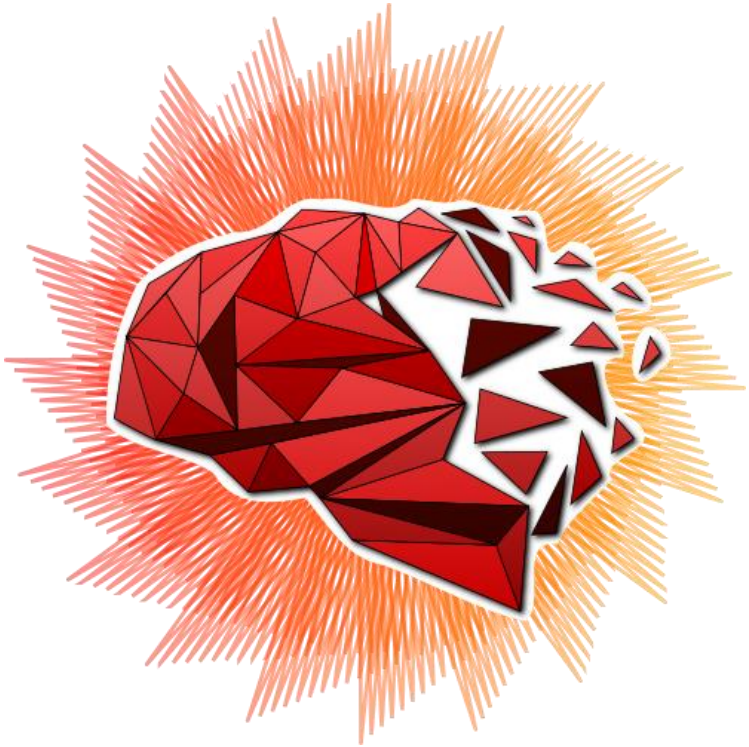


Orchestrating a brighter world

NEC

Sol: Transparent Neural Network Acceleration

Nicolas Weber
nicolas.weber@neclab.eu
NEC Laboratories Europe

AI Frameworks

PyTorch mxnet TensorFlow

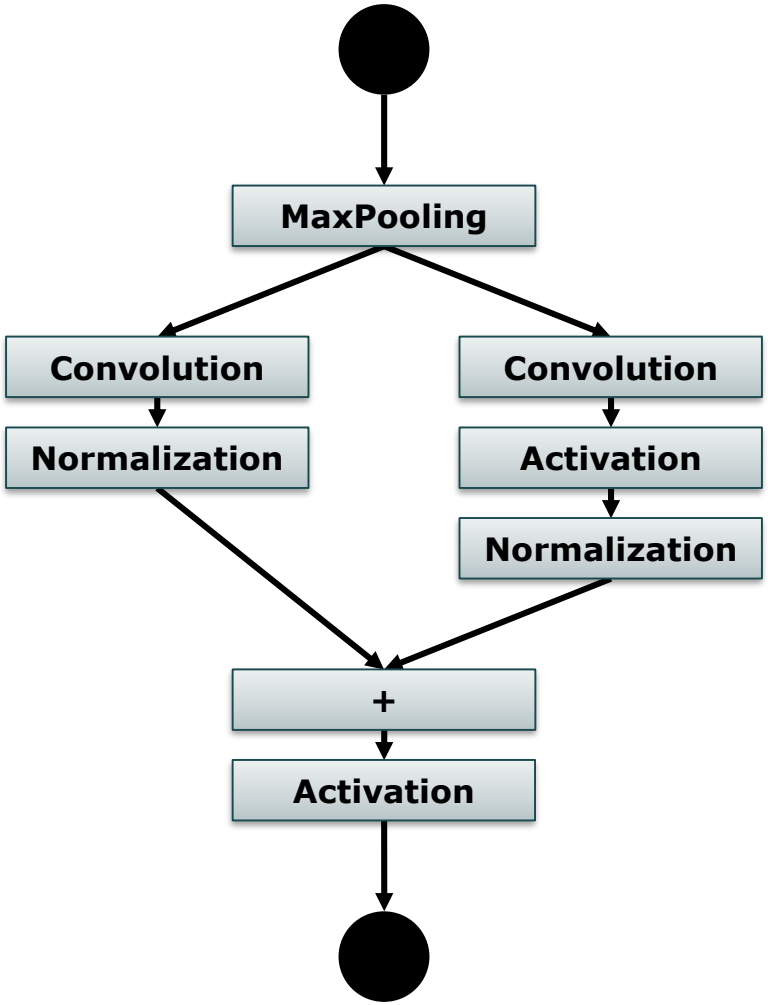
Optimizing Compilers
NGraph, TensorRT, TVM, Glow, …

Specialized Libraries
Intel MKL-DNN, cuDNN, …

Related Work

	TVM	Intel NGraph	NVIDIA TensorRT	Facebook Glow
Frameworks				
PyTorch	(ONNX)	(ONNX)	(ONNX)	✓
TensorFlow	✓	✓	✓	✗
MxNet	(ONNX)	✓	(ONNX)	✗
CNTK	(ONNX)	(ONNX)	(ONNX)	✗
Caffe2	✓	(ONNX)	✓	✗
ONNX	✓	✓	✓	✓
Devices				
X86	✓	✓	✗	✓
NVIDIA GPU	✓	✗	✓	✓
AMD GPU	✓	✗	✗	✗
NEC SX Aurora	✗	✗	✗	✗
ARM	✓	✗	✗	✗
FPGA	✓	✗	✗	✓
Operation Mode				
Inference	✓	✓	✗	✗
Training	✗	✓	✗	✗
Deployment	✓	✗	✓	✓

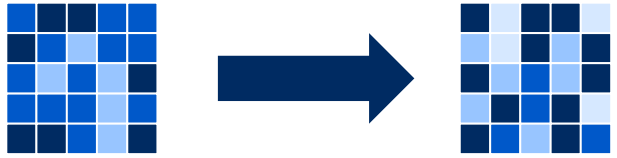
Neural Network Basics



Neural Network Layer Types

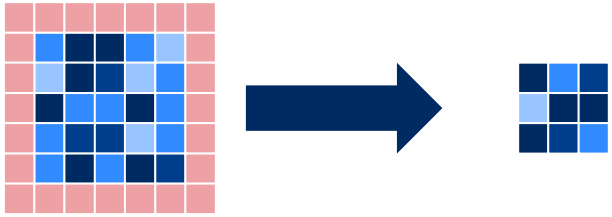
Element-wise (e.g. activation):

- ReLU, Sigmoid, BatchNorm*, …

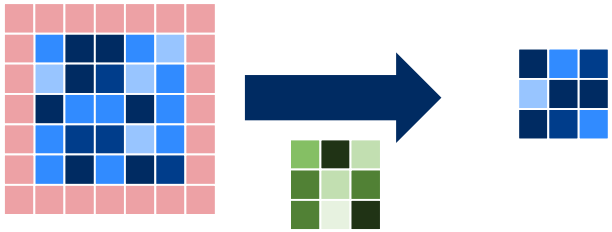


I/O Memory Bound (e.g. pooling):

- Avg-, Max-Pooling, Subsampling, …

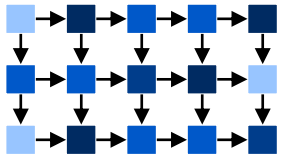


Computational Bound (e.g. conv):

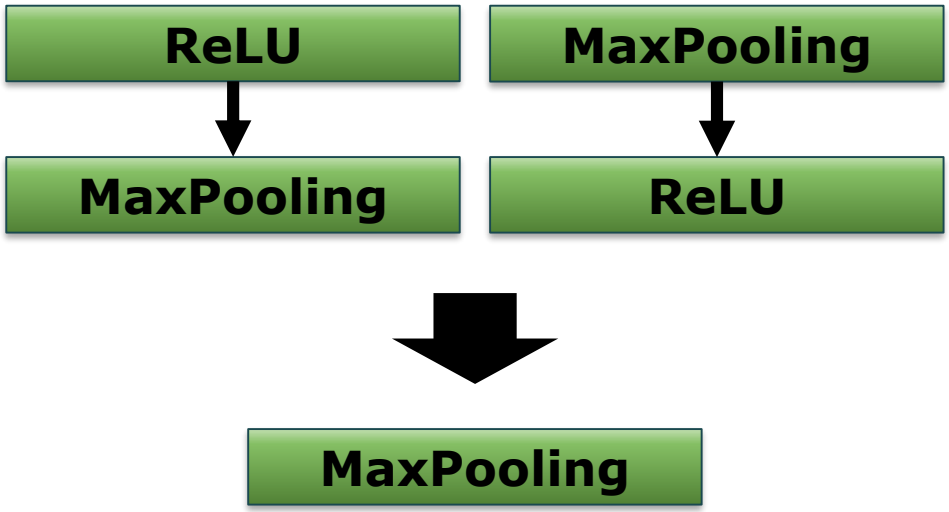


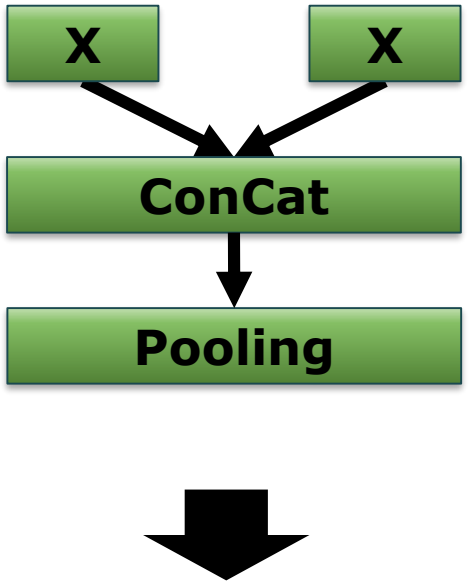
Parameter Memory Bound (e.g. dense):

Execution order constrained (e.g. RNN):

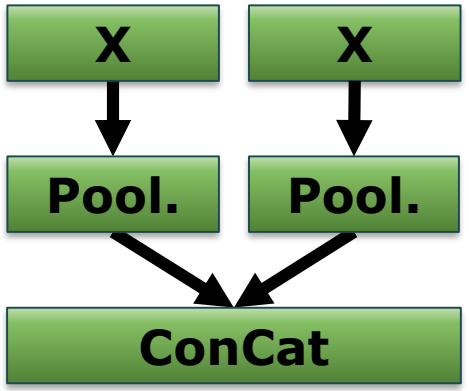


Graph Transformations

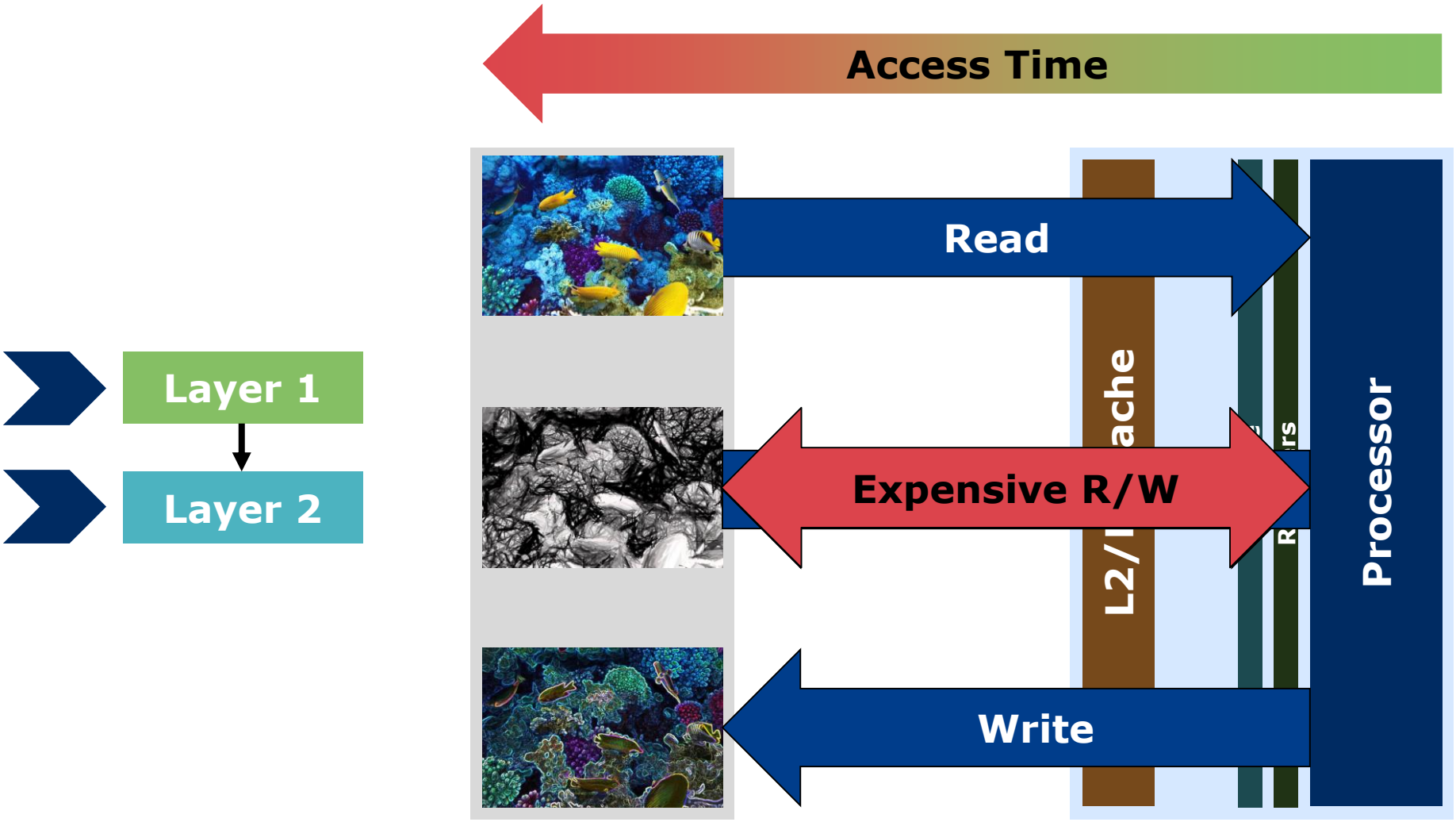




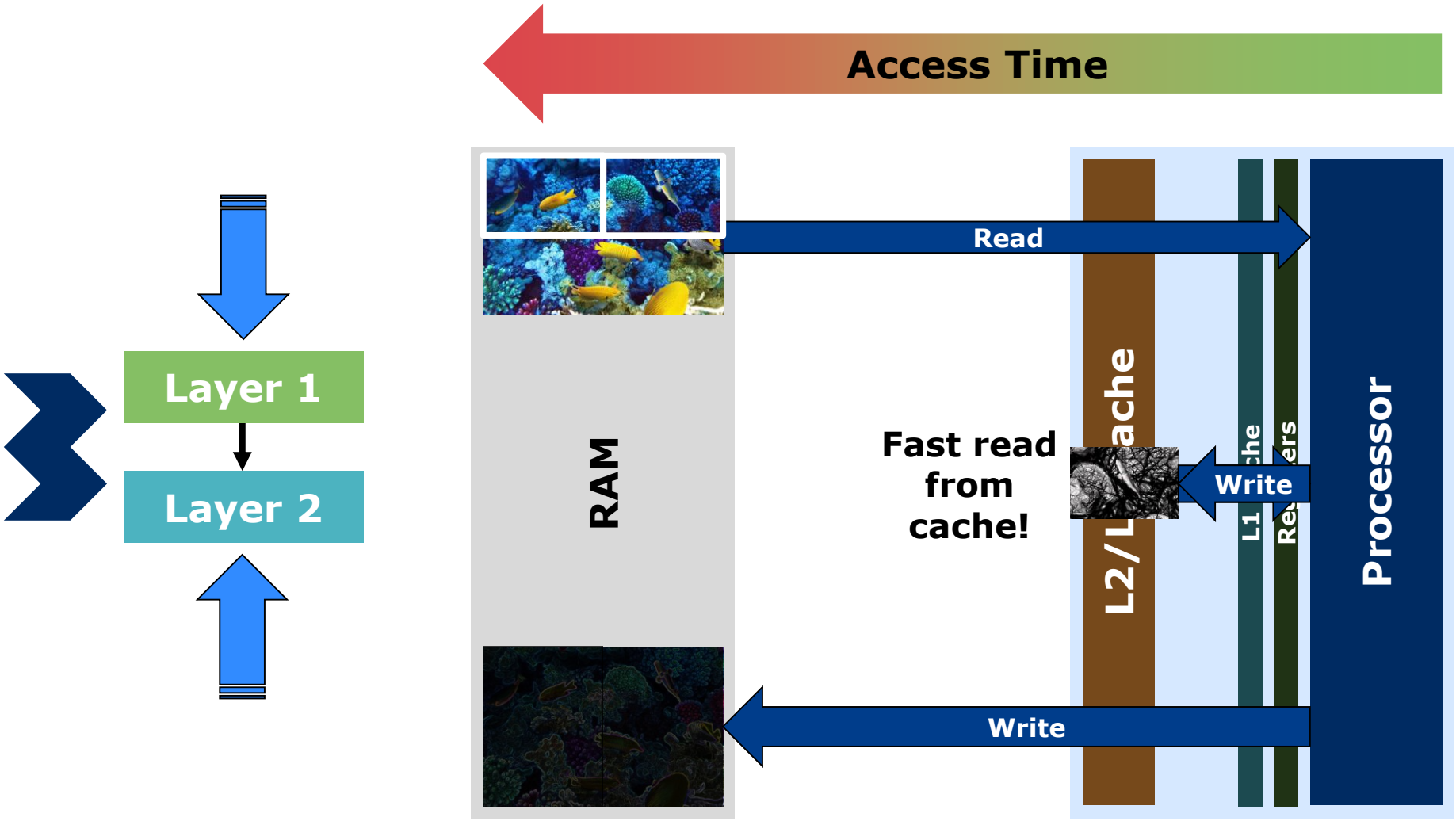

```
ReLU:
output = max(0, input)

MaxPooling:
output = -inf
for(window):
    output = max(output, input)
```



Traditional Neural Network Processing

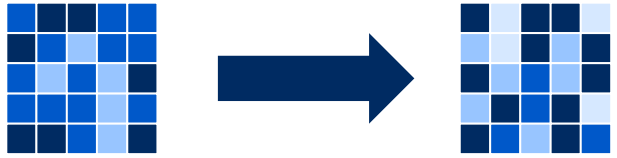


Depth-First Parallelism

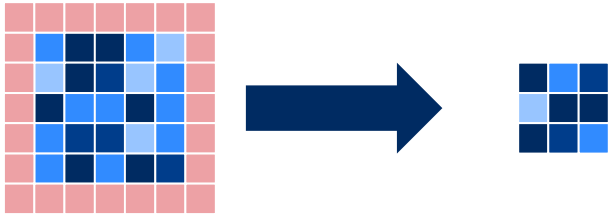


Neural Network Layer Types

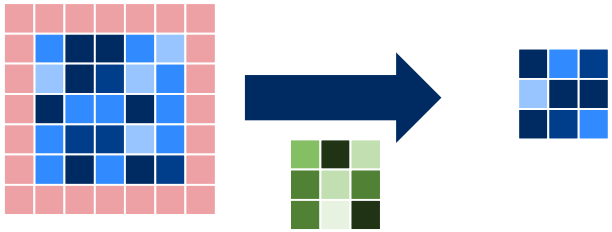
- Element-wise (e.g. activation):
  - ReLU, Sigmoid, BatchNorm*, …



- I/O Memory Bound (e.g. pooling):
  - Avg-, Max-Pooling, Subsampling, …

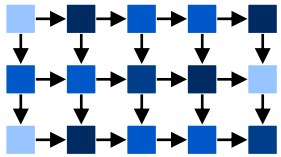


- Computational Bound (e.g. conv):



- Parameter Memory Bound (e.g. dense):

- Execution order constrained (e.g. RNN):



Why is this so difficult to optimize?

What data scientists see:

```
x = Conv(x, kernel=1x1, bias=True)
x = ReLU(x)
x = AvgPooling(x, kernel=13x13)
```

What HPC people see:

```
function(Conv):
    for(Batch, OutChannel, Y, X):
        for(InChannel, KernelY, KernelX):
            output[…] += input[…] * weight[…]
        output[…] += bias[…]

function(ReLU):
    for(Batch, OutChannel, Y, X):
        output[…] = max(0, input[…])

function(AvgPooling):
    for(Batch, OutChannel, Y, X):
        for(KernelY, KernelX):
            output[…] += input[…] / (13*13)
```

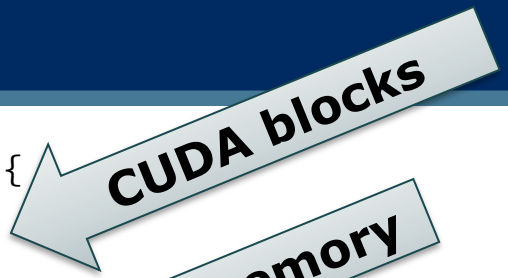
Why is this so difficult to optimize?

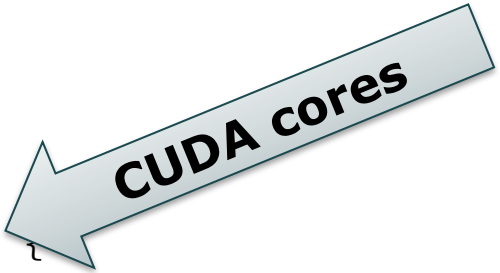
What we actually want:

```
function(FusedNetwork):
  for(Batch, OutChannel):
    float N[…]
    for(Y, X):
        for(InChannel, KernelY, KernelX):
            N[…] += input[…] * weight[…]
        N[…] += bias[…]
        N[…] = max(0, X)
    for(Y, X):
        for(KernelY, KernelX):
            output[…] += N[…] / (13*13)
```

Code Generation

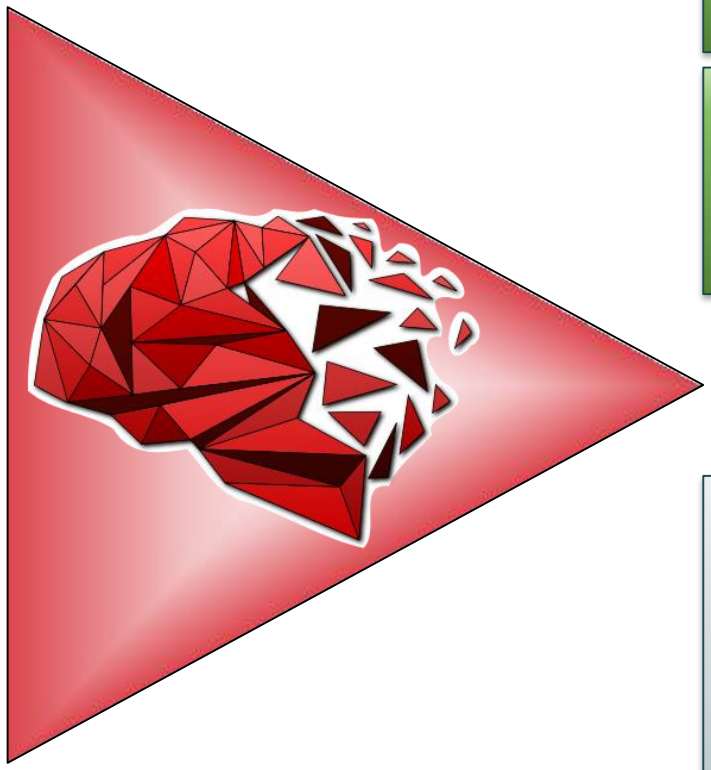
```
__global__ void F64486B08(...) {
  const int O0idx = blockIdx.x;
  const int O0 = O0idx / 256;
  const int O1 = O0idx % 256;
  __shared__ float T64[169];
  for(int O2idx = threadIdx.x; O2Idx < 169; O2Idx += 128) {
    float T63 = 0.0f;
    for(int I1 = 0; I1 < 512; I1++)          // #1 Convolution: 1x1 Pooling
      T63 += T61[O0 * 86528 + I1 * 169 + O2idx] * P63_weight[O1 * 512 + I1];
    T63 = (T63 + P63_bias[O1]);              // #1 Convolution: Bias
    T64[O2Idx] = fmaxf(T63, 0.0f);           // #2 ReLU
  }
  T66[O1] = REDUCE_ADD(T64);                 // #3 AvgPooling: 13x13 Pooling
  T66[O1] = (T66[O1] / 169.0f);              // #3 AvgPooling: Normalization
}
```





Sol Platform

PyTorch mxnet TensorFlow



Graph Transformations

Depth-First-Parallelism
I/O bound + element-wise

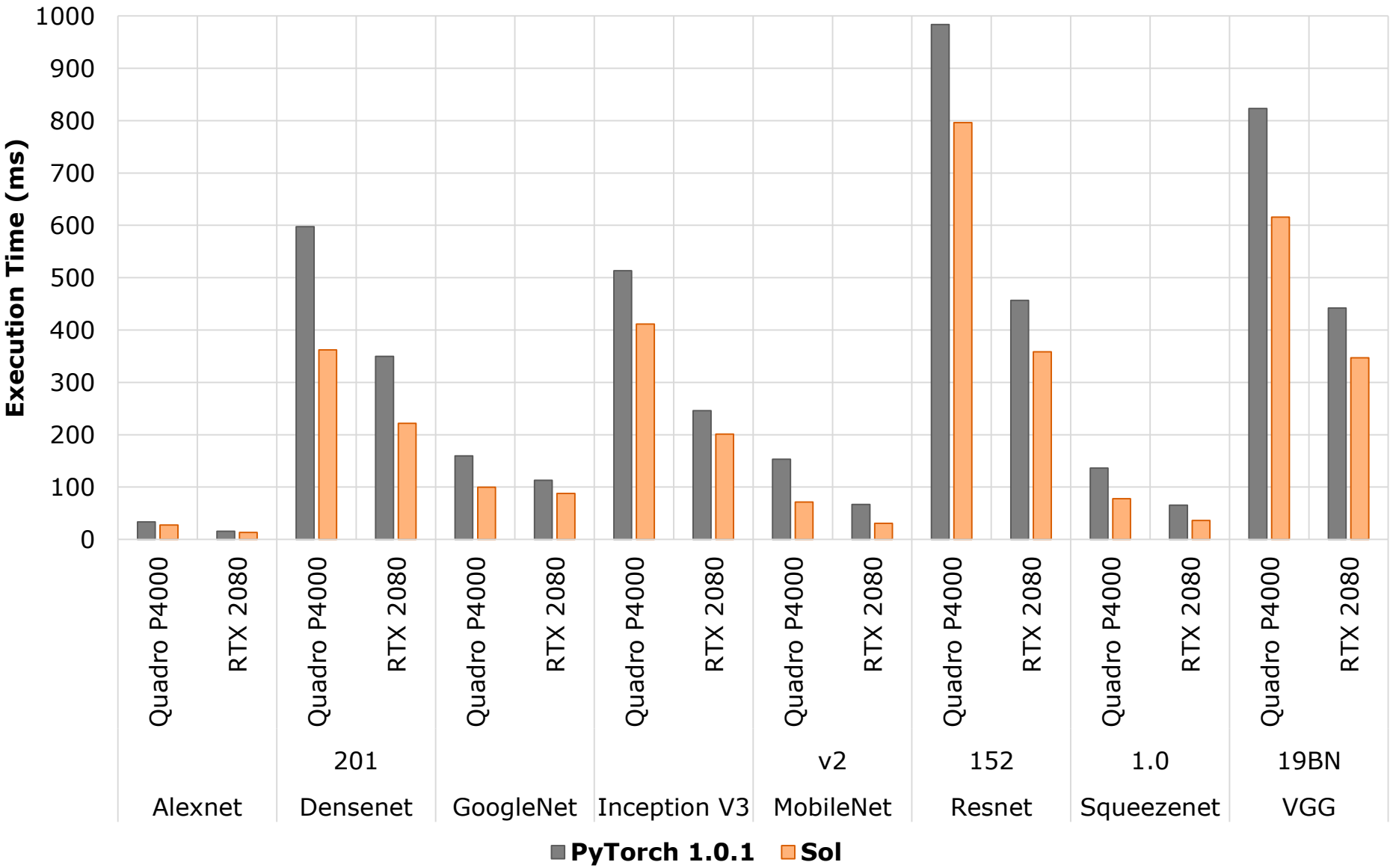
Optimizations

Specialized Libraries
Computational + Parameter bound
(e.g. Intel MKL-DNN, cuDNN,…)

Sol Usage (PyTorch)

```
import torch
from torch.autograd import Variable
from torchvision import models
from sol.pytorch import optimize

model  = models.__dict__["…"]()
input  = torch.rand(32, 32, 224, 224)
model  = optimize(model, input.size())
output = model(input)
```

Inference (128x Batched) Sol vs PyTorch v1.0.1



Related Work

	TVM	Intel NGraph	NVIDIA TensorRT	Facebook Glow	Sol
Frameworks					
PyTorch	(ONNX)	(ONNX)	(ONNX)	✓	✓
TensorFlow	✓	✓	✓	✗	✓
MxNet	(ONNX)	✓	(ONNX)	✗	✓
CNTK	(ONNX)	(ONNX)	(ONNX)	✗	✓
Caffe2	✓	(ONNX)	✓	✗	✗
ONNX	✓	✓	✓	✓	(planned)
Devices					
X86	✓	✓	✗	✓	✓
NVIDIA GPU	✓	✗	✓	✓	✓
AMD GPU	✓	✗	✗	✗	(planned)
NEC SX Aurora	✗	✗	✗	✗	✓
ARM	✓	✗	✗	✗	(planned)
FPGA	✓	✗	✗	✓	✗
Operation Mode					
Inference	✓	✓	✗	✗	✓
Training	✗	✓	✗	✗	✓
Deployment	✓	✗	✓	✓	(planned)

Sol: Transparent Neural Network Acceleration

NEC Laboratories Europe is the European research center of the NEC Group, a world leader in the computer and communications markets with a large world-wide base of R&D Laboratories. Top researchers from more than 20 countries develop, pilot and bring to reality technologies through open innovation.

NEC Laboratories Europe has an immediate opening for a

High Performance Machine Learning Systems

Research Scientist / Senior Researcher

Nicolas Weber

nicolas.weber@neclab.eu
NEC Laboratories Europe

www.neclab.eu/jobs