Carnegie Mellon University
Language Technologies Institute

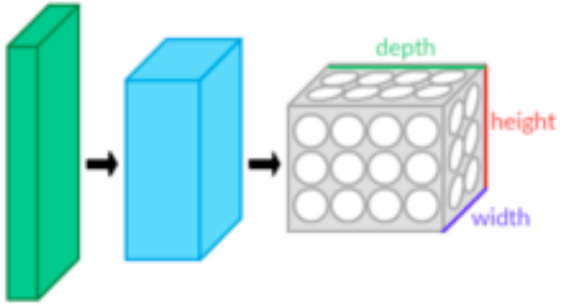
# Architecture-specific Tricks I

## CNNs

Yonatan Bisk & Emma Strubell

# CNNs

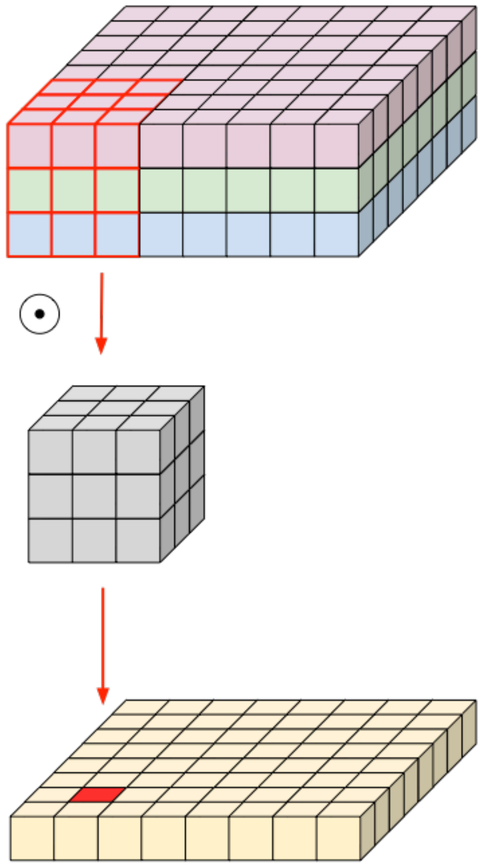
- Receptive Field
- Stride
- Dilation
- Depth-wise?
- Channels?
- Resolution?
- 1D, 2D, 3D interactions?




Which parameters and inputs need to interact?

How many do you actually need?

# Depthwise Convolutions



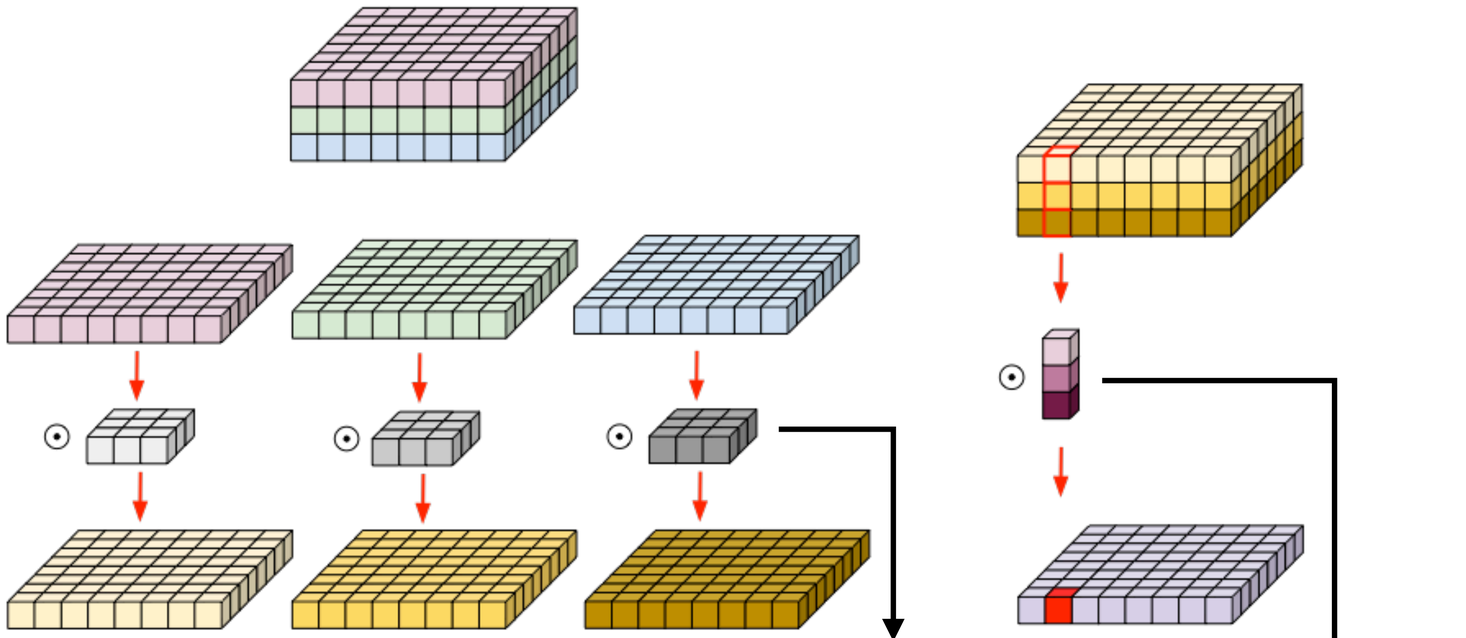
Parameters $H * W * in_C * out_C$

Compute Layer 1

$$I = 64 \times 64$$
$$in_c = 3$$
$$out_c = 16$$
$$H, W = 3,3$$
$$\Rightarrow 3 * 3 * 3 * 64 * 64 * 16$$
$$\Rightarrow 1{,}769{,}472$$

Super awesome graphics
https://eli.thegreenplace.net/2018/depthwise-separable-convolutions-for-machine-learning/

# Depthwise Separable Convolutions



$H * W * in_C$ $in_C * out_C$

Parameters

$H * W * in_C$ $in_C * out_C$

Compute Layer 1

$I = 64 \times 64$

$in_c = 3$

$out_c = 16$

$H, W = 3,3$

$\Rightarrow 3 * 3 * 3 * 64 * 64 + 64 * 64 * 3 * 16$

$\Rightarrow 307{,}200$



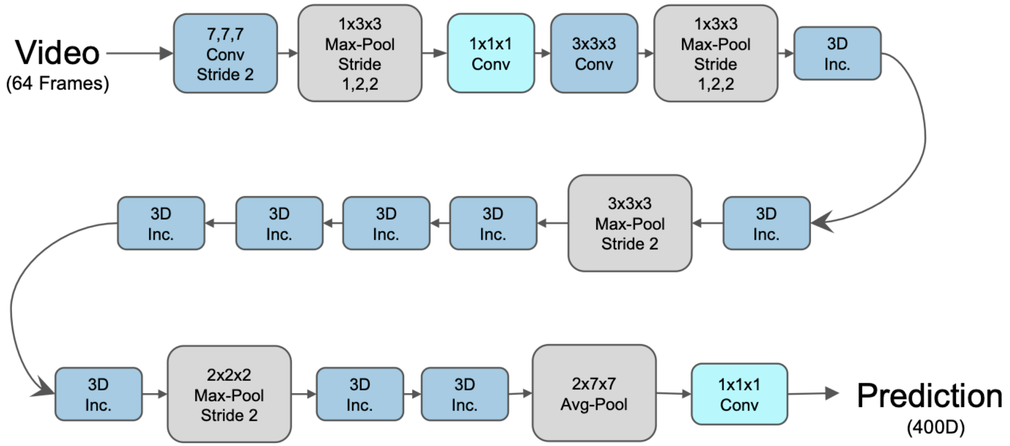
Super awesome graphics
https://eli.thegreenplace.net/2018/depthwise-separable-convolutions-for-machine-learning/

# Depthwise Separable Convolutions

| | Dense Convolution | Depthwise Separable Convolution |
|---|---|---|
| Parameters | $H * W * in_C * out_C$ | $H * W * in_C \quad in_C * out_C$ |
| Setting | $I = 64 \times 64 \quad in_c = 3 \quad out_c = 16 \quad H, W = 3,3$ | |
| Compute | $\Rightarrow 3 * 3 * 3 * 64 * 64 * 16$<br>$\Rightarrow 1{,}769{,}472$ | $\Rightarrow 3 * 3 * 3 * 64 * 64 + 64 * 64 * 3 * 16$<br>$\Rightarrow 307{,}200$ |

What did we lose?

# Video Features

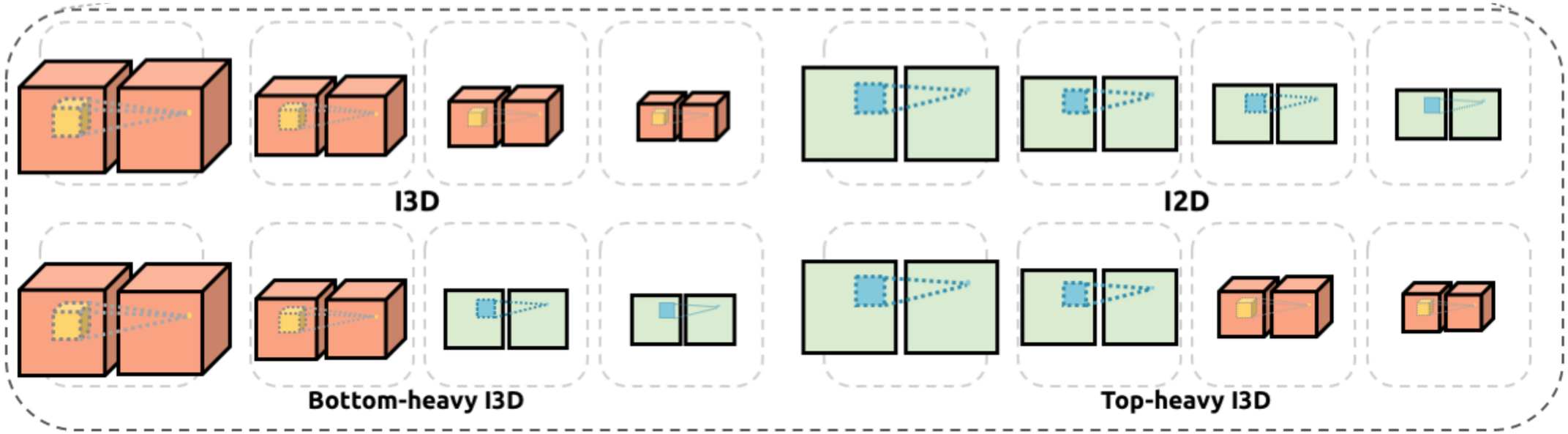



(a) I3D

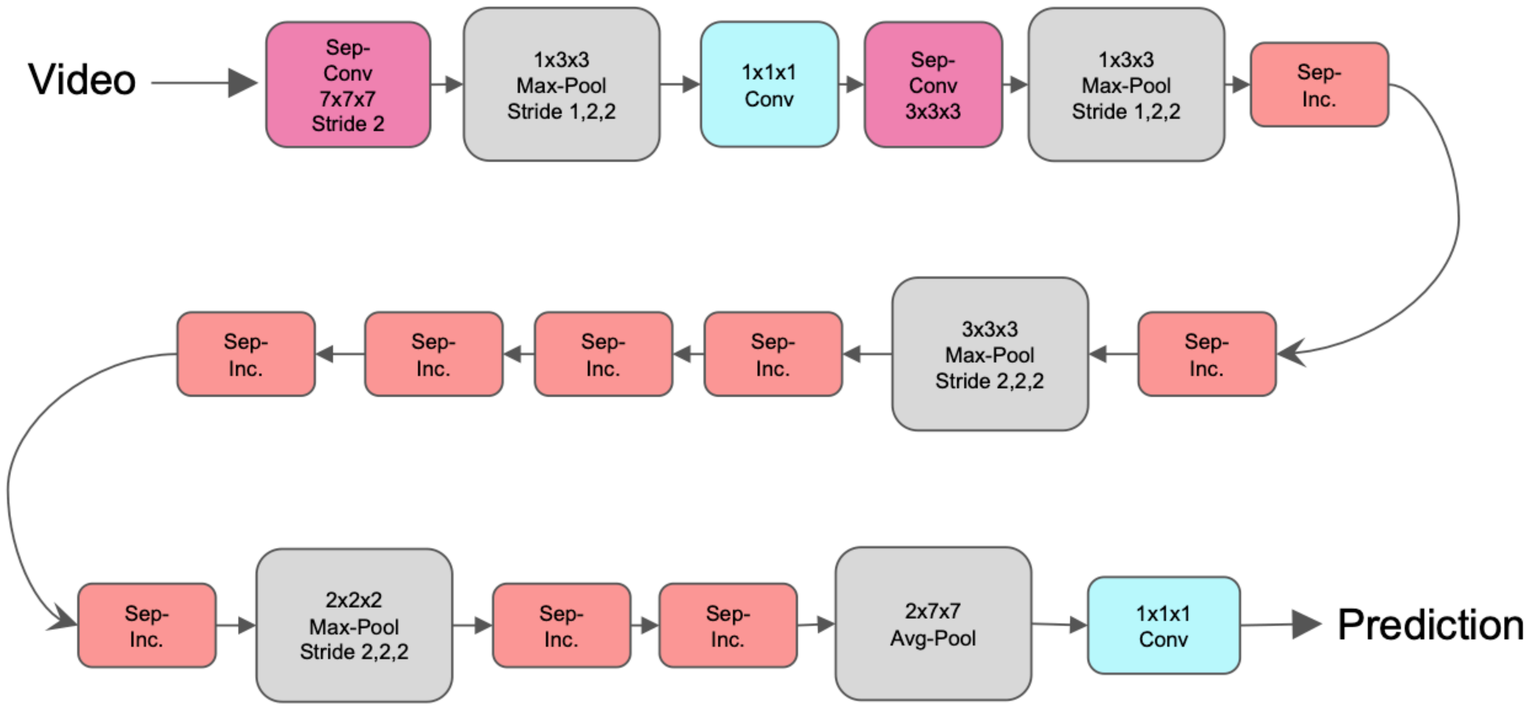


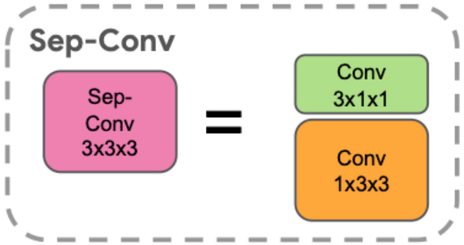
https://arxiv.org/abs/1712.04851

# I3D, I2D, …



# S3D — Separable 3D CNN




$k_t \times k \times k$

$1 \times k \times k$

$k_t \times 1 \times 1$

$k_t$ Width of the filter in time

https://arxiv.org/abs/1712.04851

# Blocks

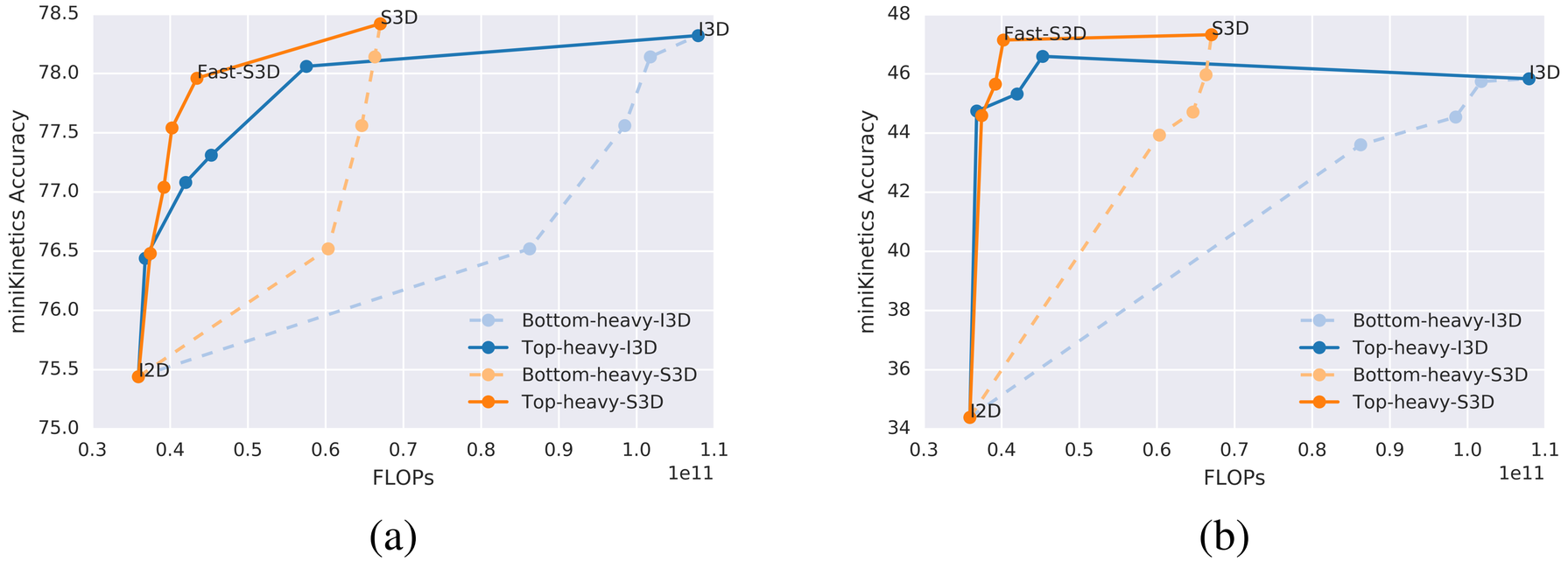



**Fig. 4.** Accuracy vs number of FLOPS needed to perform inference on 64 RGB frames. Left: Mini-Kinetics-200 dataset. Right: Something-something dataset. Solid lines denote top-heavy models, dotted lines denote bottom-heavy models. Orange denotes spatial and temporal separable 3D convolutions, blue denotes full 3D convolutions.



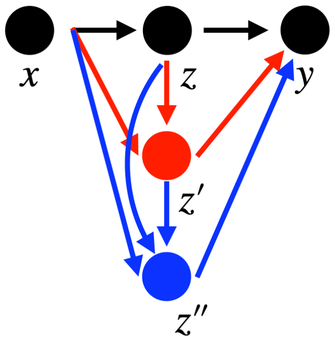
https://arxiv.org/abs/1712.04851

# More is Less? DenseNets vs ResNets

Remember this?

**Cascade Correlation**

Fahlman and Lebiere, N(eur)IPS 1989




What is the atomic “unit”?

A new layer in the hierarchy

What if feature-maps?

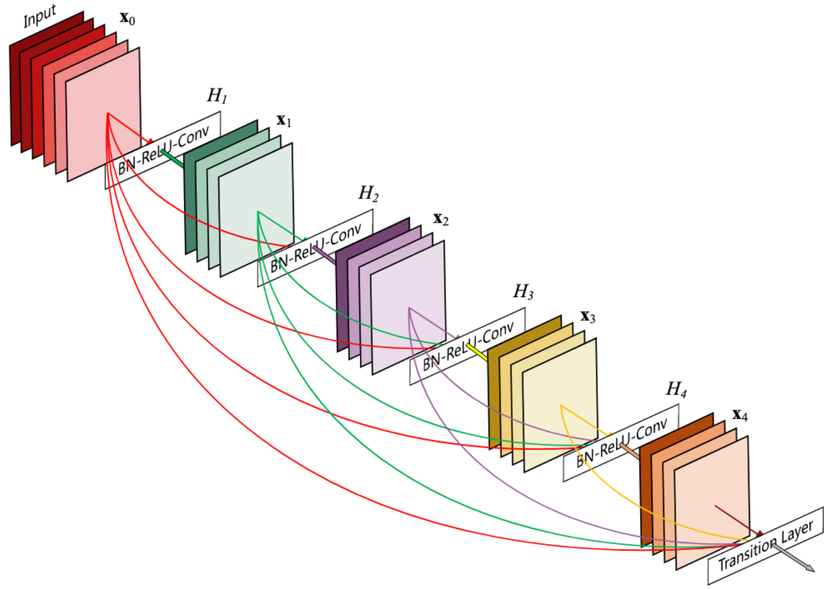



**Figure 1:** A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.



https://arxiv.org/abs/1608.06993

# More is Less? DenseNets vs ResNets

| layer name | output size | 18-layer | 34-layer | 50-layer | 101-layer | 152-layer |
|---|---|---|---|---|---|---|
| conv1 | 112×112 | 7×7, 64, stride 2 | | | | |
| conv2_x | 56×56 | 3×3 max pool, stride 2 | | | | |
| | | [3×3, 64; 3×3, 64] ×2 | [3×3, 64; 3×3, 64] ×3 | [1×1, 64; 3×3, 64; 1×1, 256] ×3 | [1×1, 64; 3×3, 64; 1×1, 256] ×3 | [1×1, 64; 3×3, 64; 1×1, 256] ×3 |
| conv3_x | 28×28 | [3×3, 128; 3×3, 128] ×2 | [3×3, 128; 3×3, 128] ×4 | [1×1, 128; 3×3, 128; 1×1, 512] ×4 | [1×1, 128; 3×3, 128; 1×1, 512] ×4 | [1×1, 128; 3×3, 128; 1×1, 512] ×8 |
| conv4_x | 14×14 | [3×3, 256; 3×3, 256] ×2 | [3×3, 256; 3×3, 256] ×6 | [1×1, 256; 3×3, 256; 1×1, 1024] ×6 | [1×1, 256; 3×3, 256; 1×1, 1024] ×23 | [1×1, 256; 3×3, 256; 1×1, 1024] ×36 |
| conv5_x | 7×7 | [3×3, 512; 3×3, 512] ×2 | [3×3, 512; 3×3, 512] ×3 | [1×1, 512; 3×3, 512; 1×1, 2048] ×3 | [1×1, 512; 3×3, 512; 1×1, 2048] ×3 | [1×1, 512; 3×3, 512; 1×1, 2048] ×3 |
| | 1×1 | average pool, 1000-d fc, softmax | | | | |
| FLOPs | | $1.8\times10^9$ | $3.6\times10^9$ | $3.8\times10^9$ | $7.6\times10^9$ | $11.3\times10^9$ |

ResNet

| Layers | Output Size | DenseNet-121 | DenseNet-169 | DenseNet-201 | DenseNet-264 |
|---|---|---|---|---|---|
| Convolution | 112 × 112 | 7 × 7 conv, stride 2 | | | |
| Pooling | 56 × 56 | 3 × 3 max pool, stride 2 | | | |
| Dense Block (1) | 56 × 56 | [1 × 1 conv; 3 × 3 conv] × 6 | [1 × 1 conv; 3 × 3 conv] × 6 | [1 × 1 conv; 3 × 3 conv] × 6 | [1 × 1 conv; 3 × 3 conv] × 6 |
| Transition Layer (1) | 56 × 56 | 1 × 1 conv | | | |
| | 28 × 28 | 2 × 2 average pool, stride 2 | | | |
| Dense Block (2) | 28 × 28 | [1 × 1 conv; 3 × 3 conv] × 12 | [1 × 1 conv; 3 × 3 conv] × 12 | [1 × 1 conv; 3 × 3 conv] × 12 | [1 × 1 conv; 3 × 3 conv] × 12 |
| Transition Layer (2) | 28 × 28 | 1 × 1 conv | | | |
| | 14 × 14 | 2 × 2 average pool, stride 2 | | | |
| Dense Block (3) | 14 × 14 | [1 × 1 conv; 3 × 3 conv] × 24 | [1 × 1 conv; 3 × 3 conv] × 32 | [1 × 1 conv; 3 × 3 conv] × 48 | [1 × 1 conv; 3 × 3 conv] × 64 |
| Transition Layer (3) | 14 × 14 | 1 × 1 conv | | | |
| | 7 × 7 | 2 × 2 average pool, stride 2 | | | |
| Dense Block (4) | 7 × 7 | [1 × 1 conv; 3 × 3 conv] × 16 | [1 × 1 conv; 3 × 3 conv] × 32 | [1 × 1 conv; 3 × 3 conv] × 32 | [1 × 1 conv; 3 × 3 conv] × 48 |
| Classification Layer | 1 × 1 | 7 × 7 global average pool | | | |
| | | 1000D fully-connected, softmax | | | |

DenseNet



https://arxiv.org/abs/1608.06993

# More is Less? DenseNets vs ResNets

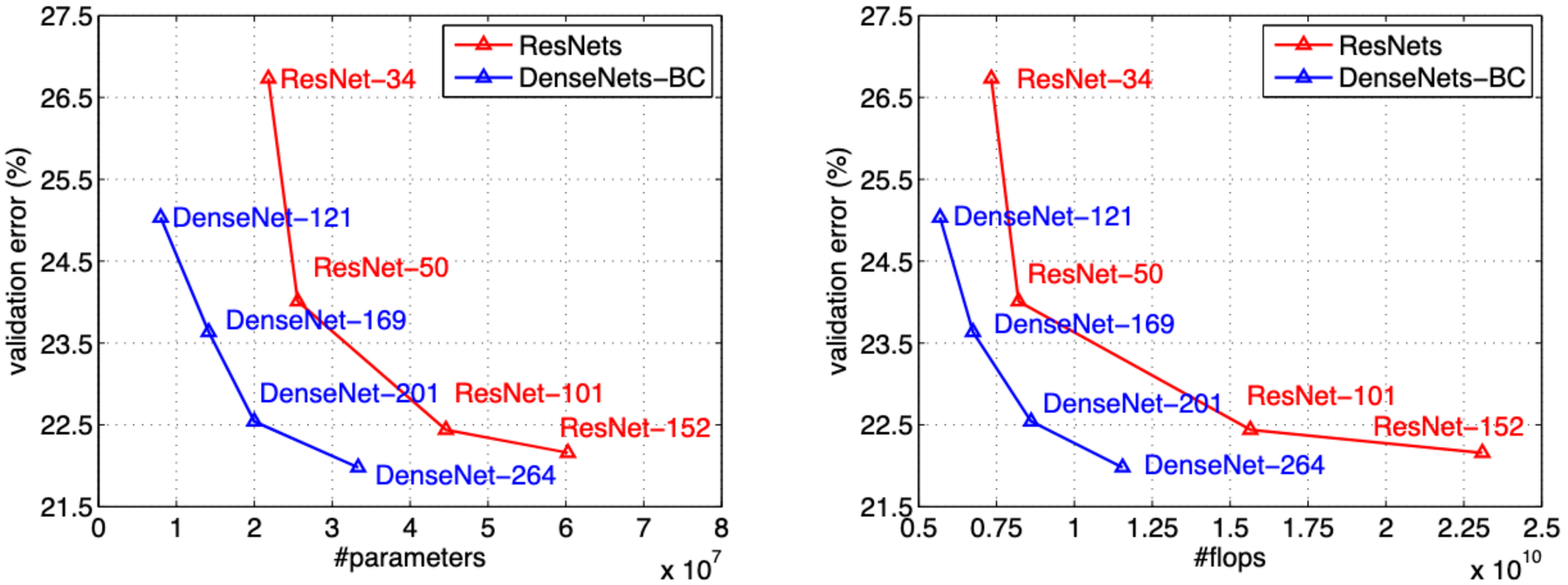



**Figure 3:** Comparison of the DenseNets and ResNets top-1 error rates (single-crop testing) on the ImageNet validation dataset as a function of learned parameters (*left*) and FLOPs during test-time (*right*).



https://arxiv.org/abs/1608.06993

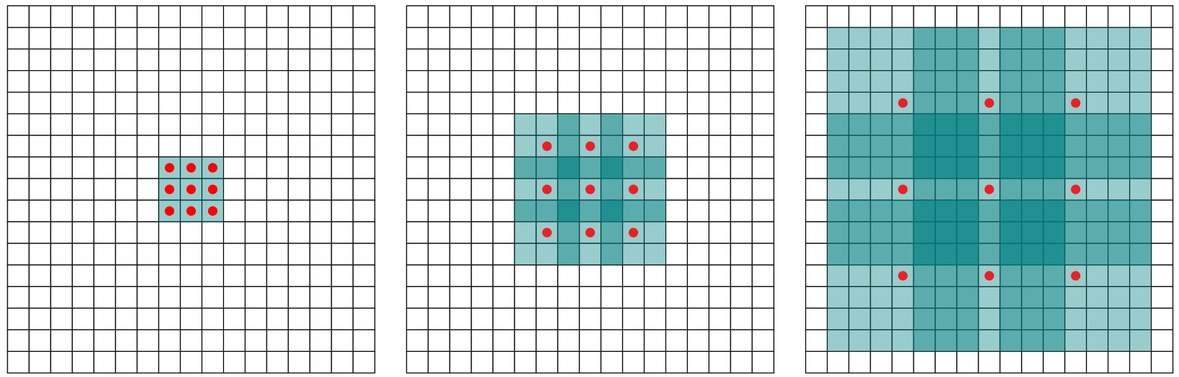
# Dilated Convolutions

Increase receptive field size exponentially as # parameters grows linearly.

Add parameter δ: dilation size (# inputs to skip); Increase δ exponentially at each layer.

## In computer vision

- Image segmentation: Yu & Koltun (2016)




## In speech synthesis

- WaveNet (van den Oord et al. 2016)

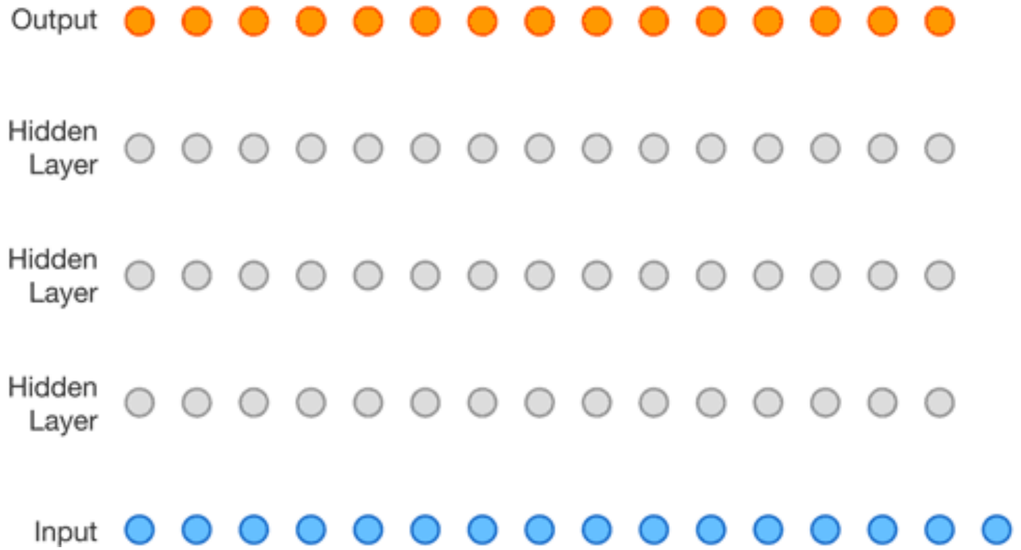
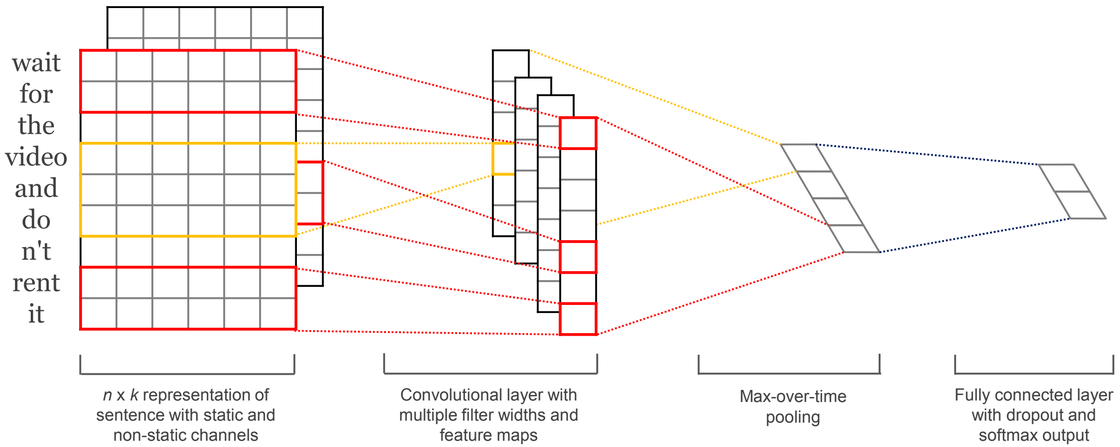




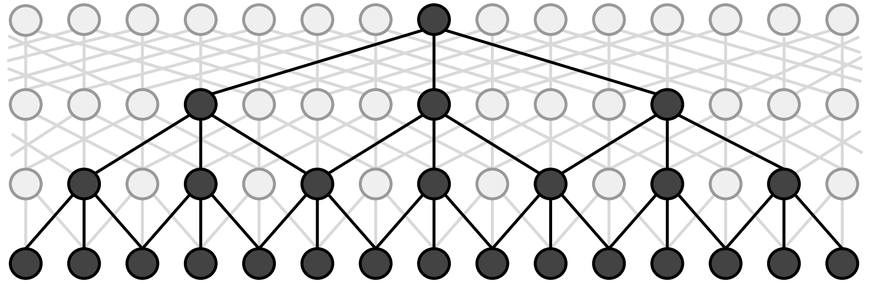

Figure: https://deepmind.com/blog/article/wavenet-generative-model-raw-audio

# CNNs for NLP?

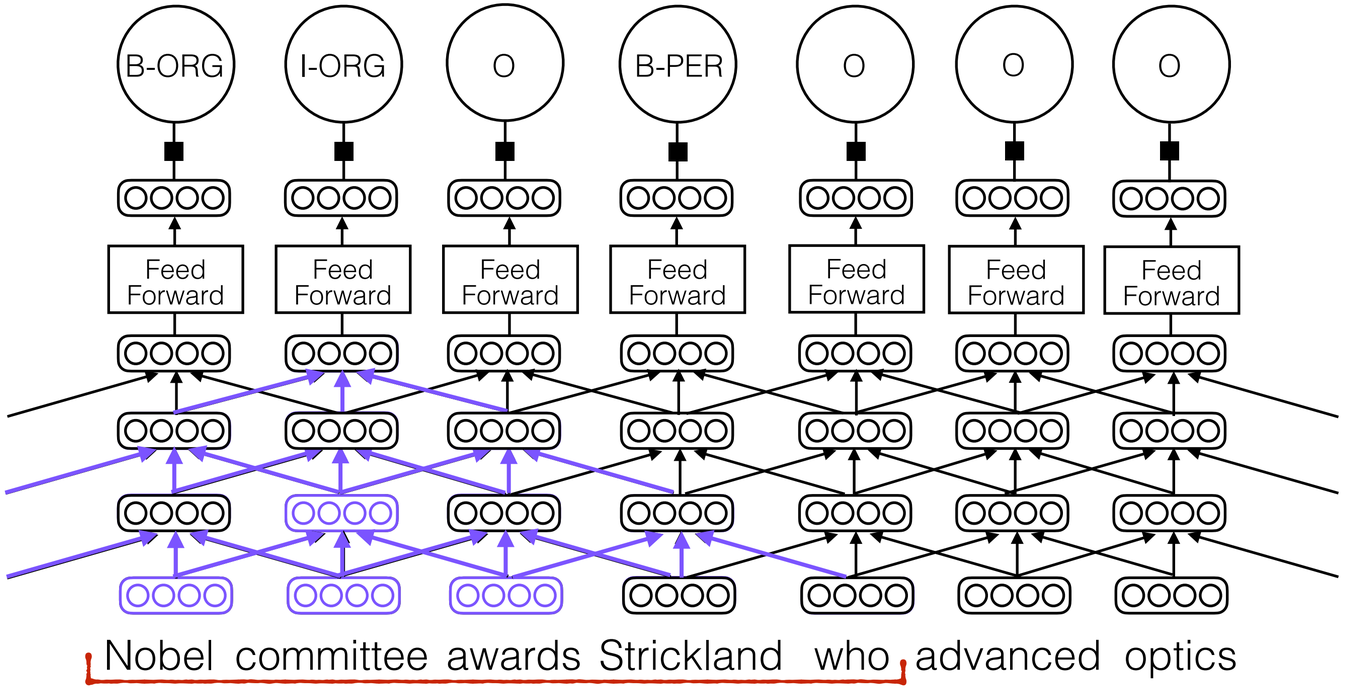
- For sentence / sequence classification: Yes! (Kim 2017)



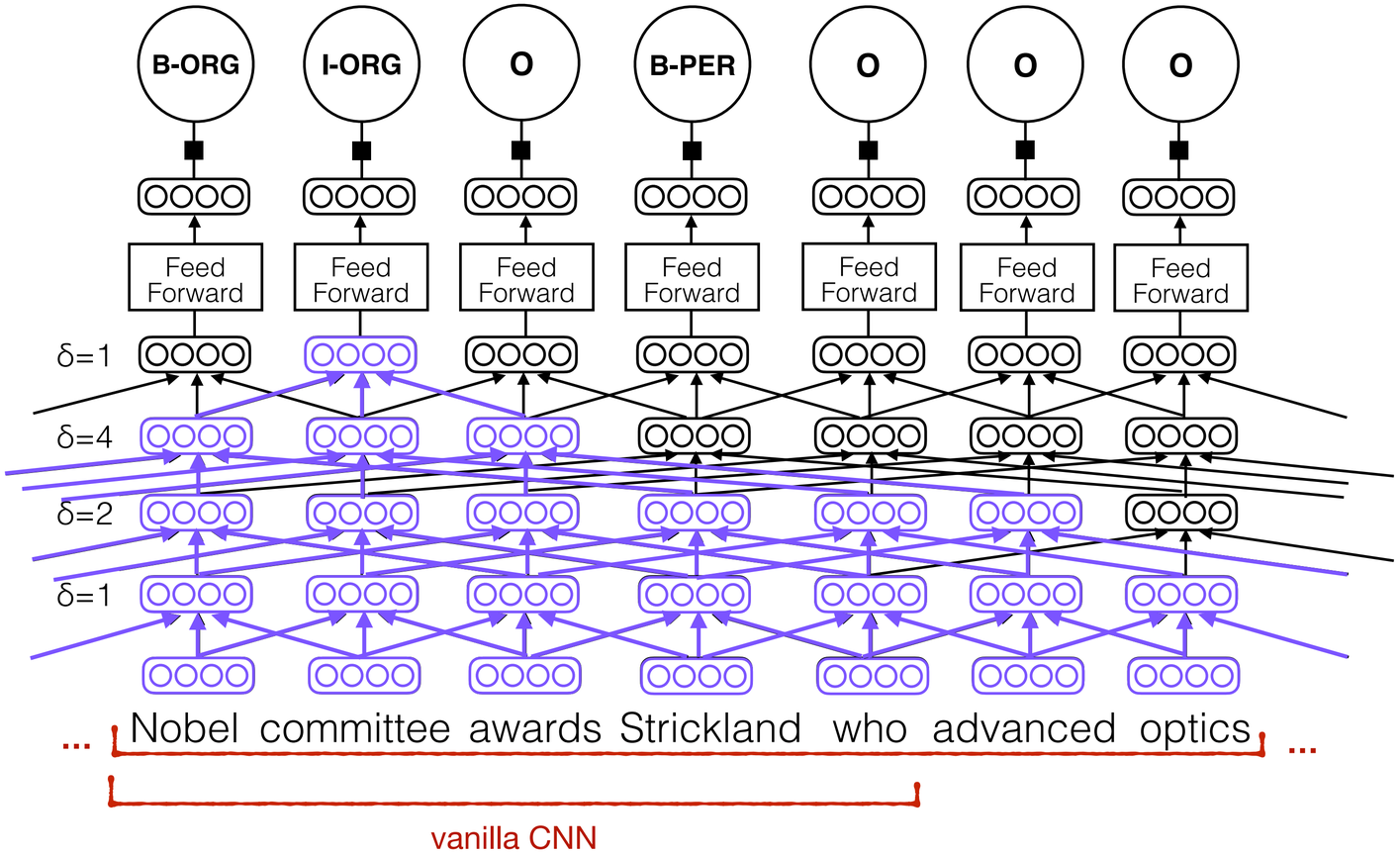

- For sequence labelling / structured outputs? Maybe!
  - Try dilated convolutions (Strubell et al. 2017).



# CNNs for NLP?


B-ORG
I-ORG
O
B-PER
O
O
O
Feed Forward
Feed Forward
Feed Forward
Feed Forward
Feed Forward
Feed Forward
Feed Forward
Nobel committee awards Strickland who advanced optics

# Dilated CNNs for NLP

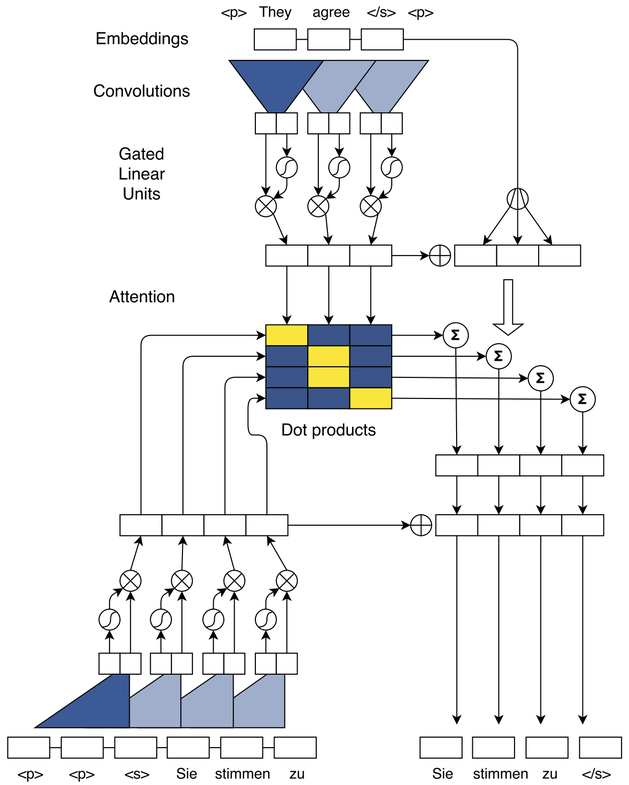


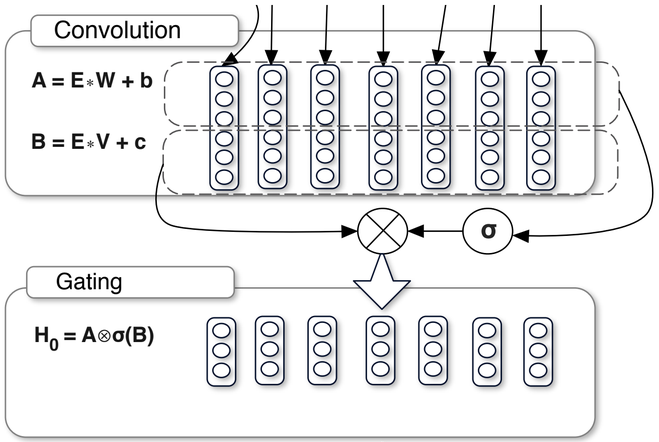
# CNNs for long-range dependencies?

May 8, 2017 Citations (Oct 2023): 3,696

## Convolutional Sequence to Sequence Learning

**Jonas Gehring** [1] **Michael Auli** [1] **David Grangier** [1] **Denis Yarats** [1] **Yann N. Dauphin** [1]




June 12, 2017 Citations: 91,974

## Attention Is All You Need

**Ashish Vaswani*** Google Brain avaswani@google.com

**Noam Shazeer*** Google Brain noam@google.com

**Niki Parmar*** Google Research nikip@google.com

**Jakob Uszkoreit*** Google Research usz@google.com

**Llion Jones*** Google Research llion@google.com

**Aidan N. Gomez*** † University of Toronto aidan@cs.toronto.edu

**Łukasz Kaiser*** Google Brain lukaszkaiser@google.com

**Illia Polosukhin*** ‡ illia.polosukhin@gmail.com

# CNNs for long-range dependencies: Now

## S4

**Algorithm 1** S4 Convolution Kernel (Sketch)

**Input:** S4 parameters $\boldsymbol{\Lambda}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{C}^N$ and step size $\Delta$
**Output:** SSM convolution kernel $\overline{\boldsymbol{K}} = \mathcal{K}_L(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \overline{\boldsymbol{C}})$ for $\boldsymbol{A} = \boldsymbol{\Lambda} - \boldsymbol{P}\boldsymbol{Q}^*$ (equation (5))

1: $\widetilde{\boldsymbol{C}} \leftarrow \left(\boldsymbol{I} - \overline{\boldsymbol{A}}^L\right)^* \overline{\boldsymbol{C}}$ ▷ Truncate SSM generating function (SSMGF) to length $L$

2: $\begin{bmatrix} k_{00}(\omega) & k_{01}(\omega) \\ k_{10}(\omega) & k_{11}(\omega) \end{bmatrix} \leftarrow \left[\widetilde{\boldsymbol{C}}\ \boldsymbol{Q}\right]^* \left(\frac{2}{\Delta}\frac{1-\omega}{1+\omega} - \boldsymbol{\Lambda}\right)^{-1} [\boldsymbol{B}\ \boldsymbol{P}]$ ▷ Black-box Cauchy kernel

3: $\hat{\boldsymbol{K}}(\omega) \leftarrow \frac{2}{1+\omega}\left[k_{00}(\omega) - k_{01}(\omega)(1 + k_{11}(\omega))^{-1}k_{10}(\omega)\right]$ ▷ Woodbury Identity

4: $\hat{\boldsymbol{K}} = \{\hat{\boldsymbol{K}}(\omega) : \omega = \exp(2\pi i \frac{k}{L})\}$ ▷ Evaluate SSMGF at all roots of unity $\omega \in \Omega_L$

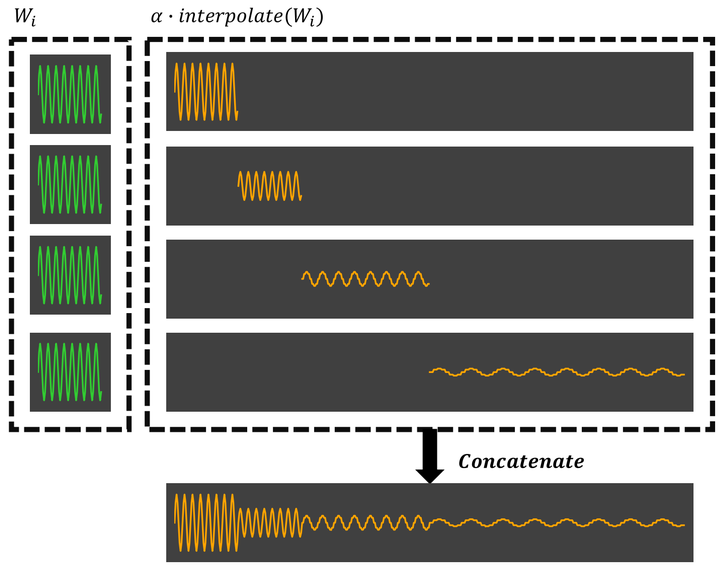
5: $\overline{\boldsymbol{K}} \leftarrow \mathsf{iFFT}(\hat{\boldsymbol{K}})$ ▷ Inverse Fourier Transform

| | Convolution[3] | Recurrence | Attention | S4 |
|---|---|---|---|---|
| Parameters | $LH$ | $\boldsymbol{H^2}$ | $\boldsymbol{H^2}$ | $\boldsymbol{H^2}$ |
| Training | $\boldsymbol{\tilde{L}H(B+H)}$ | $BLH^2$ | $B(L^2H + LH^2)$ | $\boldsymbol{BH(\tilde{H}+\tilde{L}) + B\tilde{L}H}$ |
| Space | $\boldsymbol{BLH}$ | $\boldsymbol{BLH}$ | $B(L^2 + HL)$ | $\boldsymbol{BLH}$ |
| Parallel | **Yes** | No | **Yes** | **Yes** |
| Inference | $LH^2$ | $\boldsymbol{H^2}$ | $L^2H + H^2L$ | $\boldsymbol{H^2}$ |

| Model | ListOps | Text | Retrieval | Image | Pathfinder | Path-X | Avg. |
|---|---|---|---|---|---|---|---|
| Transformer | 36.37 | 64.27 | 57.46 | 42.44 | 71.40 | ✗ | 54.39 |
| Sparse Trans. | 17.07 | 63.58 | 59.59 | 44.24 | 71.71 | ✗ | 51.24 |
| Linformer | 35.70 | 53.94 | 52.27 | 38.56 | 76.34 | ✗ | 51.36 |
| Reformer | 37.27 | 56.10 | 53.40 | 38.07 | 68.50 | ✗ | 50.67 |
| BigBird | 36.05 | 64.02 | 59.29 | 40.83 | 74.87 | ✗ | 55.01 |
| S4 (original) | 58.35 | 76.02 | 87.09 | 87.26 | 86.05 | 88.10 | 80.48 |
| S4 [Gu et al., 2022b] | 59.60 | 86.82 | **90.90** | **88.65** | 94.20 | 96.35 | 86.09 |

## SGConv

$$\mathrm{Cat}(S) = \frac{1}{Z}\left[k_0, k_1, \cdots, k_{N-1}\right], \text{ where } k_i = \alpha^i \mathrm{Upsample}_{2^{\max[i-1,0]}d}(\mathbf{w}_i).$$




| | | 256 | 512 | 1024 | 2048 | 3072 |
|---|---|---|---|---|---|---|
| Attn. Block | Inf. (ms/batch) | **2.6** | 7.3 | 23.2 | 91.7 | ✗ |
| | Mem. (GB) | **2.6** | 3.9 | 7.9 | 23.9 | OOM |
| SGConv Block | Inf. (ms/batch) | 2.7 | **5.4** | **10.9** | **21.8** | **43.6** |
| | Mem. (GB) | **2.6** | **3.4** | **5.2** | **8.7** | **15.7** |



Yuhong Li, Tianle Cai, Yi Zhang, Deming Chen, and Debadeepta Dey. What Makes Convolutional Models Great on Long Sequence Modeling? arXiv:2210.09298, 2022.
Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. ICLR, 2022.

# CNNs as X(N)ORs

- **X(N)OR-Nets:** Binary quantized (0-1) convolutional layers w/ activations computed using just X(N)OR and popcount

```
xor(00100110, 01100000) = 01000110
popcount(01000110) = 3
```

- Dot product of two binary matrices:

```
a = xnor(x, y) = not(xor(x, y))
b = popcount(a)
c = len(a)
dot(x, y) = 2b - c
```

remove NOT operation →

```
a = xor(x, y)
b = popcount(a)
c = len(a)
xor-dot(x, y) = c - 2b
```

M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. ECCV 2016.
S. Zhu, L. H. K. Duong, W. Liu. XOR-Net: An Efficient Computation Pipeline for Binary Neural Network Inference on Edge Devices. ICPADS 2020.

# CNNs as FFTs

- The convolution theorem:

convolution

Hadamard product (pointwise)

$$\mathcal{F}(\kappa * u) = \mathcal{F}(\kappa) \odot \mathcal{F}(u)$$

- fbfft: $\mathcal{O}(Sff'n^2 + (Sf + ff' + Sf')n^2 \log n)$ procedure in lieu of the original $\mathcal{O}(Sff'n^2k^2)$ (Vasilache et al. 2015)
- Winograd algorithm: same idea, fewer FLOPs (no complex #s) (Lavin & Gray 2015)
- FFT convolutions are faster than Winograd on modern CPUs (Zlateski et al. 2018)



N. Vasilache, J. Johnson, M. Mathieu, S. Chintala, S. Piantino, Y. LeCun. Fast Convolutional Nets With fbfft: A GPU Performance Evaluation. ICLR 2015.
H. Pratt, B. Williams, F. Coenen, and Y. Zheng. FCNN: Fourier Convolutional Neural Networks. ECML-PKDD 2017.

What about Distillation? Quantization? Pruning? ….

Yes, but also think about data flow and necessary interactions