Introduction to Machine Learning (CS419M)

Lecture 13:

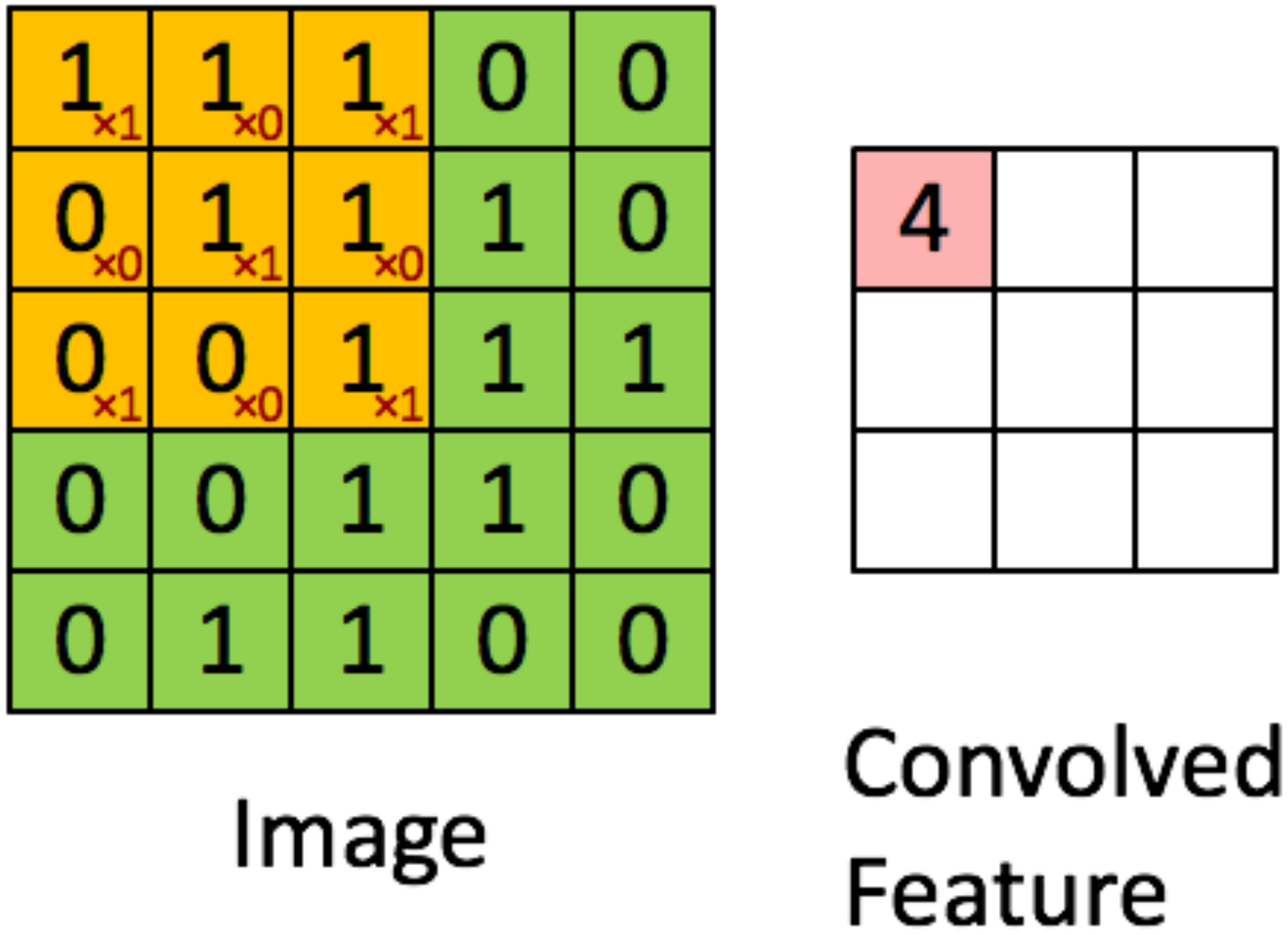
- Convolutional Neural Networks

Mar 13, 2020

Convolutional Neural Networks (CNNs)

- Fully connected (dense) layers have no awareness of spatial information
- Key concept behind convolutional layers is that of ***kernels*** or ***filters***
- Filters slide across an input space to detect spatial patterns (translation invariance) in local regions (locality)

Convolution Layer



Convolution Layer

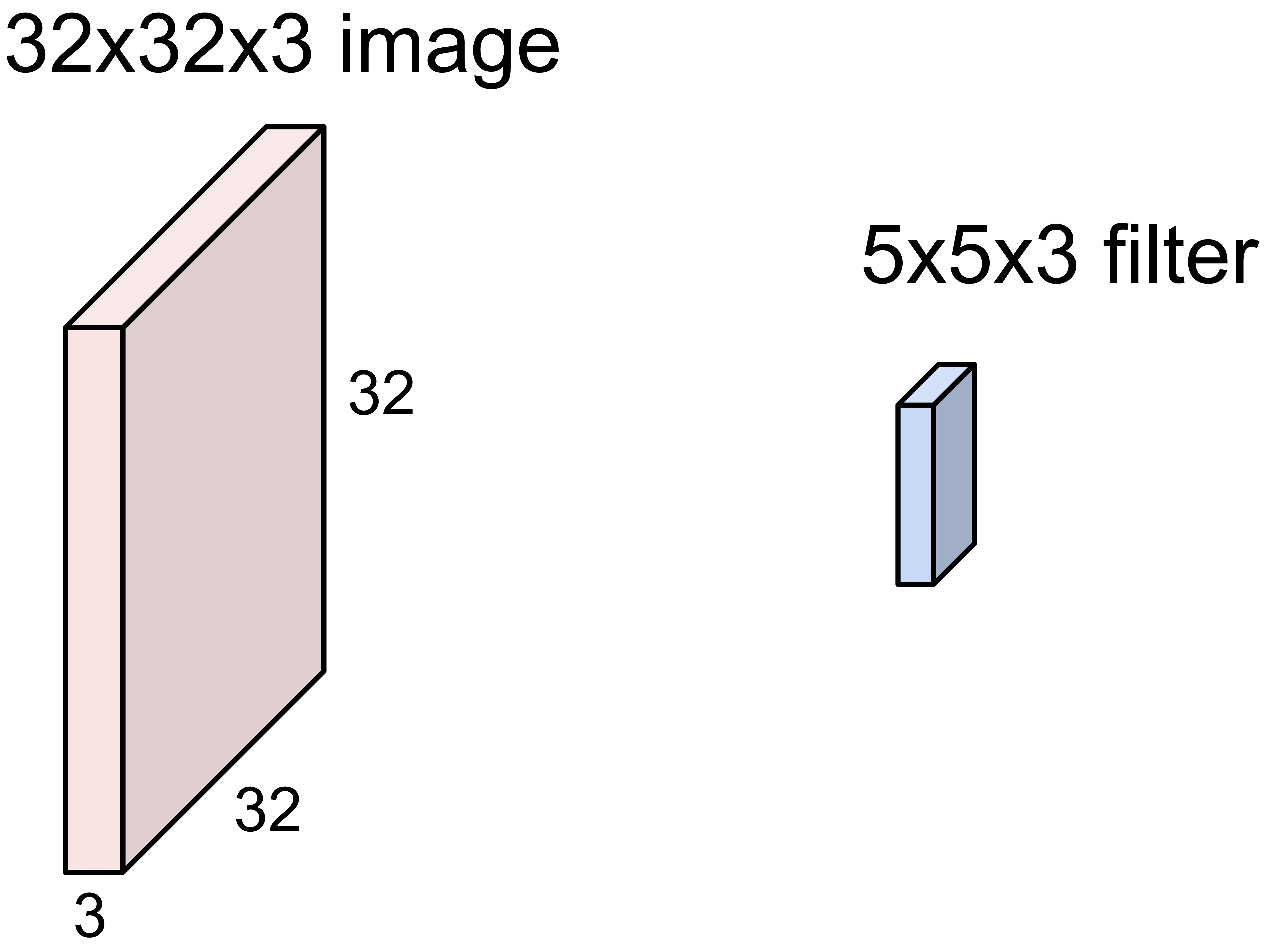
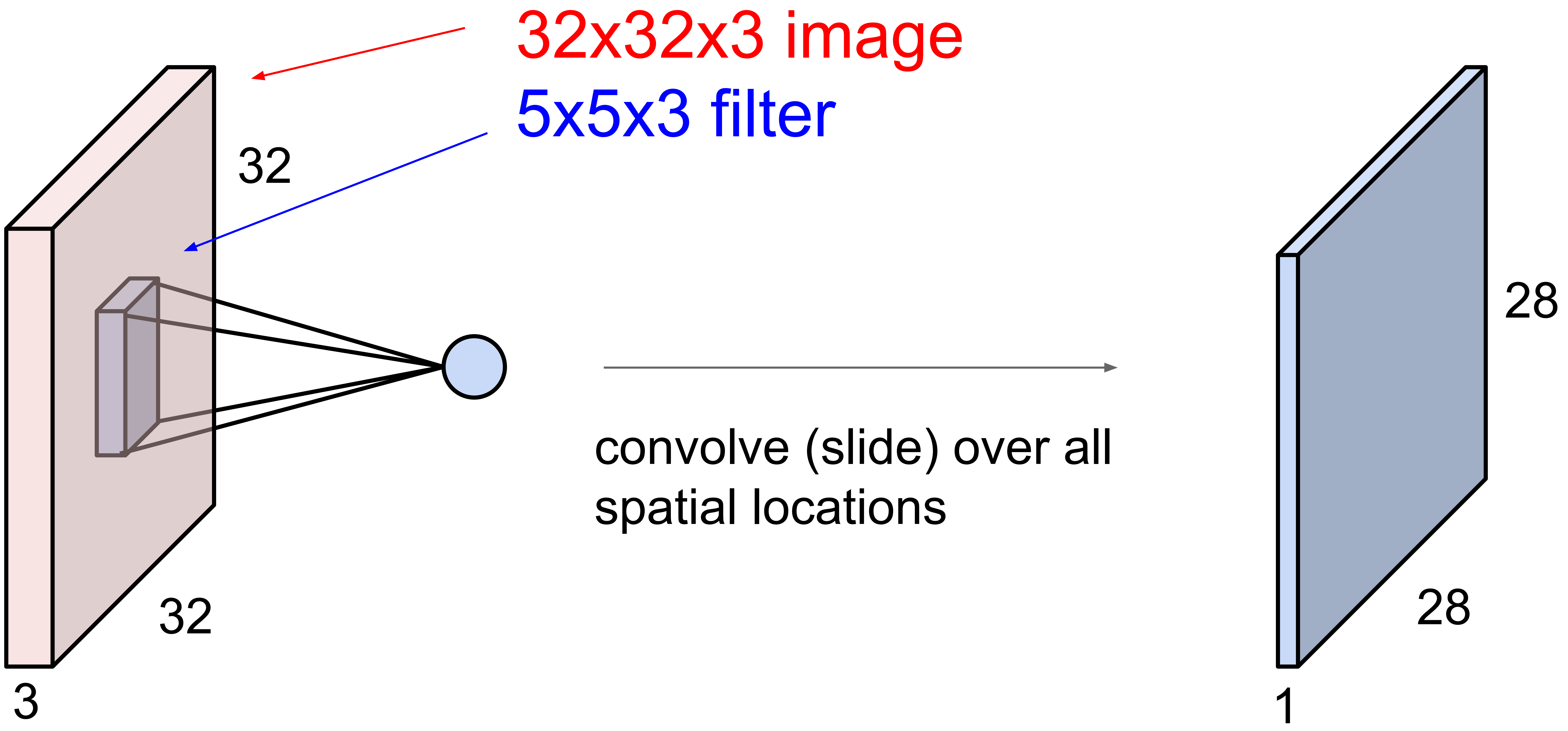


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer



Convolution Layer

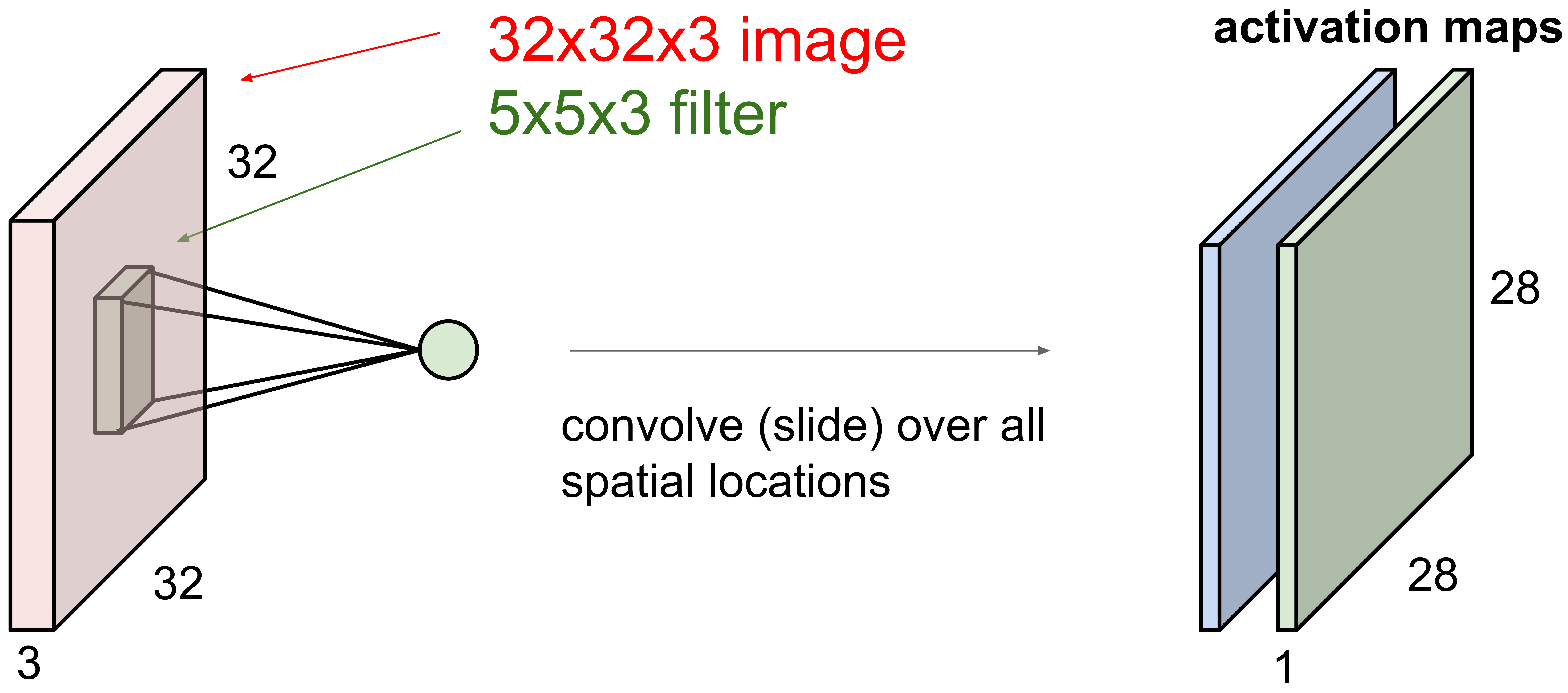


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolution Layer

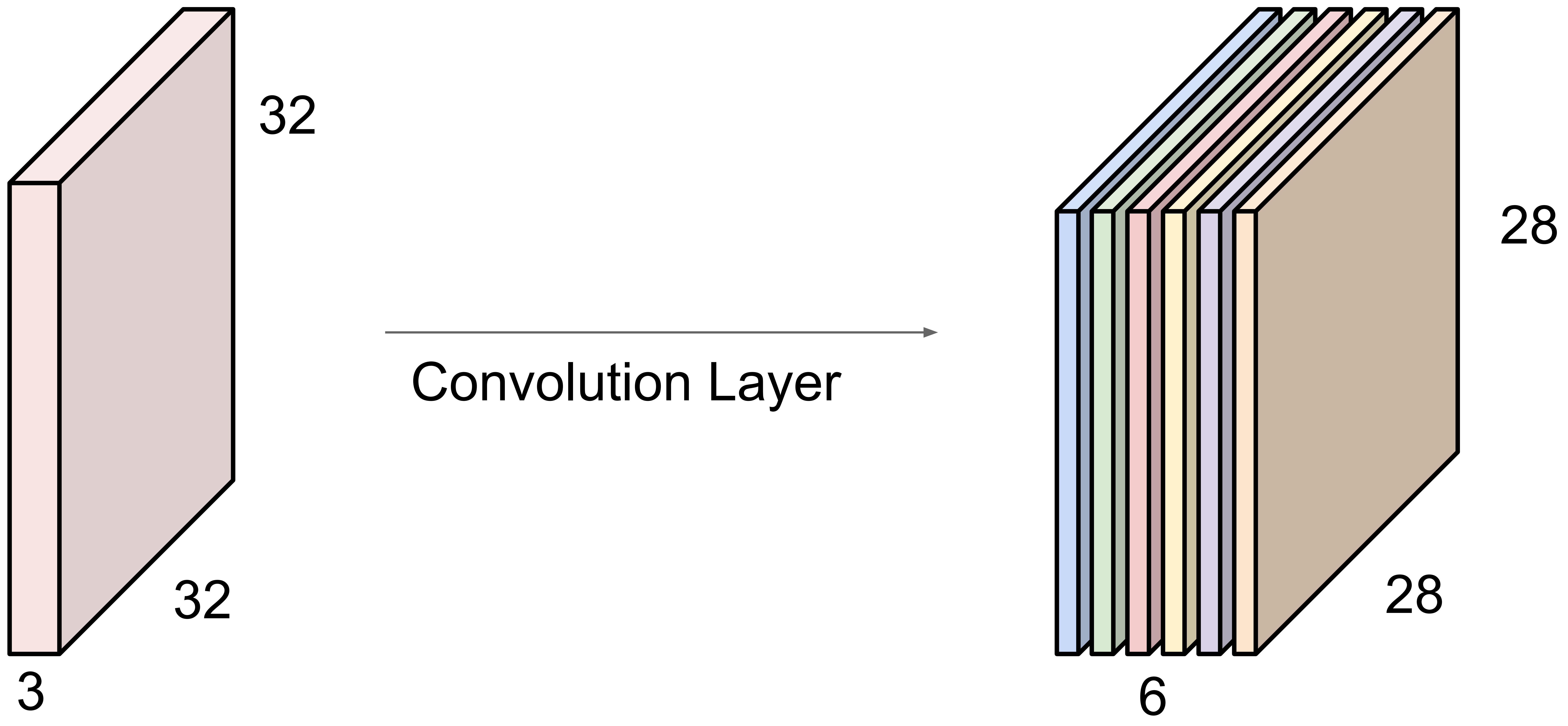


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

Convolutional Neural Network

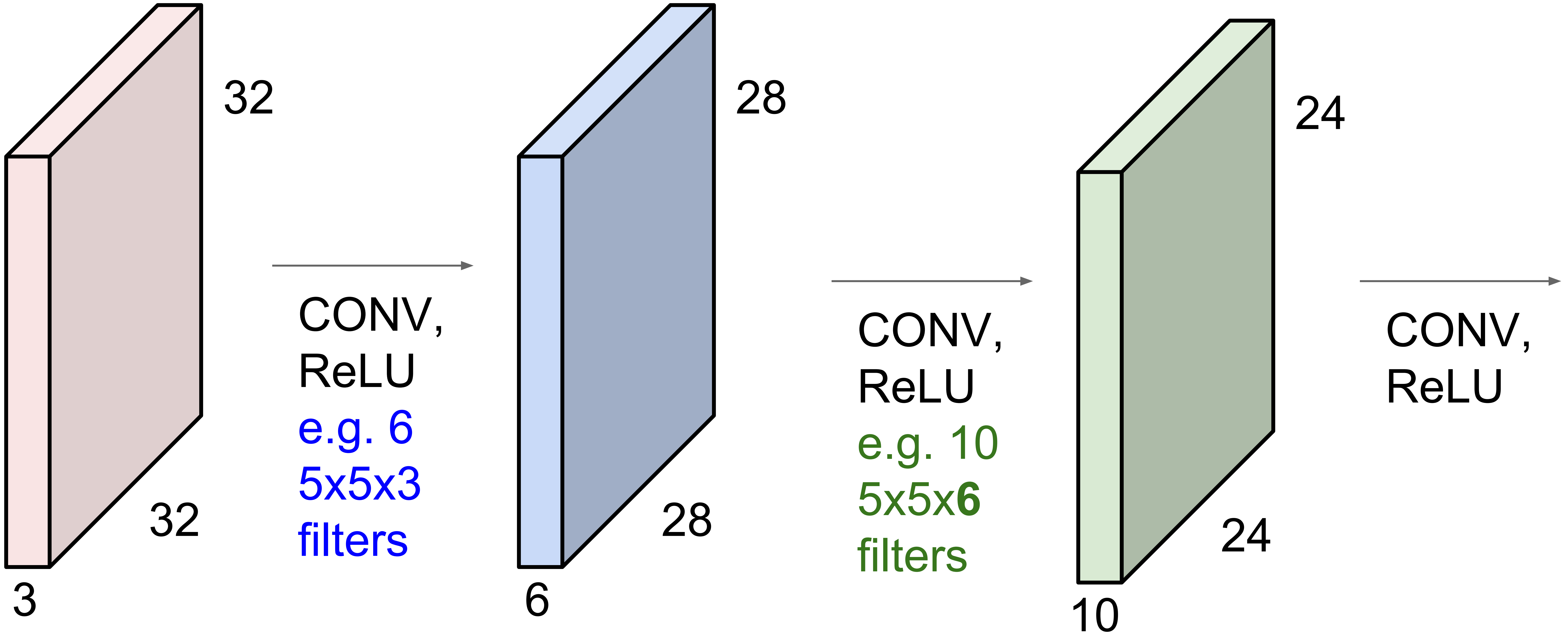


Image from:http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf

What do these layers learn?

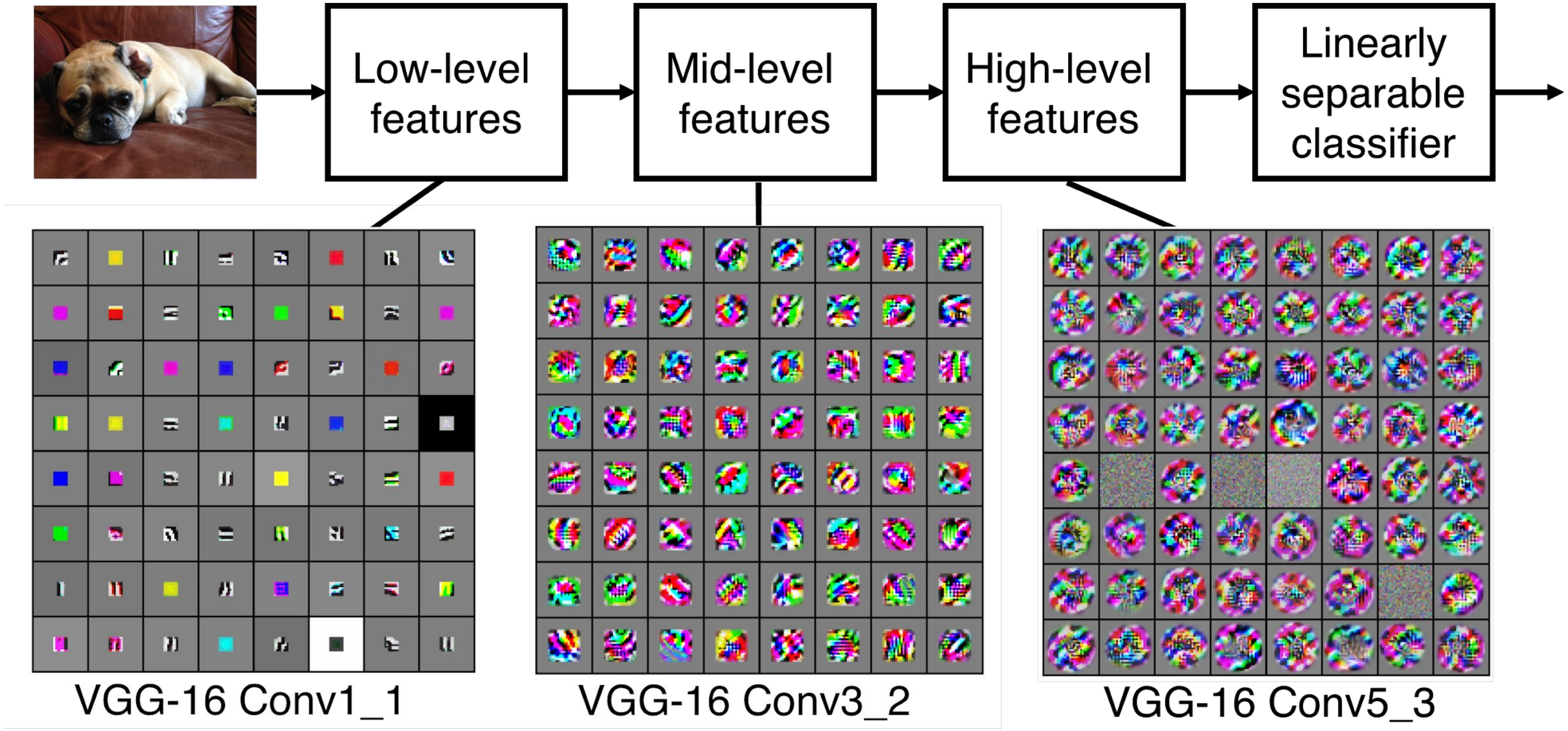
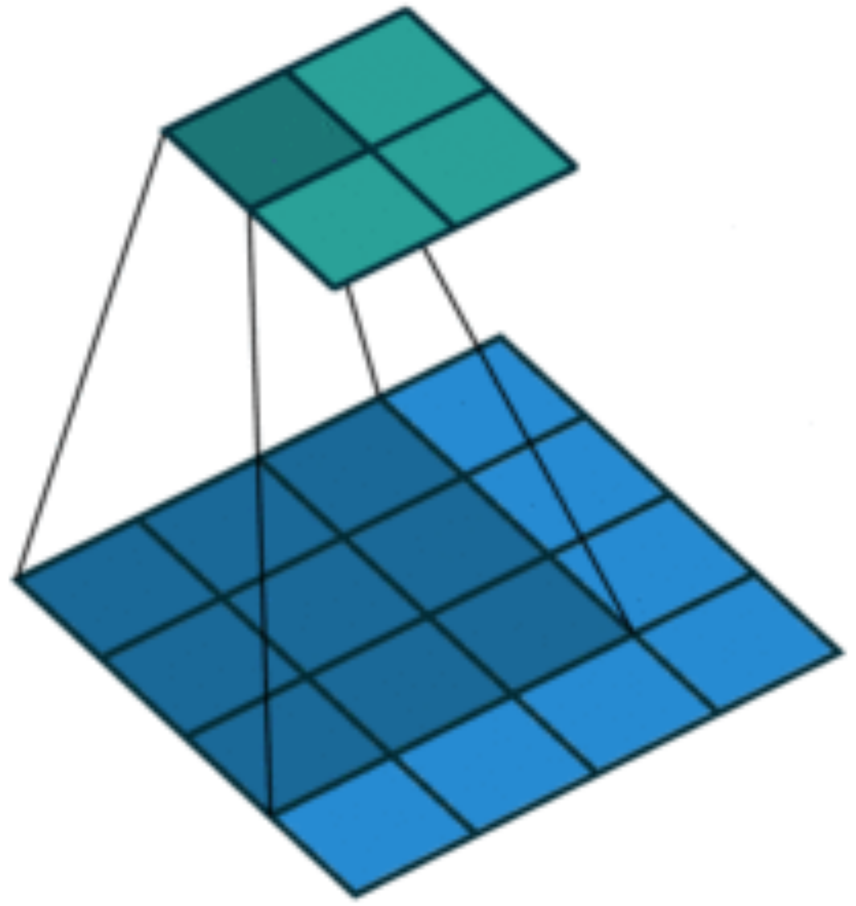


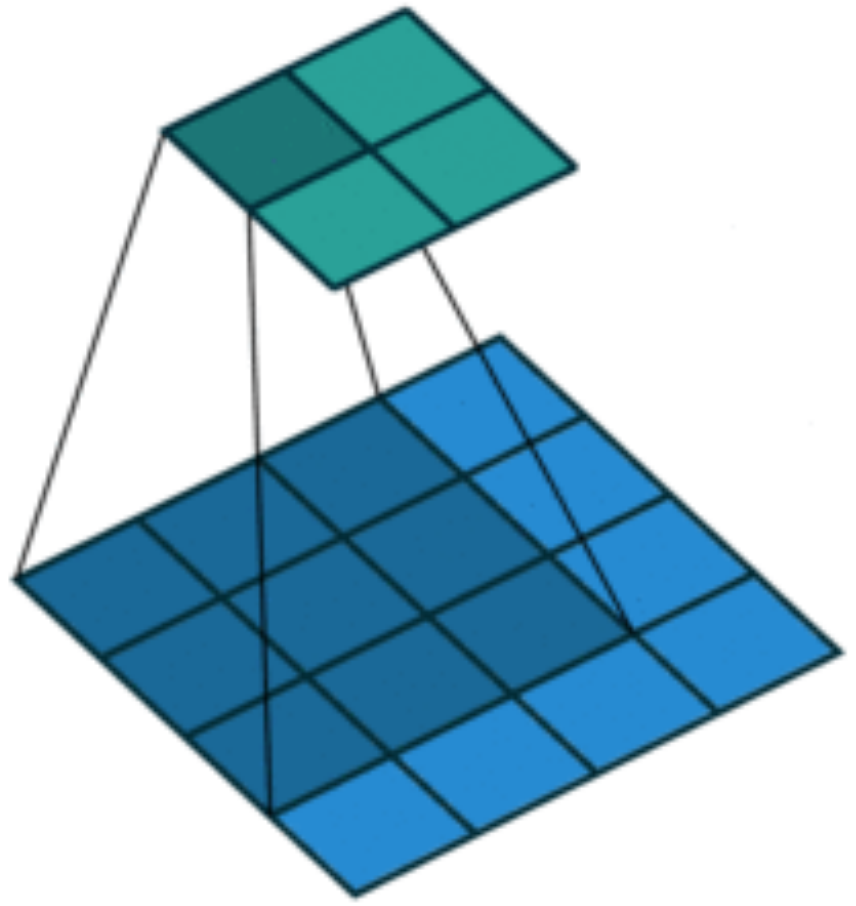
Image from: Simonyan and Zisserman, 2014

Convolutional Neural Networks (CNNs)

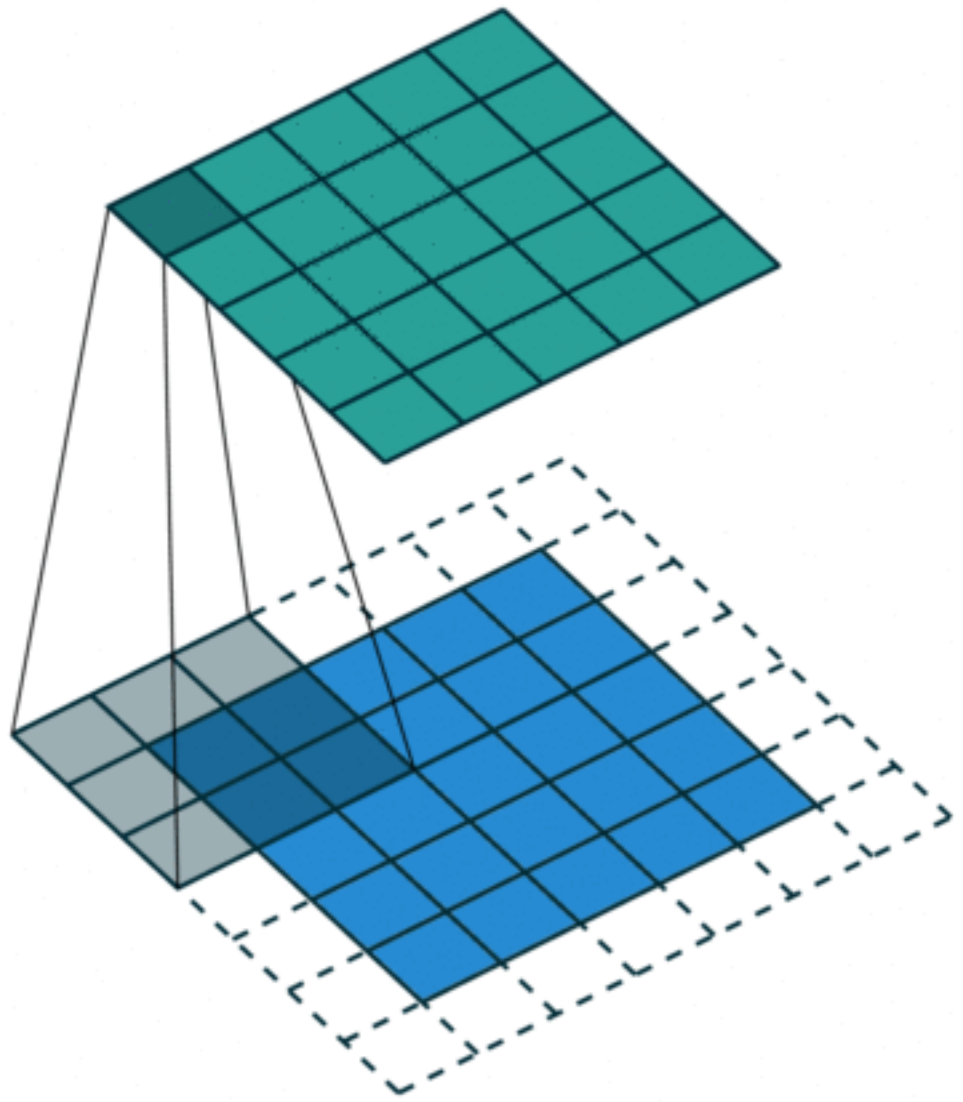


Stride=1, No padding

Convolutional Neural Networks (CNNs)



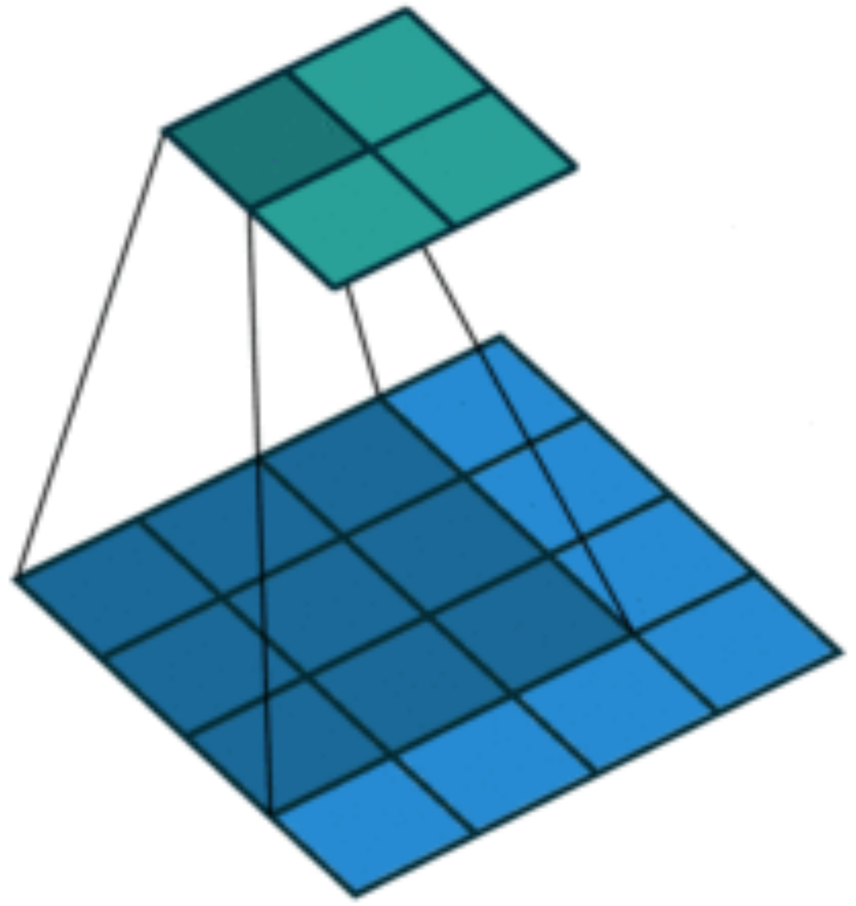
Stride=1, No padding



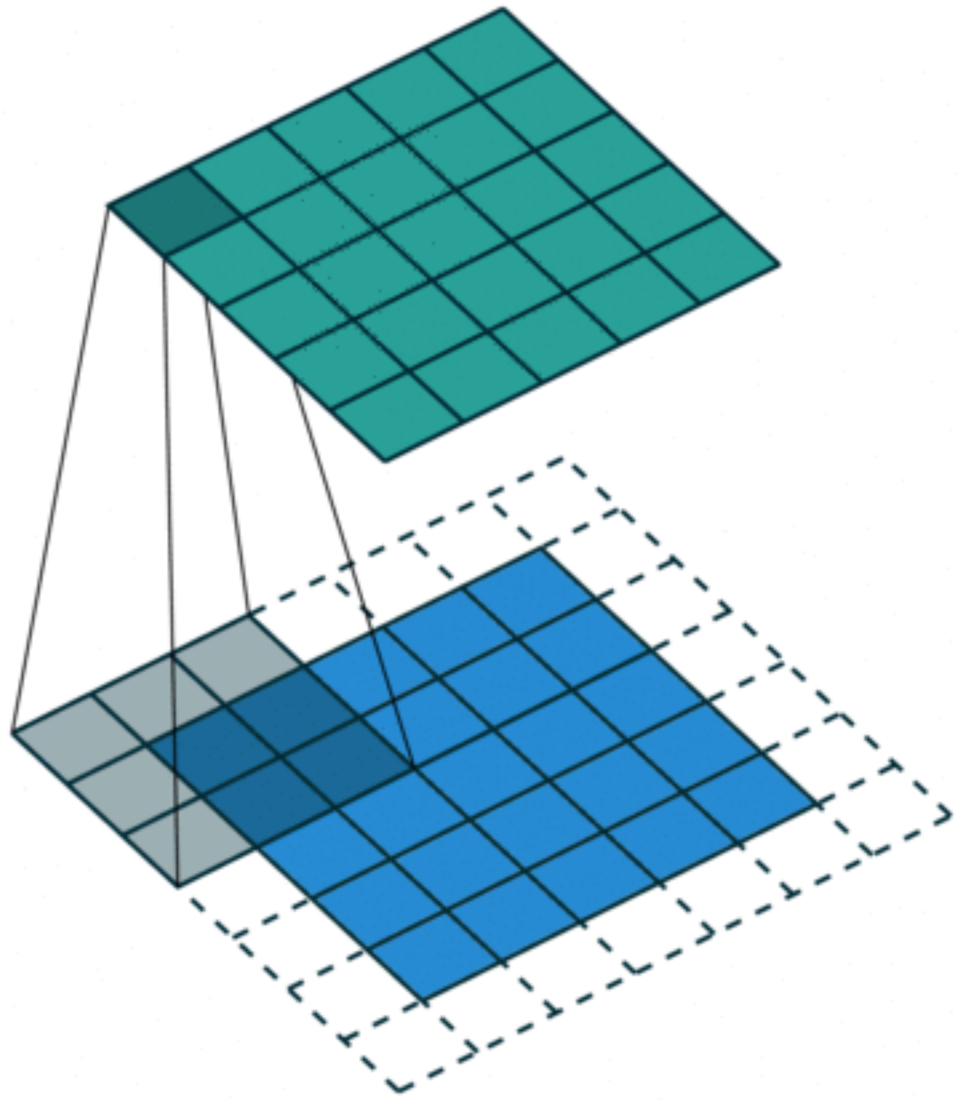
Stride=1, Padding, P=1

All animations are from: https://github.com/vdumoulin/conv_arithmetic

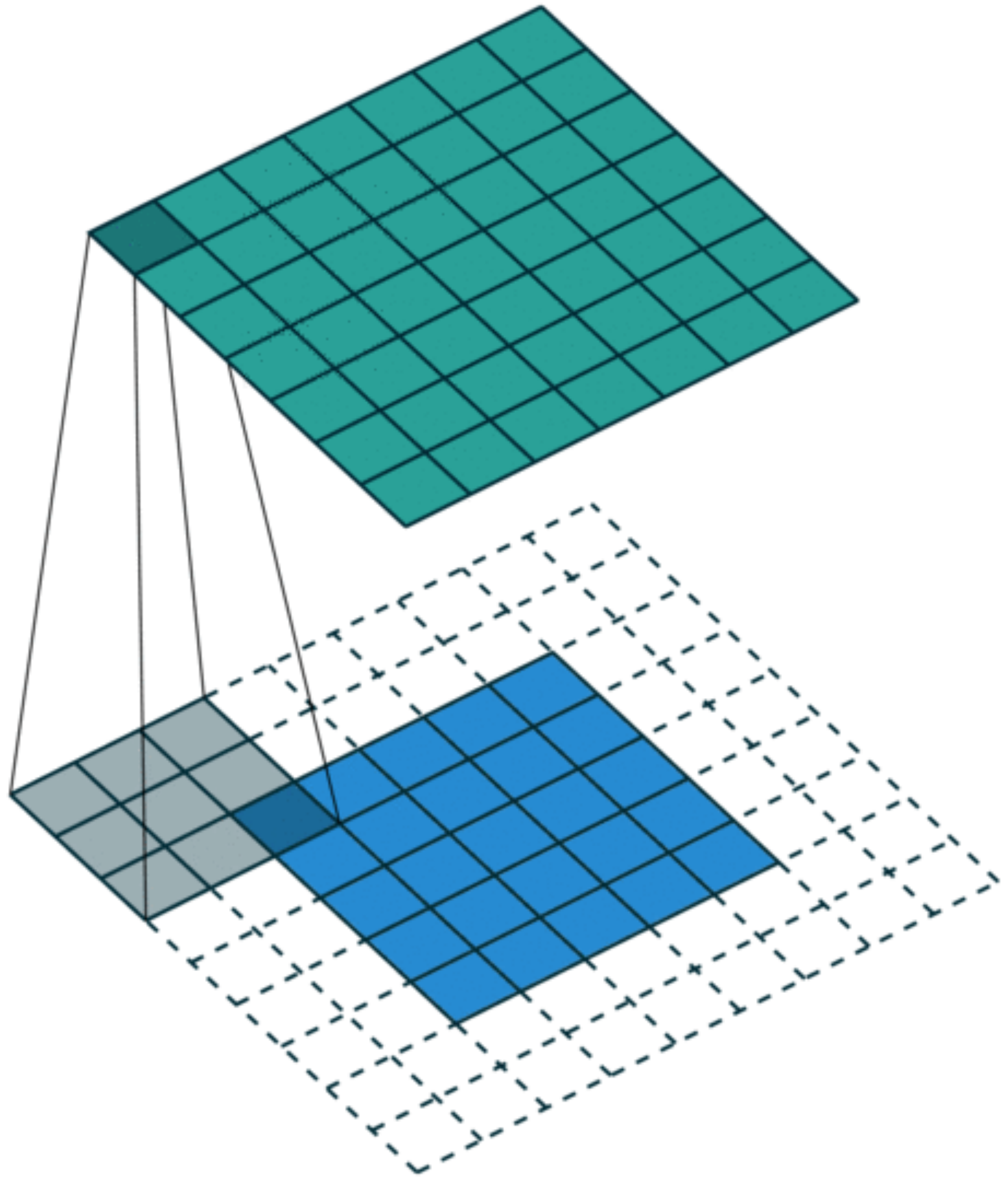
Convolutional Neural Networks (CNNs)



Stride=1, No padding



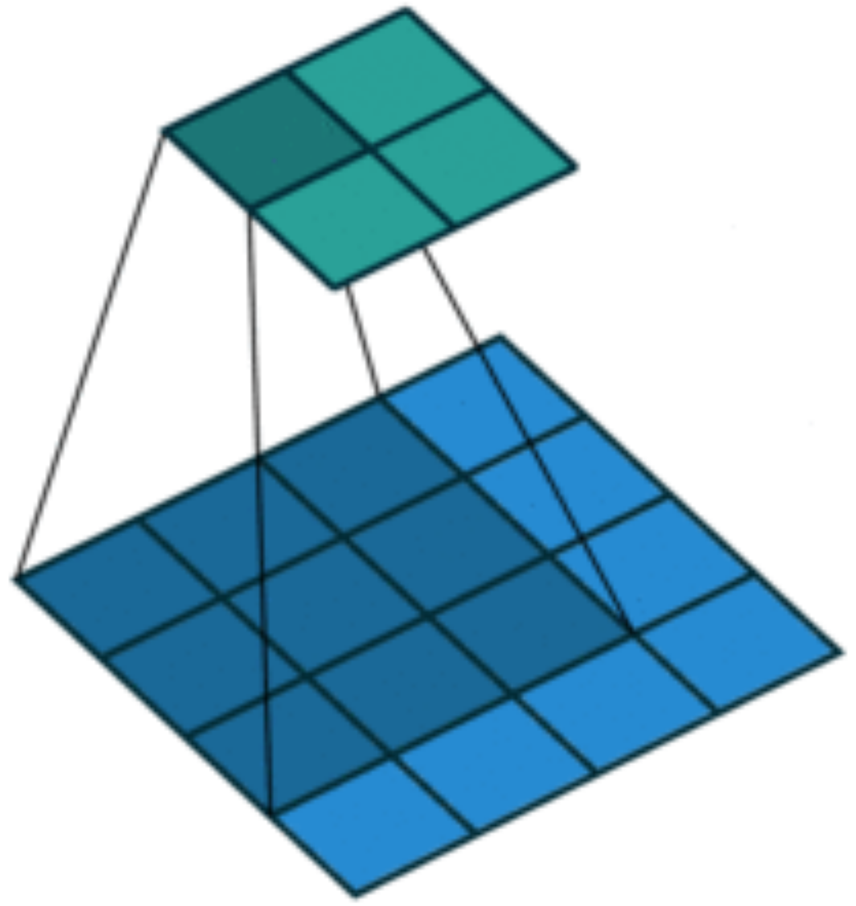
Stride=1, Padding, P=1



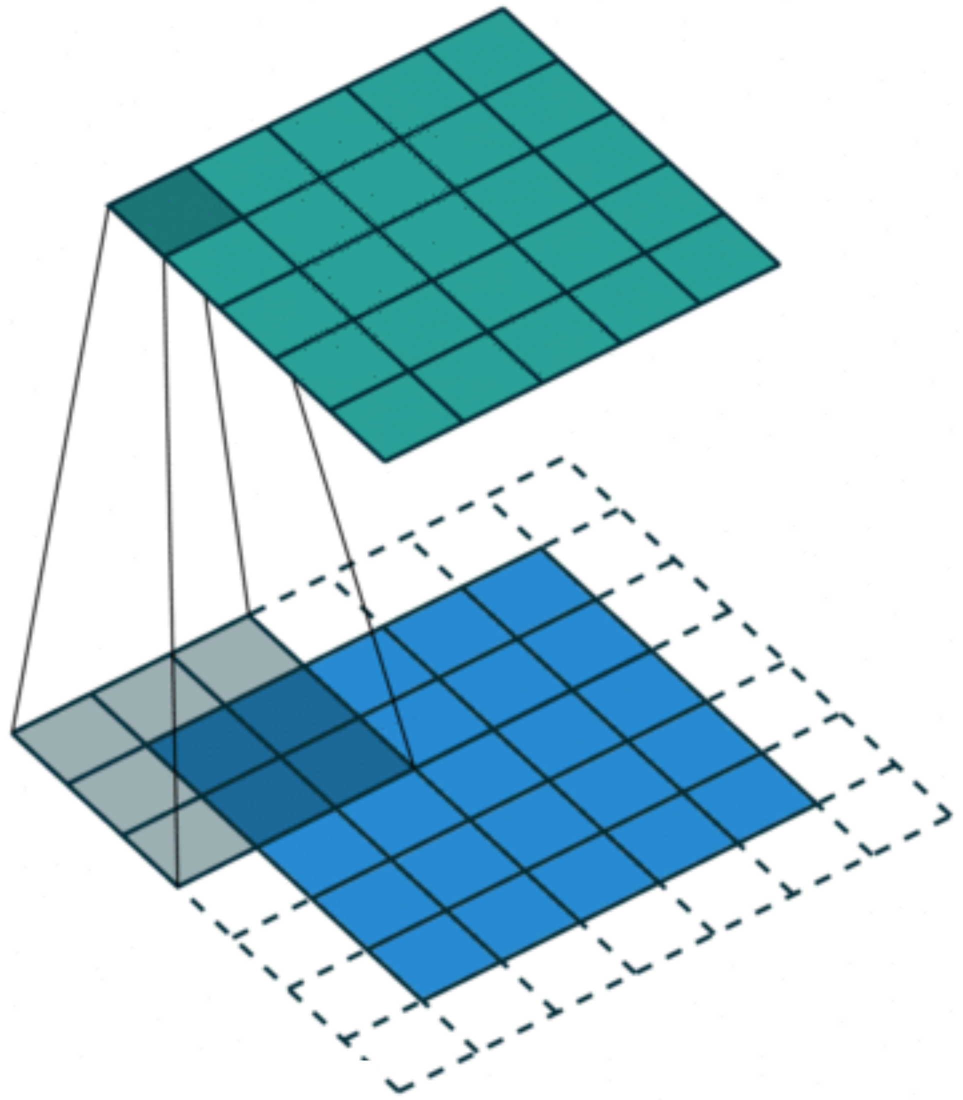
Stride=1, Padding, P=2

All animations are from: https://github.com/vdumoulin/conv_arithmetic

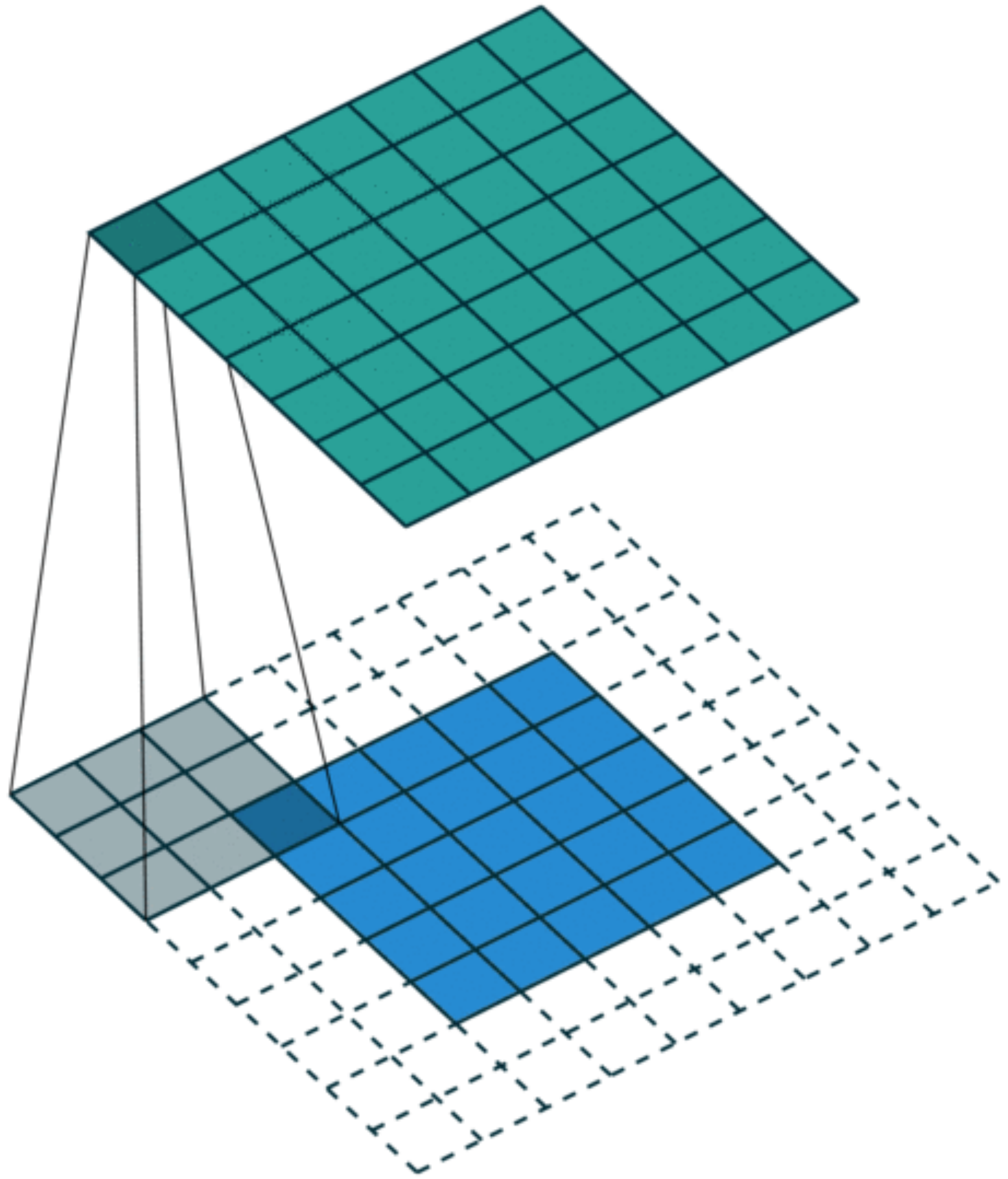
Convolutional Neural Networks (CNNs)



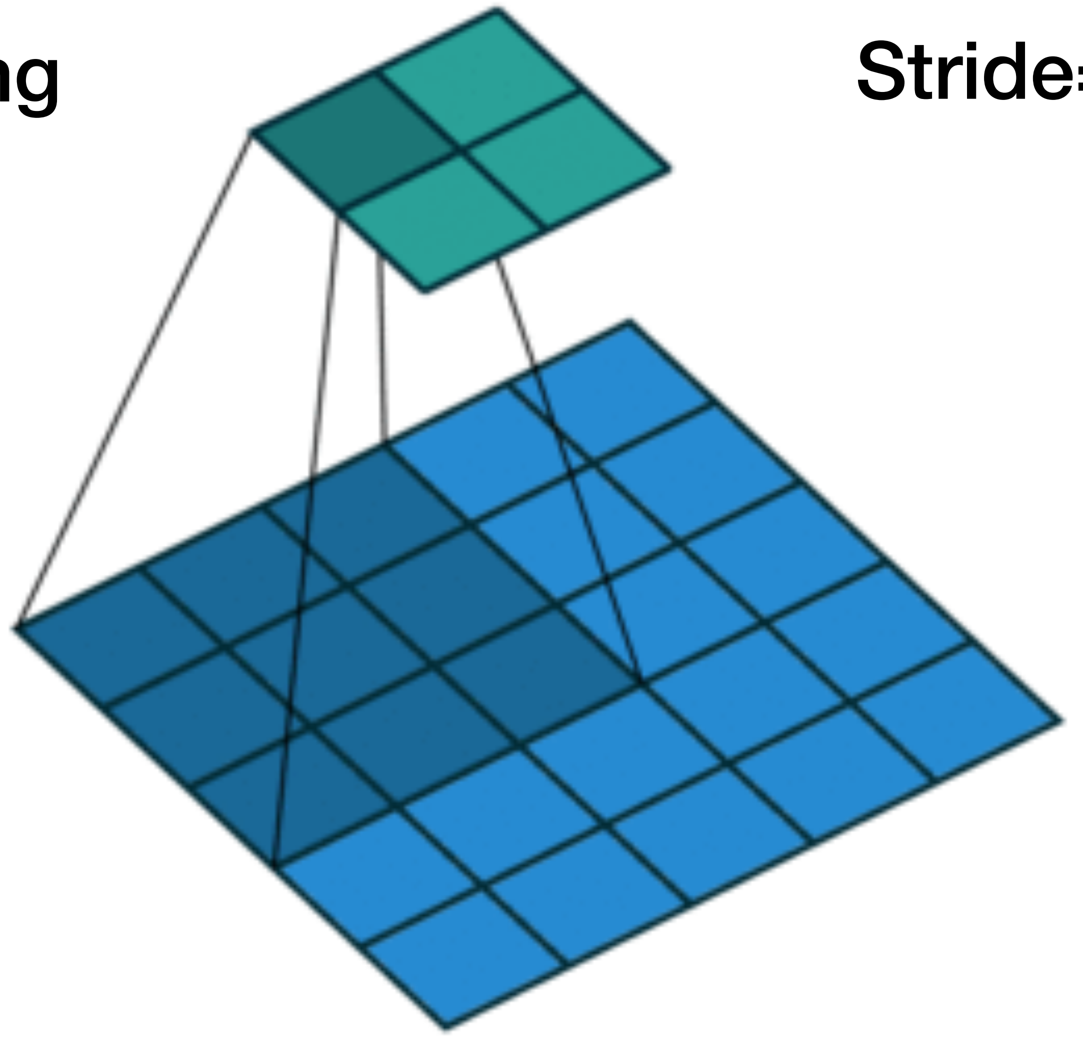
Stride=1, No padding



Stride=1, Padding, P=1



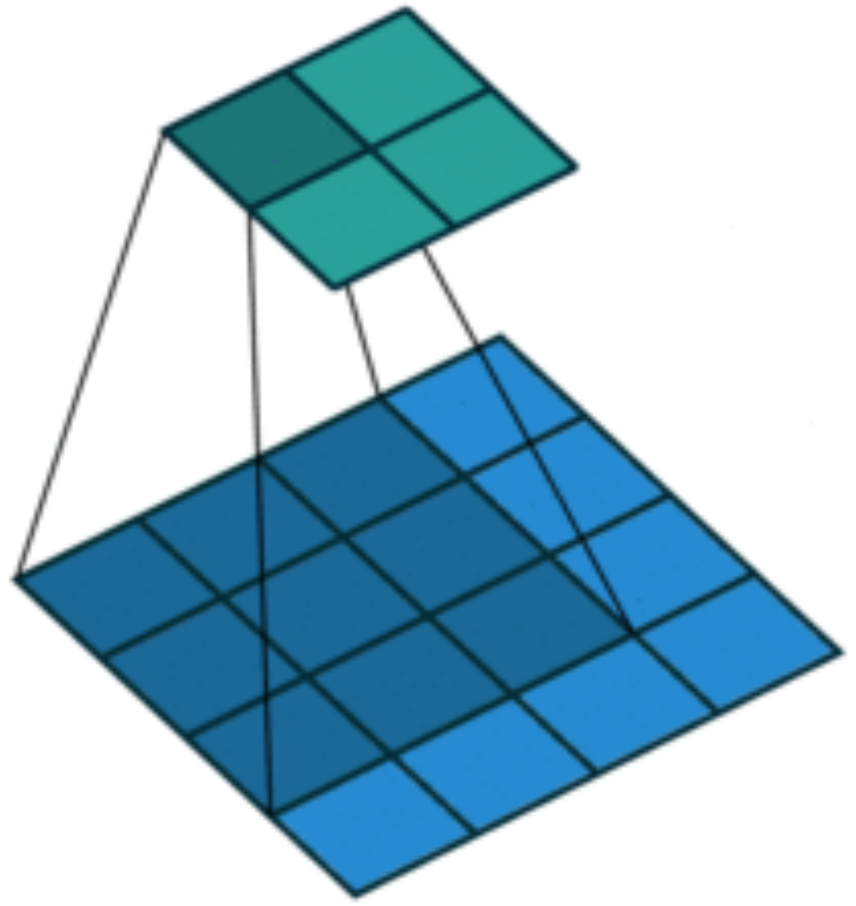
Stride=1, Padding, P=2



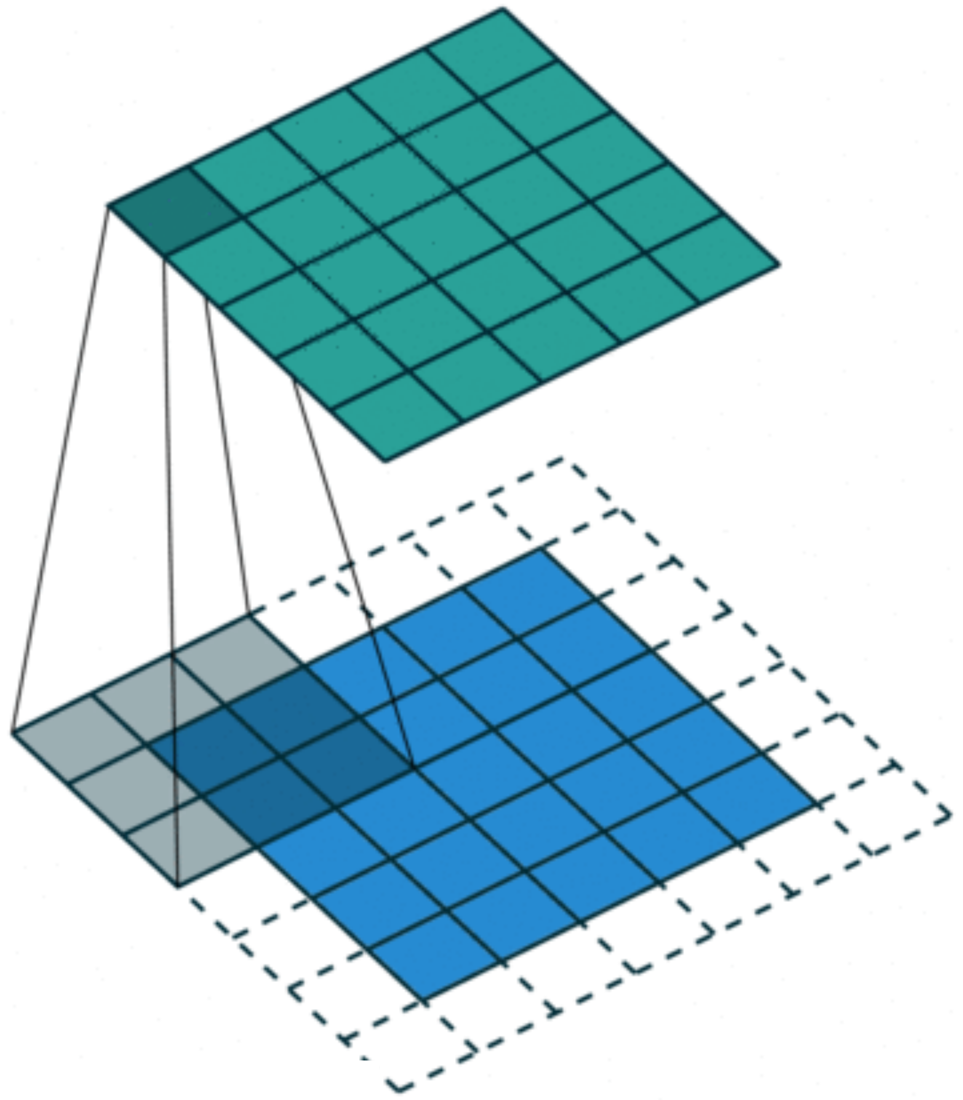
Stride=2, No padding

All animations are from: https://github.com/vdumoulin/conv_arithmetic

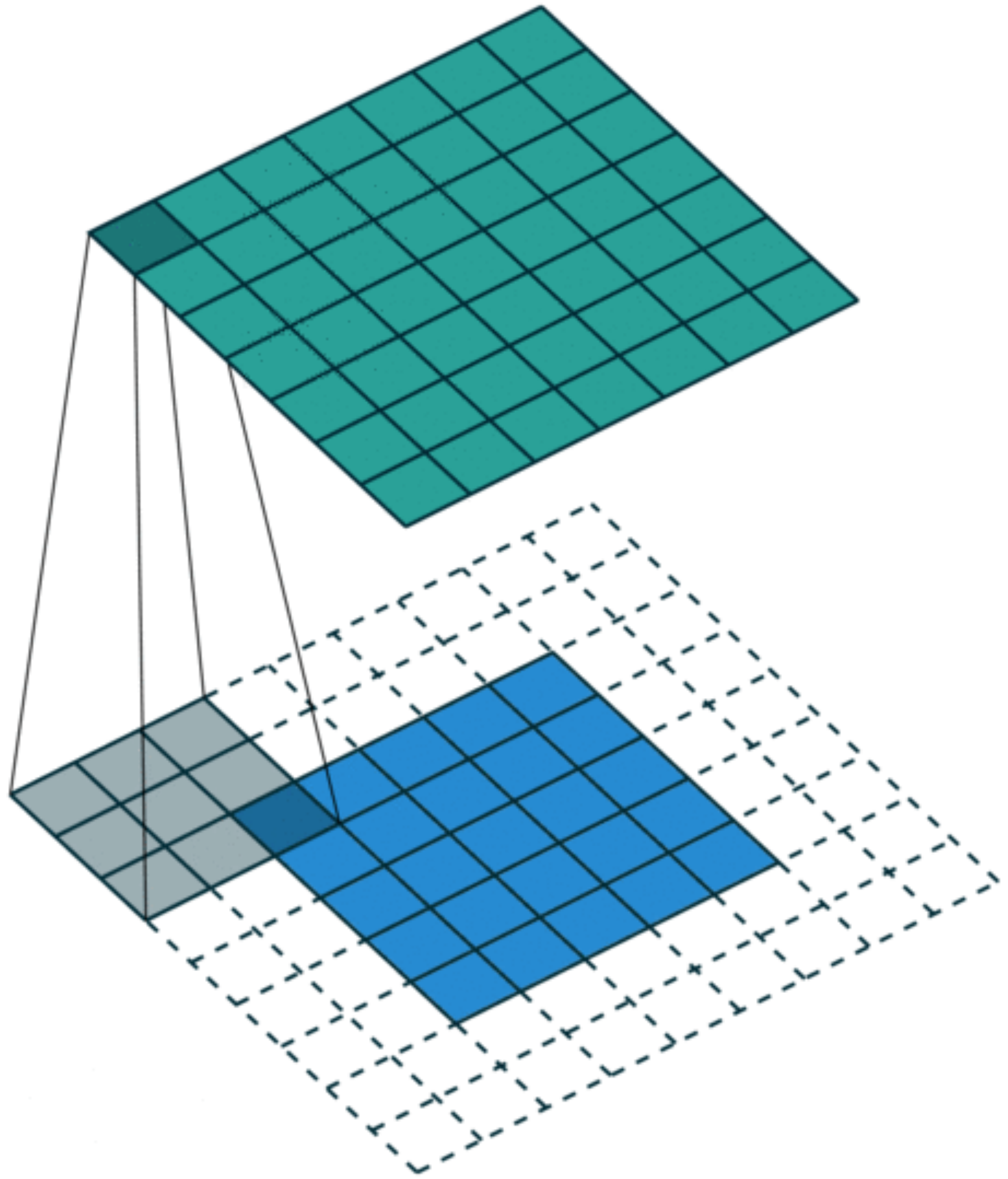
Convolutional Neural Networks (CNNs)



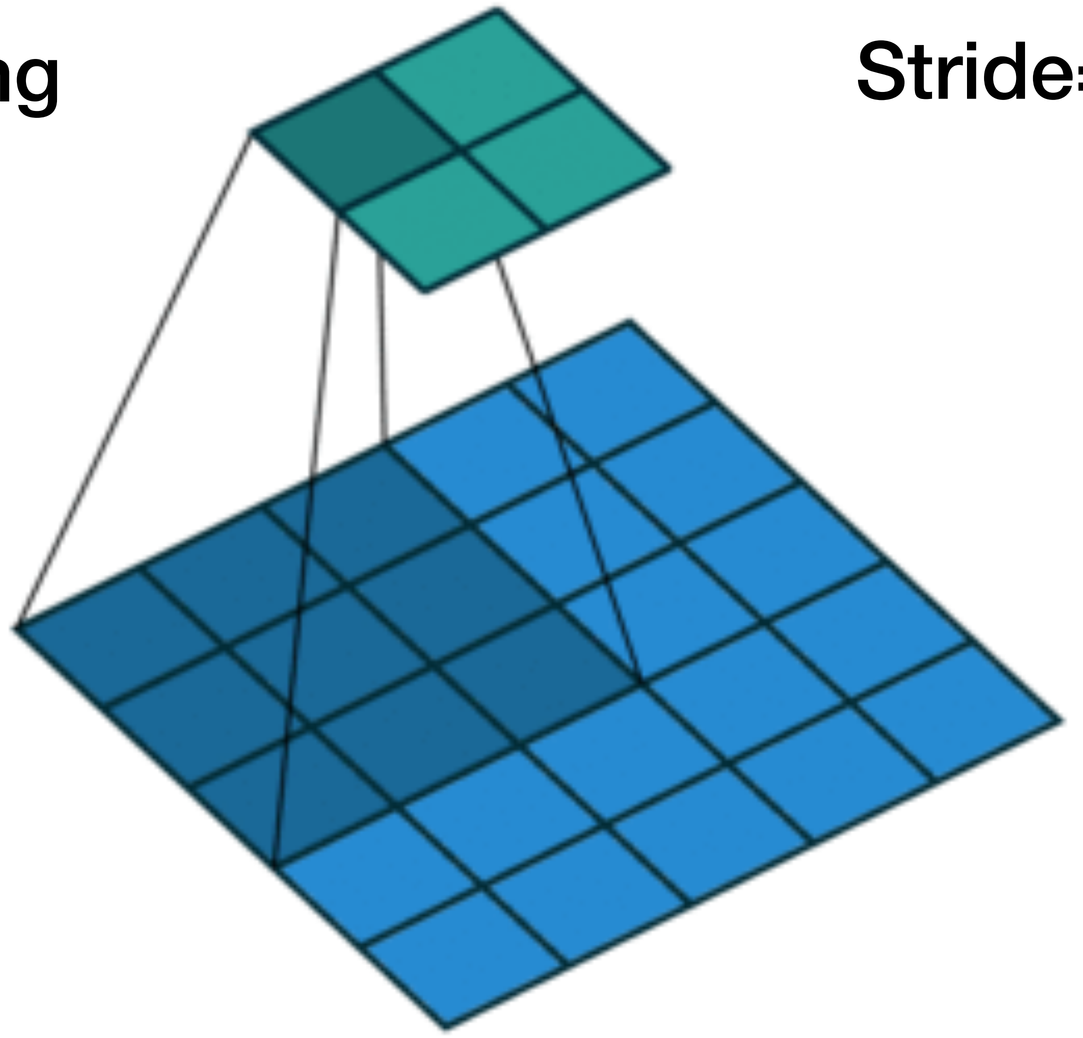
Stride=1, No padding



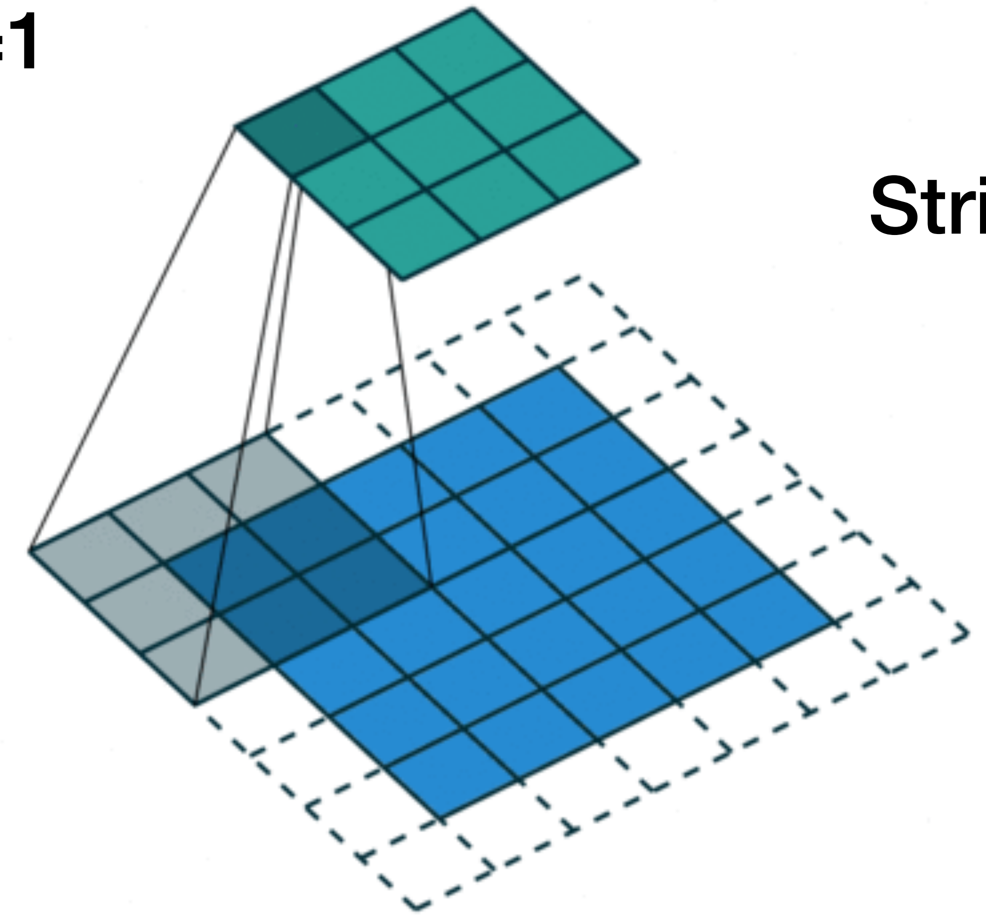
Stride=1, Padding, P=1



Stride=1, Padding, P=2



Stride=2, No padding



Stride=2, Padding, P=1

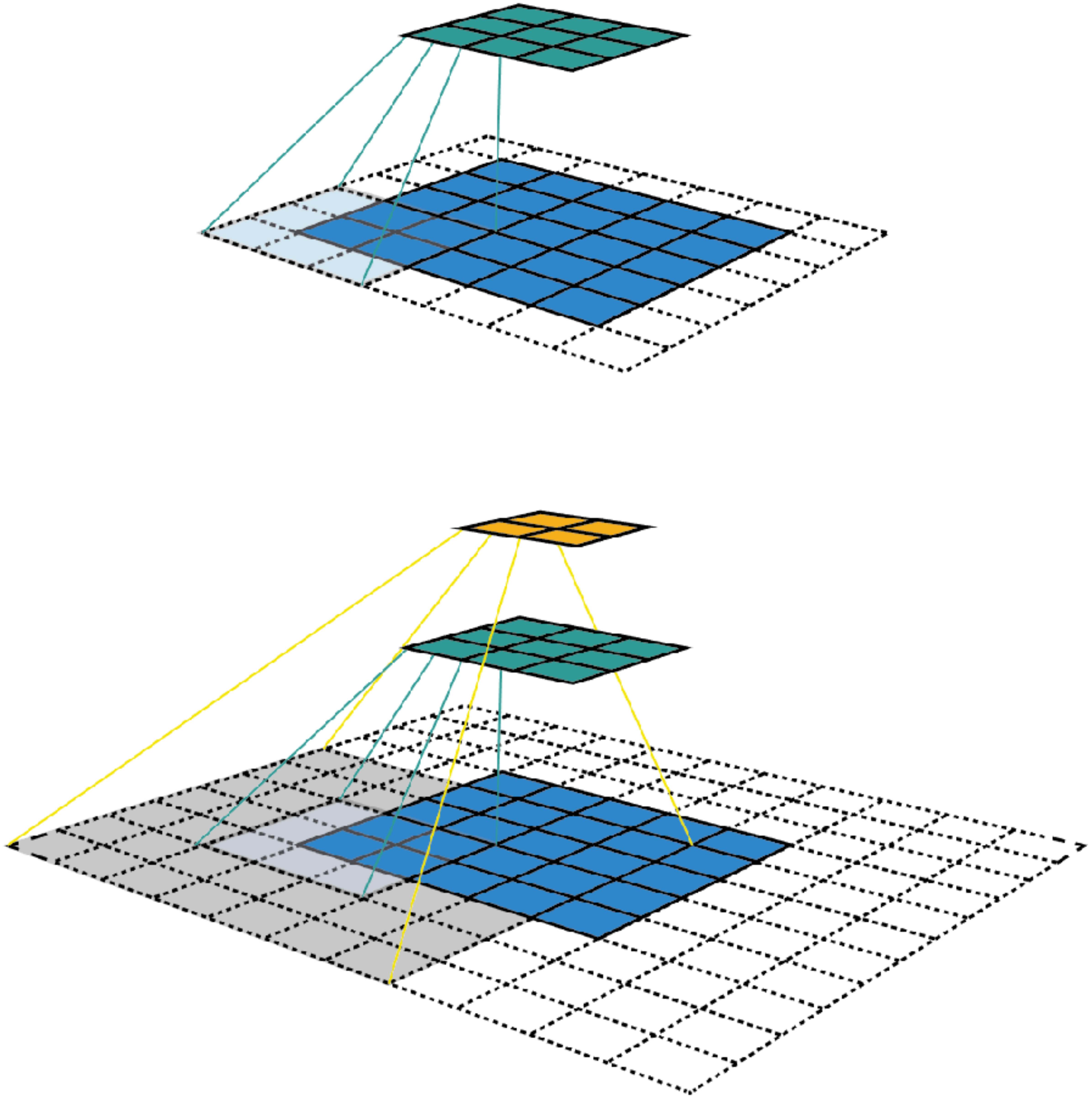
All animations are from: https://github.com/vdumoulin/conv_arithmetic

Convolution Layers: Summary

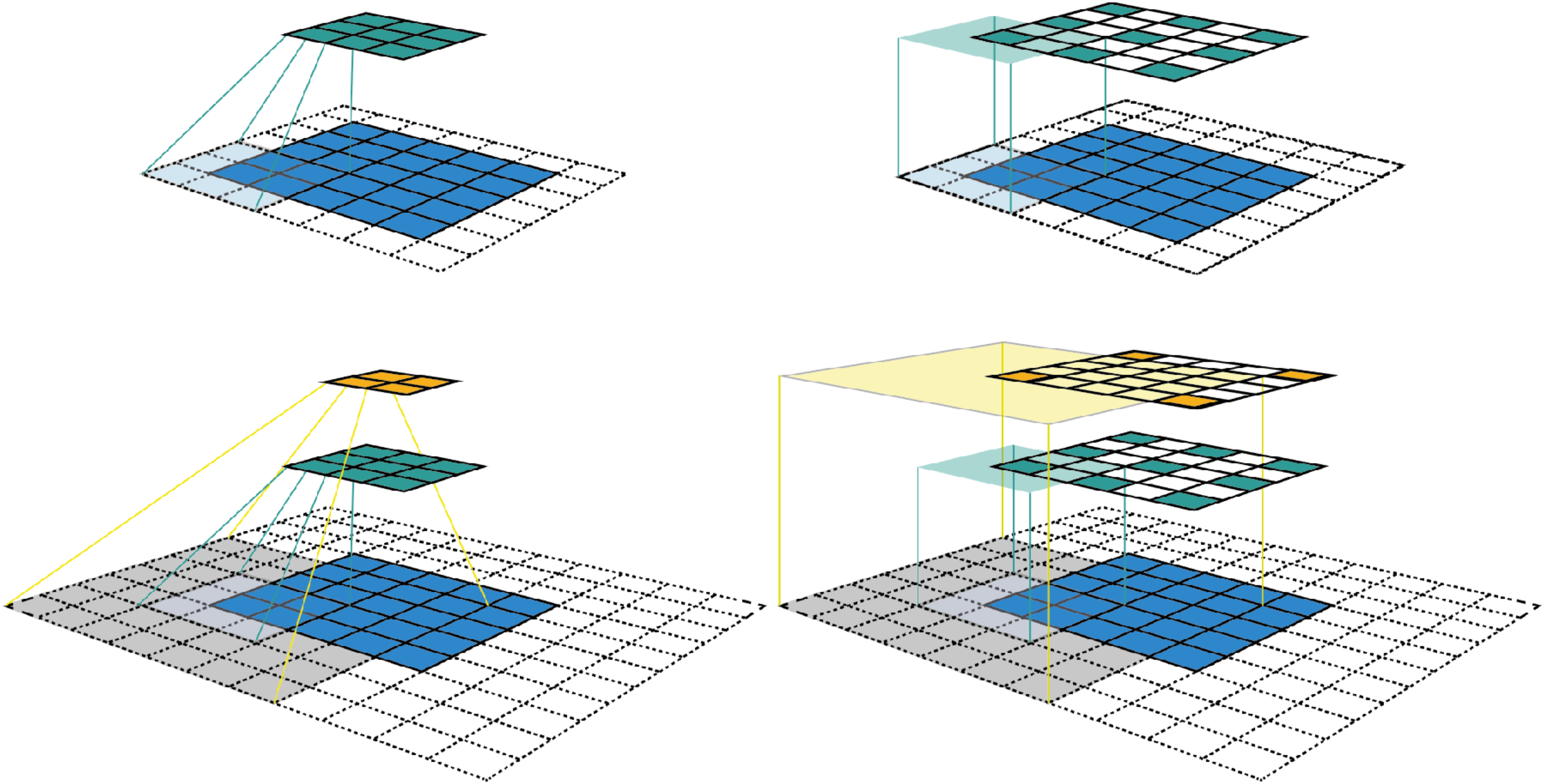
- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.

Summary from: http://cs231n.github.io/convolutional-networks/

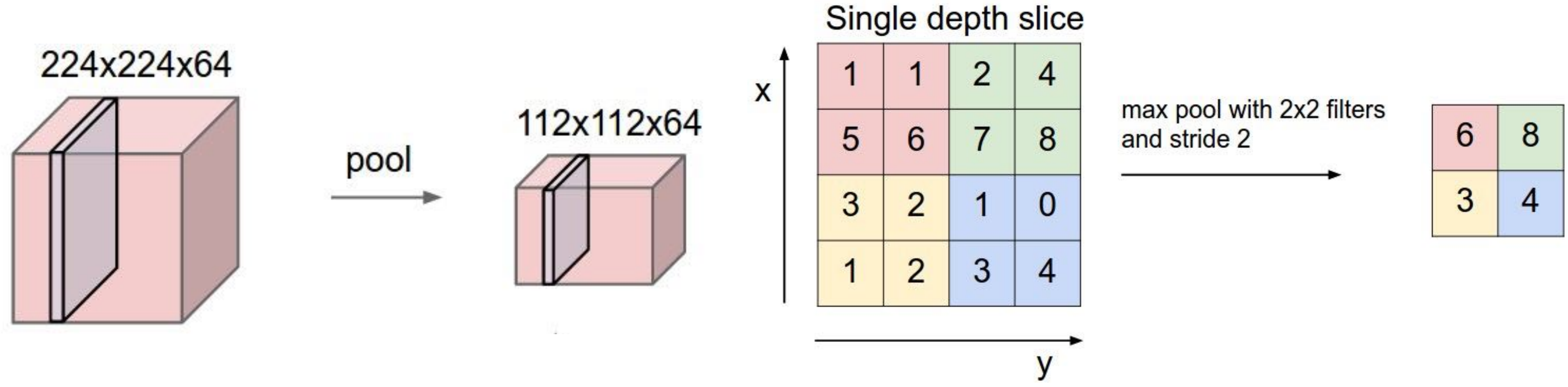
Receptive Field



Receptive Field



Pooling Layer



- Why pooling?
  Reduce the size of the representation, speed up the computations and make the features a little more robust.

- Max pooling is popularly used in CNNs.

Image from: http://cs231n.github.io/convolutional-networks/

Pooling Layer

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires two hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$

Summary from: http://cs231n.github.io/convolutional-networks/

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

- γ and β are learned parameters used across all batches

- At test time: Individual inputs, no mini-batch.

  - First, normalize inputs using training population statistics.

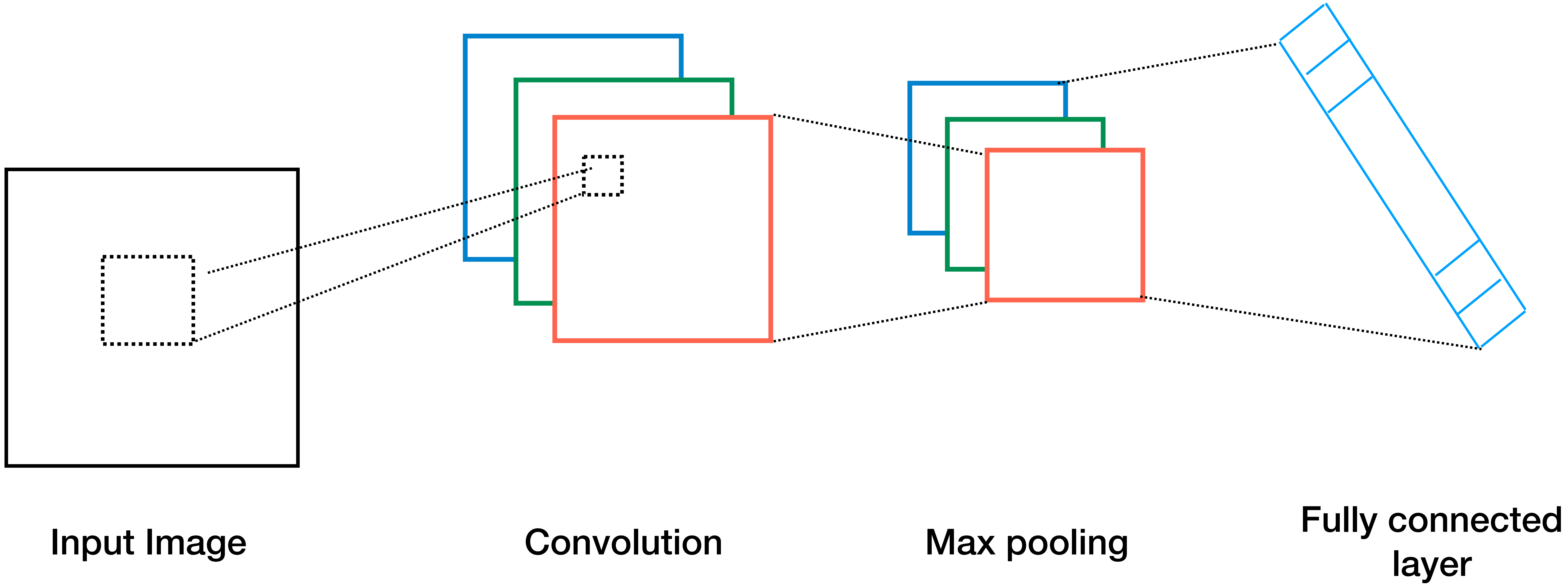
$$\hat{x} \leftarrow \frac{x - \mu_{\text{pop}}}{\sqrt{\sigma_{\text{pop}}^2 + \epsilon}}$$

  - Then, scale and shift.

$$\hat{y} \leftarrow \gamma\hat{x} + \beta$$

https://arxiv.org/pdf/1502.03167v3.pdf

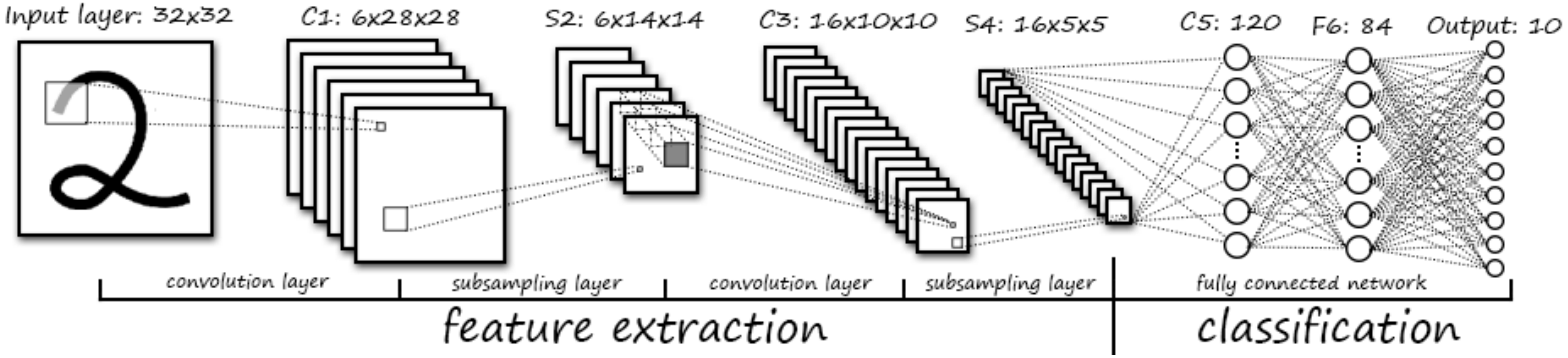
Convolutional Architectures



- Block that can be repeated: Convolutional layer, followed by non-linearity (e.g. ReLU) + Max pooling

- Fully connected layers before classification

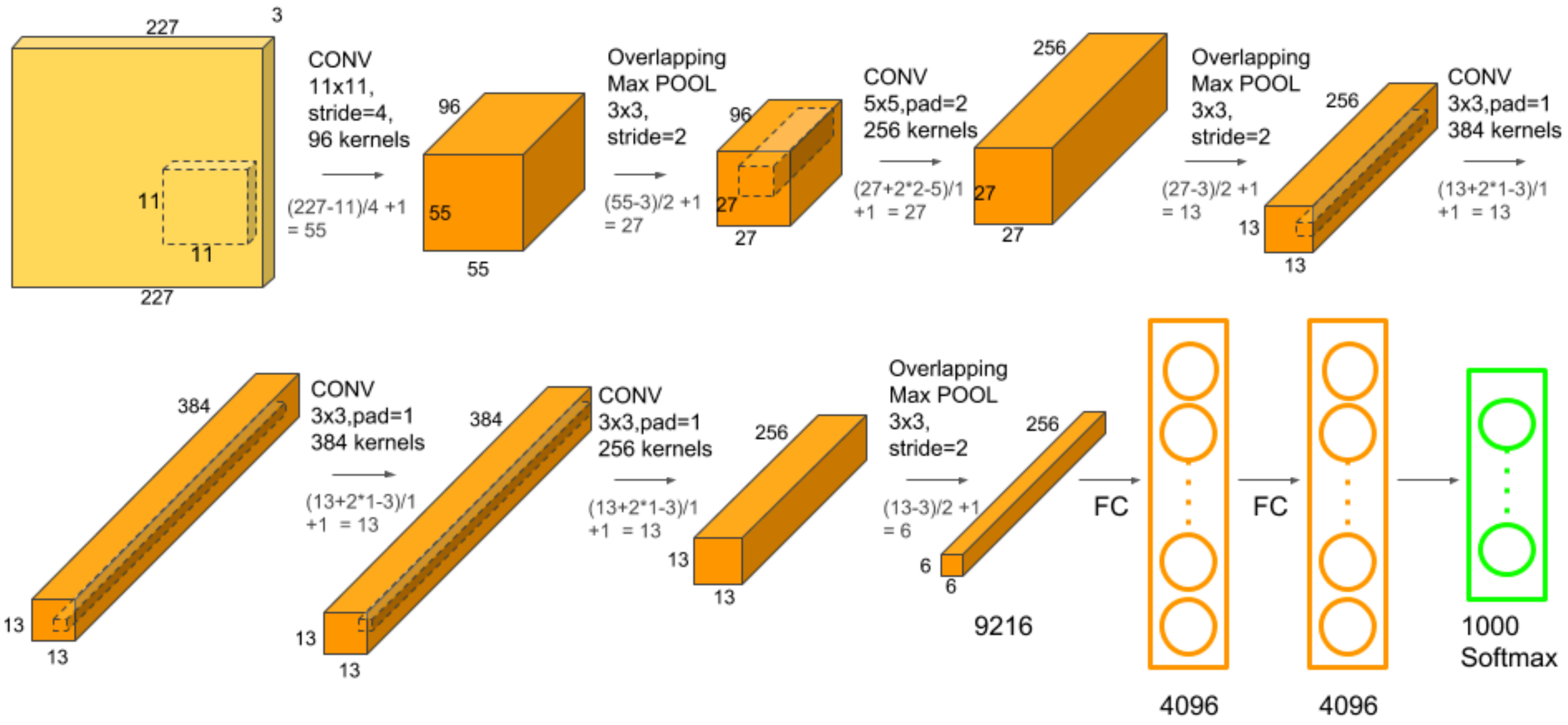
LeNet-5



- One of the first successful CNN architectures
- Used to classify images of hand-written digits

LeNet-5: LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. “Gradient-based learning applied to document recognition”. Proceedings of the IEEE, 86(11), pp.2278-2324.

AlexNet



- Winner (by a large margin) of the ImageNet challenge in 2012.
- Much larger than previous architectures.

AlexNet: ImageNet Classification with Deep Convolutional Neural Networks by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, NIPS 2012
Image from: https://www.learnopencv.com/understanding-alexnet/