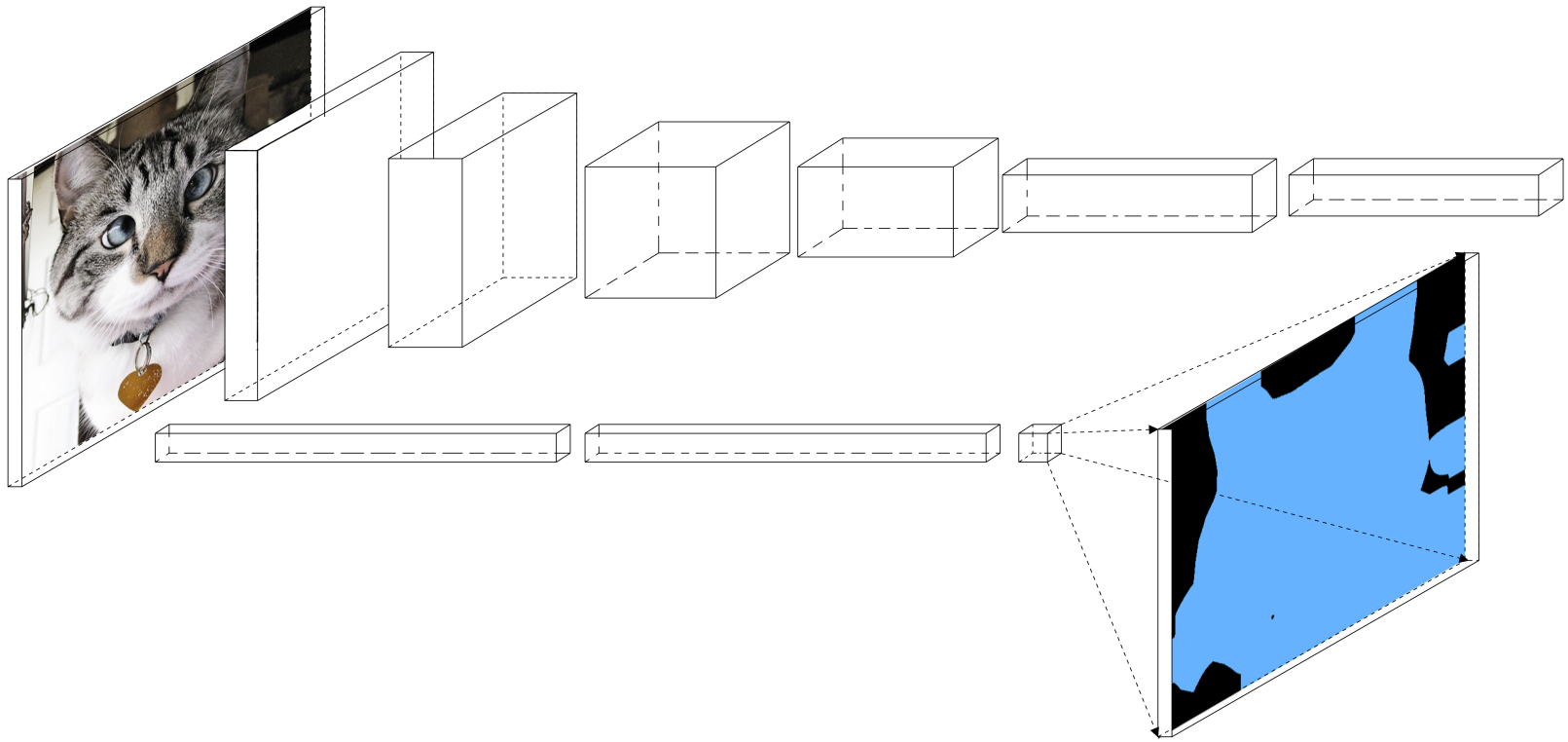
Latte: Fully Convolutional Network in Halide

Anbang Hu, X.D. Zhai

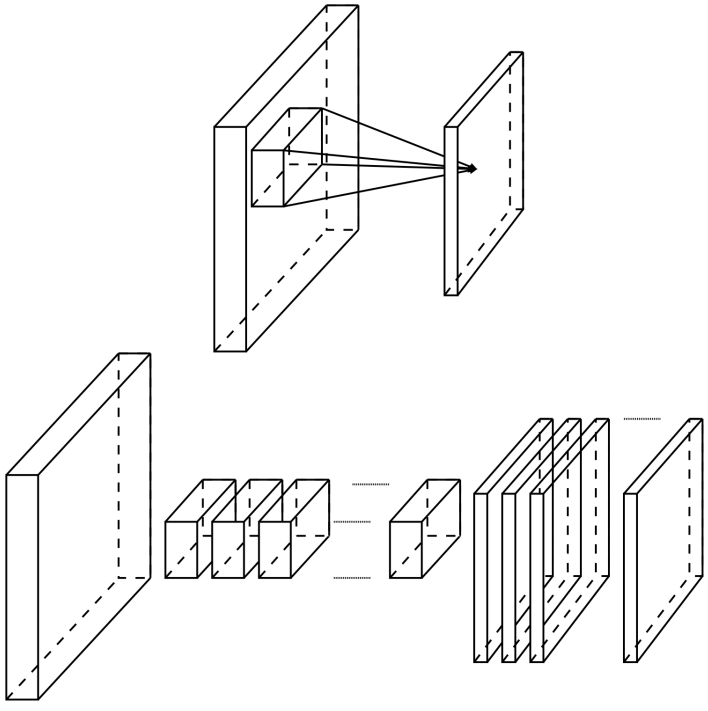
May 9, 2016

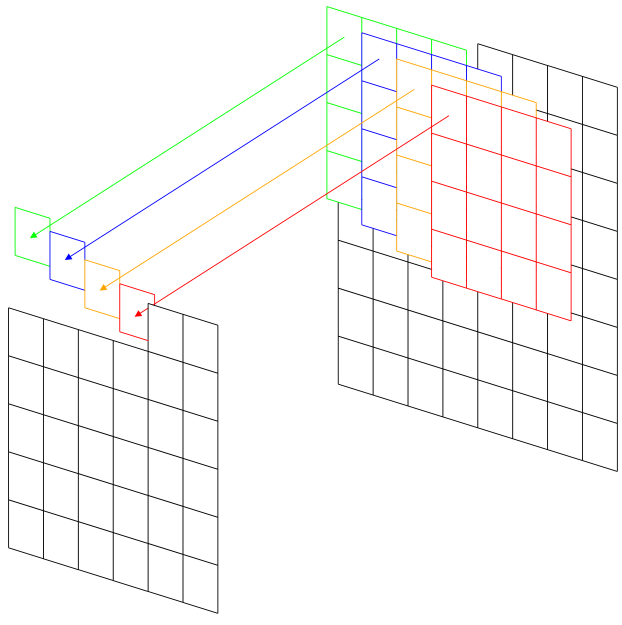
Fully Convolutional Network[2]



Convolution ReLU Pooling Deconvolution

Convolution[1]

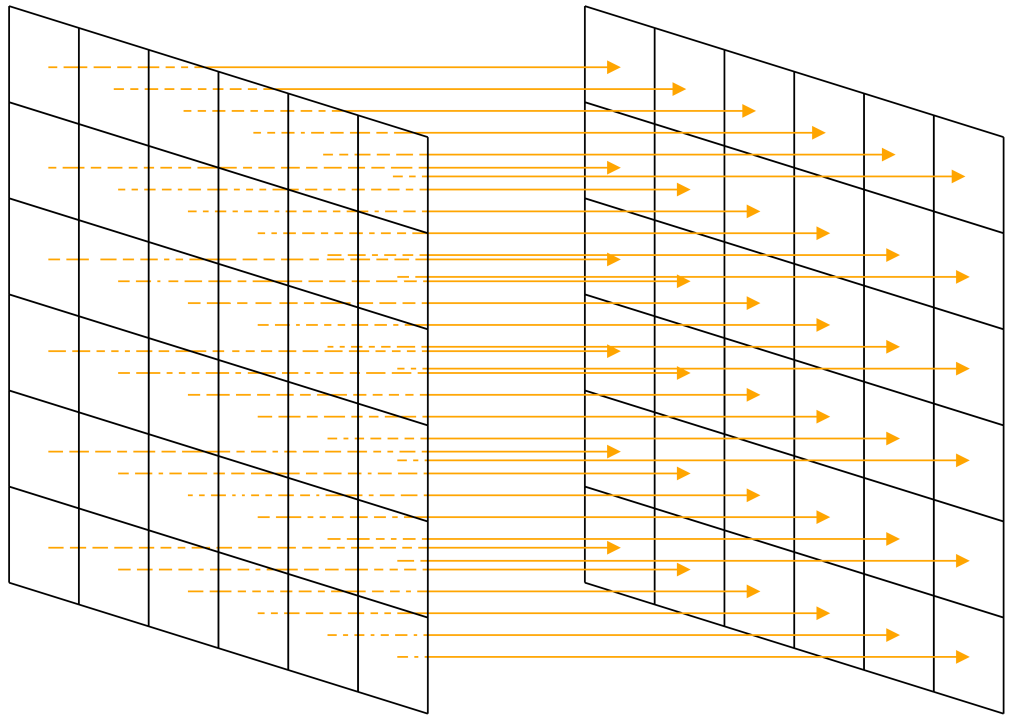




```
/* Algorithm */
output(i,j,k,l)=bias(0,0,0,k)+
  sum(kernel(r.x,r.y,r.z,k)*clamped_input(i*stride+r.x-pad,j*stride+r.y-pad,r.z,l));

/* Schedule v5 */
Var fused;
output.fuse(k,l,fused).parallel(fused);     // Parallelize over channels & batch
output.vectorize(i,16);                     // Compute in SIMD fashion
output.compute_root();                      // Compute output before next layer
clamped_input.store_root().compute_root(); // Compute once and store for lookup
```

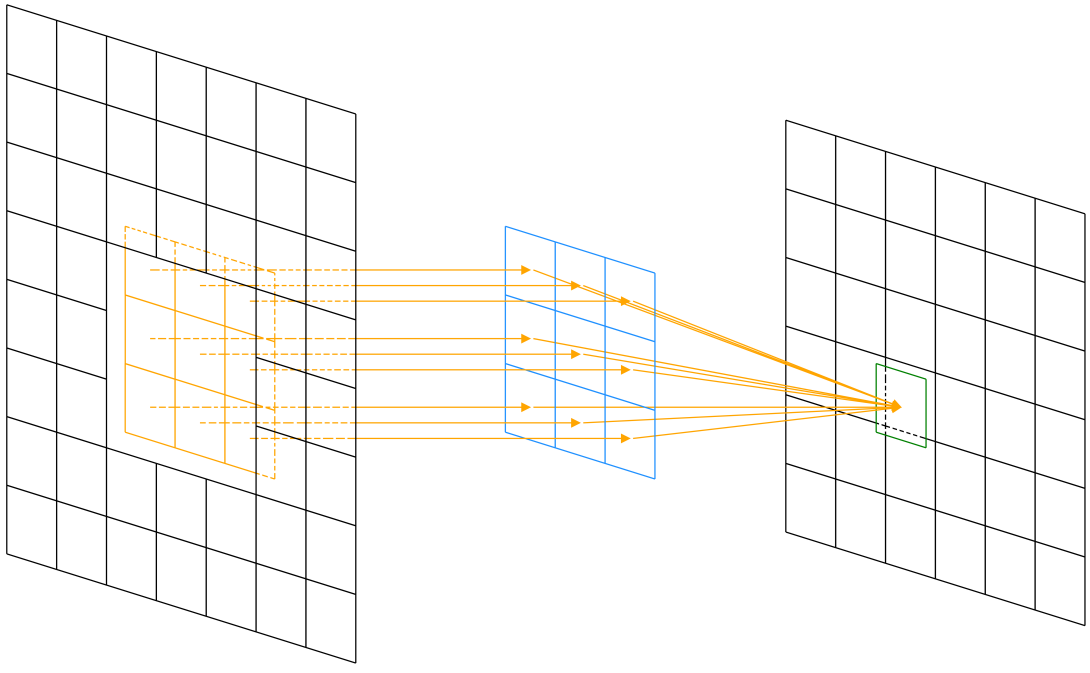
ReLU



```
/* Algorithm */
output(i,j,k,l)=max(0,input(i,j,k,l))+negative_slope*min(0,input(i,j,k,l));

/* Schedule v5 */
Var fused;
output.fuse(k,l,fused).parallel(fused);     // Parallelize over channels & batch
output.vectorize(i,16);                     // Compute in SIMD fashion
output.compute_root();                      // Compute output before next layer
```

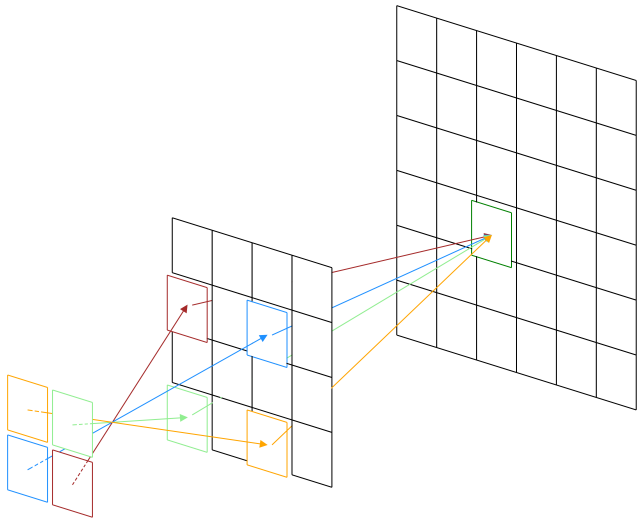
Pooling

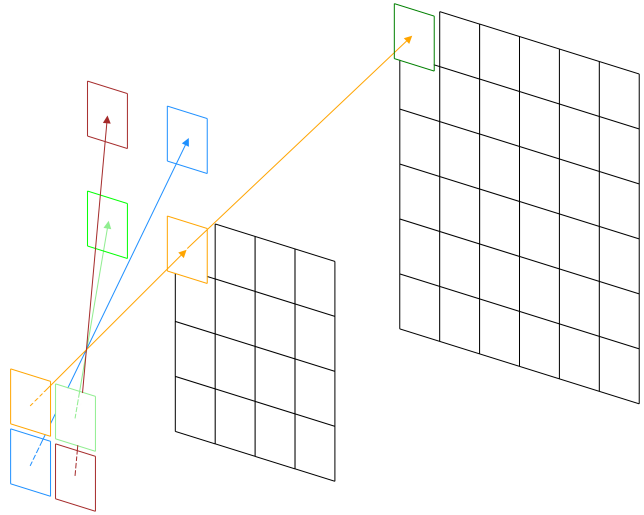


```
/* Algorithm */
output(i,j,k,l)=maximum(input(i*stride + r.x, j*stride + r.y, k, l));

/* Schedule v5 */
Var fused;
output.fuse(k,l,fused).parallel(fused);     // Parallelize over channels & batch
output.vectorize(i,16);                     // Compute in SIMD fashion
output.compute_root();                      // Compute output before next layer
```

Deconvolution[3]

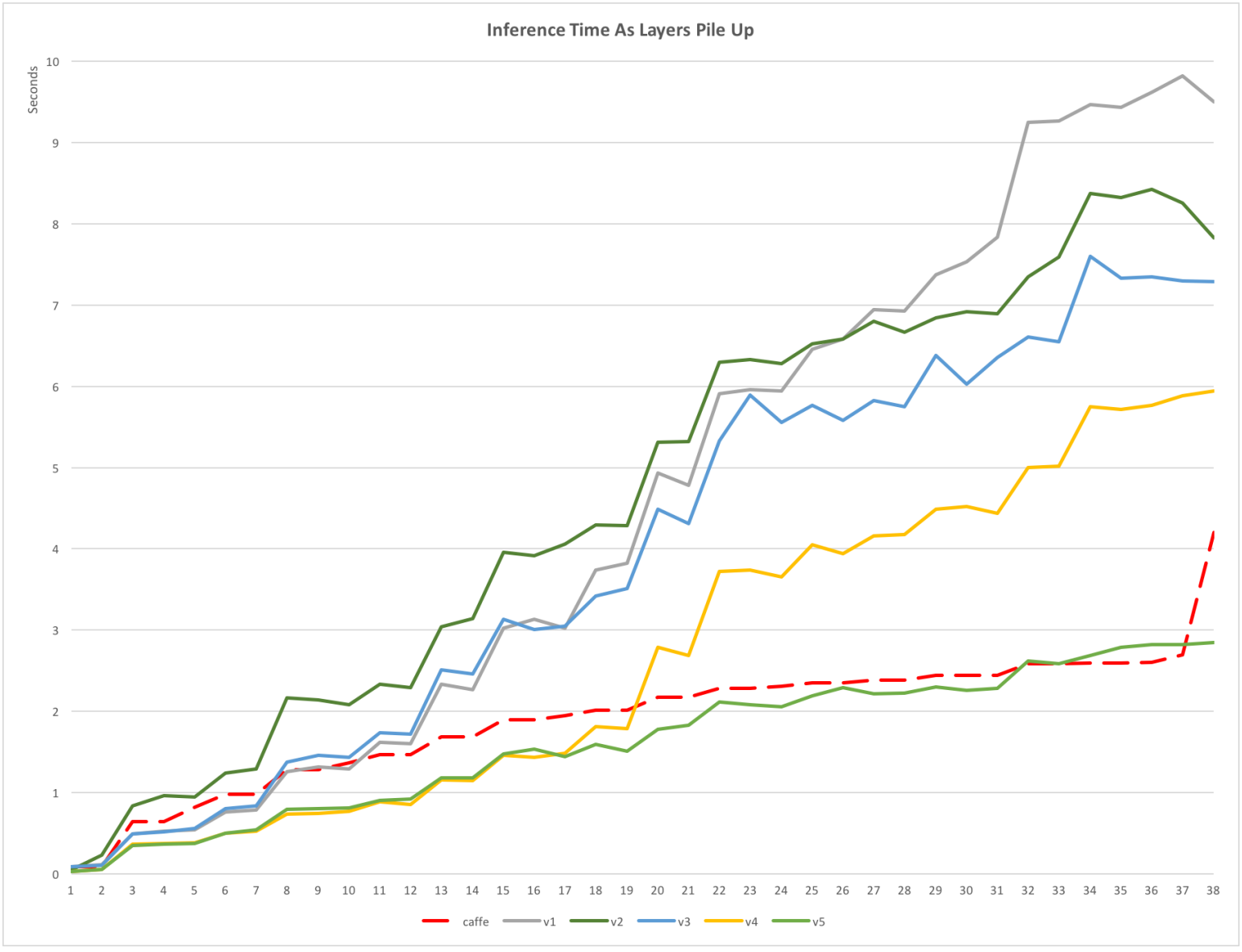




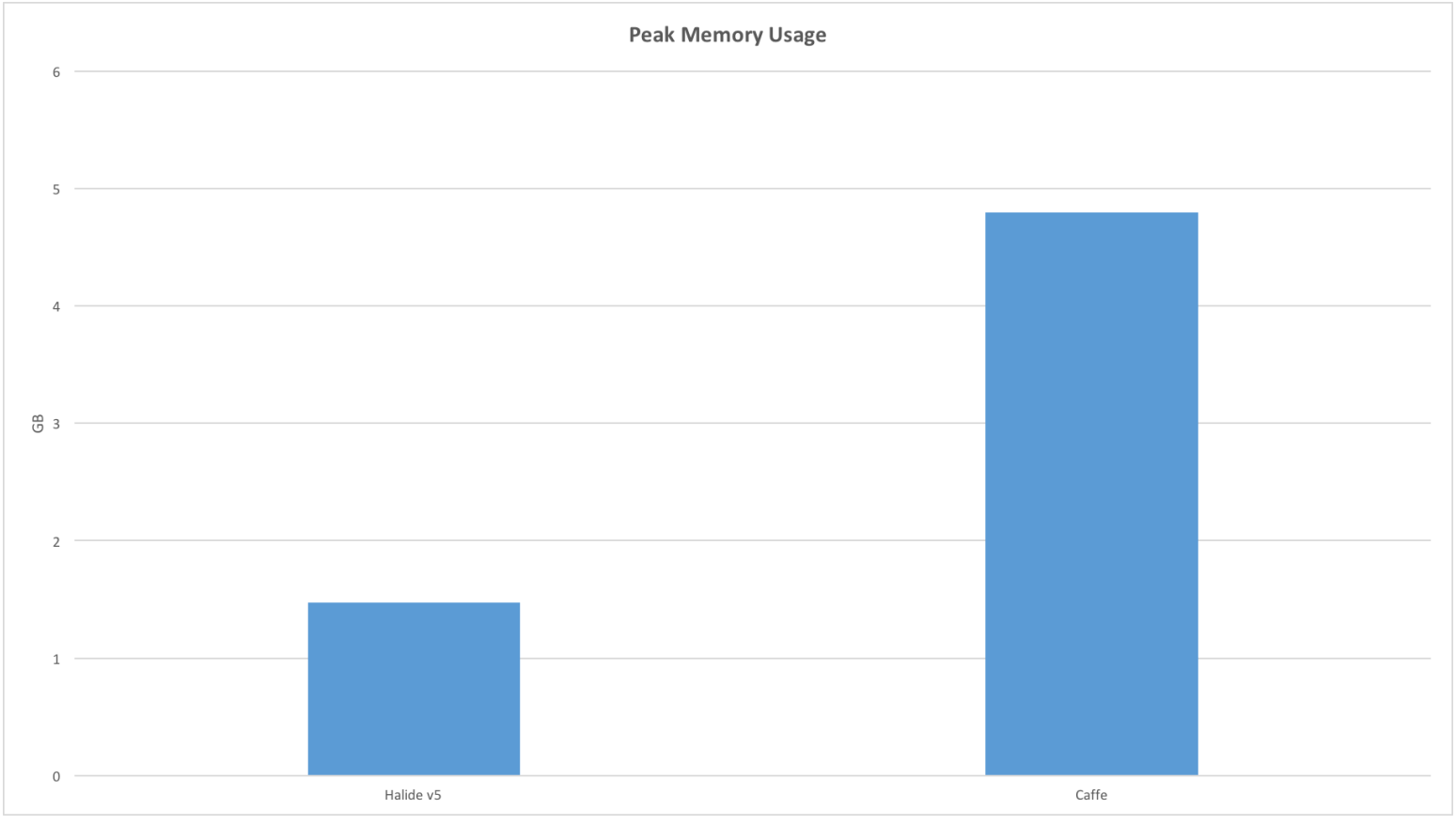
```
/* Algorithm */
int kernel_step=kernel_size/stride;
RDom r(0,kernel_step,0,kernel_step,0,input_channels);
output(i,j,k,l)=sum(kernel(r.x*stride+i%stride,r.y*stride+j%stride,r.z,k)*
  clamped_input(i/stride-r.x,j/stride-r.y,r.z,l));

/* Schedule v5 */
Var fused;
output.fuse(k,l,fused).parallel(fused);     // Parallelize over channels & batch
output.vectorize(i,16);                     // Compute in SIMD fashion
output.compute_root();                      // Compute output before next layer
clamped_input.store_root().compute_root(); // Compute once and store for lookup
```

Results : Inference Time



Results : Memory Usage



Segmentation Results

References

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton.

Imagenet classification with deep convolutional neural networks.

In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Jonathan Long, Evan Shelhamer, and Trevor Darrell.

Fully convolutional networks for semantic segmentation.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

Andrea Vedaldi and Karel Lenc.

Matconvnet - convolutional neural networks for MATLAB.

CoRR, abs/1412.4564, 2014.