Convolutional Neural Networks

อ. ปรัชญ์ ปิยะวงศ์วิศาล

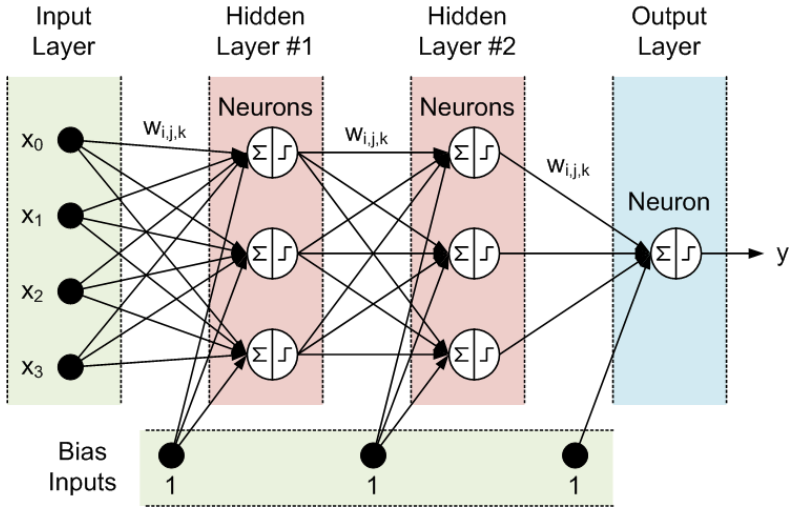
Pratch Piyawongwisal

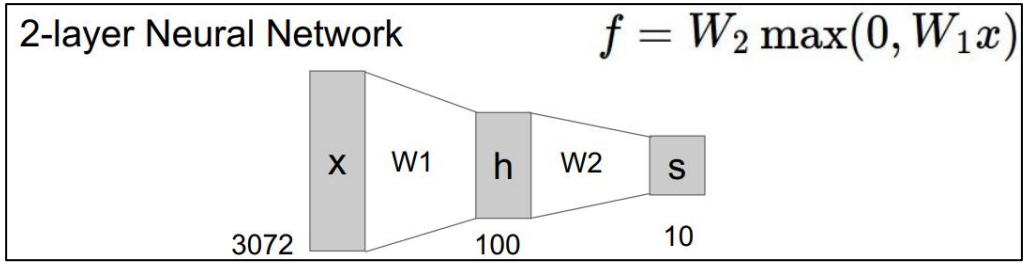
Today

- Convolutional Neural Networks
  - Recap: Fully-Connected NN
  - Applications of CNN
  - Why use CNN over Fully-Connected NN for images?
  - Convolution & Filter
  - CONV Layer
  - Padding and Stride
  - Pooling Layer
  - Counting Parameters of CNN
  - Hyperparameters of CNN
  - Basic CNN Structure & Modern Architecture
  - Lab: CIFAR10 with CNN

Recap: Feedforward/Fully-Connected NN

- Fully-Connected NN คือ NN ที่ในแต่ละ layer มีเส้น weight เชื่อมครบทุกคู่ neuron
- ทบทวน concept ที่สำคัญ
  - Neuron, Weight, Bias
  - Activation Function
  - Layer (Input/Hidden/Output)
  - ค่า Activation
  - Softmax Layer
  - Matrix Form ของการคำนวณในแต่ละ Layer:
    - $\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \sigma\left(\begin{bmatrix} W_{11} & W_{12} & \dots & W_{1n} \\ W_{21} & W_{22} & \dots & W_{2n} \\ W_{31} & W_{32} & \dots & W_{3n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}\right)$
  - Backprop & SGD Training Algorithm

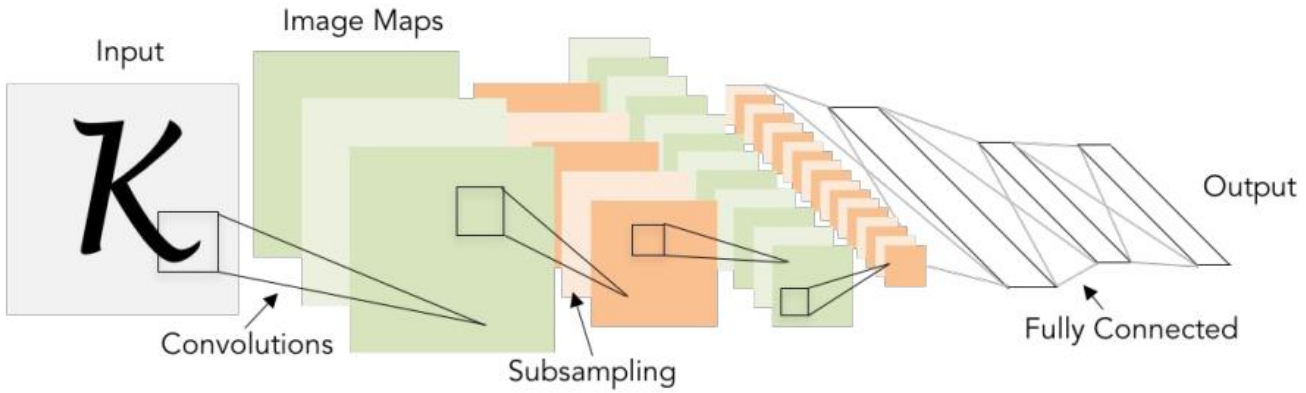




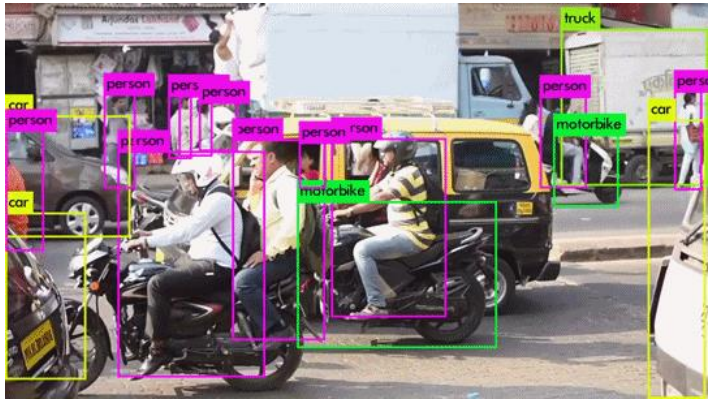
Convolutional Neural Network (CNN)

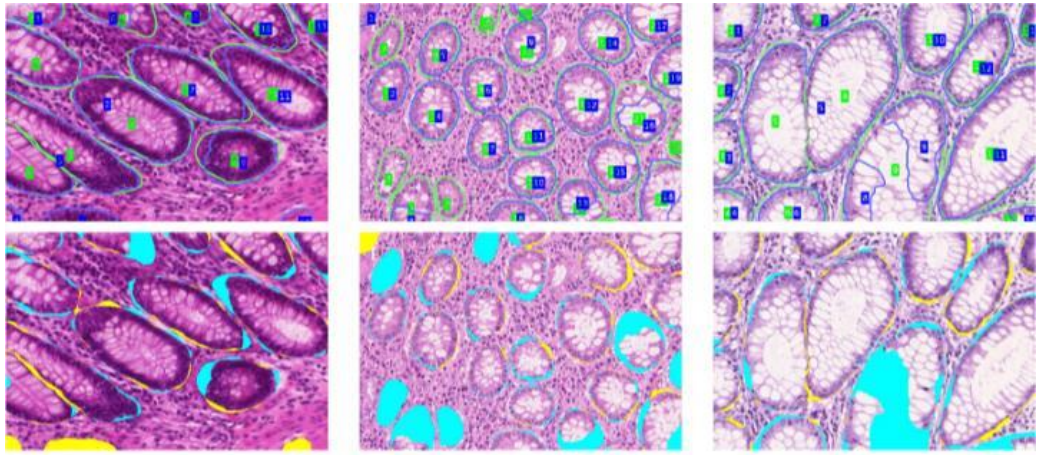
- เป็น **NN** ที่ออกแบบใหม่เพื่อให้เหมาะสมกับข้อมูล **input 2+** มิติ (เช่น รูปภาพ, วิดิโอ)
  - **เดิม:** ใช้ layer แบบ fully-connected (FC layer)
    = ระหว่าง layer มีเส้น weight เชื่อมครบทุกคู่ neuron
    นั่นคือ neuron ใน layer หลังจะสนใจ**ทุกส่วน**ของ layer ก่อนหน้า

  - **เปลี่ยนเป็น:** ใช้ layer แบบ convolution (CONV layer)
    = ใช้ weight จำนวนน้อยๆ (เช่น 5x5) กวาดหาสิ่งที่สนใจใน layer ก่อนหน้า
    นั่นคือ neuron ใน layer หลังจะกวาดดูเป็น**บริเวณเล็กๆ** ของ layer ก่อนหน้า

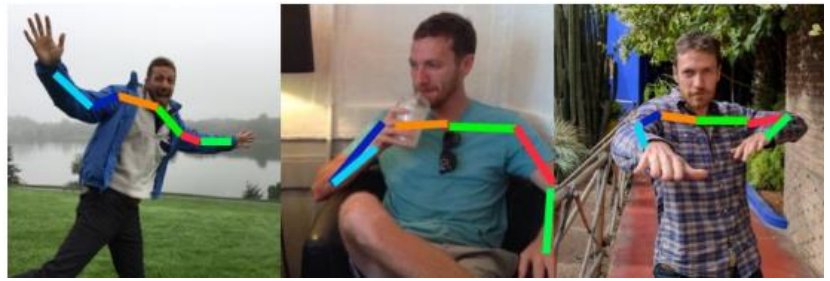


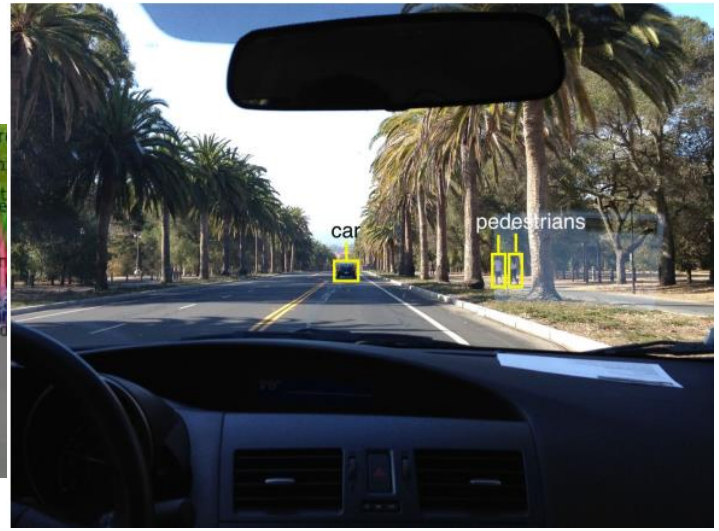
Applications of CNN





(d) benign (e) benign (f) malignant





Why not Fully-Connected NN?

- Q: ทำไม Fully-Connected NN หรือ Dense Layer จึง**ไม่เหมาะ**กับข้อมูลรูปภาพ?
  - ทั้งๆ ที่ในแล็บที่แล้ว เราก็ใช้ FC NN กับชุดข้อมูล MNIST ขนาด 32x32 แล้วได้ผลค่อนข้างดี

```
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(10, activation=tf.nn.softmax))
```

Why not Fully-Connected NN?

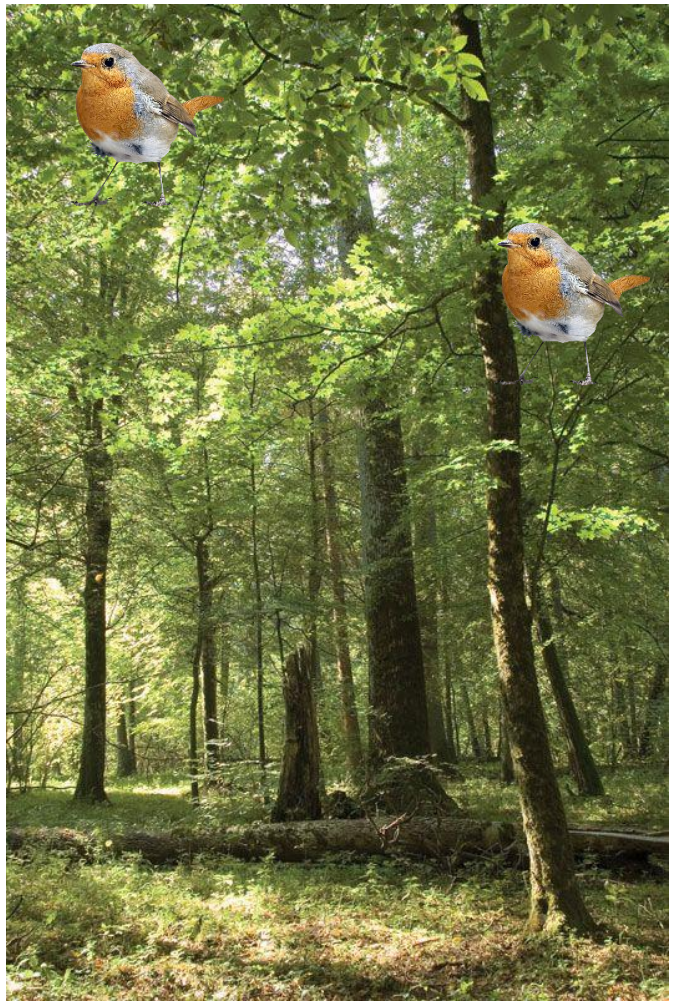
- Q: ทำไม Fully-Connected NN หรือ Dense Layer จึง**ไม่เหมาะ**กับข้อมูลรูปภาพ?
  - ทั้งๆ ที่ในแล็บที่แล้ว เราก็ใช้ FC NN กับชุดข้อมูล MNIST ขนาด 32x32 แล้วได้ผลค่อนข้างดี

- Ans:
  - ต้องใช้ parameter จำนวนมหาศาล หาก input มีขนาดใหญ่
  - พารามิเตอร์จำนวนมากนั้นเกินจำเป็นสำหรับตรวจจับ pattern ในรูปภาพ

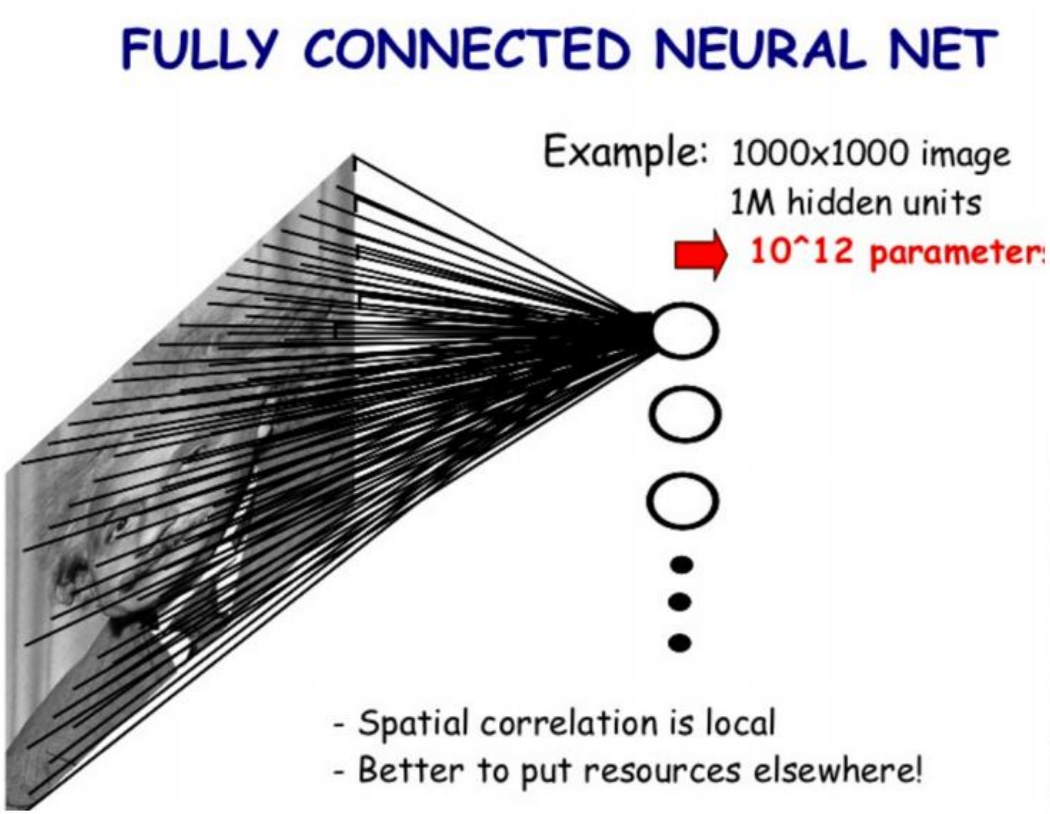
Why not Fully-Connected NN?

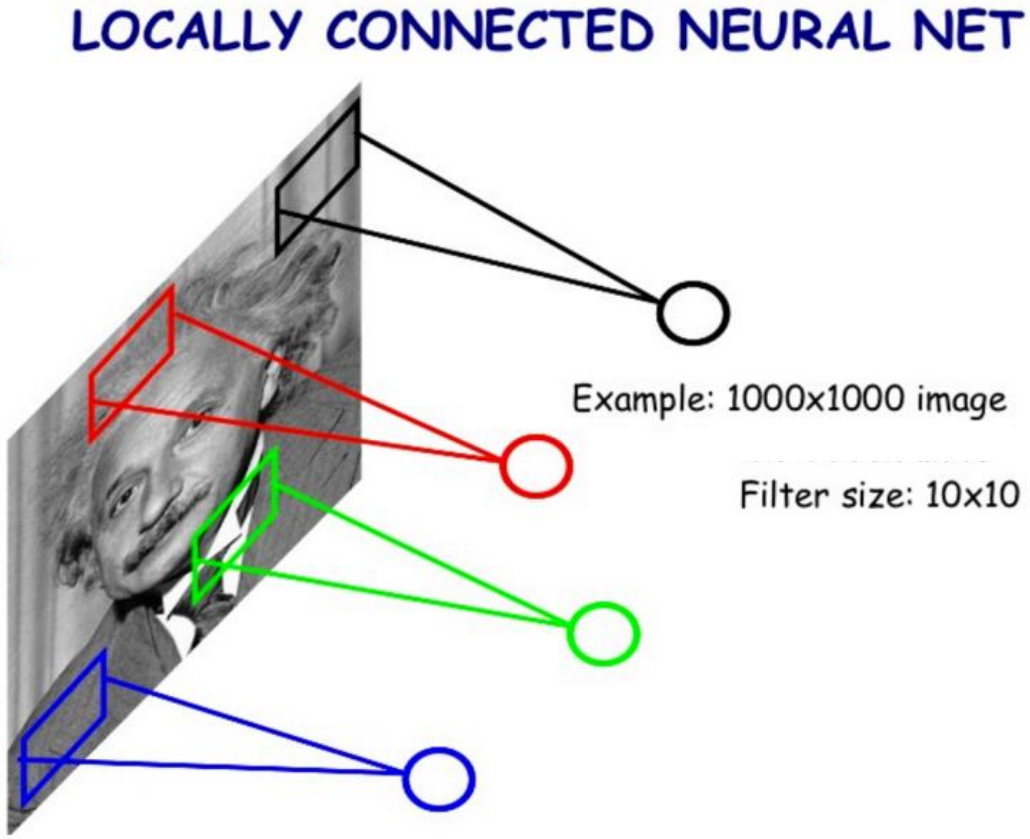
- **2 คุณสมบัติของ vision system**
  - Translation Invariance
    - การเลื่อนตำแหน่งของวัตถุในภาพไม่มีผลต่อการตรวจจับ
    - = ไม่ว่าวัตถุจะอยู่ตรงไหนก็ยังควรตรวจจับได้ไม่ต่างกัน
  - Locality
    - การตรวจจับสิ่งที่น่าสนใจในภาพ กวาดตาดูทีละบริเวณเล็กๆ ก็พอ
    - ไม่ใช่ดูทั้งภาพตลอดเวลา

- ด้วย 2 คุณสมบัตินี้ เราจึงสามารถใช้ CONV แทน FC Layer ได้
  - ลดจำนวน parameter ลงไปมาก ☺



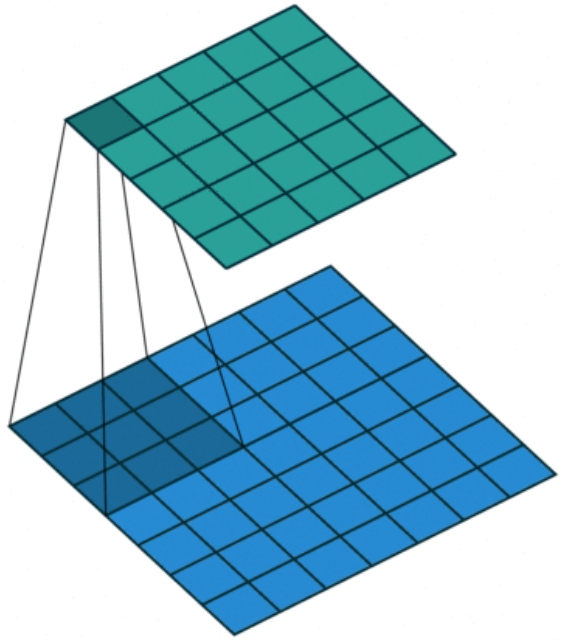
FC vs CONV





https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/deep_learning/convolutional_neural_networks

Convolution & Filter Kernels



- **Convolution** เป็น operator พื้นฐานใน image processing
  - $h * I(x, y) = \sum_m \sum_n h(m, n) I(x - m, y - n)$
  - คือการนำ Kernel h มา convolve กับภาพ I

- **Kernel/Filter**
  - คือ matrix ขนาดเล็ก (เช่น 3x3, 5x5) ที่เราสามารถออกแบบค่าเพื่อที่นำมา convolve กับภาพแล้วเกิด effect ที่น่าสนใจได้
  - เช่น Kernel สำหรับทำการ Blur, Sharpen, Detect Edges
  - Kernel in action: https://setosa.io/ev/image-kernels/

https://github.com/vdumoulin/conv_arithmetic

Computing Convolution (CONV Layer)

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

https://medium.com/@draj0718/zero-padding-in-convolutional-neural-networks-bf1410438e99

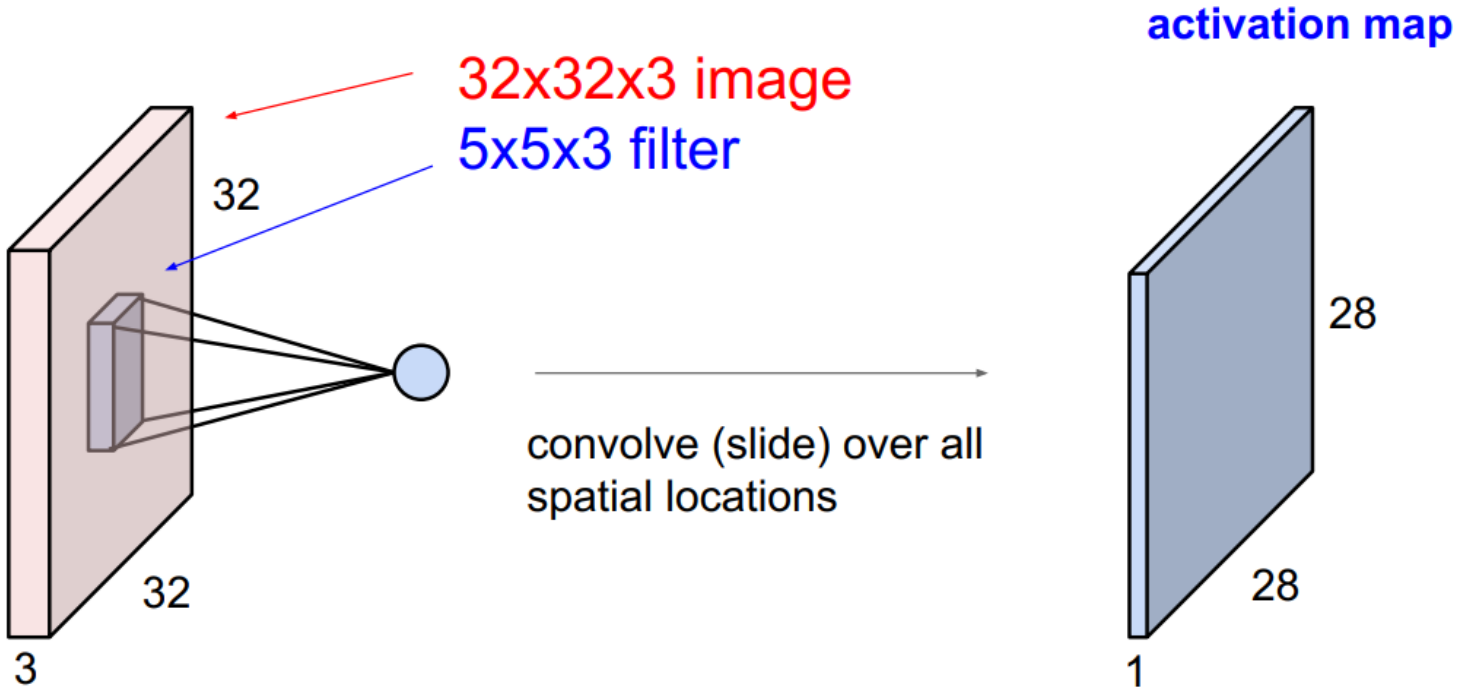
Convolution as Template Matching

- เราสามารถมอง **convolution** เป็นการทำ **template matching** ได้

https://github.com/ekapolc/pattern_course18/blob/master/slides/L09-CNN-RNN.pdf

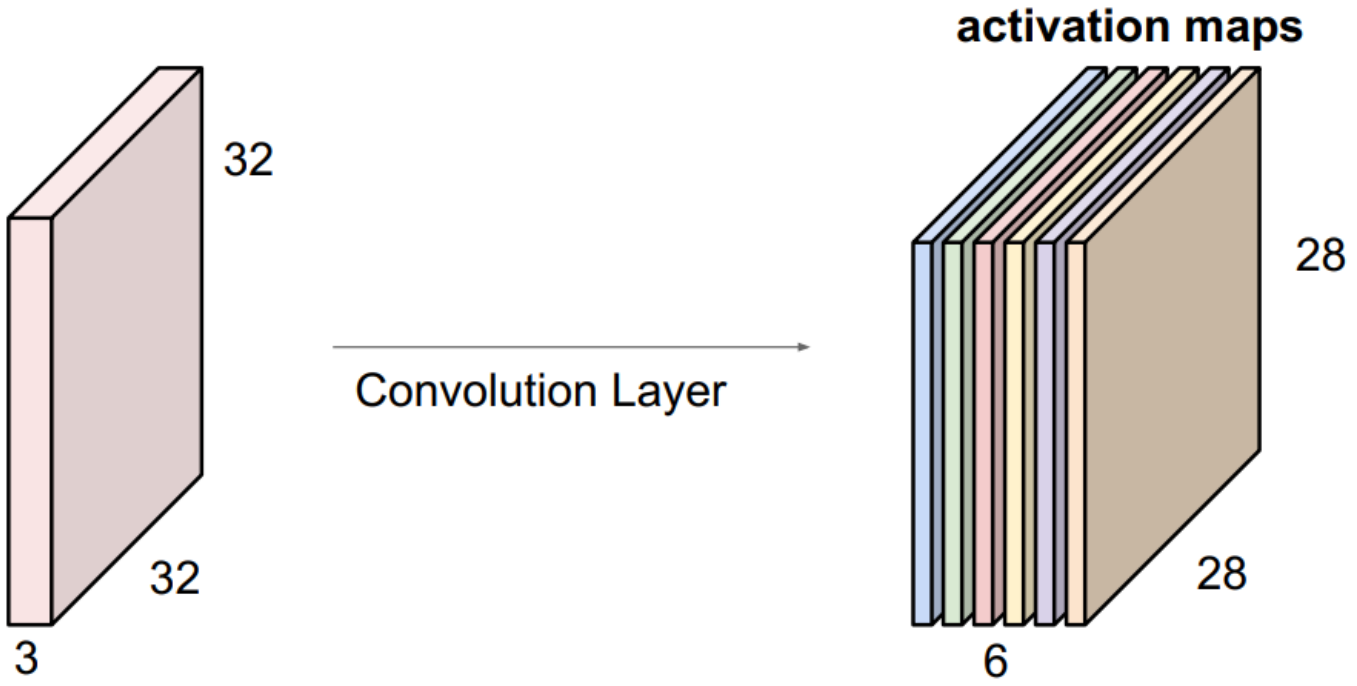
CONV Layer

- หากภาพ **input** มี **3 channel (R,G,B)** เราจะต้องใช้ **filter** ที่มีความหนา **3** ด้วย
  - นั่นคือ **depth** ของ **input** และ **filter** ต้องเท่ากันเสมอ
- สังเกตว่าการ **convolve** จะยุบมิติความหนาของ **output** กลับเป็น **1**



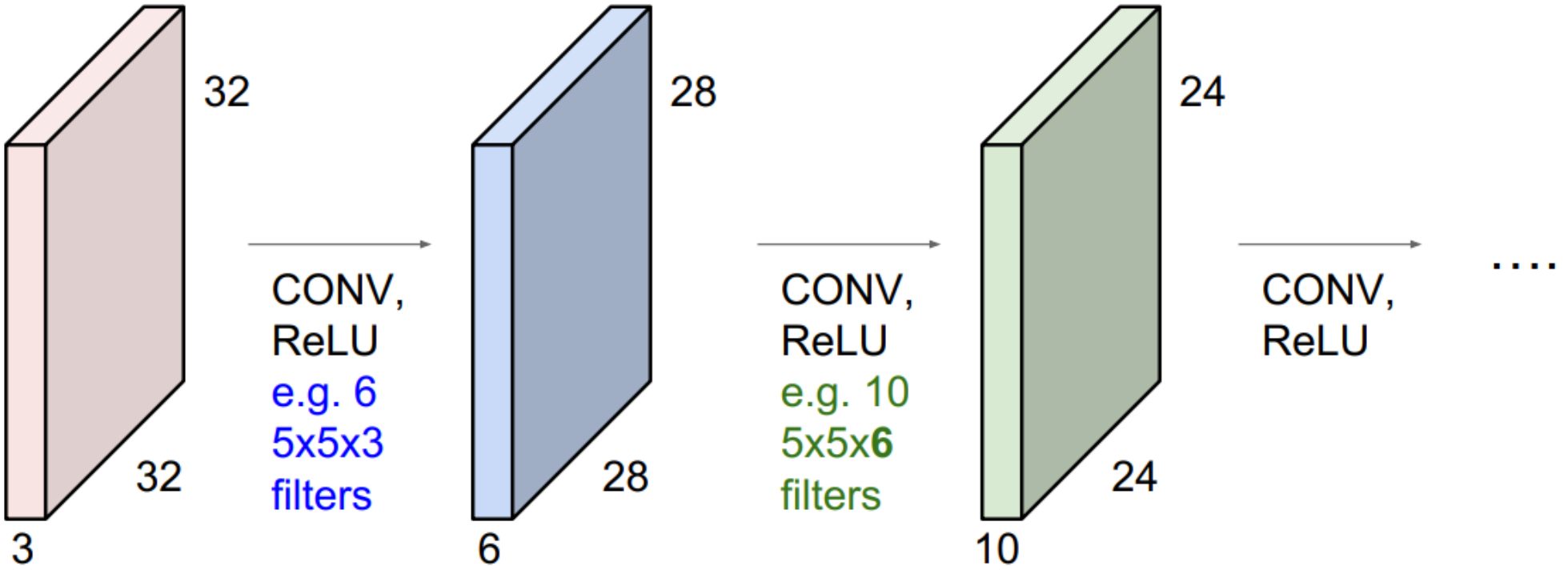
CONV Layer

- โดยปกติใน **CONV Layer** เราจะใช้ **Filter** หลายๆ ตัวช่วยกันได้
- เช่น หากเราใช้ **5x5 filter 6** ตัว ก็จะได้ผลดังภาพ

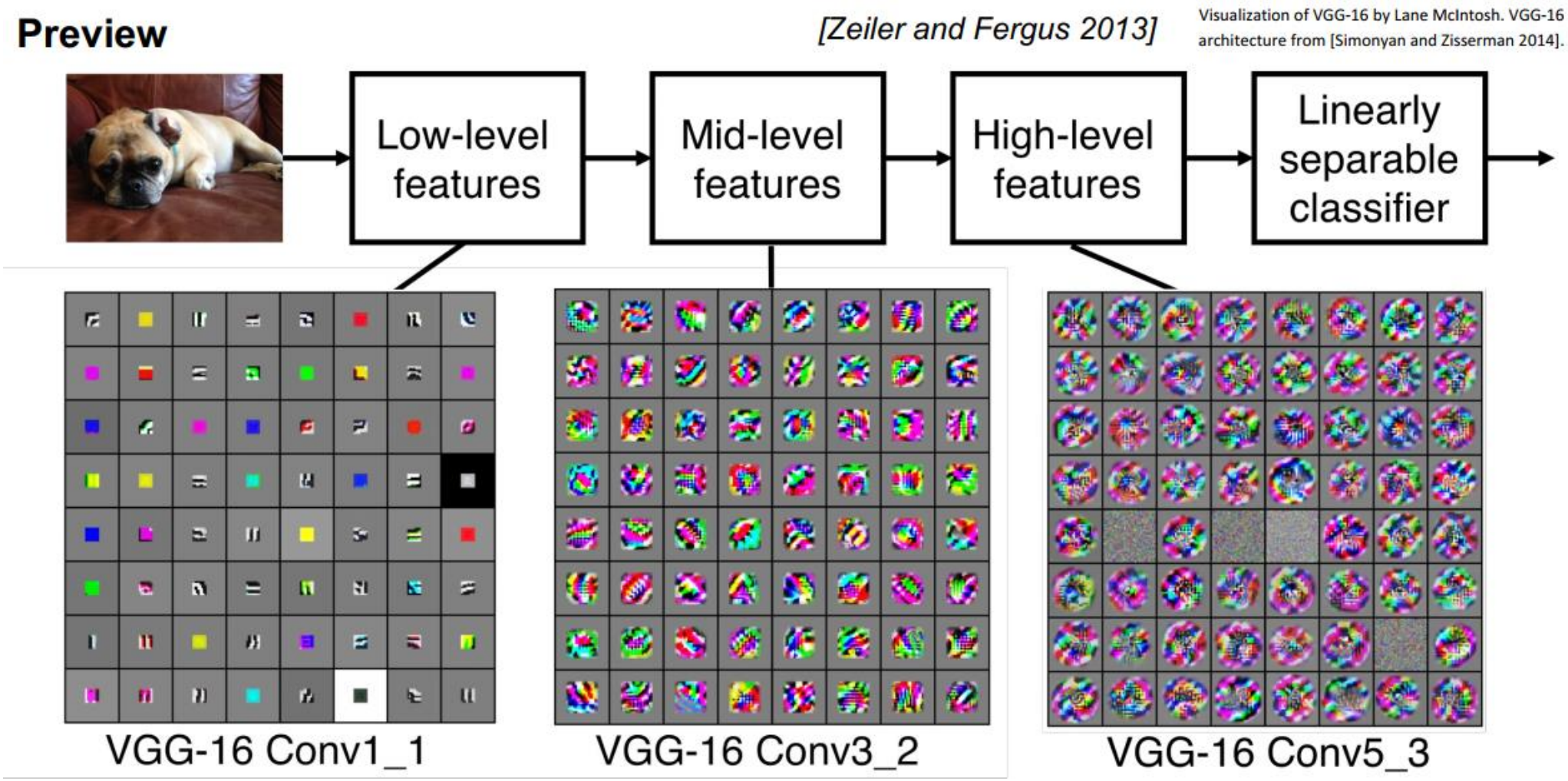


มอง **output** ที่ได้เป็นภาพขนาด **28x28x6**

Stacking CONV Layers

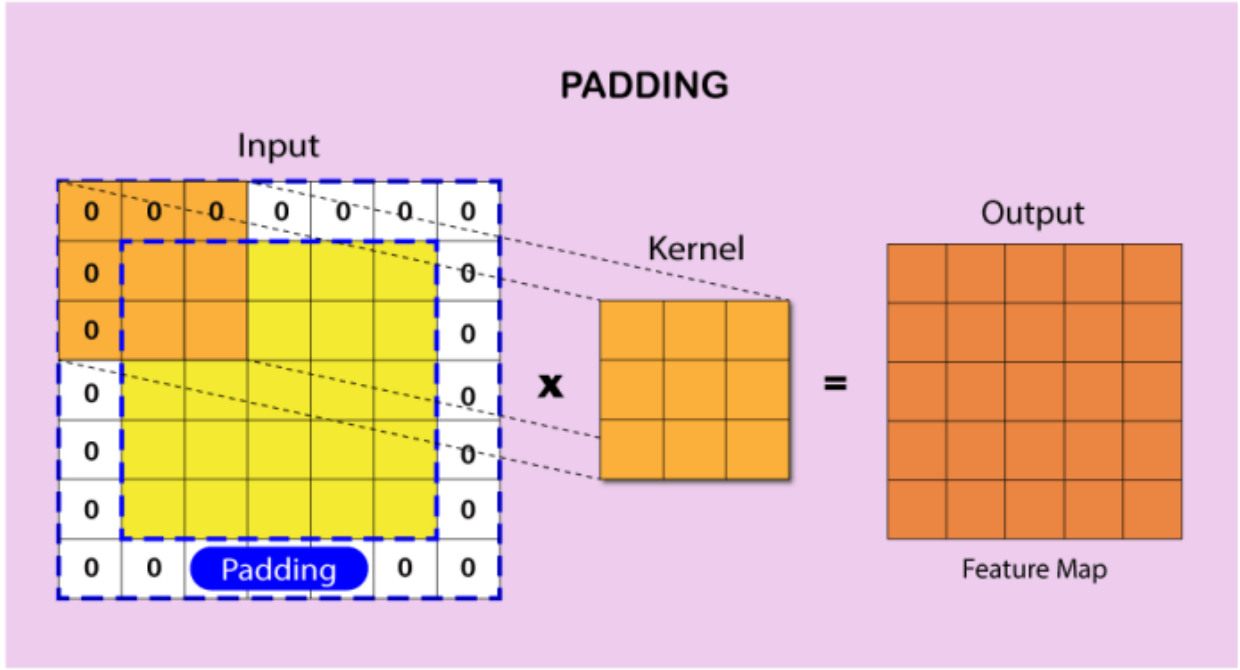


What do multiple CONV layers do?



Padding

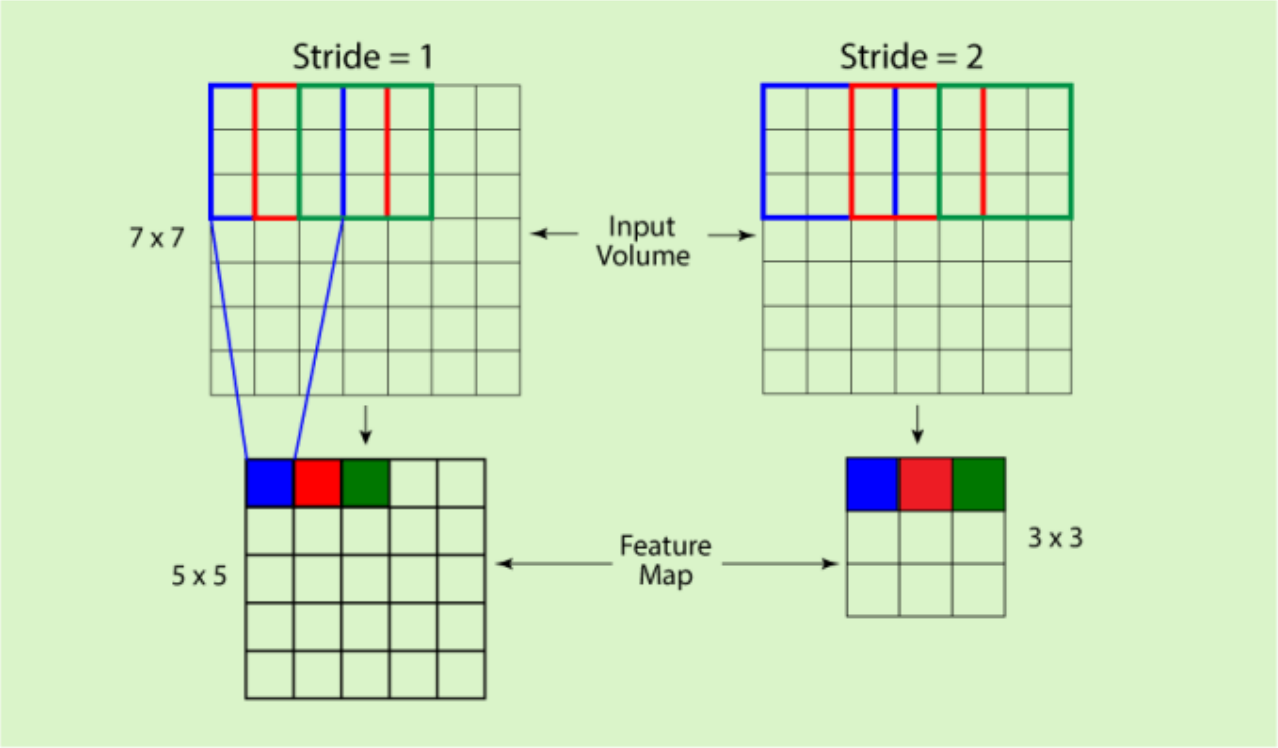
- **Padding** (ใน CONV Layer) คือการเติมค่าในบริเวณขอบของภาพ เพื่อให้ convolve แล้วได้ output ที่มีขนาดเท่าเดิมได้
  - เช่น Zero-Padding



https://developersbreach.com/convolution-neural-network-deep-learning/

Stride

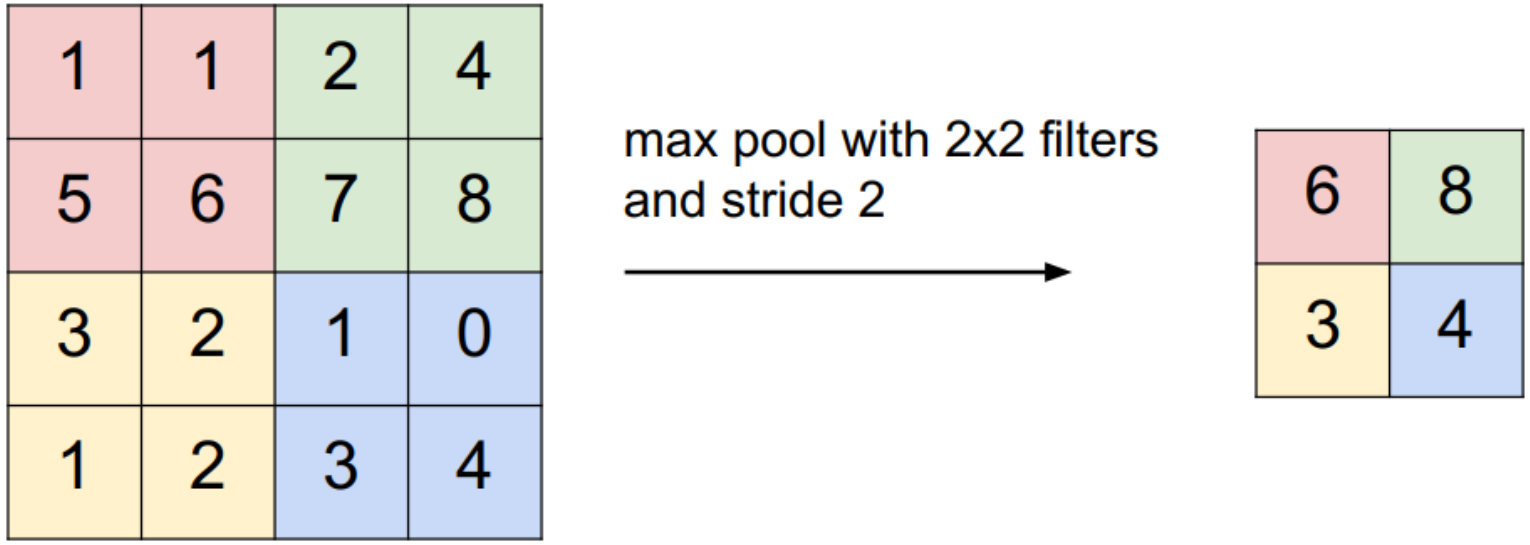
- **Stride** (ใน CONV Layer) คือ การตั้งให้ **Filter** เดินไปก้าวละหลาย **pixel** ได้
  - เพื่อประหยัดการคำนวณ ☺



https://developersbreach.com/convolution-neural-network-deep-learning/

Pooling Layer

- เรานิยมทำ **Max Pooling** หลัง **CONV Layer** เพื่อลดขนาด **output** ลง
  - ช่วยประหยัดการคำนวณไปได้มาก ☺
- **Q1:** ข้อมูลสูญหาย มีผลเสียหรือไม่?
- **Q2:** ทำไมใช้ **Max** ไม่ใช้ **Average/Min**?



Counting CNN Parameters

- จำนวน parameter ใน CONV Layer ≈ ขนาดของ Filter (+ 1 bias ต่อ Filter)
- สังเกตว่า:
  - น้อยมากเมื่อเทียบกับ fully-connected NN
  - จำนวน parameter ไม่ขึ้นกับ height/width ของภาพ input ☺
  - Pooling Layer ไม่มี parameter
- Ex:
  - จงหาจำนวน parameter ของ CONV Layer ที่ใช้ Filter ขนาด 3x3x3 จำนวน 16 ชิ้น

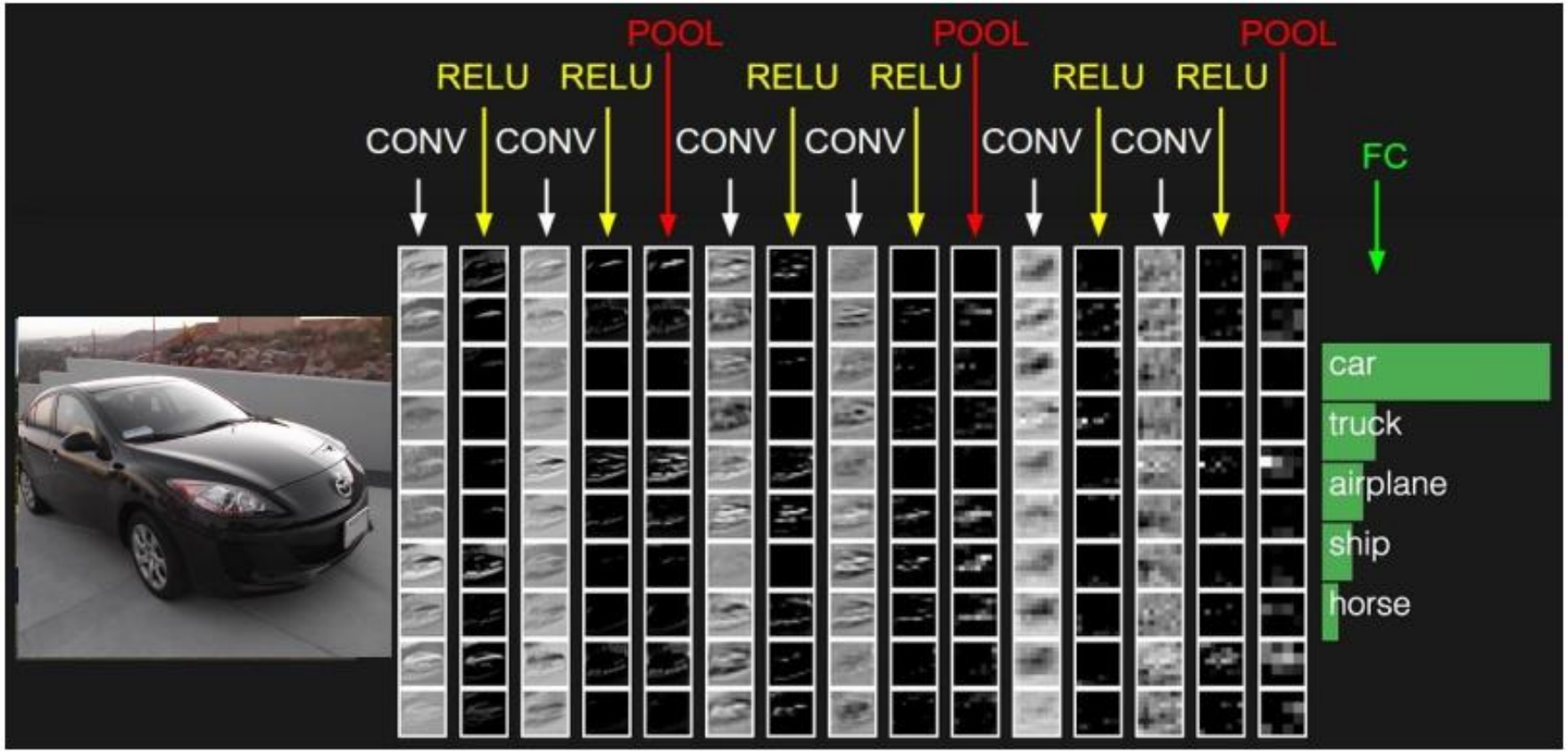
CNN Hyperparameters

- สิ่งเหล่านี้ถือเป็น hyperparameter ของ CNN
  - (CONV) Filter size
  - (CONV) Number of Filters
  - (CONV) Padding
  - (CONV) Stride
  - (POOL) Pooling Size
  - (POOL) Pooling Strategy

  - … และอื่นๆ เช่น activation function, learning rate, number of layers, ….

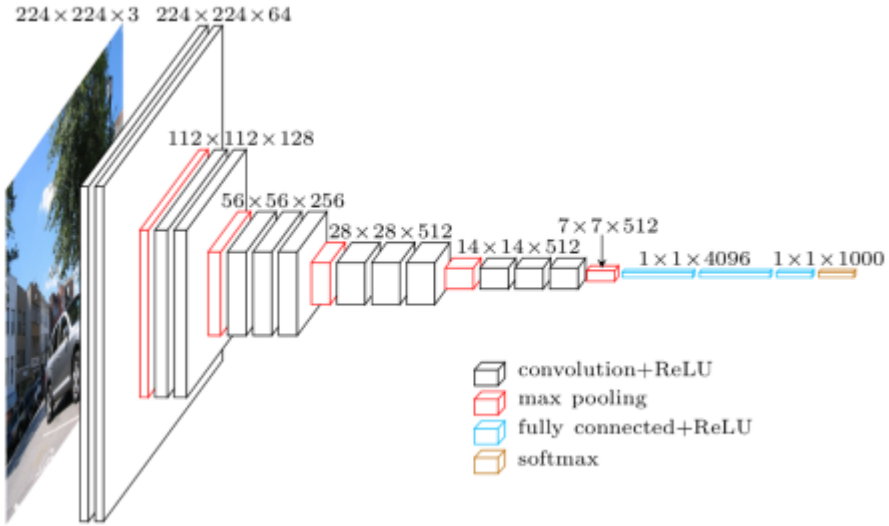
Basic CNN Structure

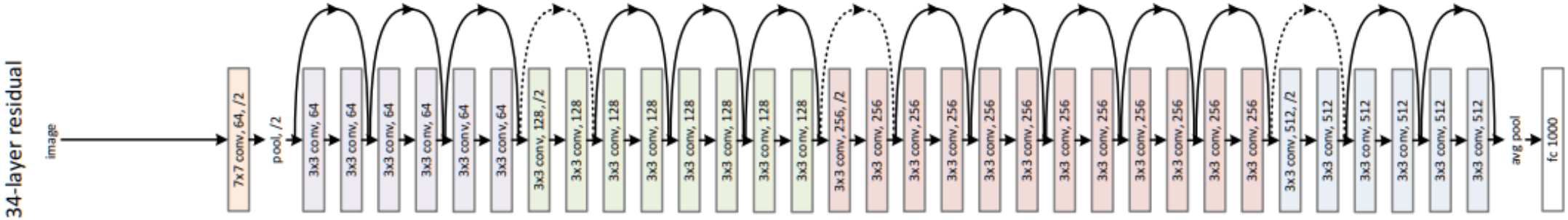
- INPUT → [CONV → RELU → CONV → RELU → POOL]*N → [FC → RELU]*M → FC → SOFTMAX



(ไม่ออกสอบ) Modern CNN Architectures

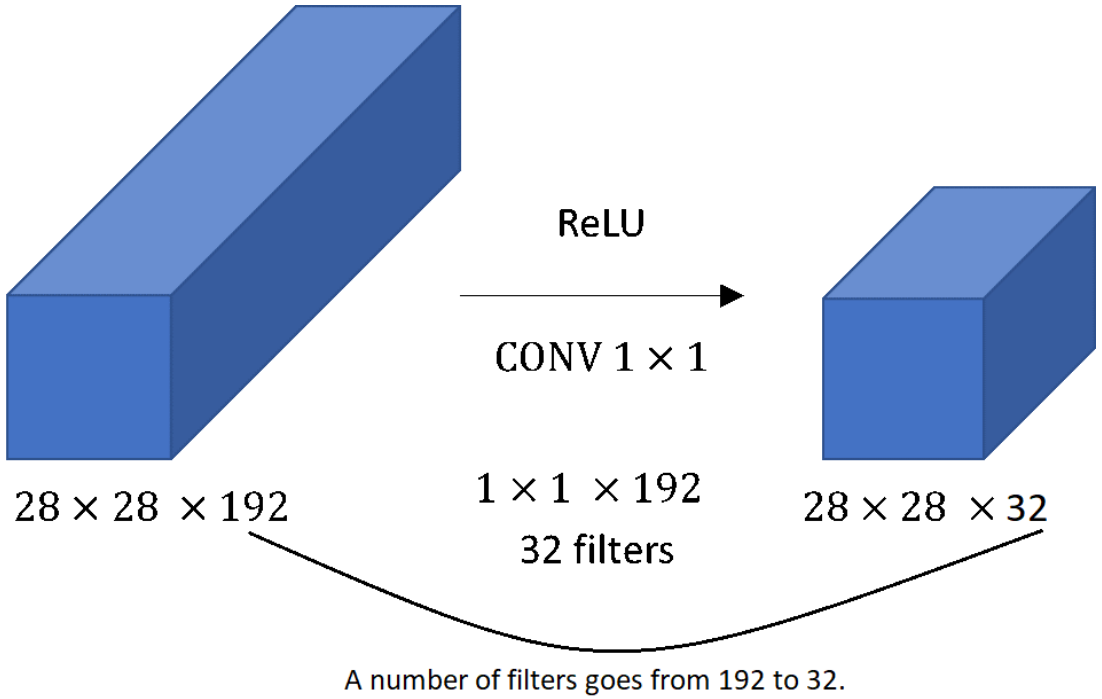
- architecture ของ CNN ในปัจจุบัน พัฒนาไปไกลมาก
  - มีจำนวน layer มากเป็น 100+
  - มีความซับซ้อนสูง และมีกลไกพิเศษ เช่น Skip Connection
- หากใช้ CNN เพื่อสกัดฟีเจอร์ภาพ ควรใช้ตัวที่เป็นที่นิยม เช่น
  - 2014: VGG, Inception
  - 2015: ResNet
  - 2017: MobileNet
  - 2019: EfficientNet





(ไม่ออกสอบ) 1x1 Convolution

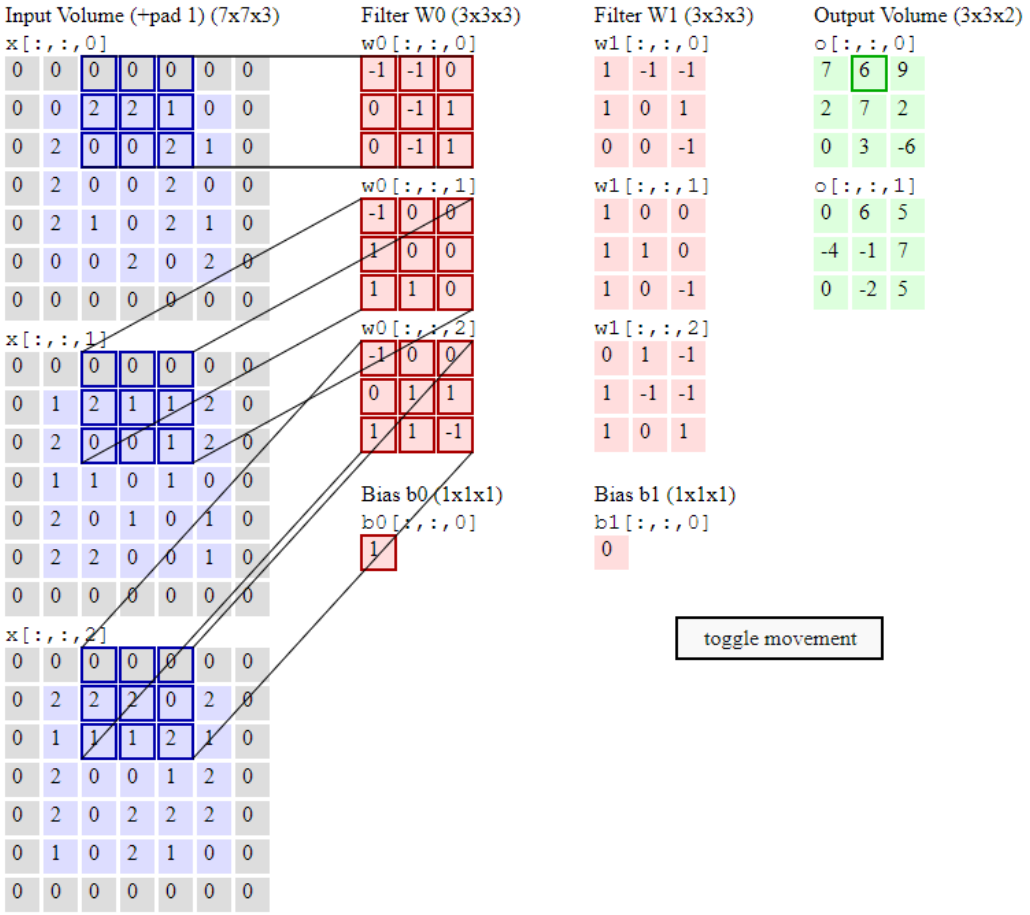
- **1x1 Convolution** อาจดูแปลกตา แต่จริงๆ แล้วมีประโยชน์ในการช่วยยุบ/ลดมิติ **channel** ของ **input** ได้



https://datahacker.rs/introduction-into-1x1-convolution/

Recap: Convolution Computation

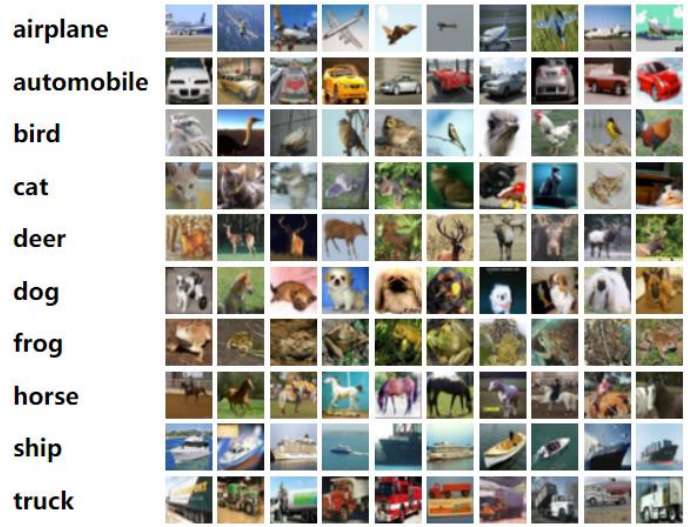
- https://cs231n.github.io/convolutional-networks/



Lab: CIFAR10 with CNN

Lab: CIFAR10 with CNN

1. ให้ copy โค้ดของ lab ที่แล้ว (MNIST with Feedforward NN) มาเป็นจุดเริ่มต้น
2. เปลี่ยนไปใช้ชุดข้อมูล CIFAR10 แล้วสำรวจรูปภาพ
   - tensorflow.keras.datasets.cifar10
3. ลองสำรวจข้อมูลรูปภาพ
   - x_train.shape
   - x_test.shape
   - plt.imshow(x_train[0])
   - print(y_train[0])
4. ทดลองฝึก Feedforward NN กับข้อมูลชุดใหม่โดยไม่ต้องแก้ไขโค้ดใดๆ
   - (!) เพิ่ม step การ normalize ค่า pixel ภาพโดยการหารด้วย 255
   - สังเกตค่า loss, train accuracy, test accuracy



ทดลองปรับเปลี่ยน model ให้เป็นแบบ CNN

5. เริ่มจากใส่ layers.Conv2D โดยใช้ filter ขนาด 3x3 จำนวน 1 filter
   โดย Conv2D layer แรกสุดจะต้องตั้ง input_shape ให้ตรงกับขนาดภาพ input
   - Q: เราควรใส่ layer ดังกล่าวไว้ตรงบรรทัดใดในโค้ด? ก่อนหรือหลัง Flatten()?
   - ใช้คำสั่ง model.summary() ในการตรวจสอบจำนวน parameter ใน layer CONV
   - ทำการ train ใหม่แล้วสังเกตค่า train/test accuracy
   - หาก train accuracy ยังมีแนวโน้มเพิ่มขึ้นได้อีก อาจลองเพิ่ม epochs เป็น 7 หรือ 10

ทดลองปรับเปลี่ยนโครงสร้างของ CNN

6. ทดลองปรับเปลี่ยนโครงสร้างของ CNN ดังต่อไปนี้

   โดยในแต่ละข้อ ให้สังเกตการเปลี่ยนแปลงของ train/test acc และจำนวน parameters

   - ลองเพิ่มจำนวน filter ของ CONV Layer จาก 1 เป็น 6 และ 32
   - ลองเพิ่ม CONV Layer อีก 1 ชั้น เป็น CONV-CONV
   - ลองใส่ Max Pooling ด้วย layers.MaxPooling2D ในหลายๆ รูปแบบ เช่น
     - CONV-CONV-MAXPOOL
     - CONV-MAXPOOL-CONV-MAXPOOL
   - ลองลดขนาด Dense Layer ให้เหลือ 1 Layer
   - ลองลดจำนวน neuron ใน Dense Layer ให้เหลือ 64
   - ลองใช้ CONV(32)-MAXPOOL-CONV(64)-MAXPOOL-CONV(64)-MAXPOOL

- Q: โมเดลใดได้ test accuracy ที่ดีที่สุด โดยที่ใช้จำนวนพารามิเตอร์ไม่มากเกินไป?

คำศัพท์ที่ควรรู้เกี่ยวกับการ implement NN

- เรามักเก็บข้อมูลไว้ในตัวแปรชนิด tensor
  - 2D tensor
    - สำหรับเก็บ tabular data (samples, features)
  - 3D tensor
    - สำหรับเก็บ time series data (samples, timesteps, features)
  - 4D tensor
    - สำหรับเก็บ images (samples, height, width, channel)
  - 5D tensor
    - สำหรับเก็บ videos (samples, frames, height, width, channel)

คำศัพท์ที่ควรรู้เกี่ยวกับการ implement NN

- batch size
  - จำนวนข้อมูลที่ส่งเข้าไปในแต่ละ iteration (default: 32) สำหรับทำ mini-batch SGD
- iteration
  - การส่งข้อมูล 1 batch เข้าไปใน NN เพื่อทำการ train ปรับค่า param (backprop)
- epoch
  - การส่งข้อมูล training set เข้าไปใน NN เพื่อทำการ train ครบทั้ง training set 1 รอบ
- **ตัวอย่าง:**
  - จำนวน training data: 50,000 ภาพ
  - batch_size: 32 (default) ➔ แสดงว่าในการ train ส่งข้อมูล iteration ละ 32 ภาพ
  - ดังนั้น ใน 1 epoch จึงต้องทำทั้งหมด 50,000/32 = 1563 iterations
    - จึงจะส่งข้อมูลให้ NN ครบทั้ง 50,000 ภาพ
  - ปกติเราอาจต้อง train หลาย epoch จึงจะได้ loss ที่พอใจ