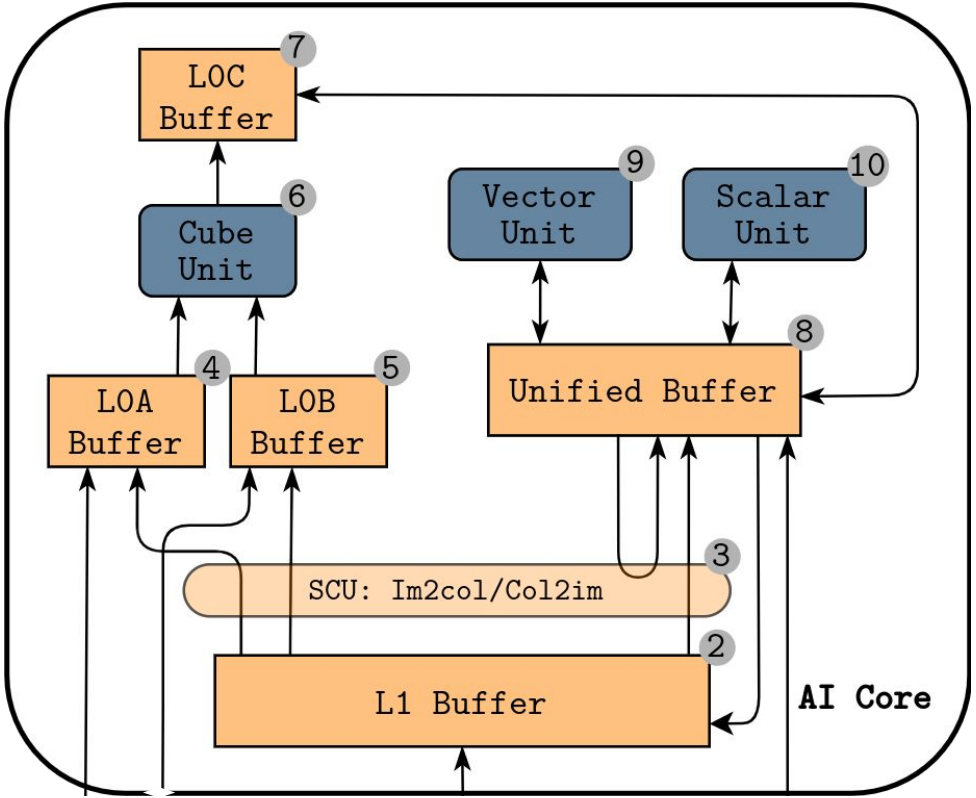
DaVinci - AI Core

- Scratch-Pad Memories



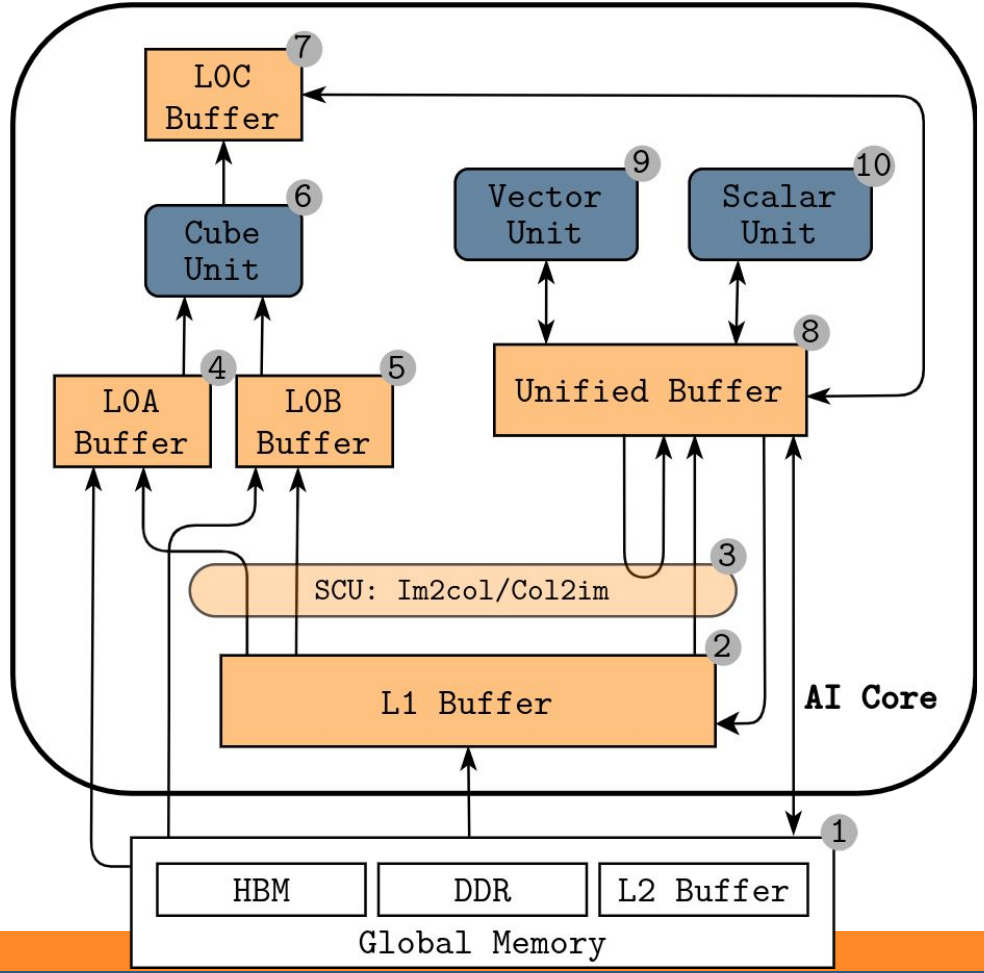
DaVinci - AI Core

- Scratch-Pad Memories
- SCU: Storage Conversion Unit



DaVinci - AI Core

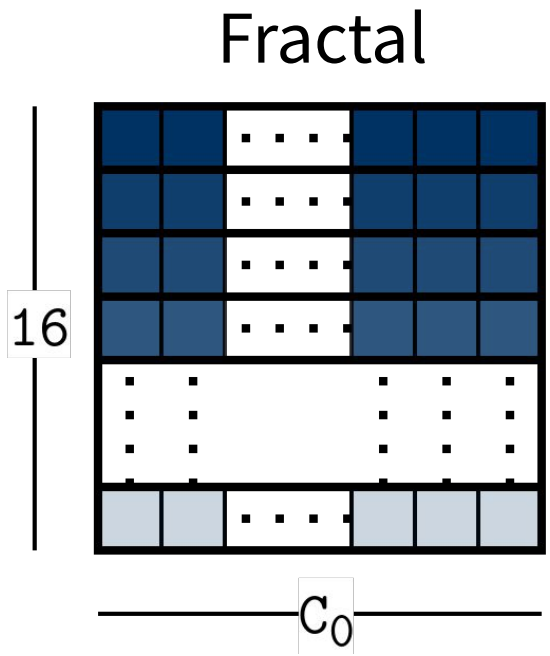
- Global Memory



DaVinci - Fractal Memory Layout

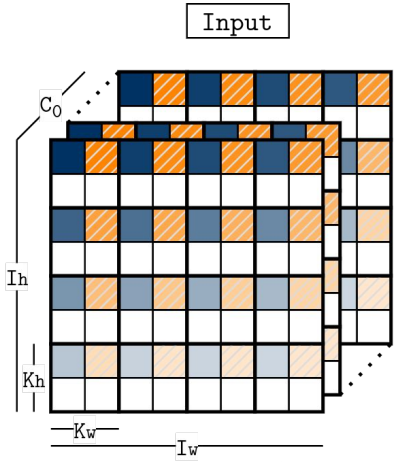
$NCHW \rightarrow NC_1HWC_0$

- $C1 = \lceil C/C0 \rceil$
- $C0 = 16$
  - (Float16)



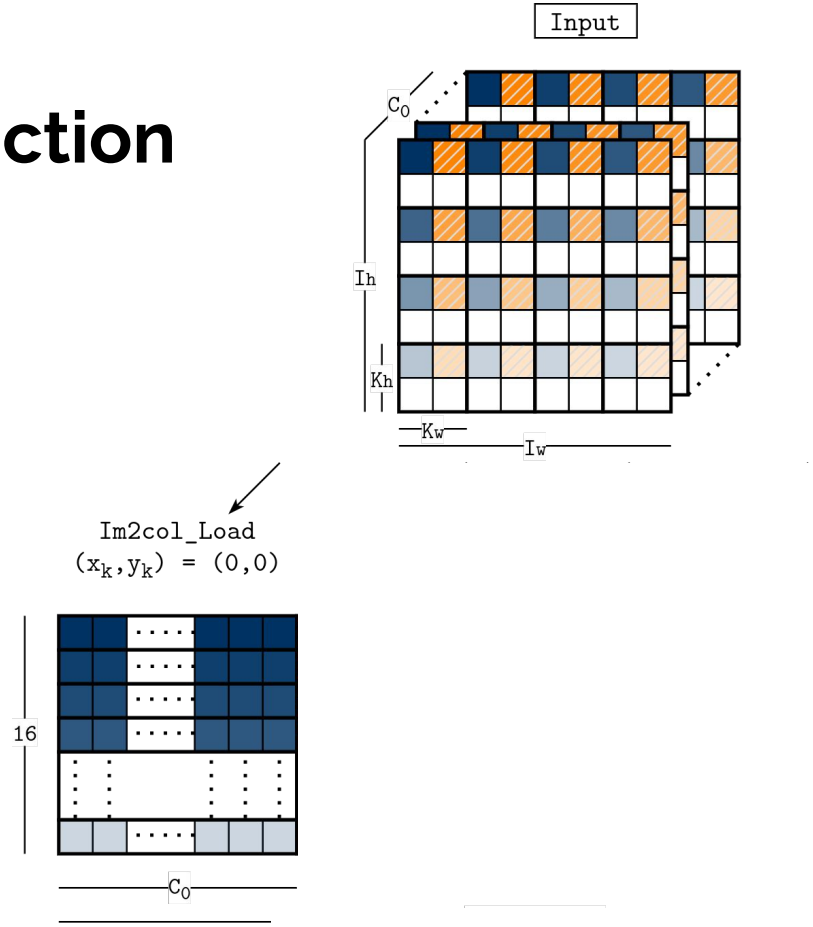
Im2col Instruction

- Loads fractals
  - NC_1HWC_0



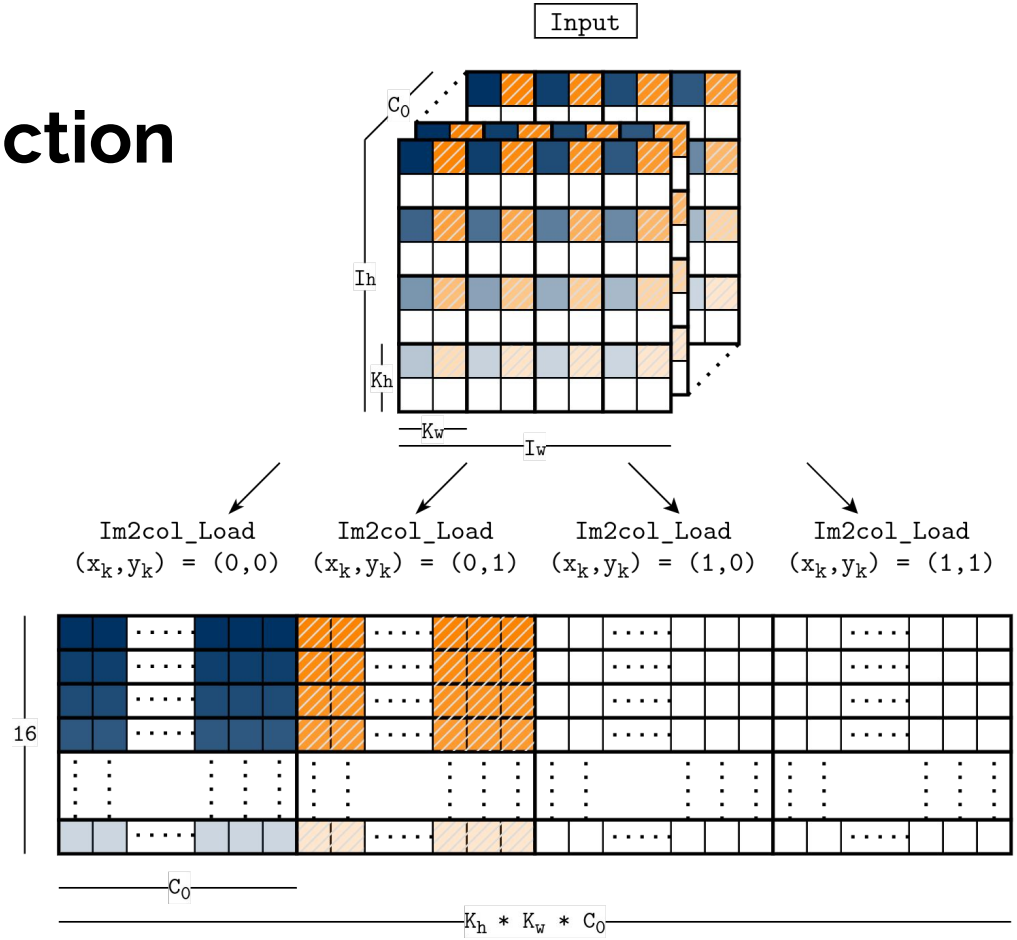
Im2col Instruction

- Loads fractals
  - NC_1HWC_0



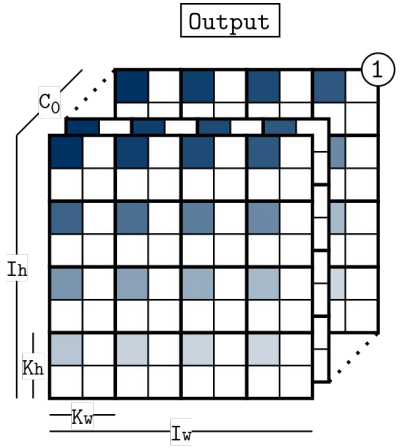
Im2col Instruction

- Loads fractals
  - NC_1HWC_0



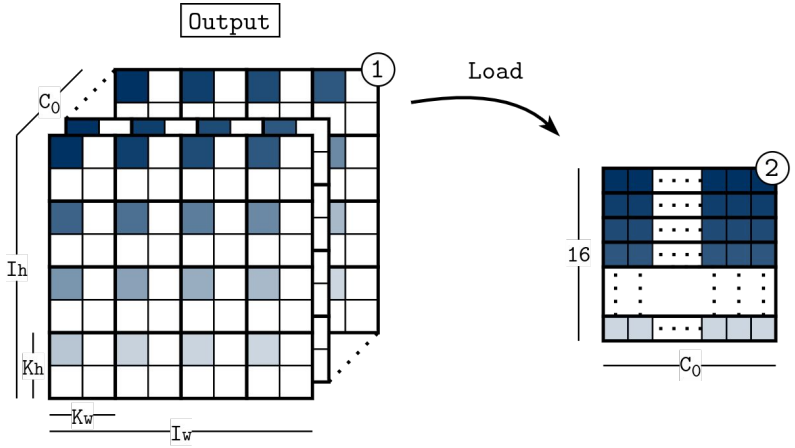
Col2im Instruction

- Inputs are fractals
  - NC_1HWC_0



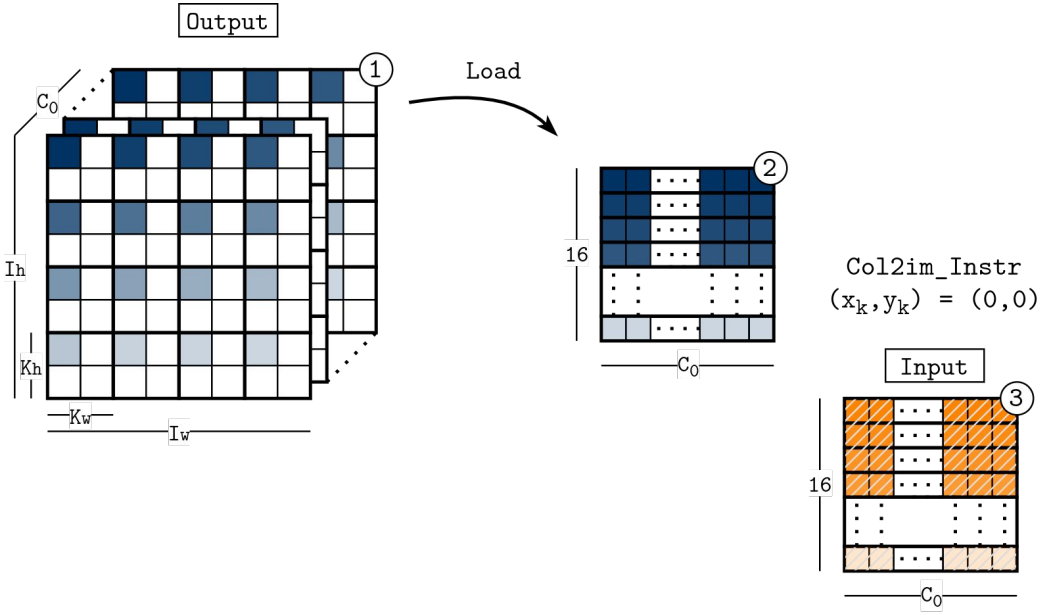
Col2im Instruction

- Inputs are fractals
  - NC_1HWC_0



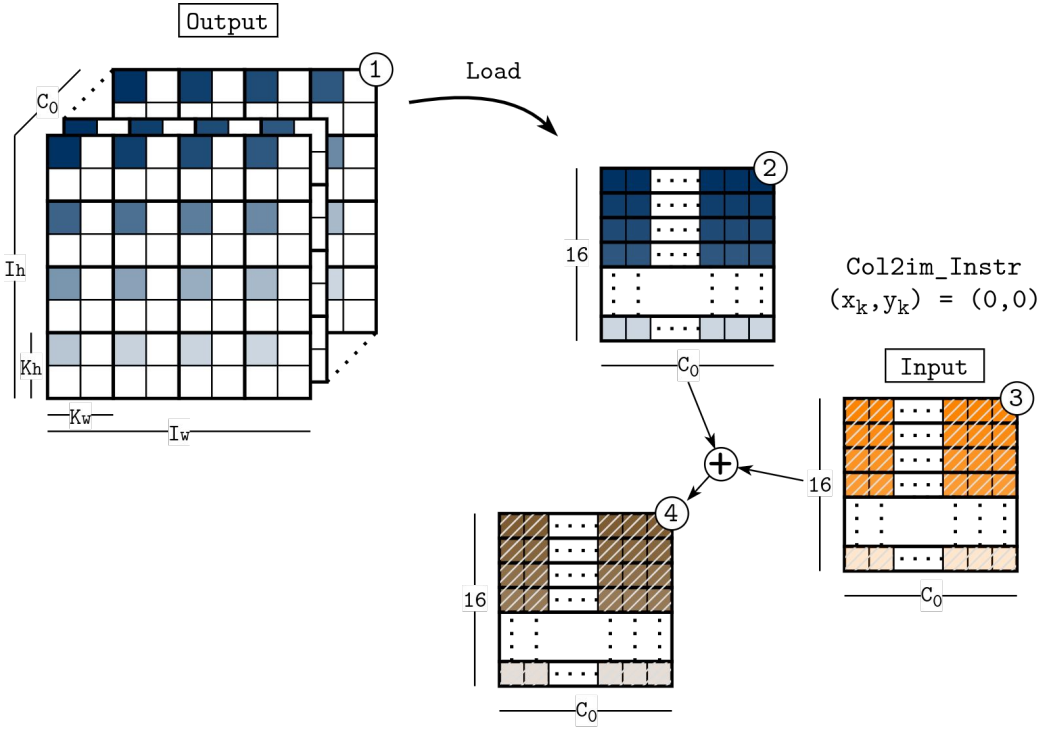
Col2im Instruction

- Inputs are fractals
  - NC_1HWC_0



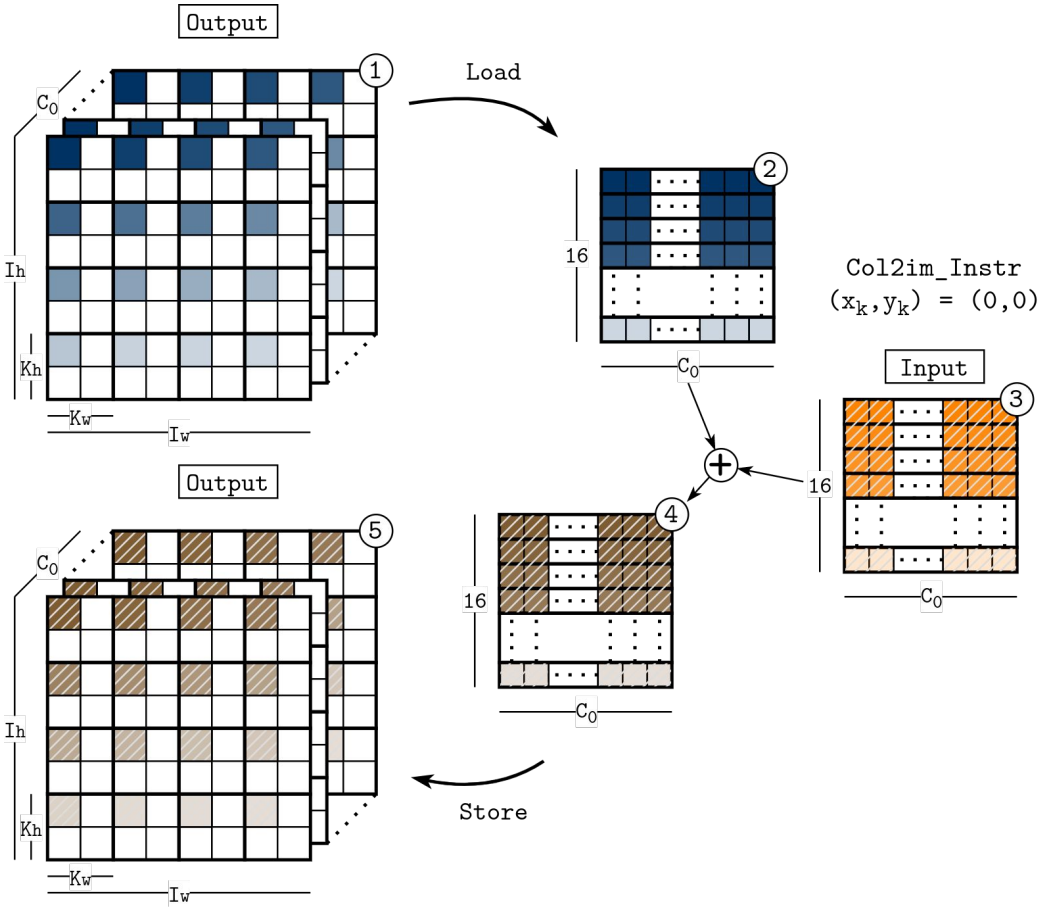
Col2im Instruction

- Inputs are fractals
  - NC_1HWC_0



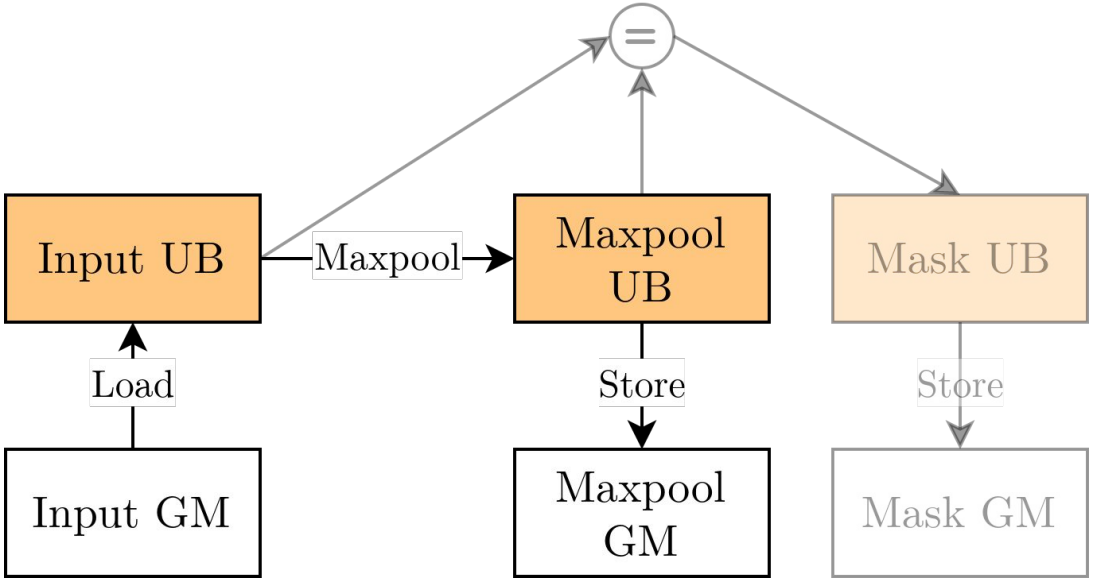
Col2im Instruction

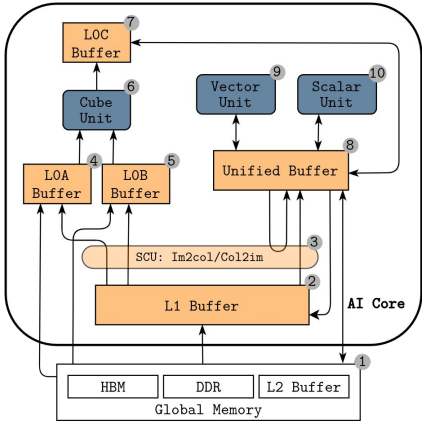
- Inputs are fractals
  - NC_1HWC_0



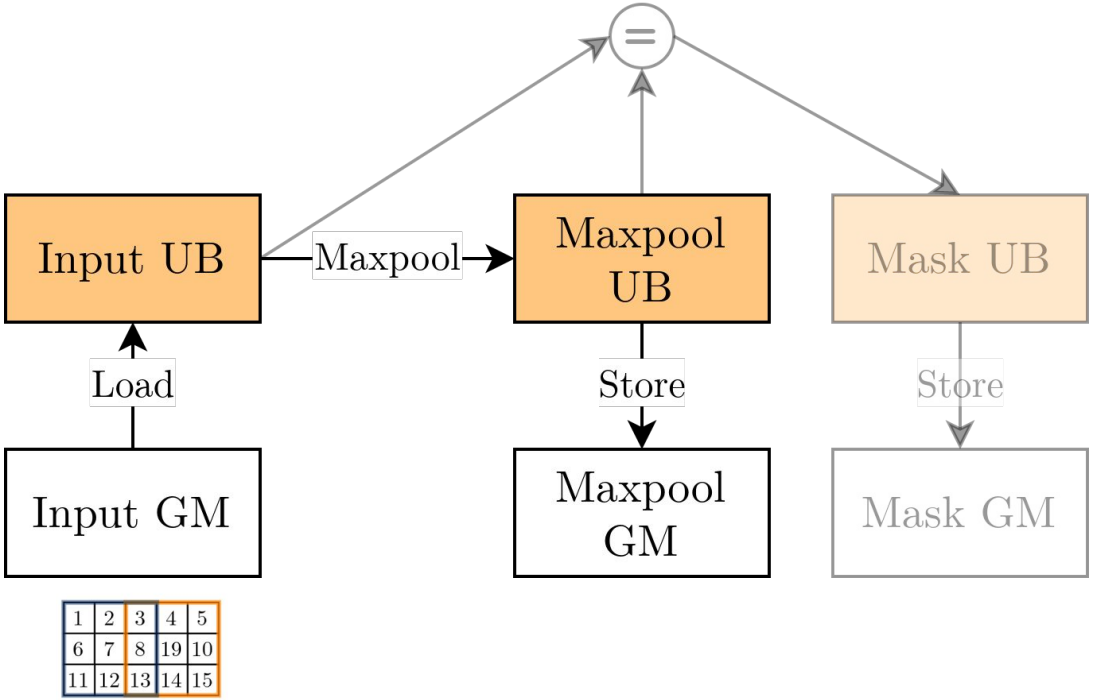
Im2col & Col2im Based Pooling

Forward Pooling

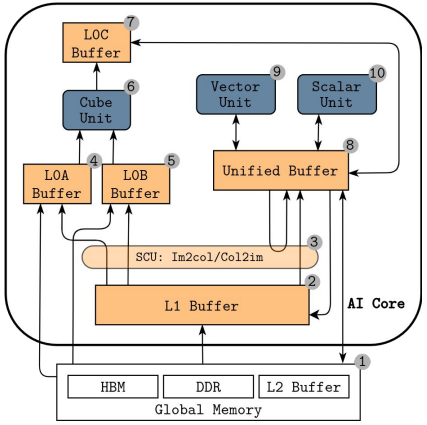




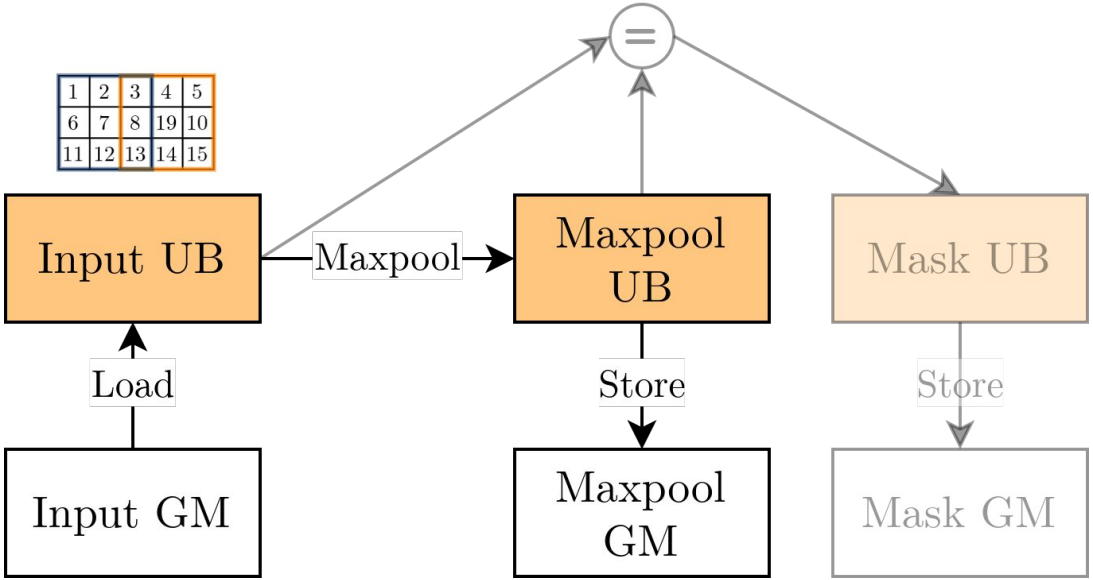
Forward Pooling

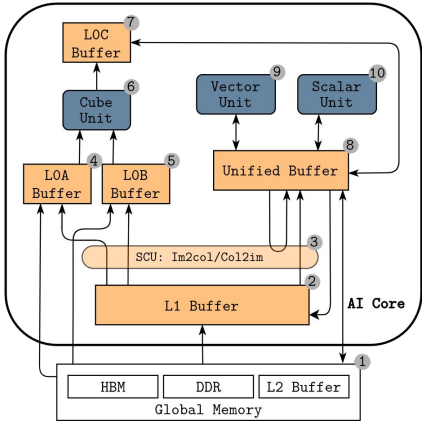


1	2	3	4	5
6	7	8	19	10
11	12	13	14	15

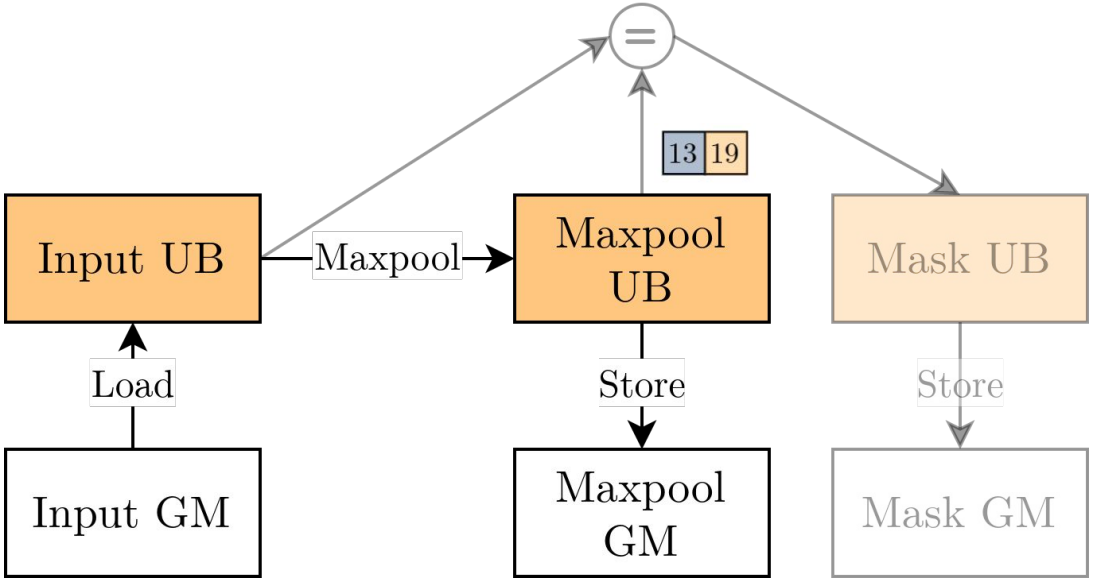


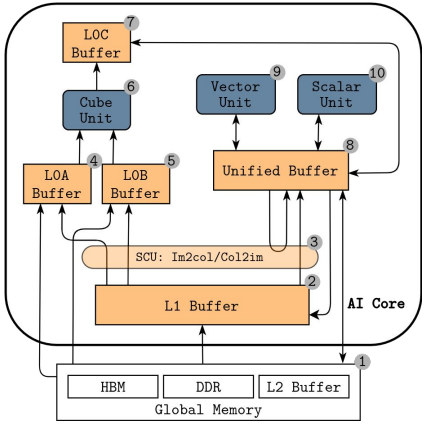
Forward Pooling



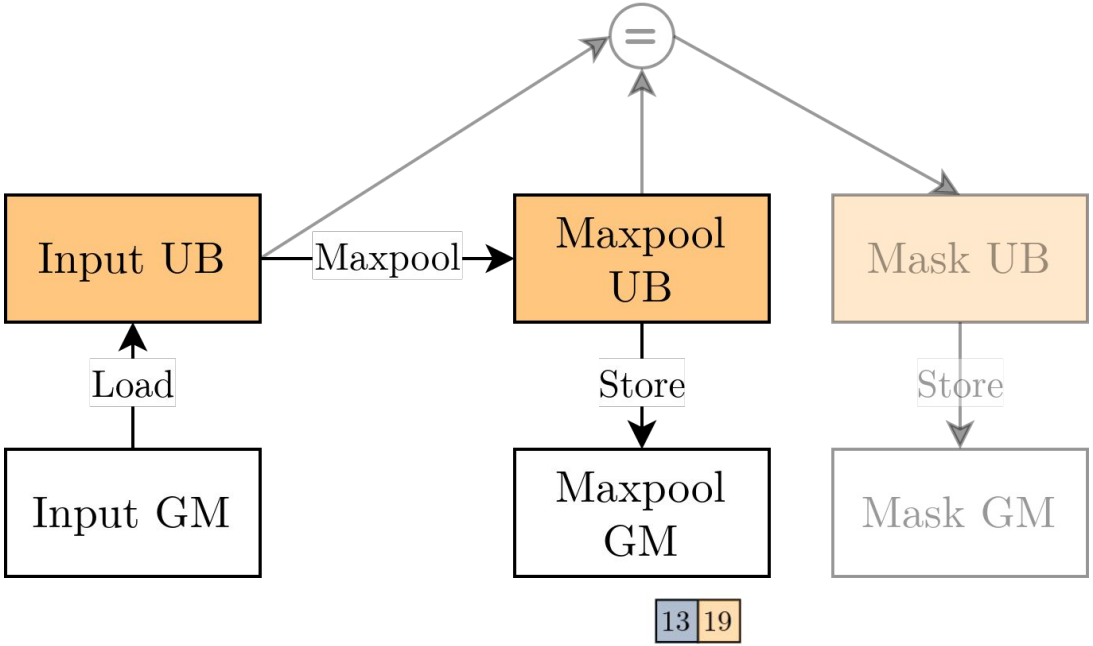


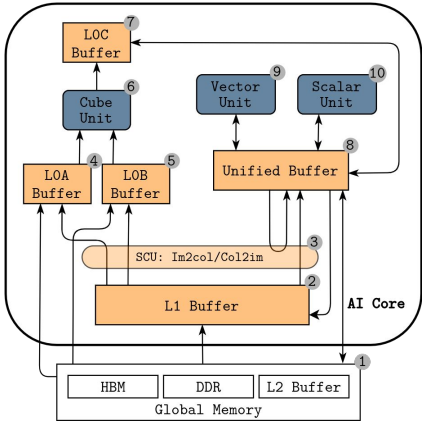
Forward Pooling



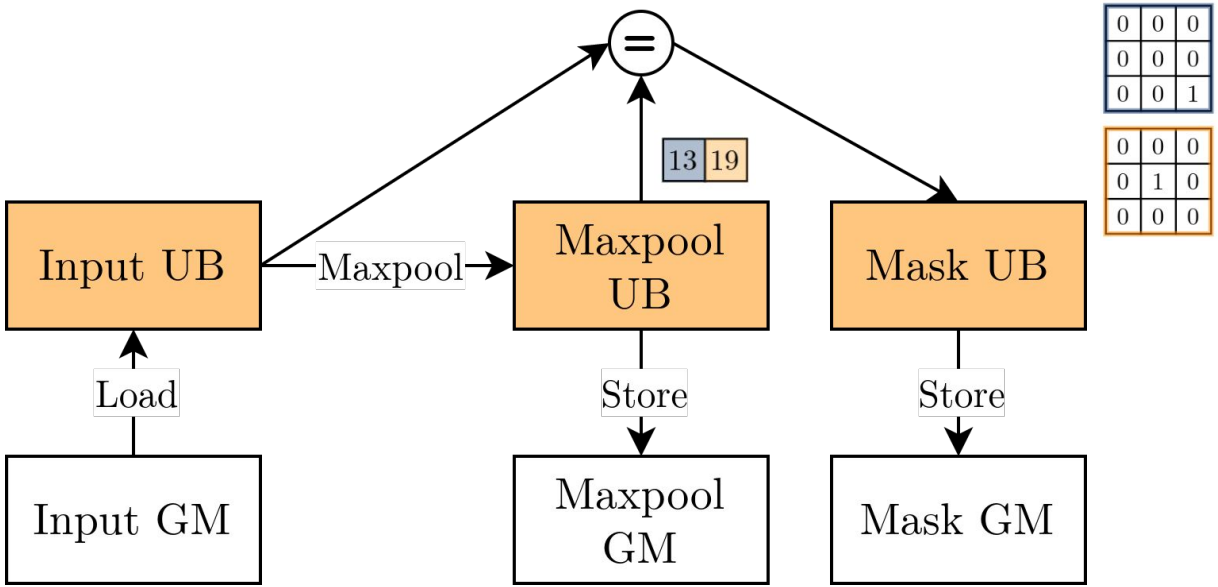


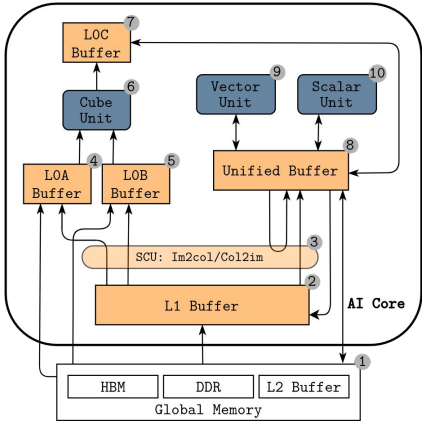
Forward Pooling



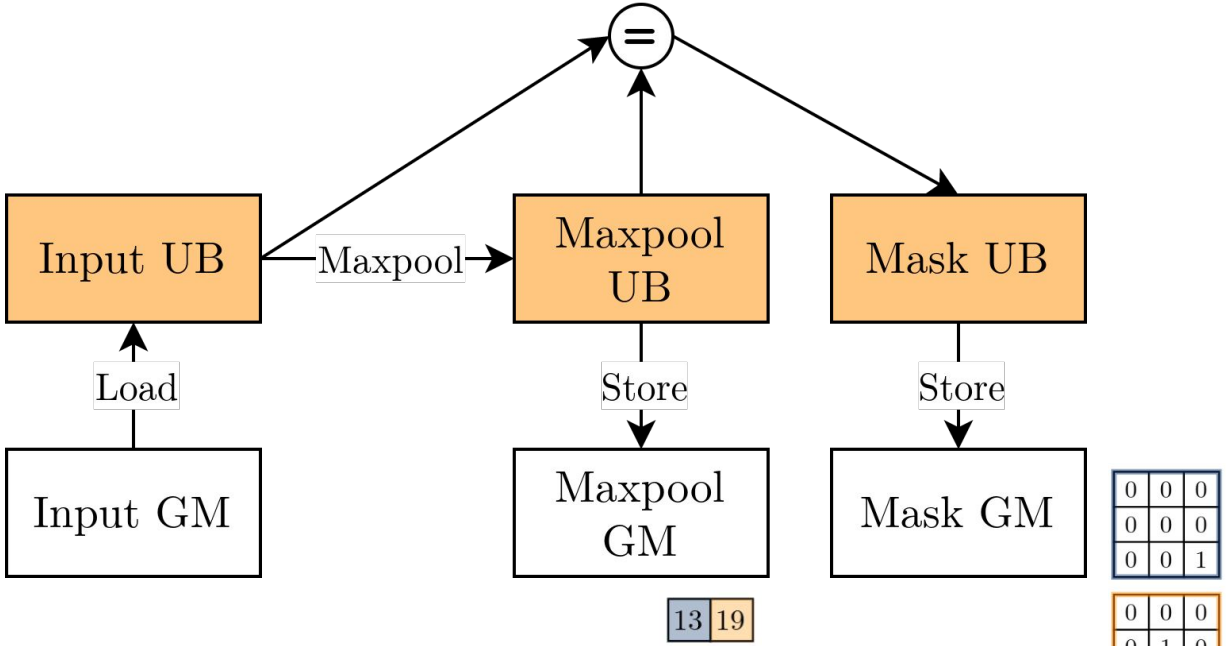


Forward Pooling + Mask





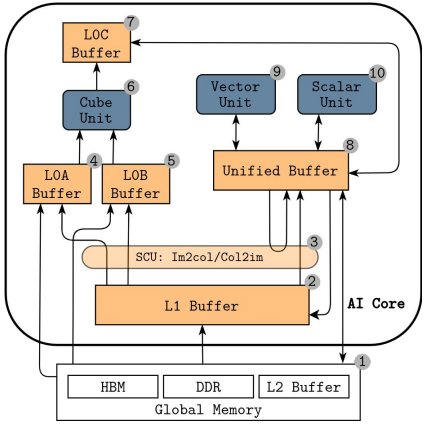
Forward Pooling + Mask



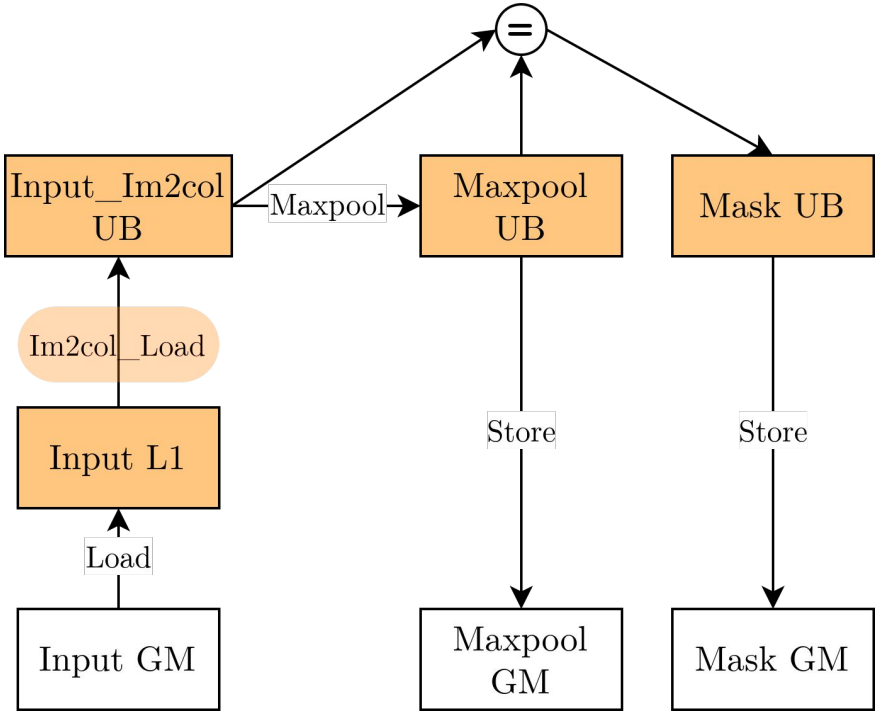
0	0	0
0	0	0
0	0	1

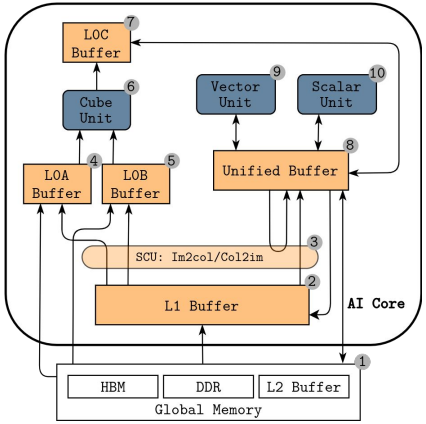
13	19

0	0	0
0	1	0
0	0	0

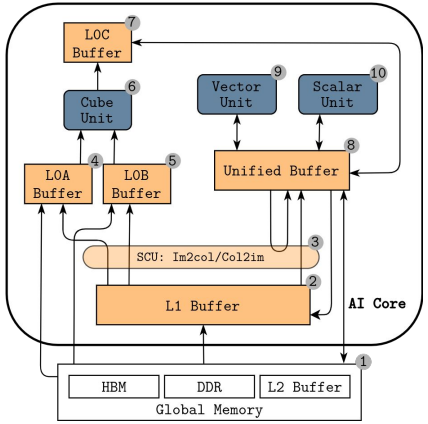


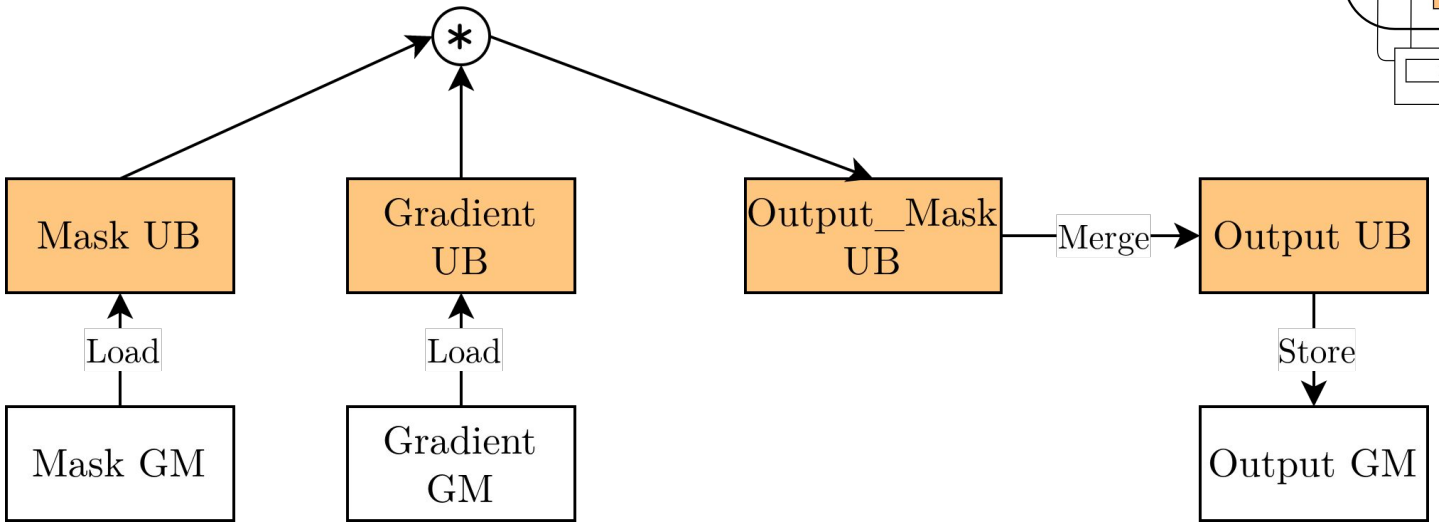
Im2col-Based Forward Pooling (+ Mask)



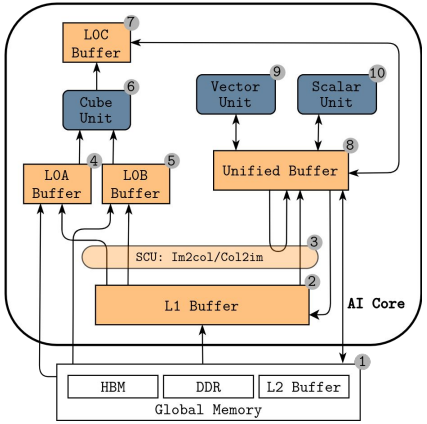


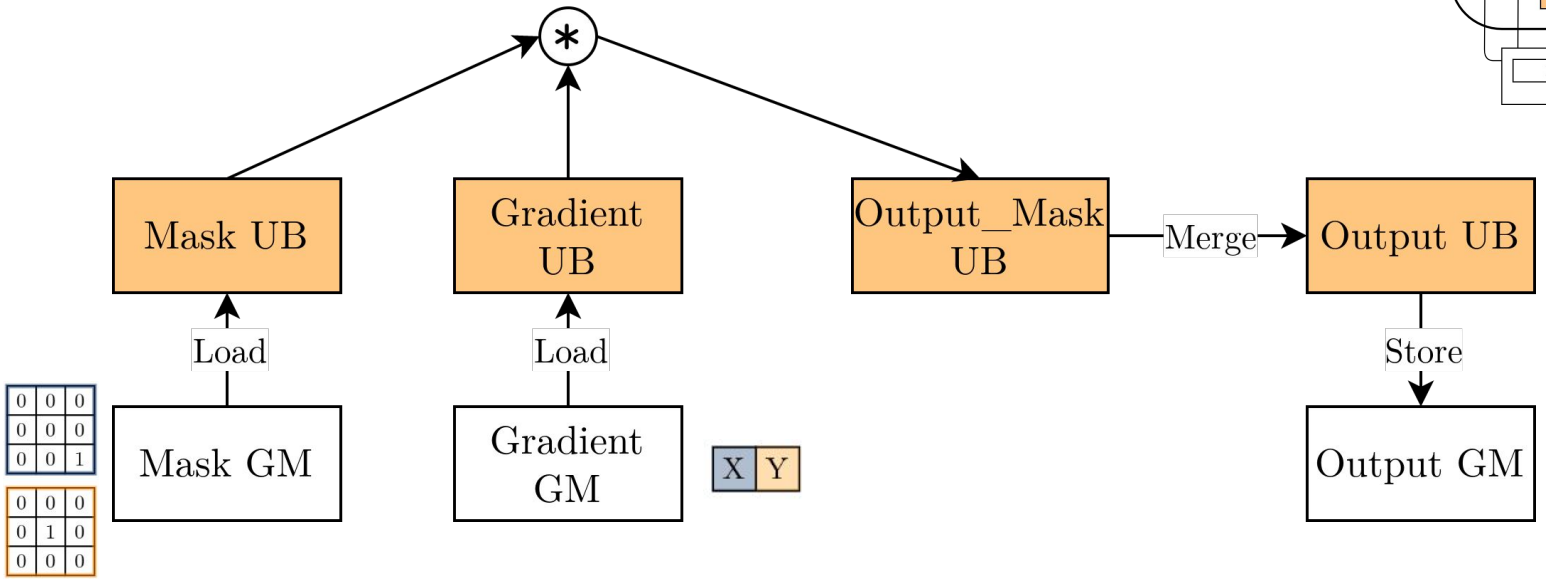
Backward Pooling



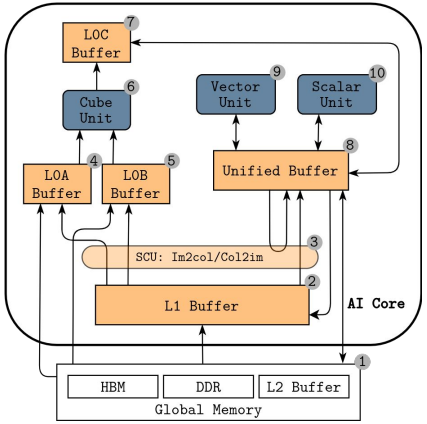


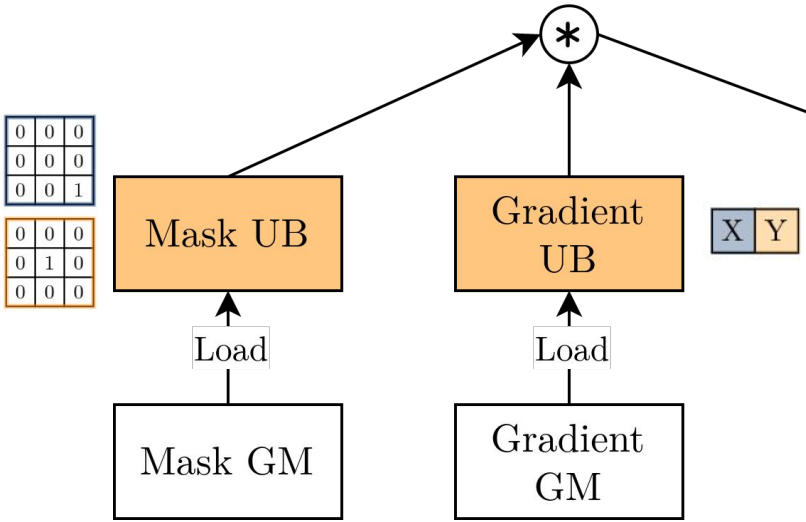
Backward Pooling



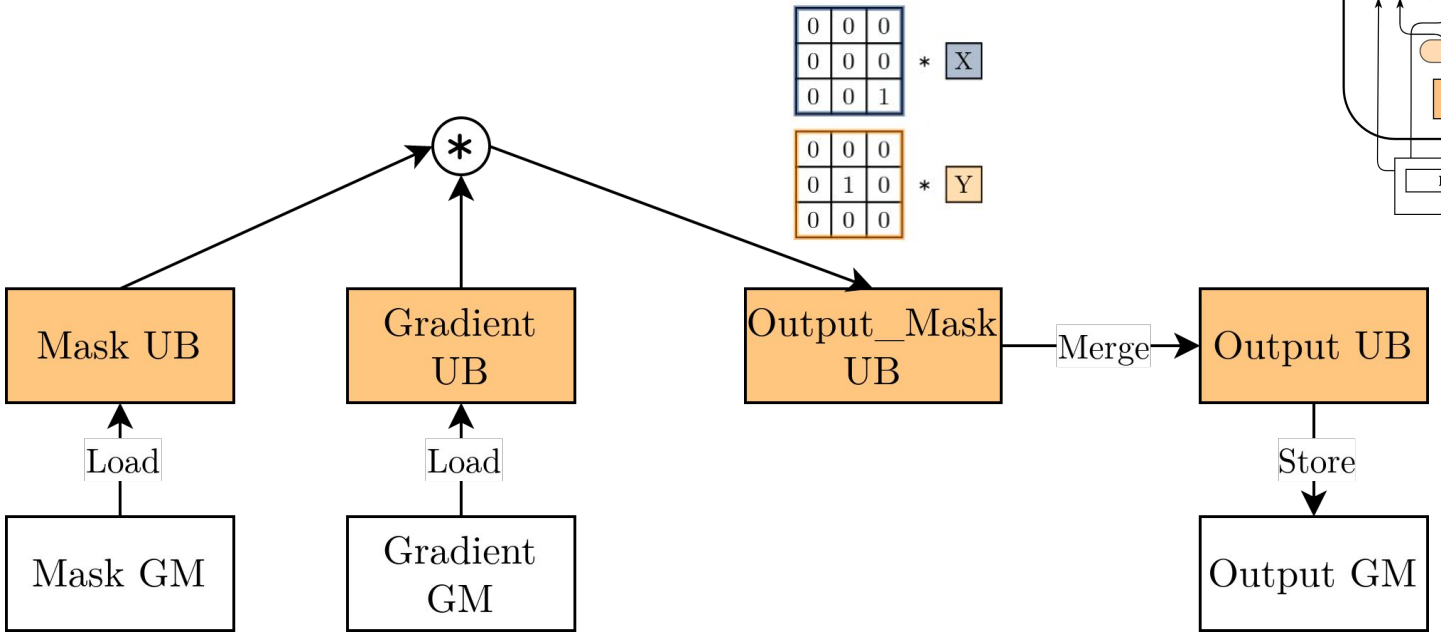


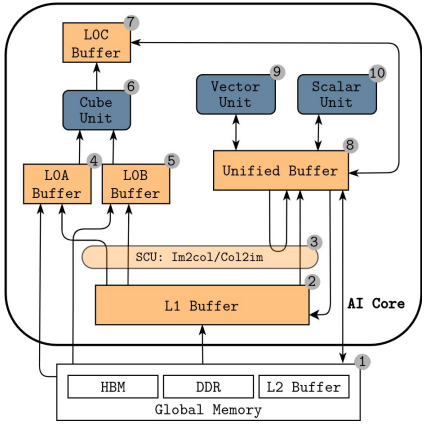
Backward Pooling



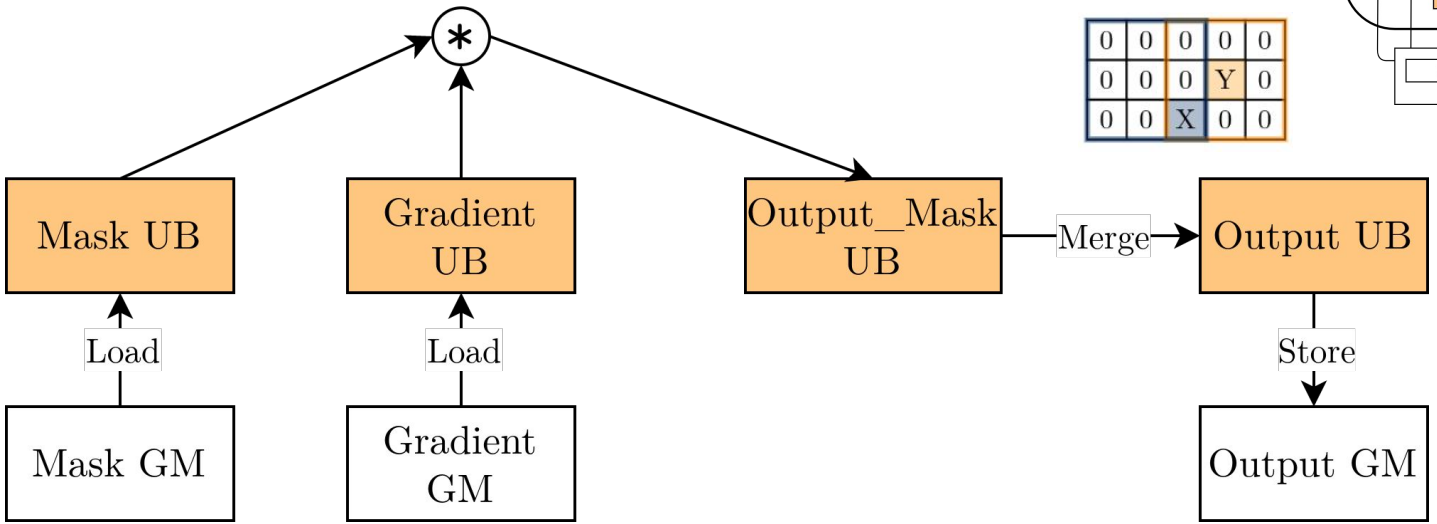


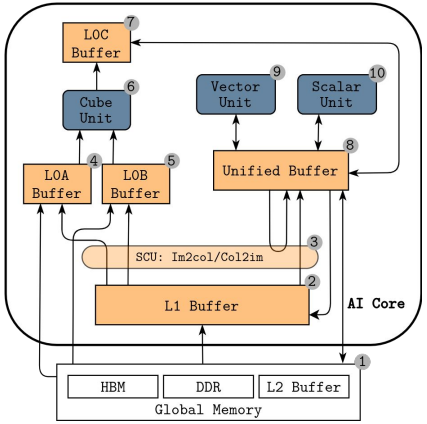
Backward Pooling



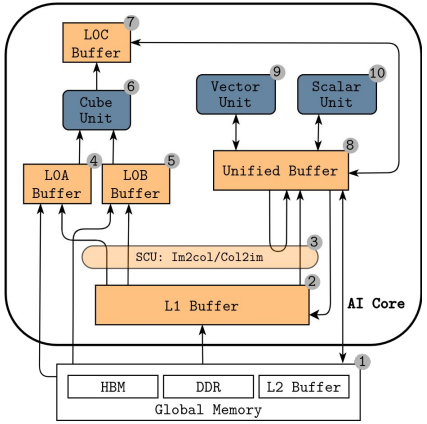


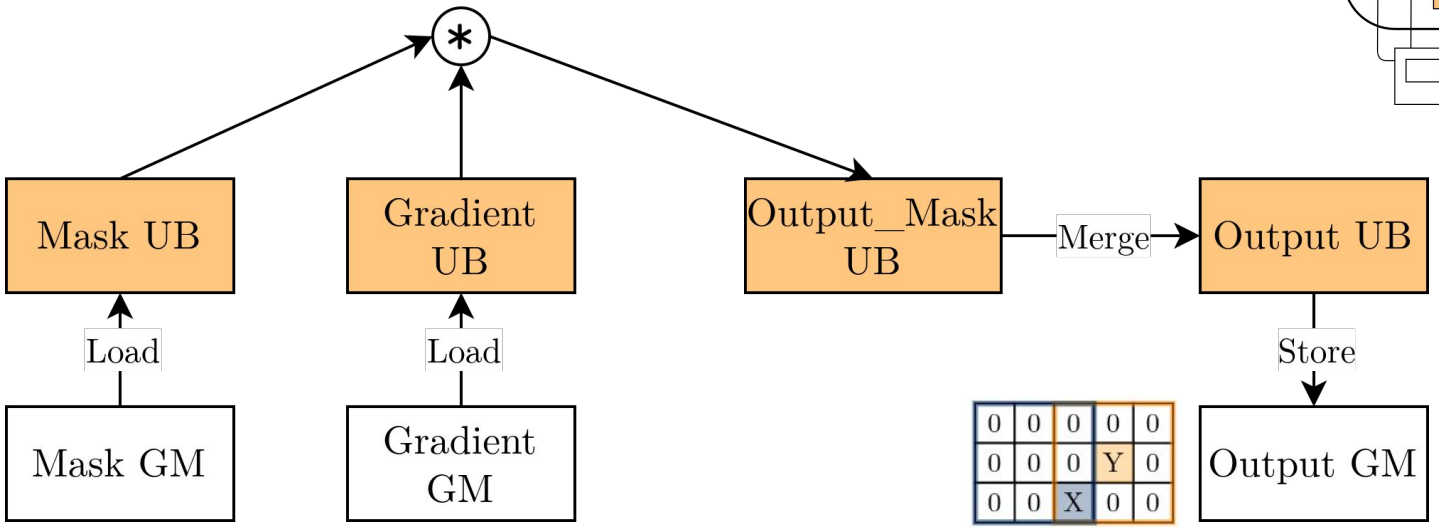
Backward Pooling



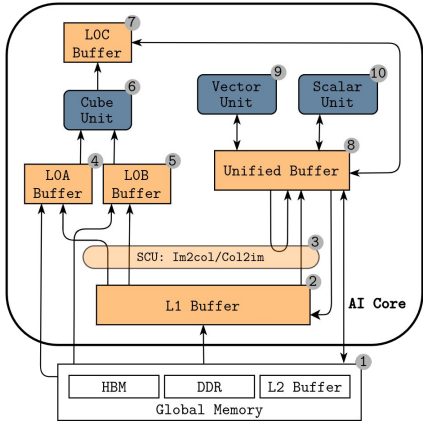


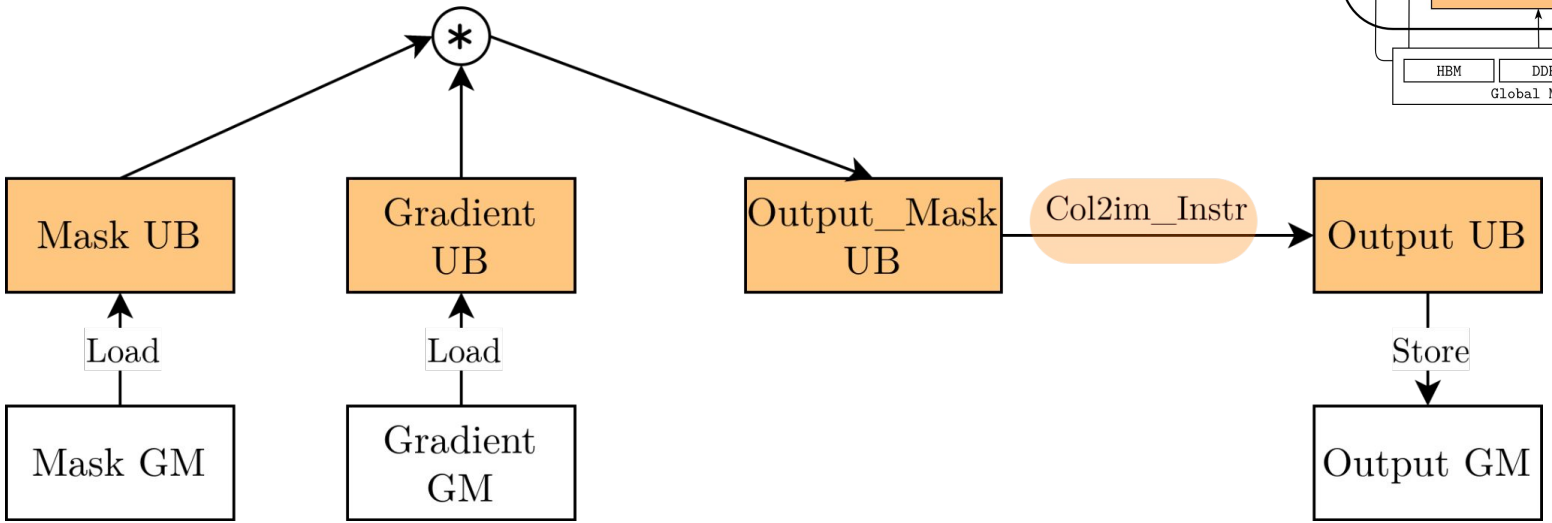
Backward Pooling



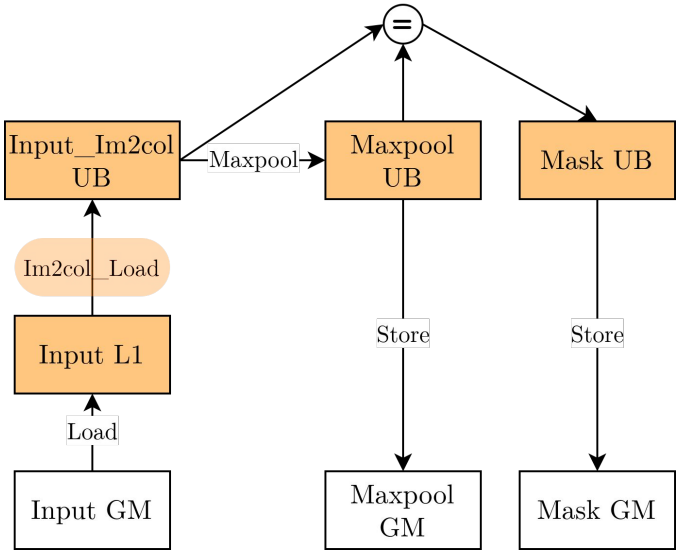


Col2im-Based Backward Pooling

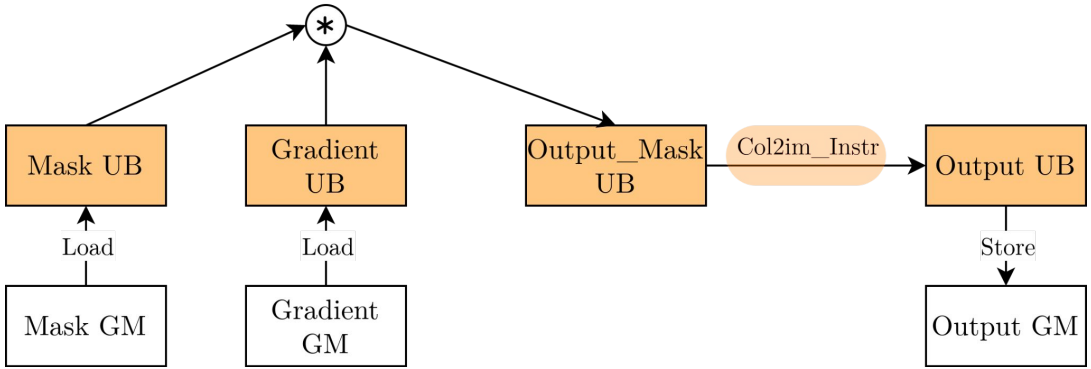




Avgpool


=
Input_Im2col UB
Maxpool
Maxpool UB
Mask UB
Im2col_Load
Input L1
Store
Store
Load
Input GM
Maxpool GM
Mask GM

Avgpool


*
Mask UB
Gradient UB
Output_Mask UB
Col2im_Instr
Output UB
Load
Load
Store
Mask GM
Gradient GM
Output GM

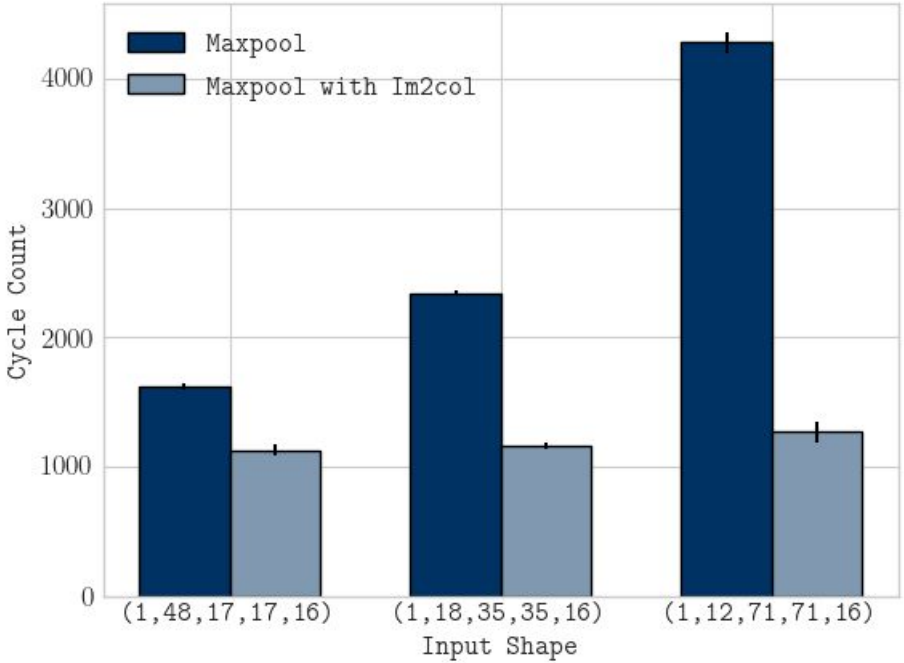
Experimental Evaluation

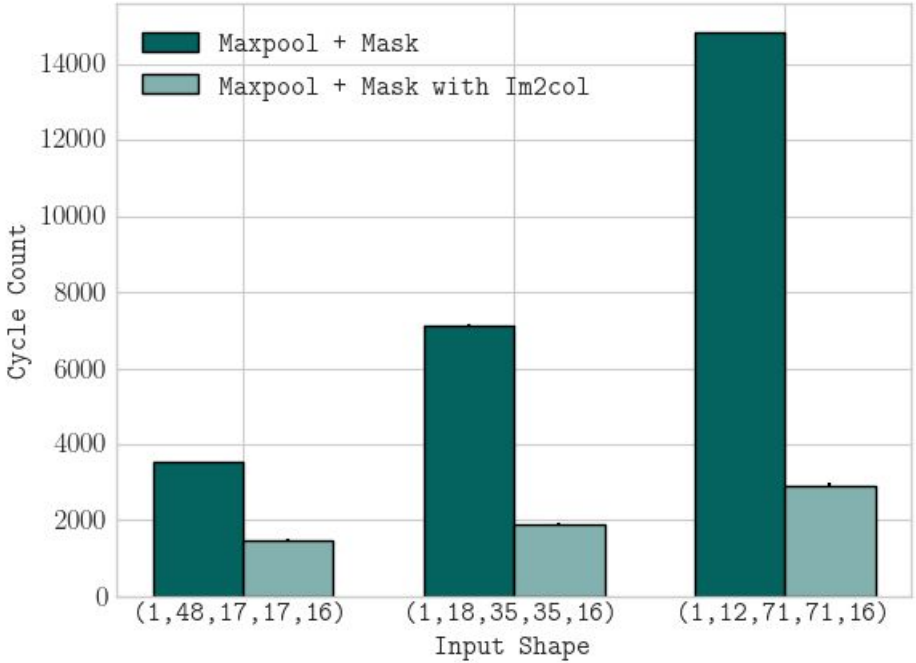
- InceptionV3 Comparison
- Stride Tests

Maxpool Inputs in CNNs

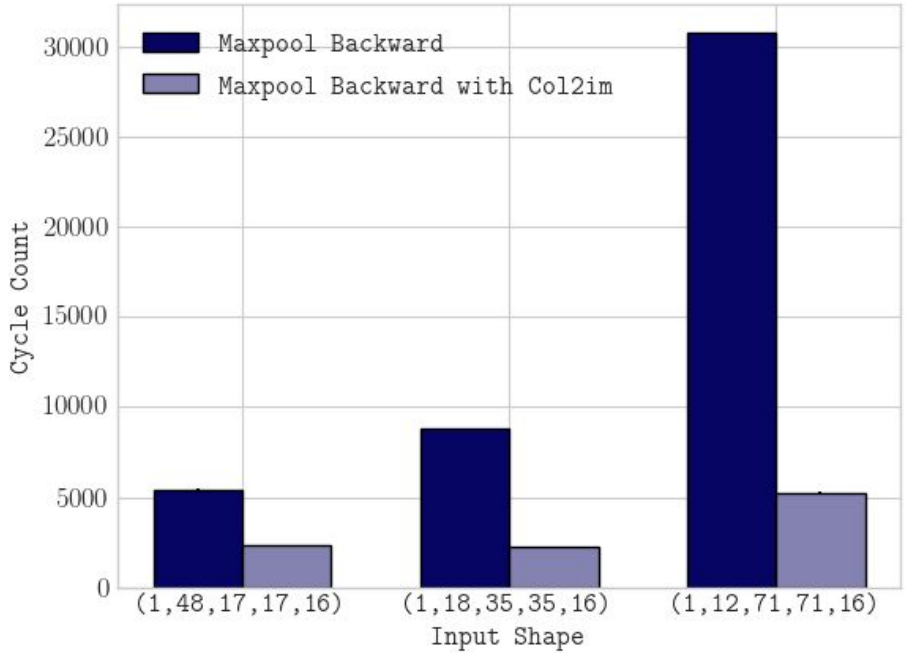
CNN	Input 1	Input 2	Input 3	Input 4
InceptionV3	147,147,64	**71,71,192**	**35,35,288**	**17,17,768**
Xception	147,147,128	74,74,192	37,37,728	19,19,1024
Resnet50	112,112,64	-	-	-
VGG16	224,224,64	112,112,64	56,56,256	28,28,512

InceptionV3 - Maxpool Forward



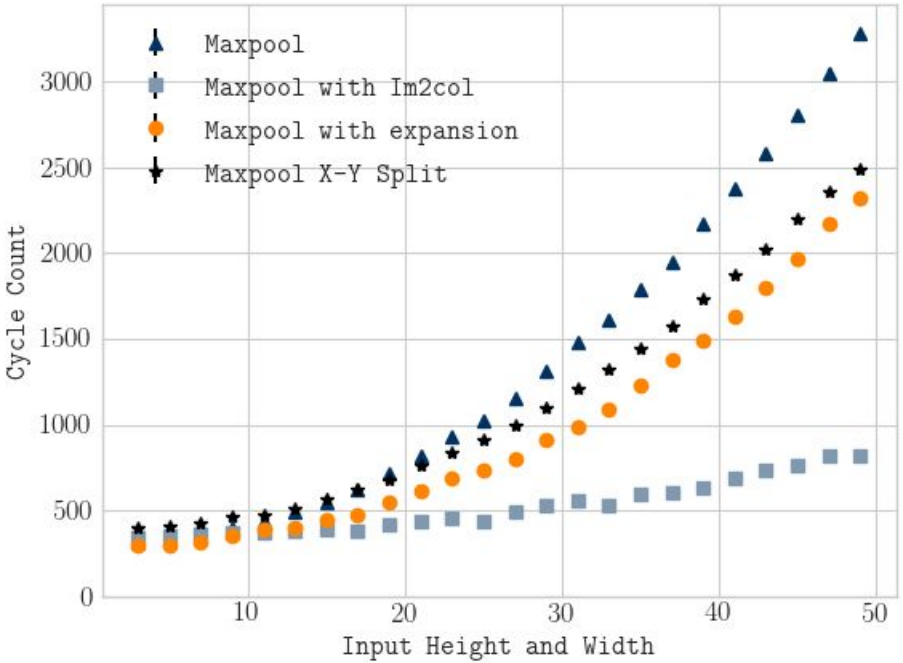


InceptionV3 - Maxpool Backward

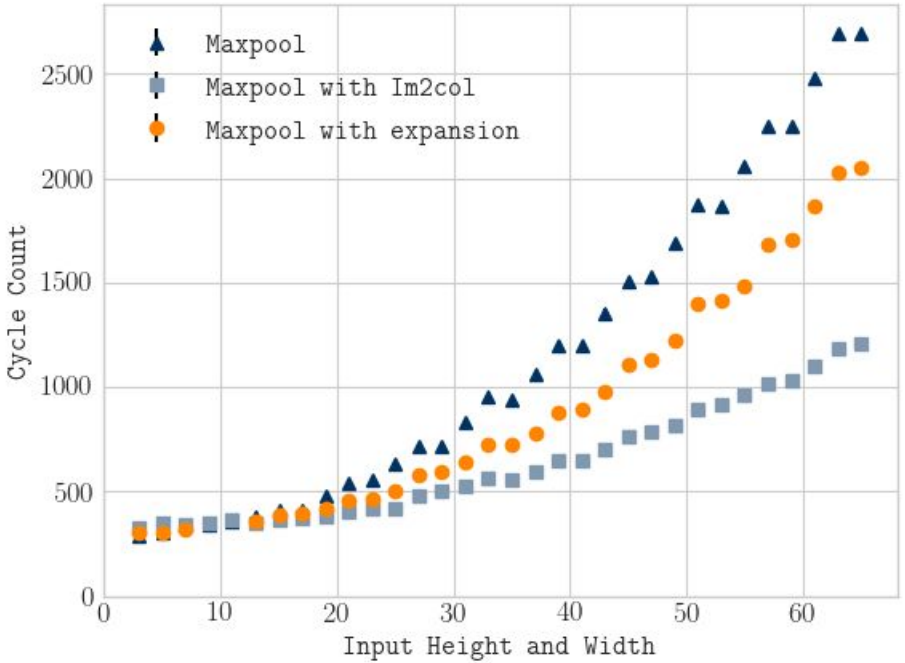


Stride Tests - Speedup

Stride = (2,2)

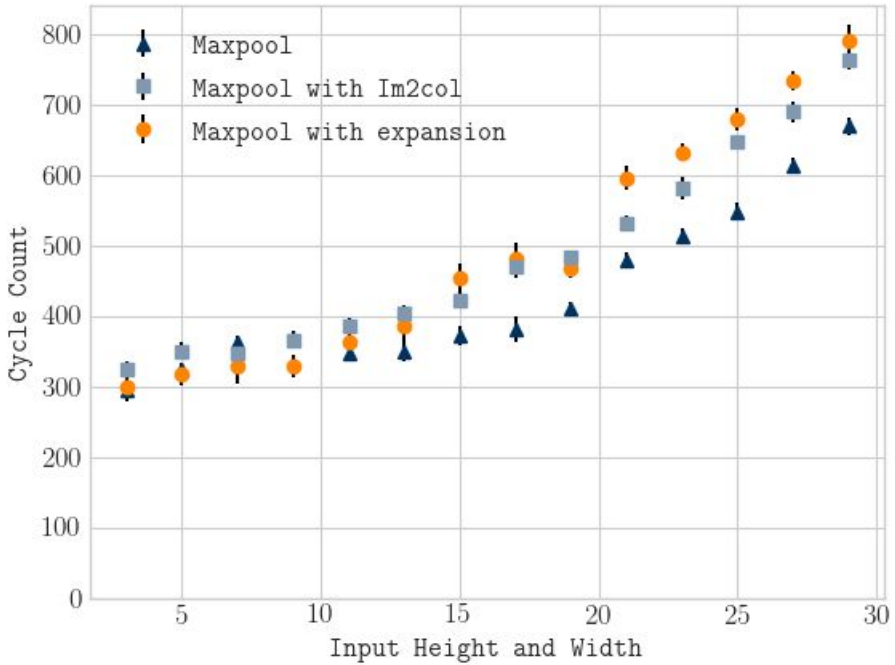


Stride = (3,3)



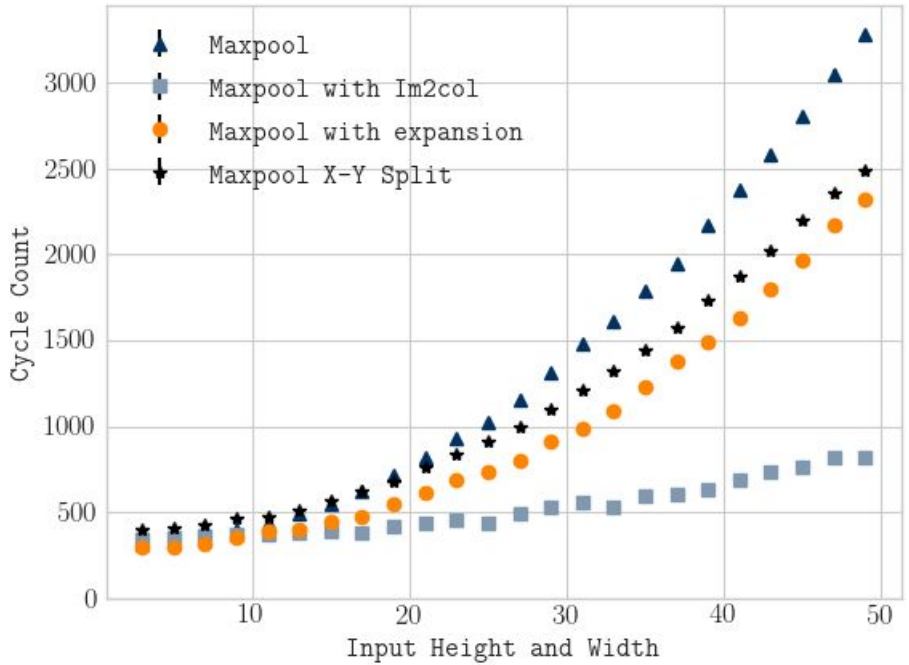
Stride Tests - Slowdown

Stride = (1,1)



Stride Tests - XY Split □

Stride = (2,2)



Conclusion

- DaVinci's Im2col & Col2im Instructions
- Im2col/Col2im Based Pooling in DaVinci
- Experimental Evaluation
  - Speedups up to 5.8x
  - Improvements for all but (1,1) stride
  - Superior to the X-Y split

Thank you!

Caio Salvador Rohwedder
c157754@dac.unicamp.br

- Overview
- Background
  - Convolution & Im2col
  - Col2im & Backward Operators
  - Pooling Operators
- DaVinci
  - AI Core
  - Fractal Memory Layout
  - Im2col & Col2im Instruction
- Im2col/Col2im Based Pooling
  - Forward & Backward
- Experiments
  - InceptionV3 Comparison
  - Stride Tests
- Conclusion