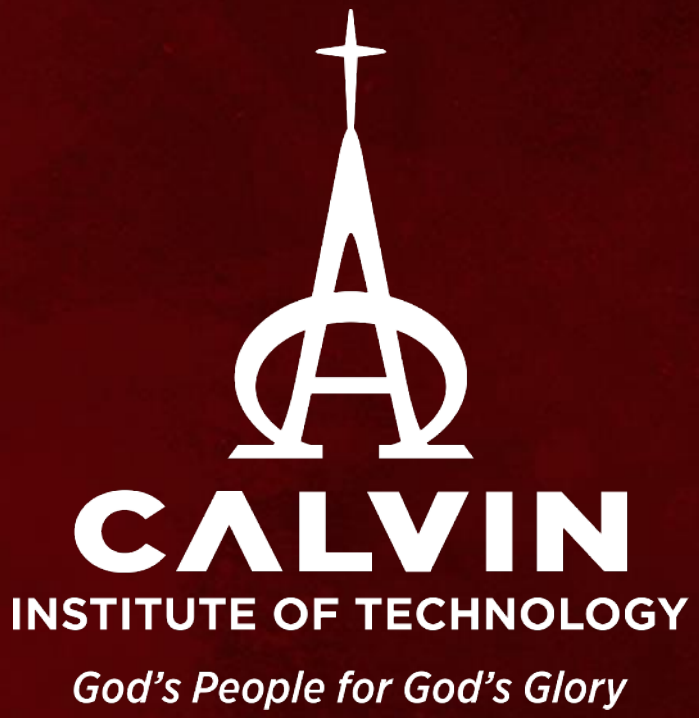


Convolution Network

Hendrik Santoso Sugiarto

Capaian Pembelajaran

- Convolution Network
  - Mengerti dan mengaplikasikan operasi konvolusi dan pooling (dengan padding dan stride)
- Convolutional Models
  - Menggunakan operasi konvolusi untuk merancang permodelan deep learning (sistem klasifikasi gambar)

Convolutional Network

Motivasi

- Berapa input pada gambar:
  - 64x64 pixel? 12288
  - 1000x1000 pixel? 3000000

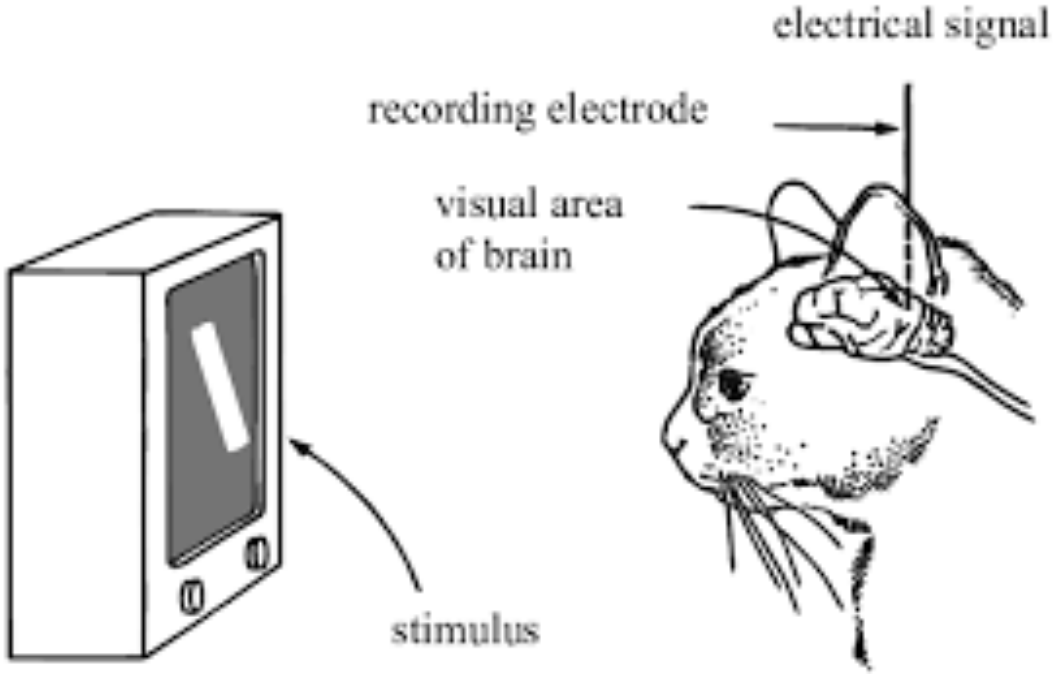
→dog? (0/1)

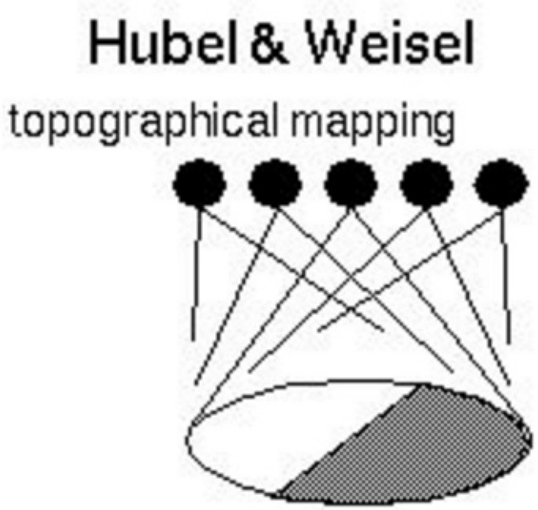
- Berapa parameter untuk input tersebut jika hidden layernya berukuran sama dengan input?
  - 64x64 pixel? 150,994,944
  - 1000x1000 pixel? 9x10^12

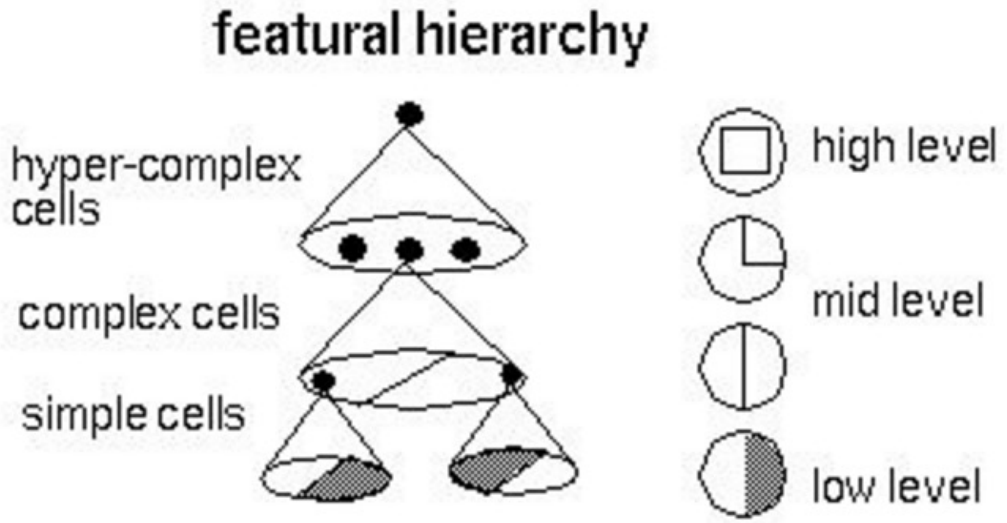
- Bagaimana cara memilih fitur yang relevan?
  - Problem rata-rata?
  - Problem translasi?

Insight dari Neurosains

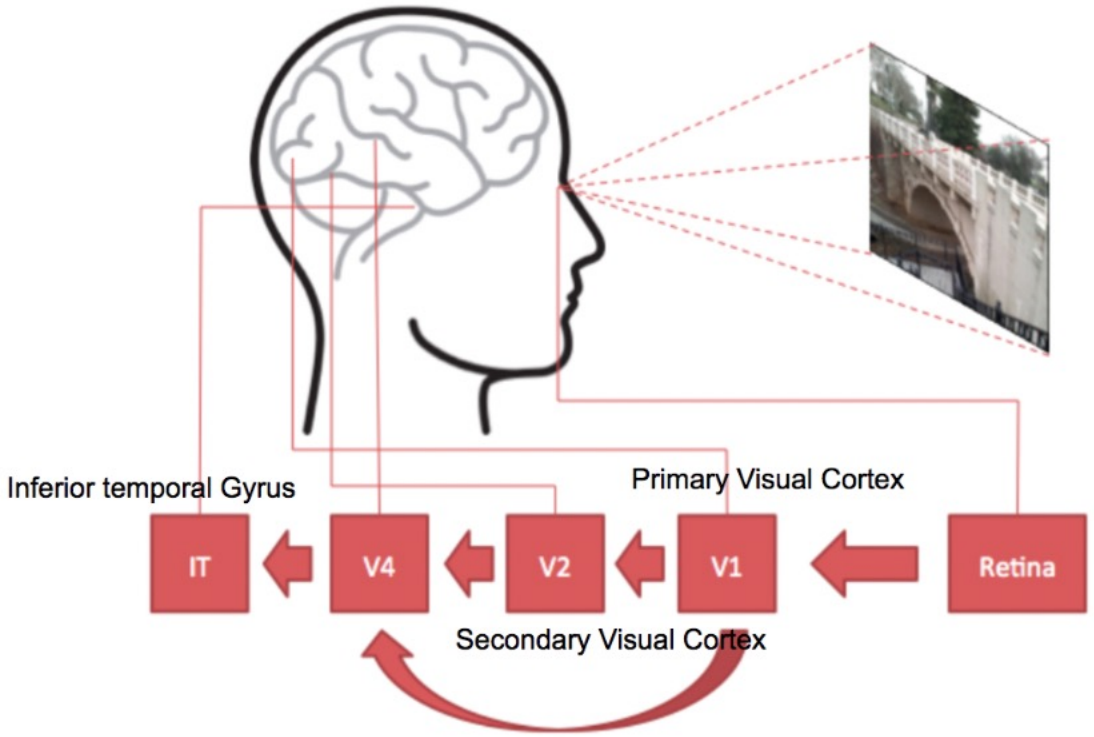
- **1959**: Hubel & Wiesel
  - Medan resepsi dari satu neuron di korteks striate pada kucing
  - https://www.youtube.com/watch?v=IOHayh06LJ4
- **1962**: Receptive Fields
  - Interaksi binocular dan arsitektur fungsional di korteks visual pada kucing
- Hierarchical Organizations
- **2005**: Neural abstraction
  - Medial temporal lobe
  - Menangkap abstraksi
  - “Halle Berry Neuron”







Visual Cortex



V1: Edge detection, etc.

V2: Extract simple visual properties (orientation, spatial frequency, color, etc)

V4: Detect object features of intermediate complexity

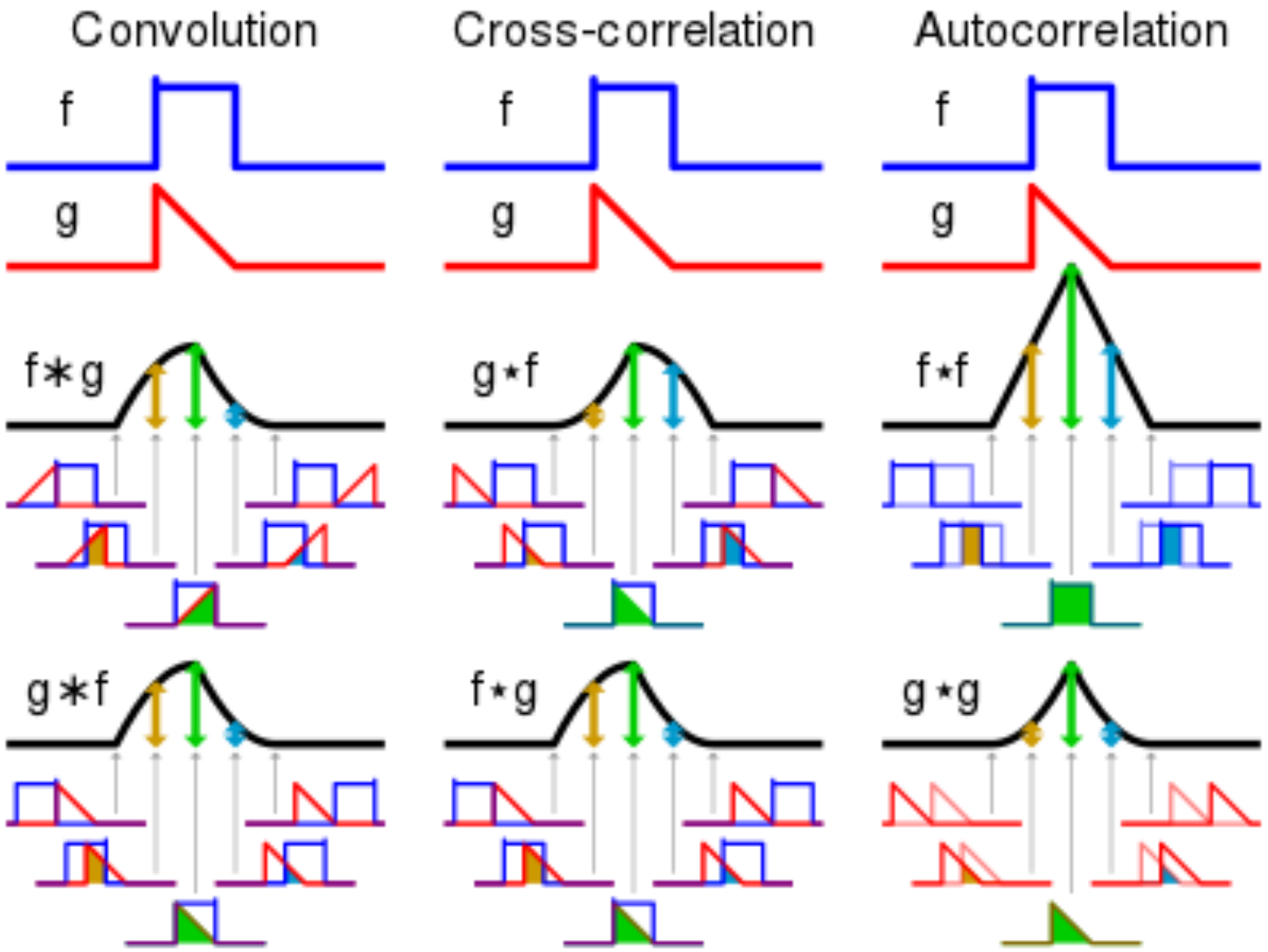
TI: Object recognition.

Fig. 3. Illustration of the human visual cortex system. (Image source: Wang & Raj 2017)

Konvolusi

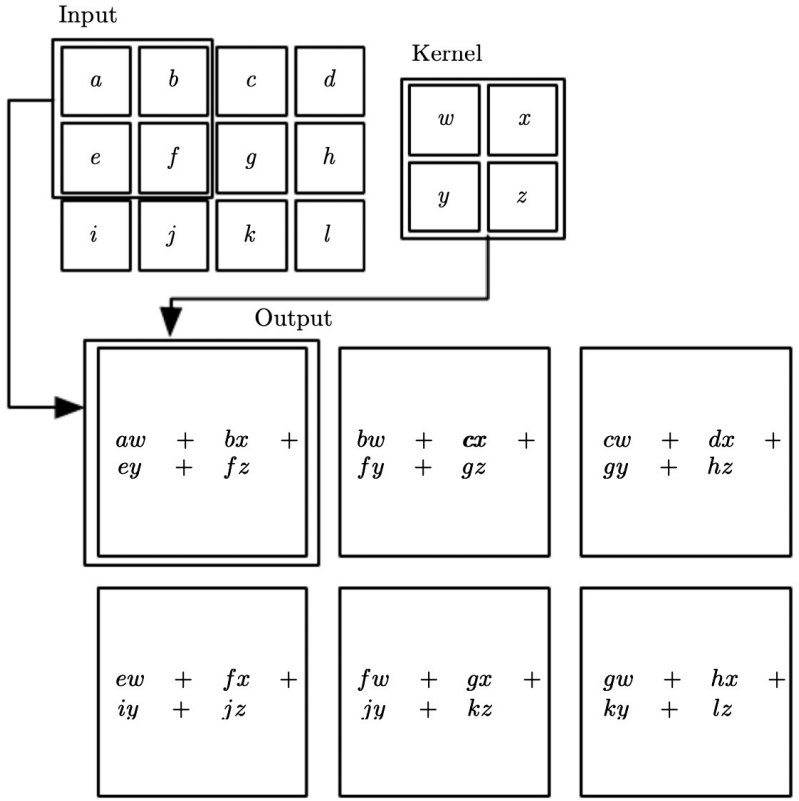
- Operasi matematis pada 2 fungsi $(f \ \& \ g)$ yang menghasilkan fungsi $(f * g)$
- $s(t) = (x * w)(t)$
- $s(t) = \int x(a)w(t-a)da$

- Cross-correlation disebut juga convolution (dalam convolutional neural network)



Konvolusi Diskrit 2D

- Konvolusi diskrit dapat dilihat sebagai perkalian tensor (**Toeplitz Matrix**)
- Ukuran:
  - Jika ukuran input: $n_h \times n_w$
  - Dan ukuran kernel/filter: $k_h \times k_w$
  - Maka ukuran output menjadi:
    - $(n_h - k_h + 1) \times (n_w - k_w + 1)$
- Konvolusi mempunyai beberapa sifat utama:
  - Sparse connection
  - Parameter sharing
  - Equivariance

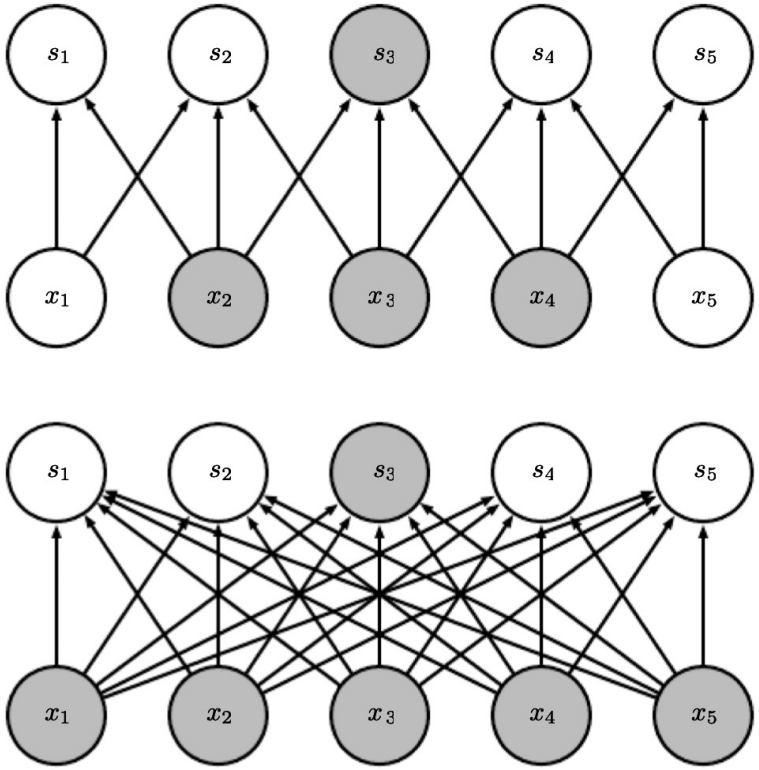


Deep learning, p334

Figure 9.1: An example of 2-D convolution without kernel-flipping. In this case we restrict the output to only positions where the kernel lies entirely within the image, called "valid" convolution in some contexts. We draw boxes with arrows to indicate how the upper-left element of the output tensor is formed by applying the kernel to the corresponding upper-left region of the input tensor.

Sparse Connection

- Konvolusi memaksa neuron hanya dipengaruhi oleh neuron terdekat
- Intuisi: kita mencari fitur lokal misalnya garis, tekstur, pola



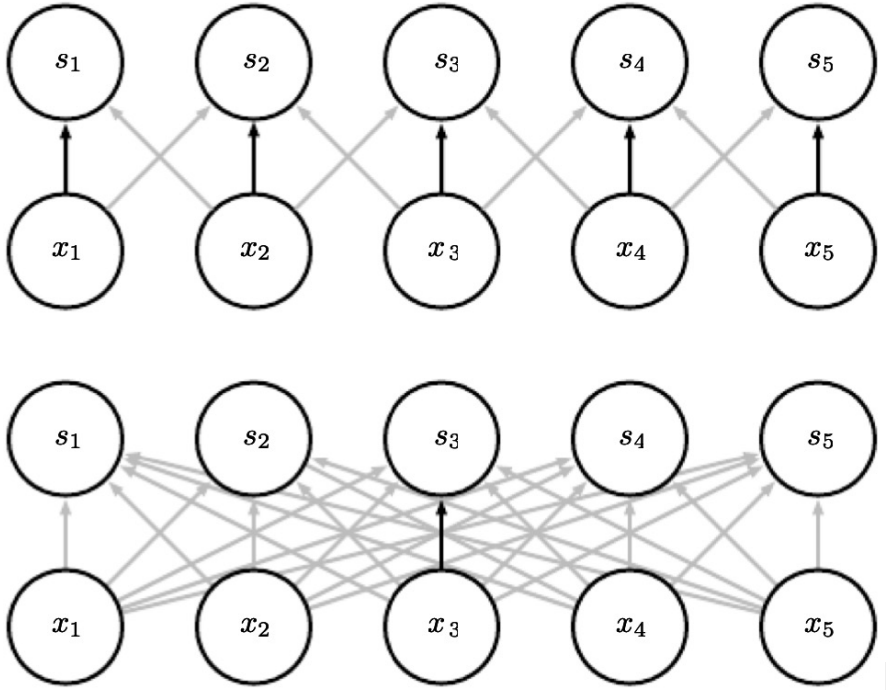
Deep learning, p337

Figure 9.3: *Sparse connectivity, viewed from above:* We highlight one output unit, s_3, and also highlight the input units in $\boldsymbol{x}$ that affect this unit. These units are known as the **receptive field** of s_3. *(Top)*When $\boldsymbol{s}$ is formed by convolution with a kernel of width 3, only three inputs affect s_3. *(Bottom)*When $\boldsymbol{s}$ is formed by matrix multiplication, connectivity is no longer sparse, so all of the inputs affect s_3.

CALVIN INSTITUTE OF TECHNOLOGY

Parameter Sharing

- Konvolusi memakai parameter yang sama untuk semua input
- Intuisi: jika kita tau cara mendeteksi fitur lokal di salah satu area gambar (misalnya orientasi), cara yang sama bisa dipakai untuk untuk area lainnya

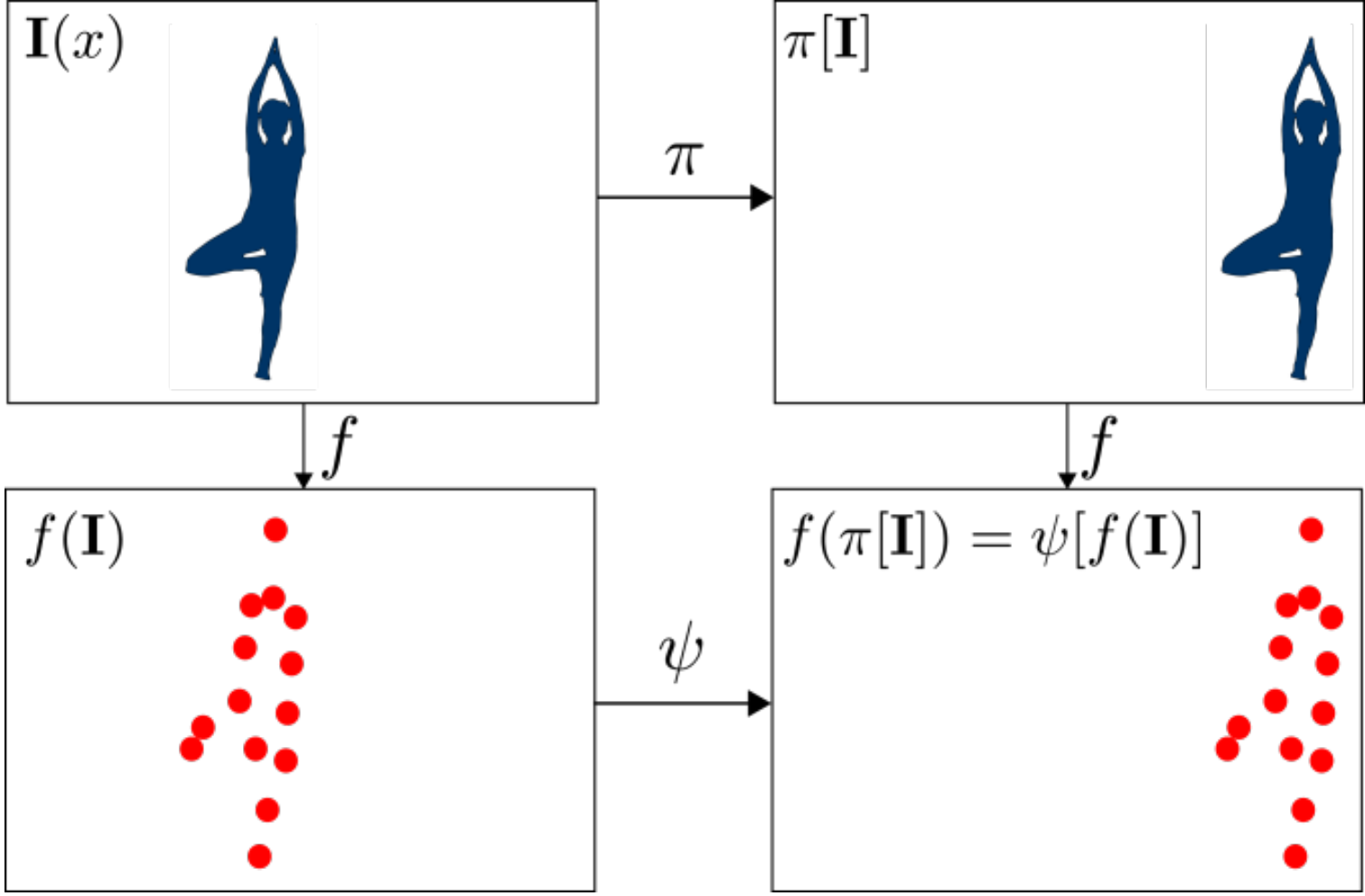


Deep learning, p338

Figure 9.5: Parameter sharing: Black arrows indicate the connections that use a particular parameter in two different models. *(Top)*The black arrows indicate uses of the central element of a 3-element kernel in a convolutional model. Due to parameter sharing, this single parameter is used at all input locations. *(Bottom)*The single black arrow indicates the use of the central element of the weight matrix in a fully connected model. This model has no parameter sharing so the parameter is used only once.

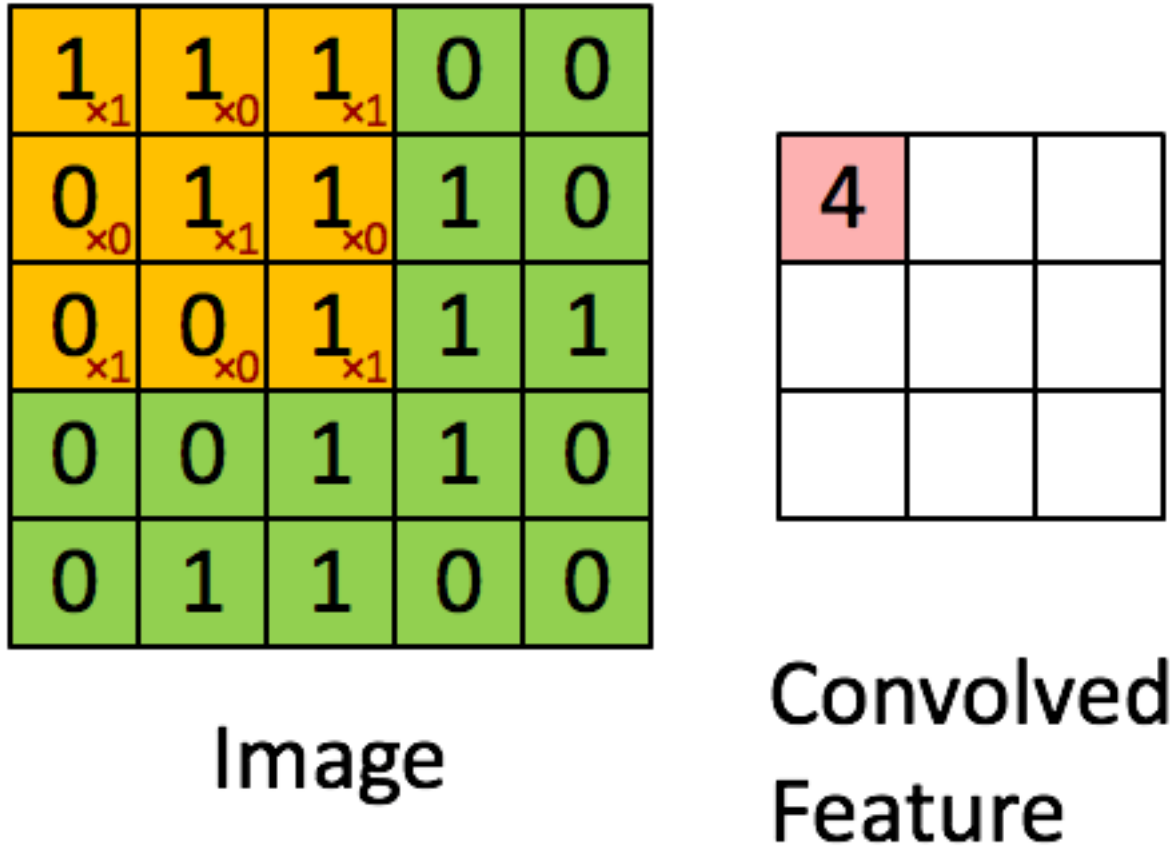
Equivariance

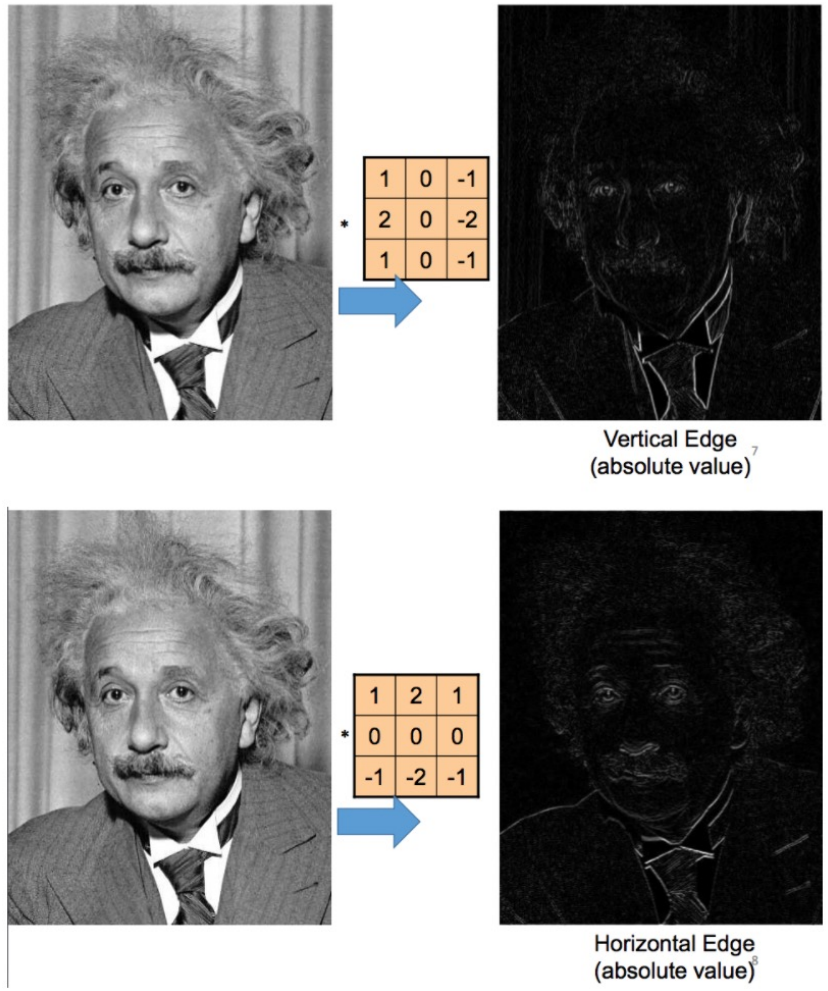
- $f\big(g(x)\big) = g\big(f(x)\big)$
- Intuisi: lokasi gambar tidak mempengaruhi fitur lokal



Konvolusi dan Filter

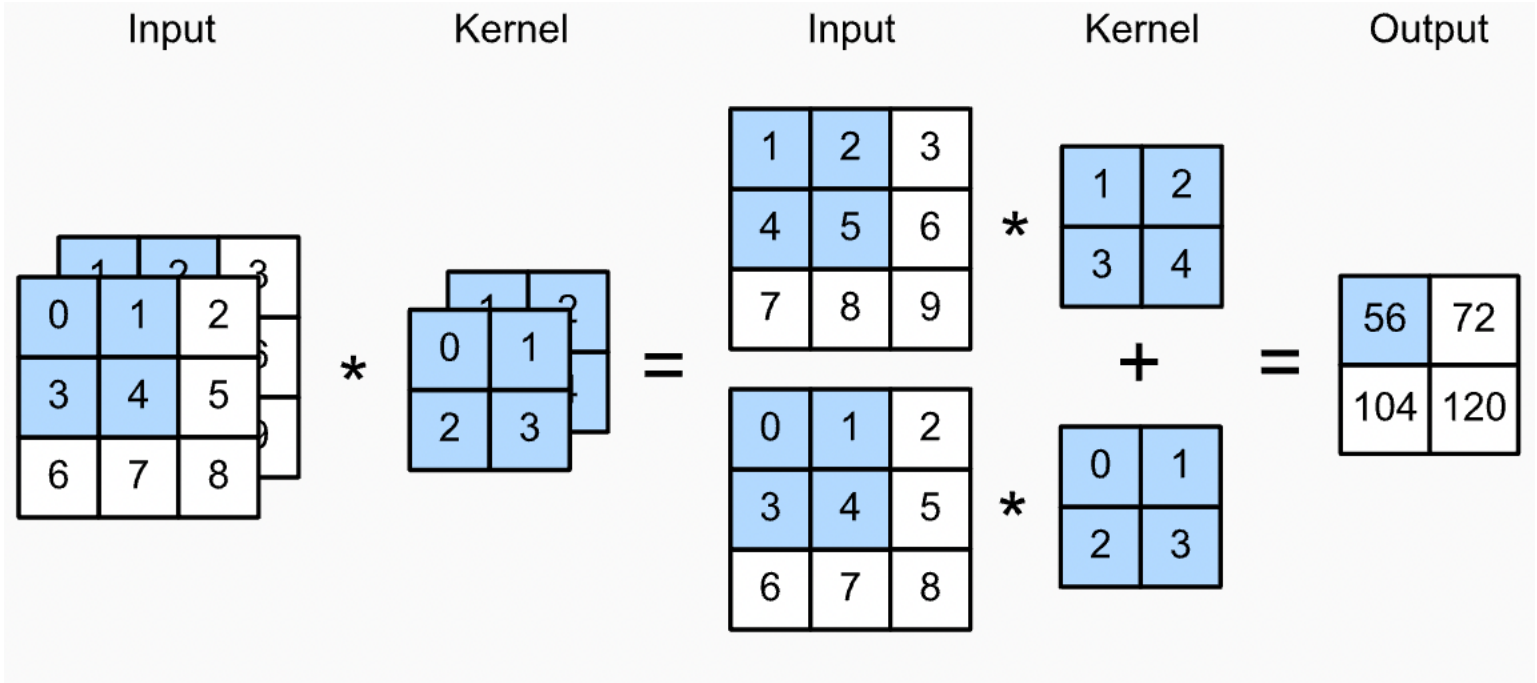
- Filter tertentu akan menangkap fitur tertentu





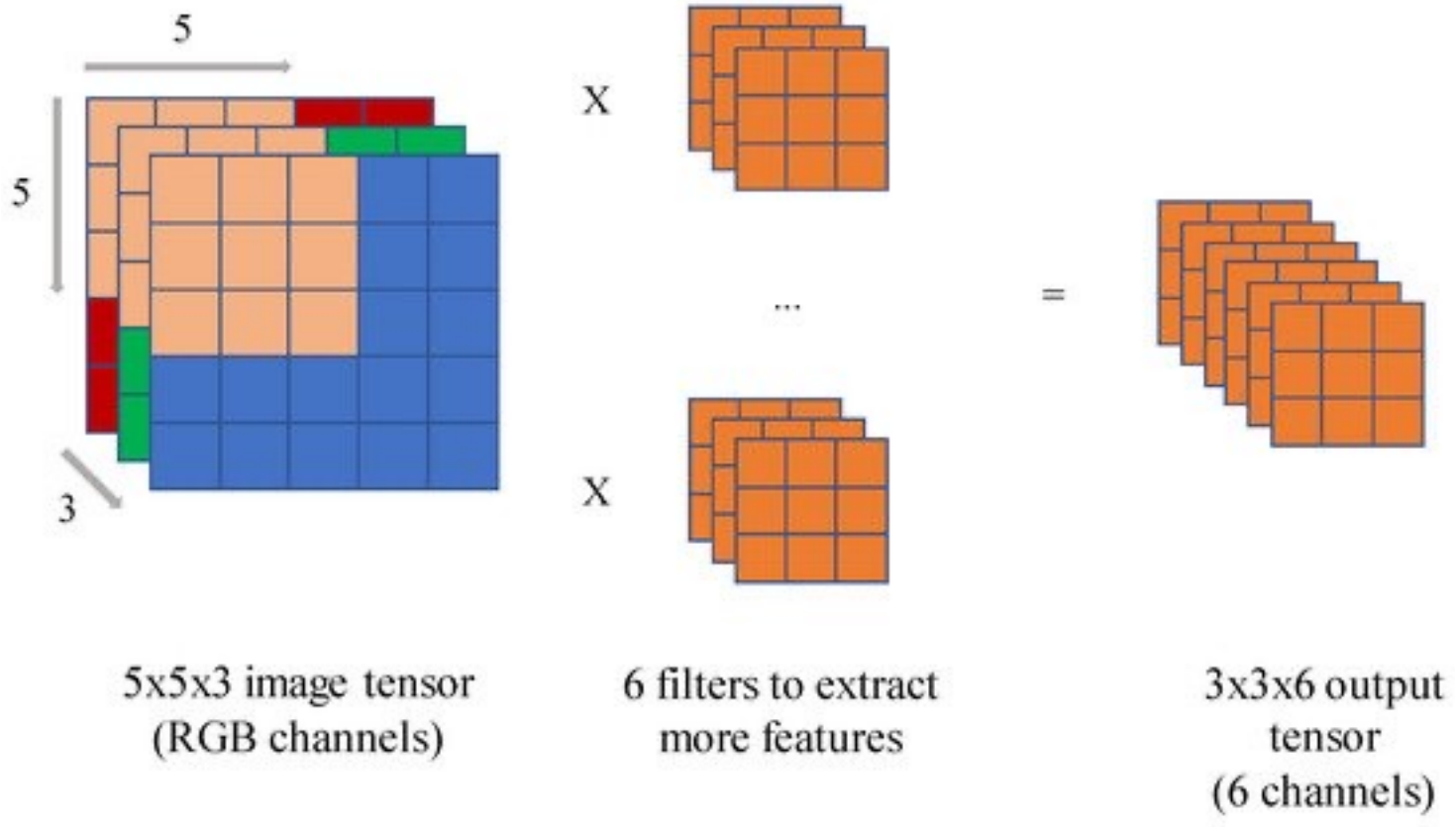
Multichannel input

- Operasi konvolusi dapat menerima multichannel (lebih dari satu) input, misalnya bada contoh berikut



Multichannel output

- Operasi konvolusi juga dapat digunakan untuk menghasilkan multichannel output
- Arsitektur dengan multichannel output sangatlah popular karena dapat digunakan untuk menyimpan informasi tentang beragama fitur



CALVIN INSTITUTE OF TECHNOLOGY

Uji Pemahaman

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

$*$

1	0	-1
1	0	-1
1	0	-1

$=$

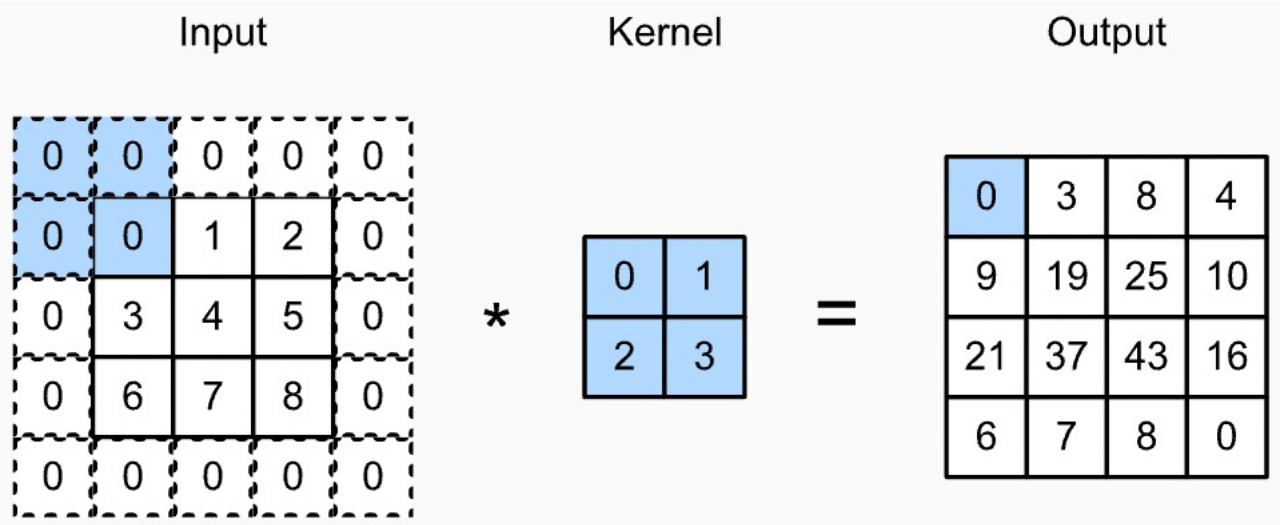
Apakah hasil dari operasi konvolusi ini?

Aktivitas: menghitung konvolusi dengan PyTorch

- Hitung konvolusi 2D:
  - Normal (manual dan library)
  - Dengan padding
  - Dengan stride
  - Dengan padding dan stride
- Hitung pooling:
  - Max pooling (manual dan library)
  - Dengan padding
  - Dengan stride
  - Dengan padding dan stride

Padding

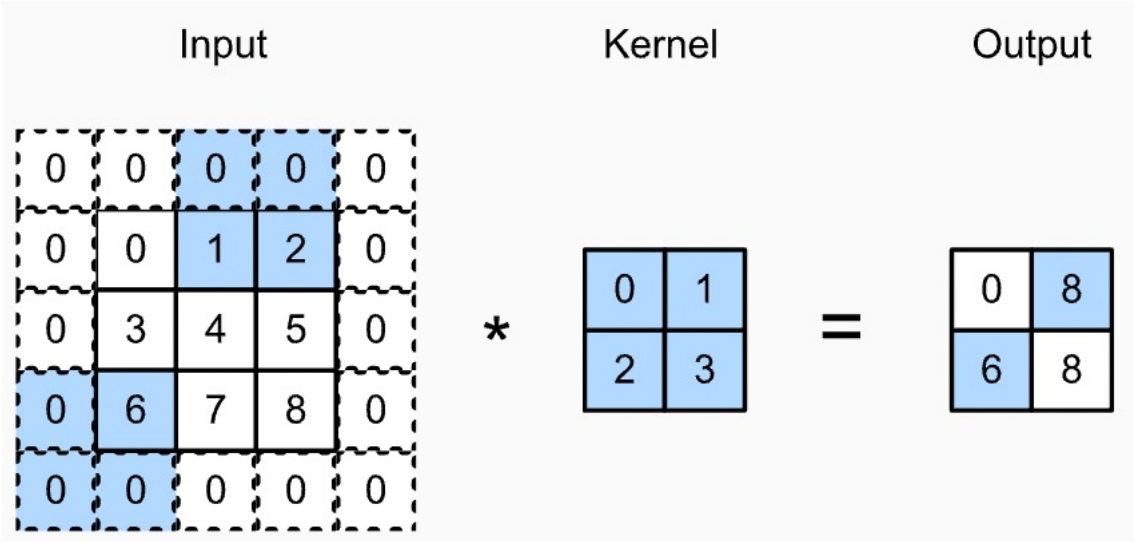
- Salah satu isu konvolusi adalah hilangnya informasi pada keliling gambar
- Salah satu trik adalah dengan menambah keliling gambar dengan tambahan pixel bernilai nol
- Jika ukuran input: $n_h \times n_w$
- Dan ukuran kernel/filter: $k_h \times k_w$
- Maka ukuran output menjadi: $(n_h - k_h + p_h + 1) \times (n_w - k_w + p_w + 1)$



CALVIN INSTITUTE OF TECHNOLOGY

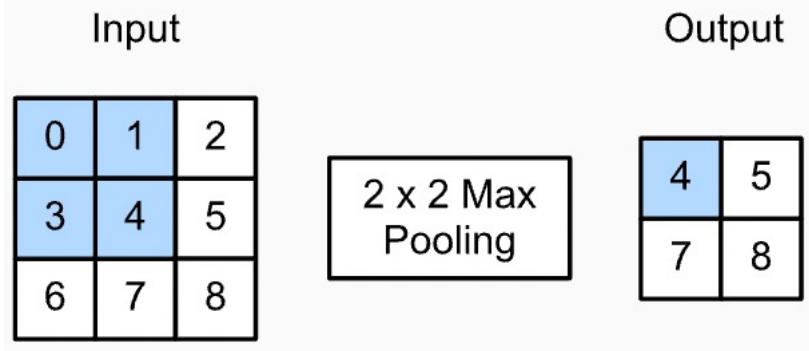
Stride

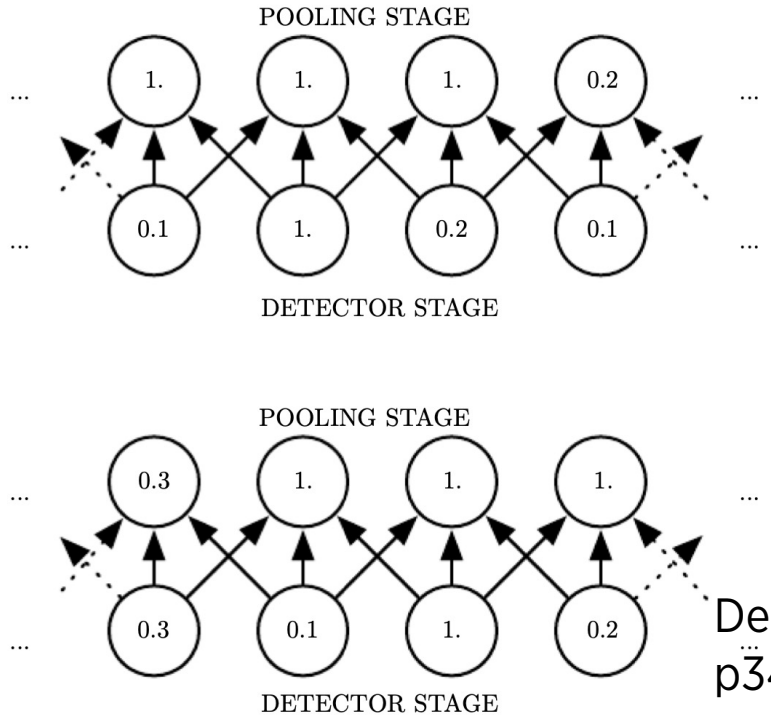
- Default konvolusi adalah menggeser 1 pixel
- Tapi dapat juga untuk menggeser beberapa pixel sekaligus sekali melangkah
- Jika ukuran input: $n_h \times n_w$
- Dan ukuran kernel/filter: $k_h \times k_w$
- Maka ukuran output menjadi: $\lfloor (n_h - k_h + p_h + s_h)/s_h \rfloor \times \lfloor (n_w - k_w + p_w + s_w)/s_w \rfloor$



Pooling

- Agregat lokal diperlukan untuk mengecilkan resolusi data, tanpa kehilangan banyak informasi
- Agregat bisa dihitung menggunakan dengan operasi lokal misalnya rata-rata, maksimum, minimum, dll
- Max pooling bersifat stabil pada pergeseran
- Padding dan stride dapat juga digunakan pada pooling



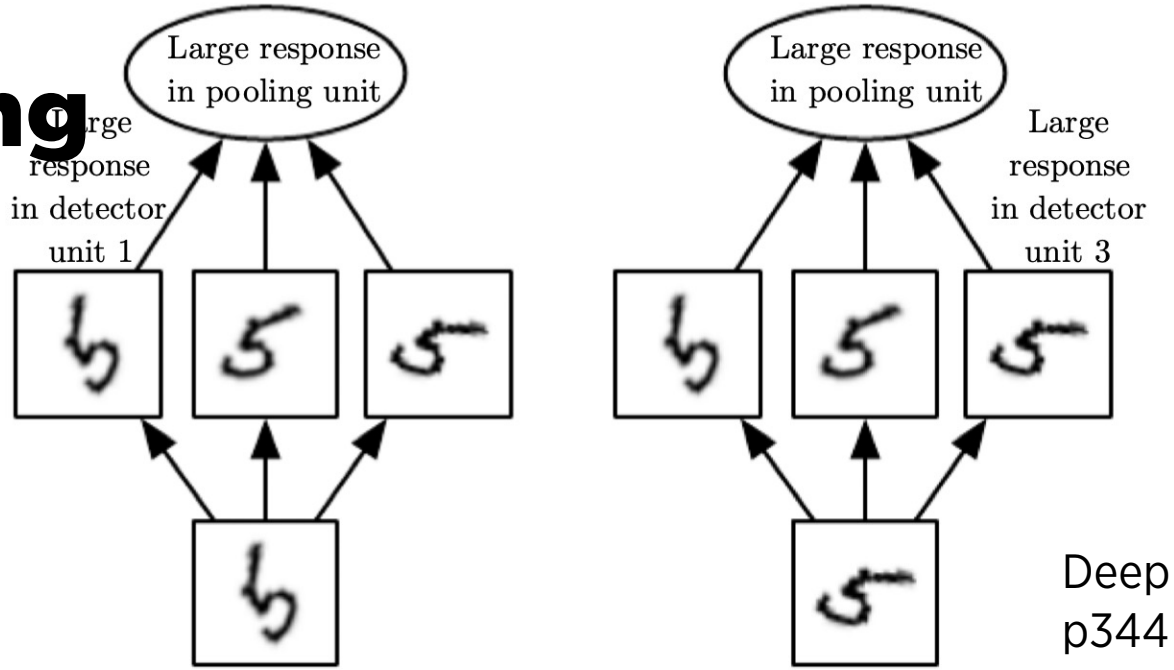


Deep learning, p343

Figure 9.8: Max pooling introduces invariance. *(Top)*A view of the middle of the output of a convolutional layer. The bottom row shows outputs of the nonlinearity. The top row shows the outputs of max pooling, with a stride of one pixel between pooling regions and a pooling region width of three pixels. *(Bottom)*A view of the same network, after the input has been shifted to the right by one pixel. Every value in the bottom row has changed, but only half of the values in the top row have changed, because the max pooling units are only sensitive to the maximum value in the neighborhood, not its exact location.

Kelebihan max pooling

- Invarian pada detector yang berbeda

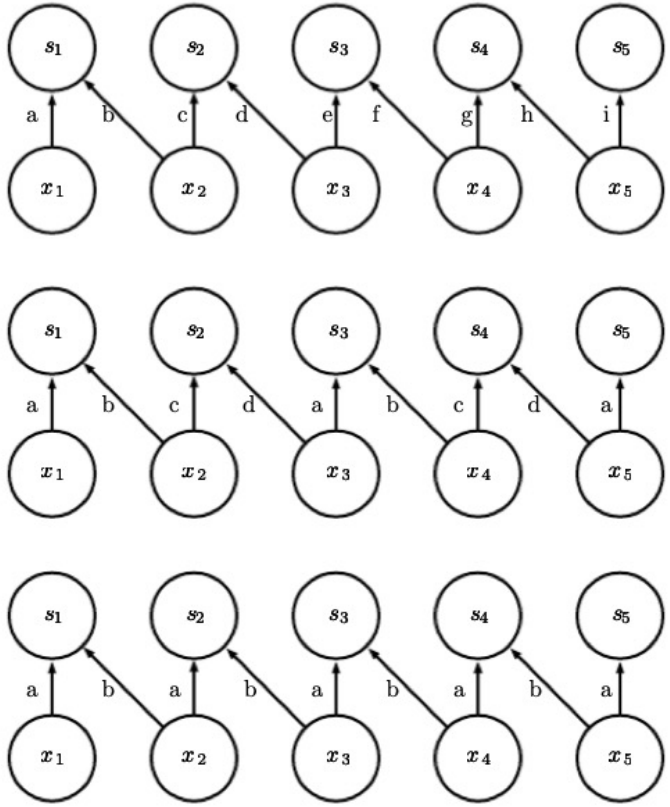


Deep learning, p344

Figure 9.9: *Example of learned invariances:* A pooling unit that pools over multiple features that are learned with separate parameters can learn to be invariant to transformations of the input. Here we show how a set of three learned filters and a max pooling unit can learn to become invariant to rotation. All three filters are intended to detect a hand-written 5. Each filter attempts to match a slightly different orientation of the 5. When a 5 appears in the input, the corresponding filter will match it and cause a large activation in a detector unit. The max pooling unit then has a large activation regardless of which detector unit was activated. We show here how the network processes two different inputs, resulting in two different detector units being activated. The effect on the pooling unit is roughly the same either way. This principle is leveraged by maxout networks (Goodfellow *et al.*, 2013a) and other convolutional networks. Max pooling over spatial positions is naturally invariant to translation; this multi-channel approach is only necessary for learning other transformations.

Konvolusi berpola (tiled)

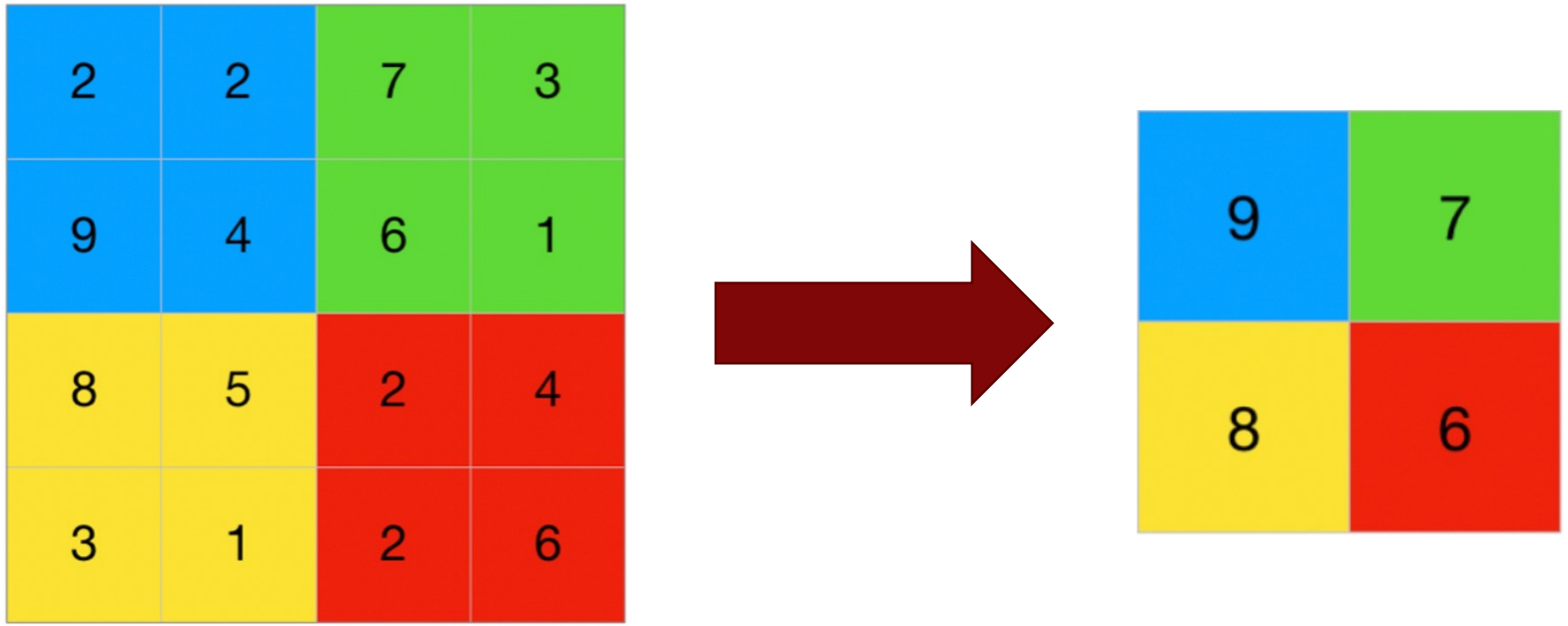
- Semi-parameter-sharing



Deep learning, p355

Figure 9.16: A comparison of locally connected layers, tiled convolution, and standard convolution. All three have the same sets of connections between units, when the same size of kernel is used. This diagram illustrates the use of a kernel that is two pixels wide. The differences between the methods lies in how they share parameters. *(Top)*A locally connected layer has no sharing at all. We indicate that each connection has its own weight by labeling each connection with a unique letter. *(Center)*Tiled convolution has a set of t different kernels. Here we illustrate the case of $t = 2$. One of these kernels has edges labeled "a" and "b," while the other has edges labeled "c" and "d." Each time we move one pixel to the right in the output, we move on to using a different kernel. This means that, like the locally connected layer, neighboring units in the output have different parameters. Unlike the locally connected layer, after we have gone through all t available kernels, we cycle back to the first kernel. If two output units are separated by a multiple of t steps, then they share parameters. *(Bottom)*Traditional convolution is equivalent to tiled convolution with $t = 1$. There is only one kernel and it is applied everywhere, as indicated in the diagram by using the kernel with weights labeled "a" and "b" everywhere.

Uji Pemahaman



Operasi apakah ini?

1. Normal Convolution
2. Convolution with Padding and Stride
3. Max Pooling
4. Average Pooling

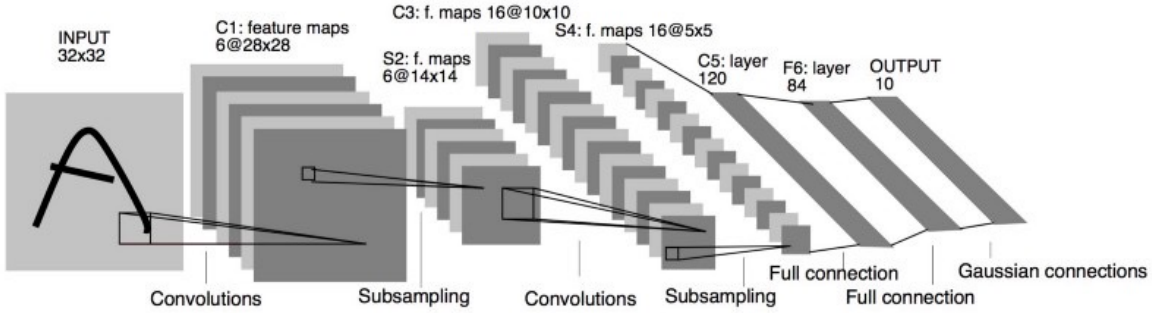
CALVIN INSTITUTE OF TECHNOLOGY

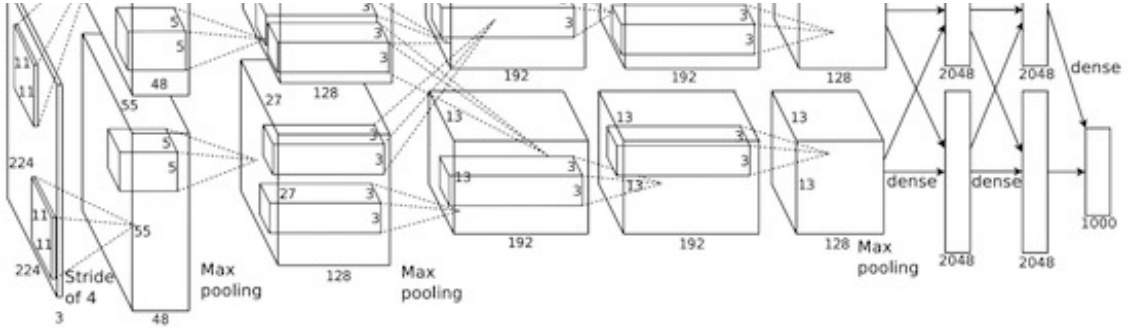
Convolutional Models

Sejarah CNN

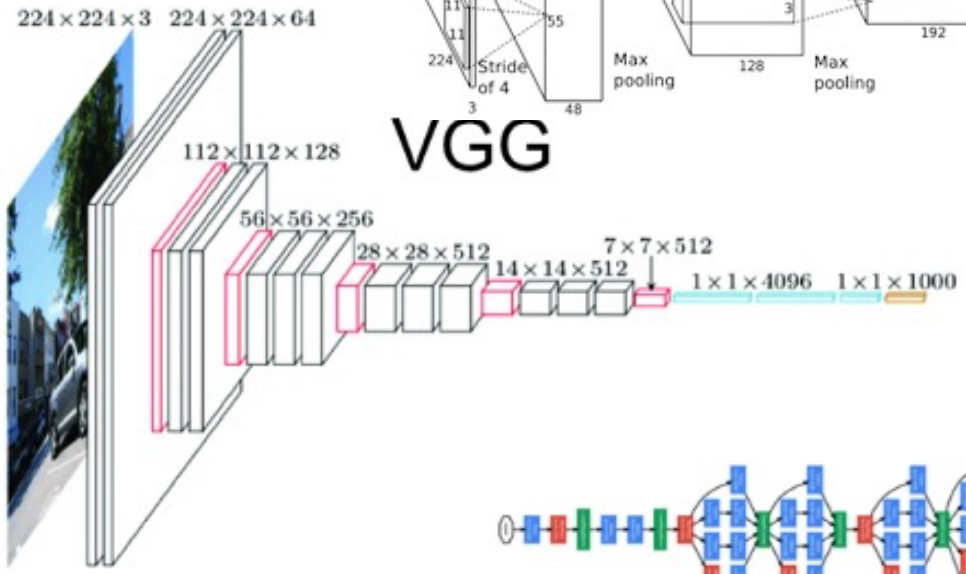
- 1998: LeNet
  - 2 Lapisan Konvolusi
  - 2 Lapisan MLP
- 2012: AlexNet
  - 5 Lapisan Konvolusi
  - 3 Lapisan MLP
- 2014: VGG
  - Sangat Dalam (19 lapisan)
- 2014: GoogleNet
  - Menggunakan Modul Insepsi
  - 22 Lapisan
- 2015: ResNet
  - Menggunakan Modul Residual
  - 152 lapisan
- 2016: SqueezeNet, ENET, Xception
  - Memiliki semua fitur sebelumnya
  - Lebih efisien

LeNet pernah kita gunakan pada LKM sebelumnya ☺

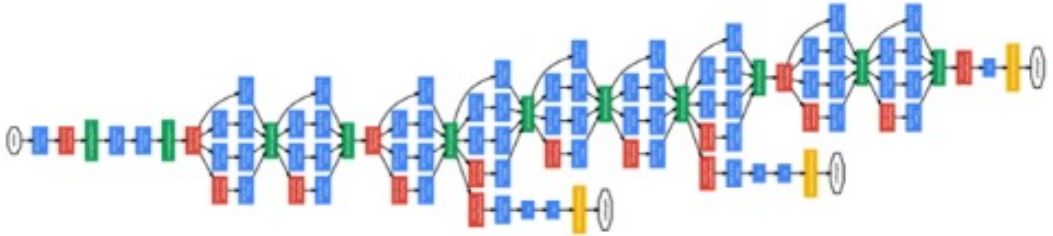




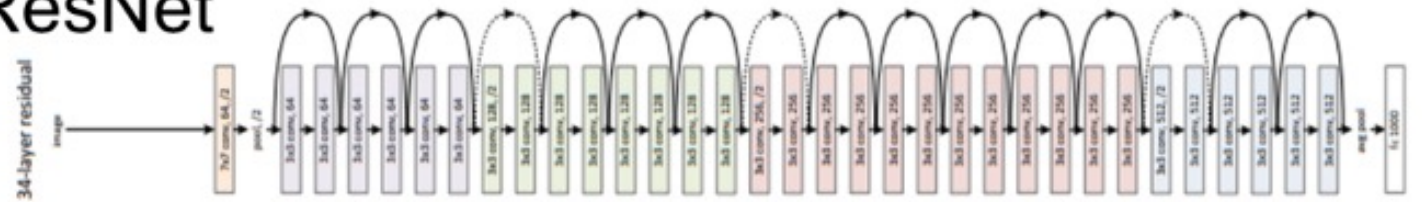
VGG



GoogLeNet



ResNet



Akurasi dan Operasi

- “An Analysis of Deep Neural Network Models for Practical Applications” Alfredo Canziani, Adam Paszke, Eugenio Culurciello Published 2016 in ArXiv

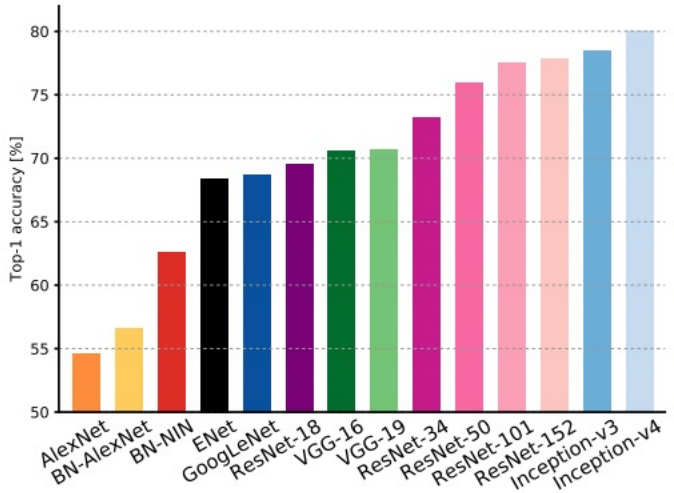


Figure 1: **Top1 *vs.* network.** Single-crop top-1 validation accuracies for top scoring single-model architectures. We introduce with this chart our choice of colour scheme, which will be used throughout this publication to distinguish effectively different architectures and their correspondent authors. Notice that networks of the same group share the same hue, for example ResNet are all variations of pink.

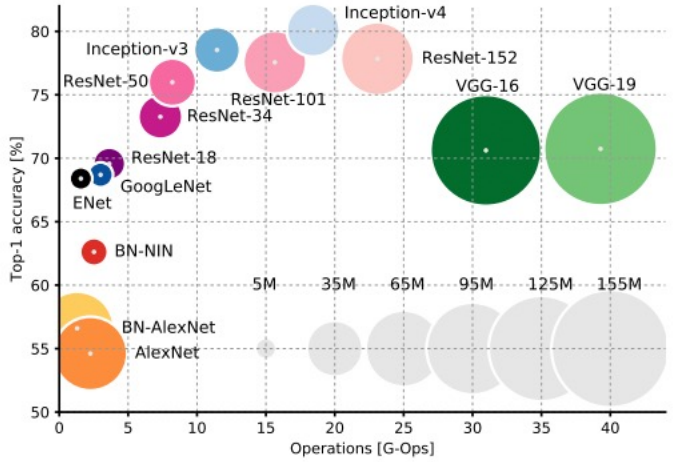
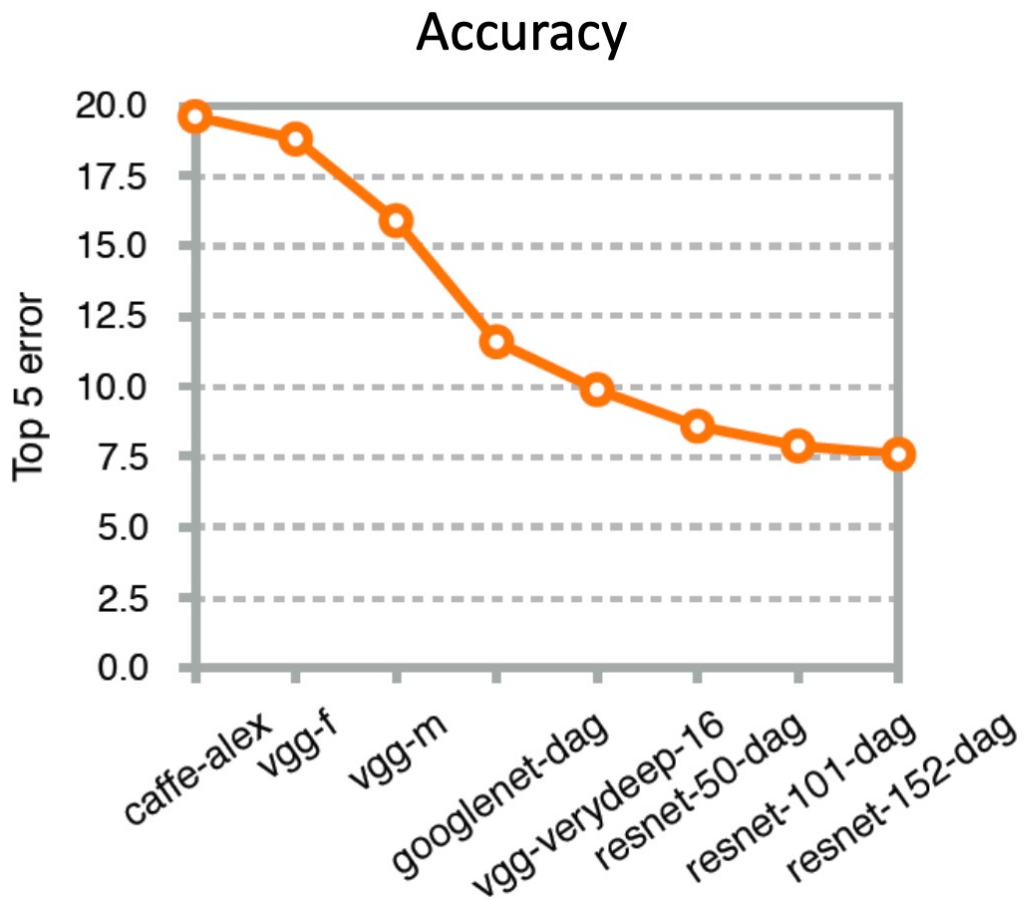


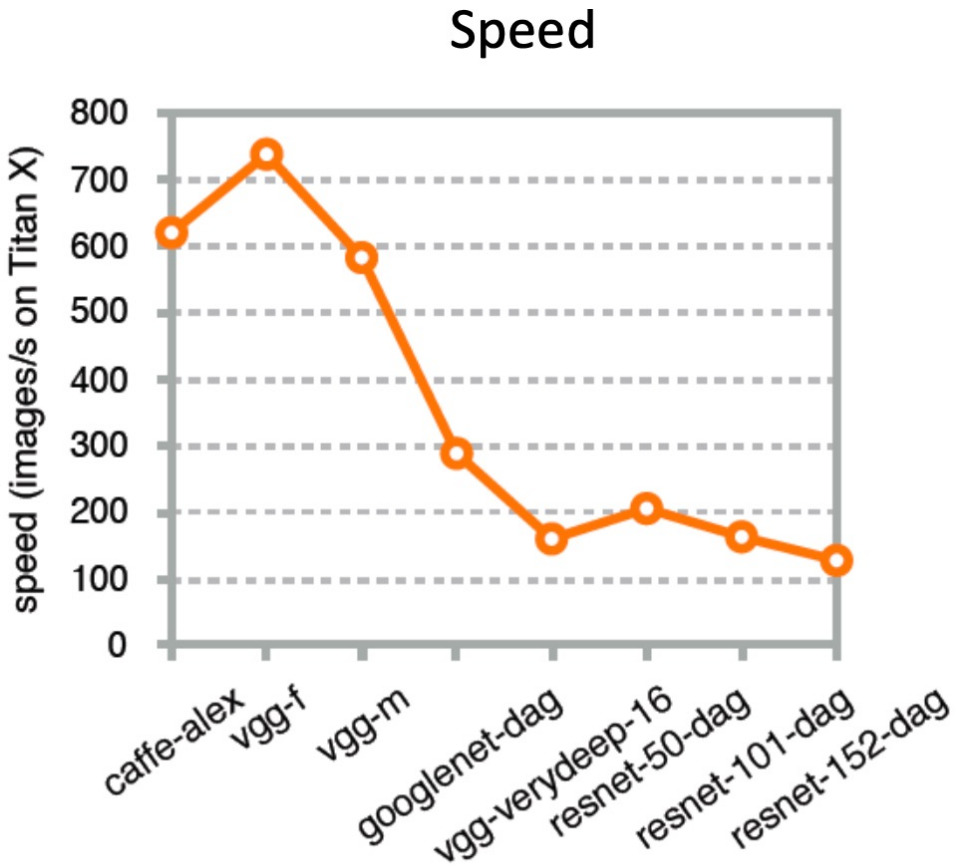
Figure 2: **Top1 *vs.* operations, size $\propto$ parameters.** Top-1 one-crop accuracy versus amount of operations required for a single forward pass. The size of the blobs is proportional to the number of network parameters; a legend is reported in the bottom right corner, spanning from 5×10^6 to 155×10^6 params. Both these figures share the same y-axis, and the grey dots highlight the centre of the blobs.

CALVIN INSTITUTE OF TECHNOLOGY

Akurasi vs Kecepatan

- 3x lebih akurat dalam 3 tahun
- 5x lebih lambat





CALVIN INSTITUTE OF TECHNOLOGY

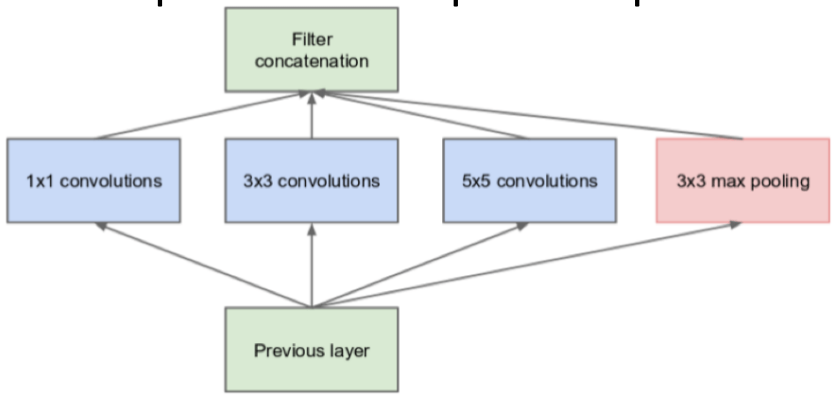
Uji Pemahaman

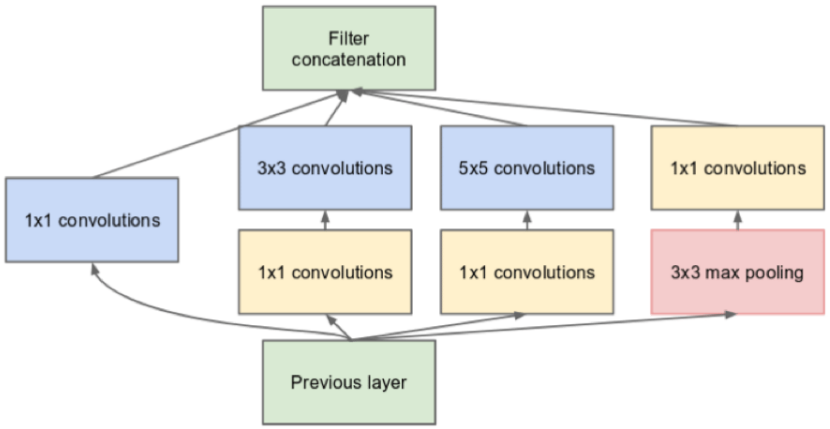
Manakah yang bukan golongan CNN?

1. AlexNet
2. VGG
3. ResNet
4. LSTM

Inception Blocks

- Problem CNN klasik:
  - Memilih ukuran kernel yang tepat untuk operasi *convolution* menjadi sulit karena banyak variasi informasi dari lokasi di dalam gambar:
    - Kernel besar cocok untuk menyalurkan informasi secara global,
    - Kernel kecil cocok untuk menyalurkan informasi secara lokal.
  - Jaringan yang sangat dalam mudah *overfitting* → sulit update gradien seluruh *network*
  - Operasi konvolusi yang menumpuk dalam jumlah besar juga dapat menyebabkan waktu komputasi yang lama/ mahal
- Solusi insepsi:
  - Operasi beberapa filter pada level sama membuat network sedikit lebih lebar daripada lebih dalam





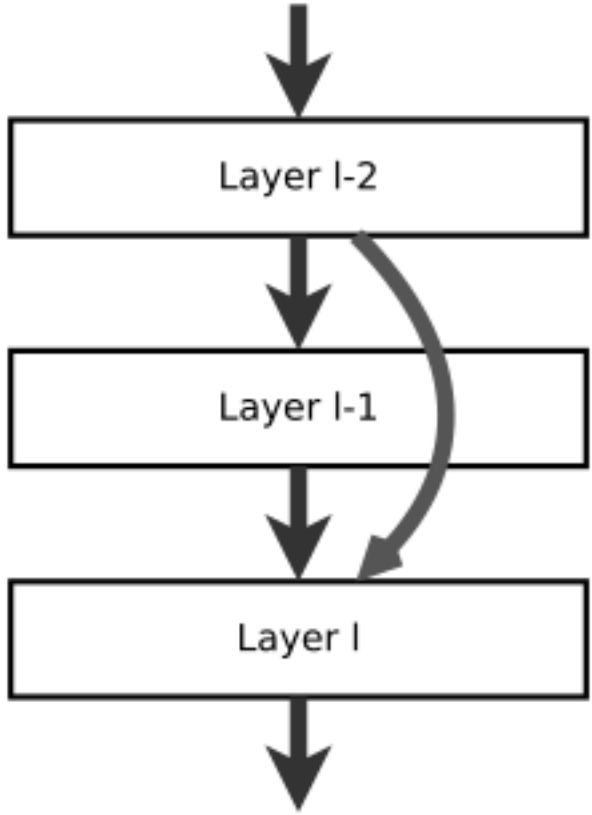
CALVIN INSTITUTE OF TECHNOLOGY

Residual Blocks

- Residual blocks membuat propagasi informasi melewati beberapa layer sekaligus
- Jika tensor $W^{(l-2,\,l)}$ adalah bobot dari layer $l-2$ ke layer l:

$$a^{(L)} = f(W^{(l-1,\,l)}.a^{(l-1)} + b^{(l)} + W^{(l-2,l)}.a^{(l-2)})$$

- Ada beberapa alasan menggunakan skip connections:
  - Menghindari vanishing gradients
  - Mitigasi degradasi (meningkatnya training error dengan menambah layer)
  - Proses learning lebih efisien



Aktivitas: membuat CNN dengan PyTorch

- Transfer pretrained AlexNet, VGG, googleNet, resNet dari PyTorch
- Lihat strukturnya dan coba buat AlexNet secara manual dengan PyTorch
- Tampilkan operasi konvolusi lapisan pertama resNet pada gambar

Uji Pemahaman

- Apa saja parameter pada CNN?

Summary

- Masing-masing sebutkan konsep utama yang diingat

Tuhan Memberkati

God's People for God's Glory
CALVIN
INSTITUTE OF TECHNOLOGY