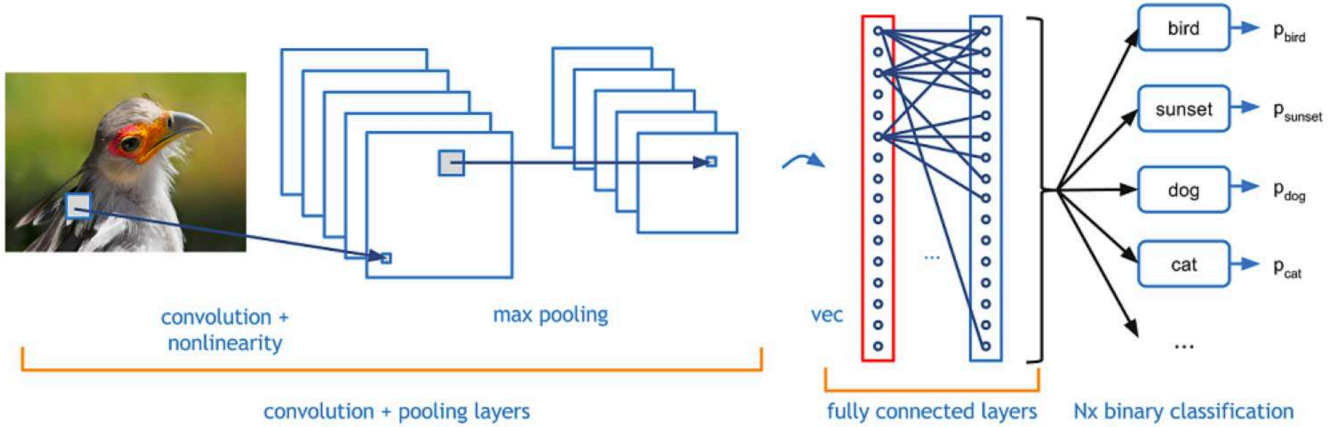
COMP9444
Neural Networks and Deep Learning

6. Convolutional Networks

Textbook, Sections 7.9, 7.11-7.13, 9.1-9.5

Convolutional Networks

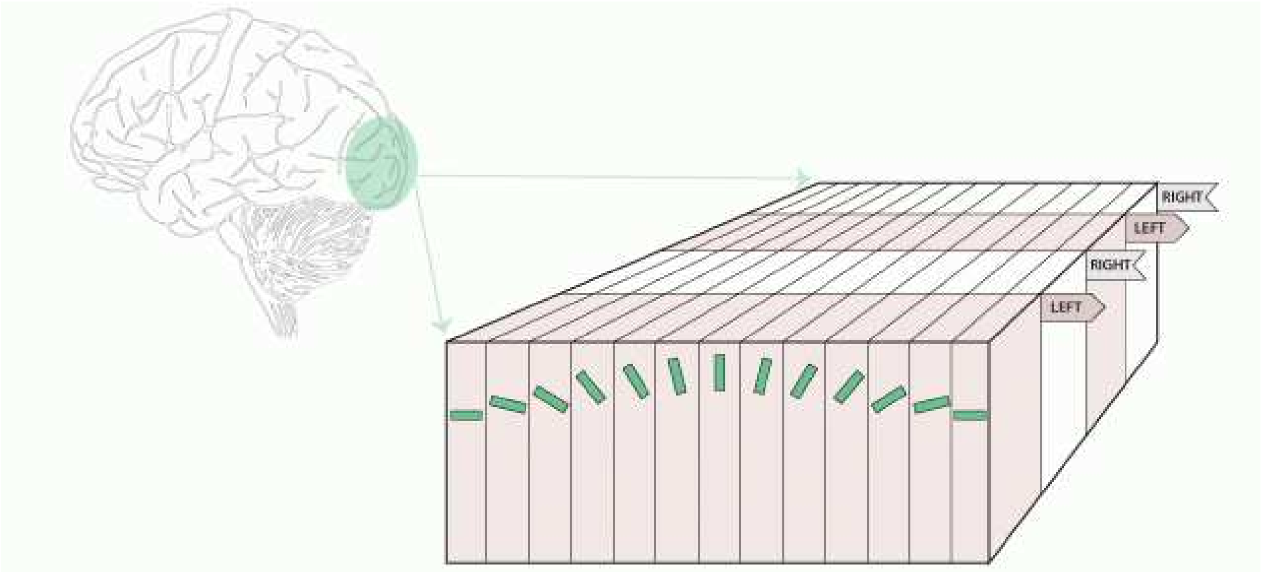


Suppose we want to classify an image as a bird, sunset, dog, cat, etc.

If we can identify features such as feather, eye, or beak which provide useful information in one part of the image, then those features are likely to also be relevant in another part of the image.

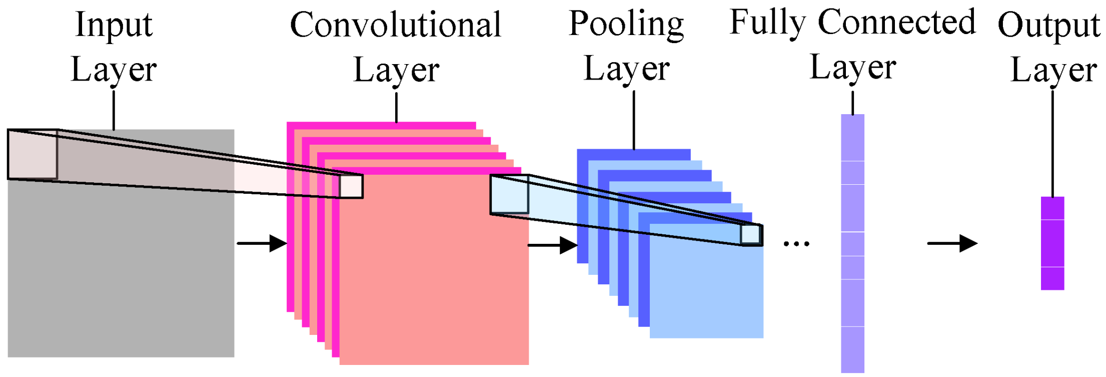
We can exploit this regularity by using a convolution layer which applies the same weights to different parts of the image.

Hubel and Weisel – Visual Cortex



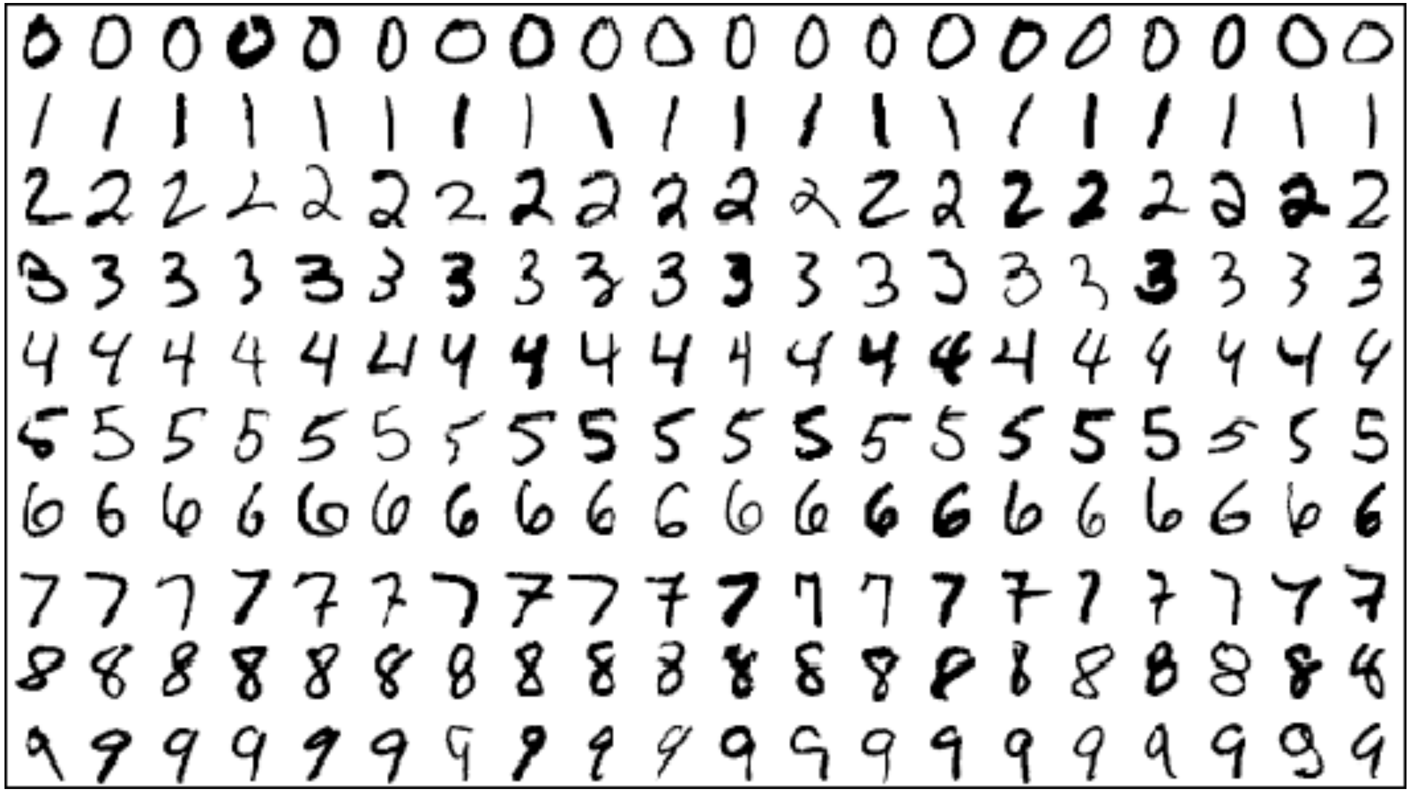
- cells in the visual cortex respond to lines at different angles
- cells in V2 respond to more sophisticated visual features
- Convolutional Neural Networks are inspired by this neuroanatomy
- CNN’s can now be simulated with massive parallelism, using GPU’s

Convolutional Network Components

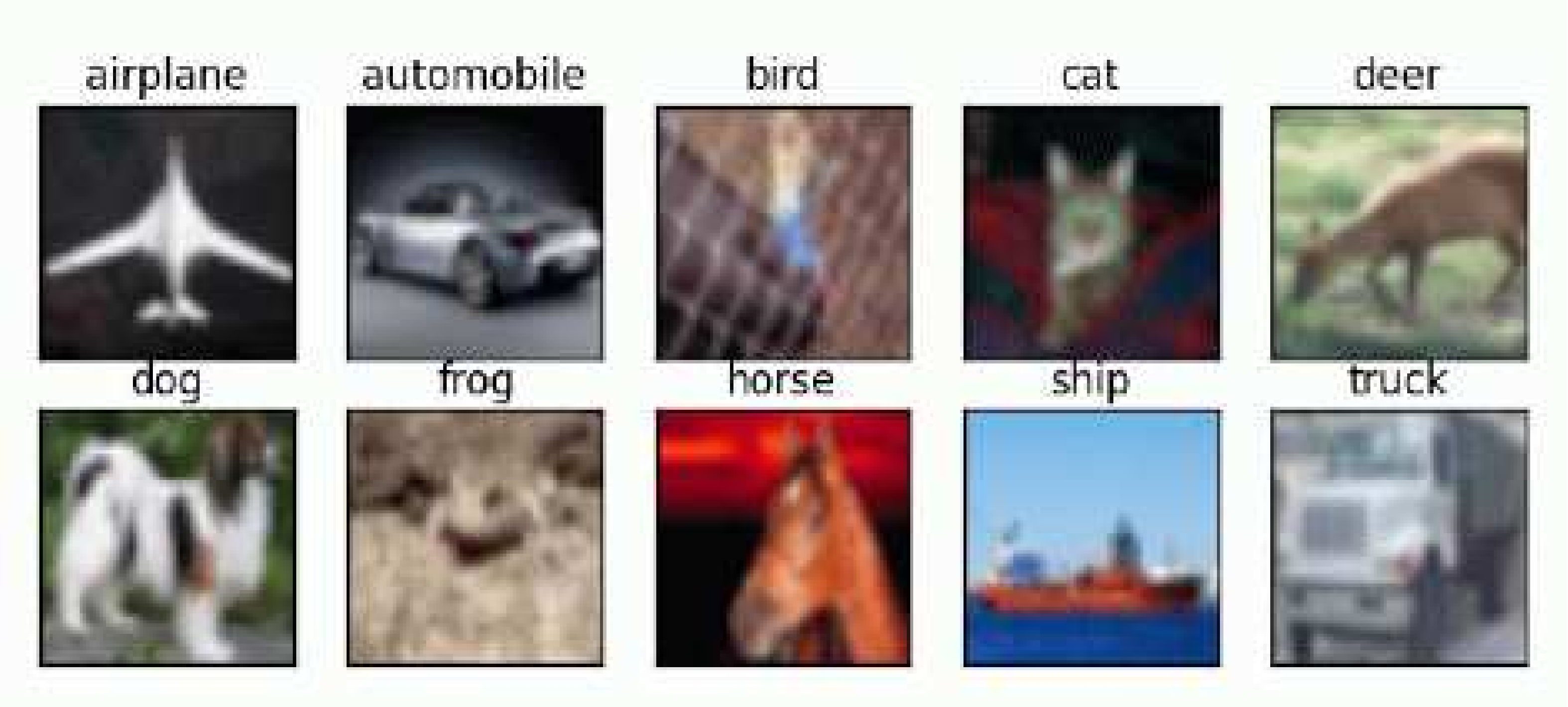


- **convolution layers**: extract shift-invariant features from the previous layer
- **subsampling or pooling layers**: combine the activations of multiple units from the previous layer into one unit
- **fully connected layers**: collect spatially diffuse information
- **output layer**: choose between classes

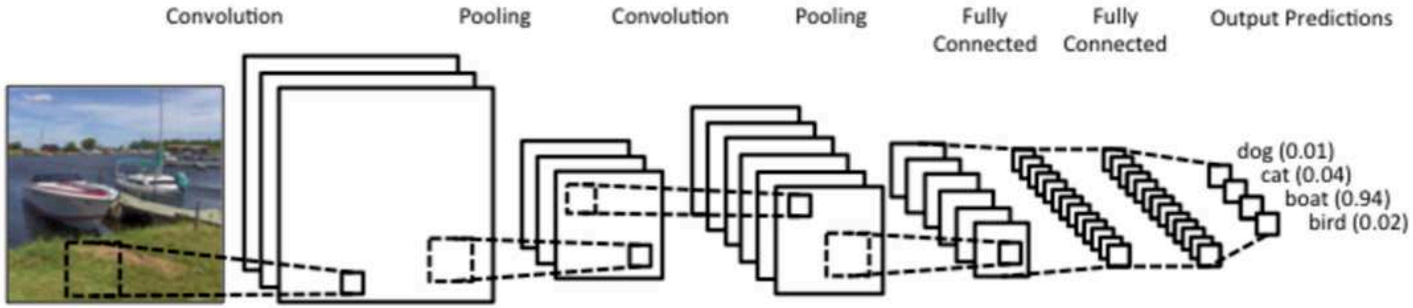
MNIST Handwritten Digit Examples



CIFAR Image Examples



Convolutional Network Architecture



There can be multiple steps of convolution followed by pooling, before reaching the fully connected layers.

Note how pooling reduces the size of the feature map (usually, by half in each direction).

Softmax (6.2.2)

Consider a classification task with N classes, and assume z_j is the output of the unit corresponding to class j.

We assume the network's estimate of the probability of each class j is proportional to $\exp(z_j)$. Because the probabilites must add up to 1, we need to normalize by dividing by their sum:

$$\text{Prob}(i) = \frac{\exp(z_i)}{\sum_{j=1}^{N} \exp(z_j)}$$

$$\log \text{Prob}(i) = z_i - \log \sum_{j=1}^{N} \exp(z_j)$$

If the correct class is i, we can treat $-\log \text{Prob}(i)$ as our cost function. The first term pushes up the correct class i, while the second term mainly pushes down the incorrect class j with the highest activation (if $j \neq i$).

Convolution Operator

Continuous convolution

$$s(t) = (x * w)(t) = \int x(a)w(t-a)da$$

Discrete convolution

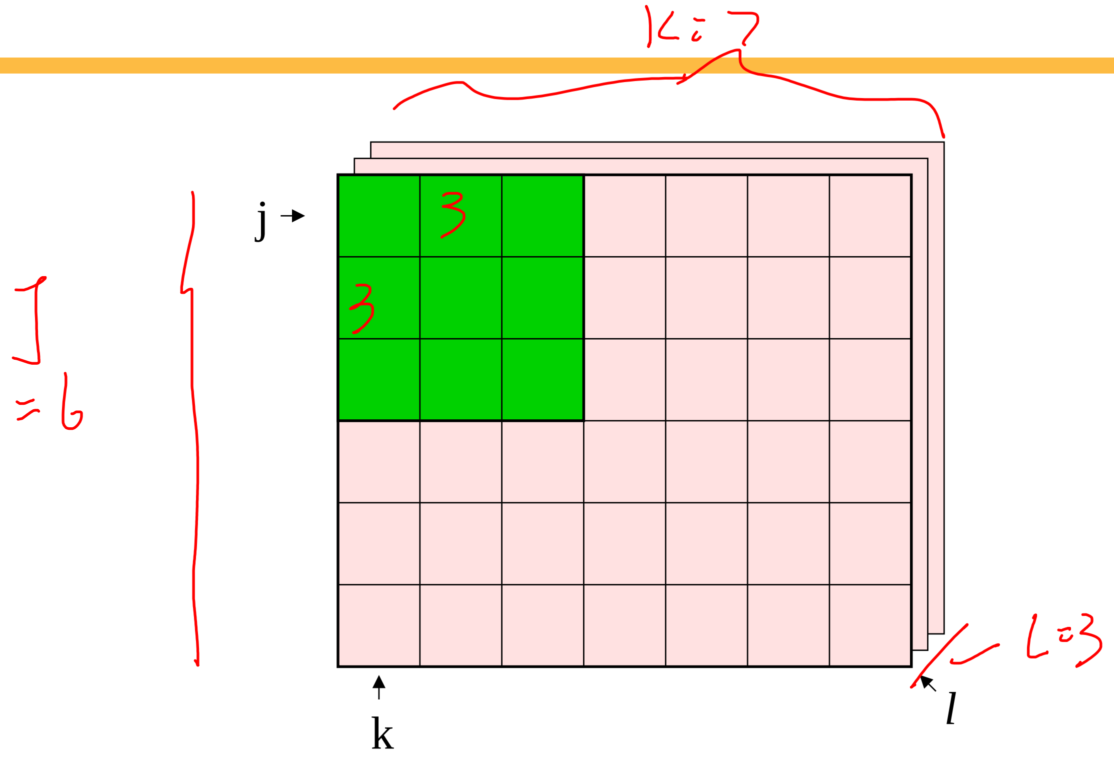
$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

Two-dimensional convolution

$$S(j,k) = (K * I)(j,k) = \sum_m \sum_n K(m,n)I(j+m,k+n)$$

Note: Theoreticians sometimes write $I(j-m,k-n)$ so that the operator is commutative. But, computationally, it is easier to write it with a plus sign.

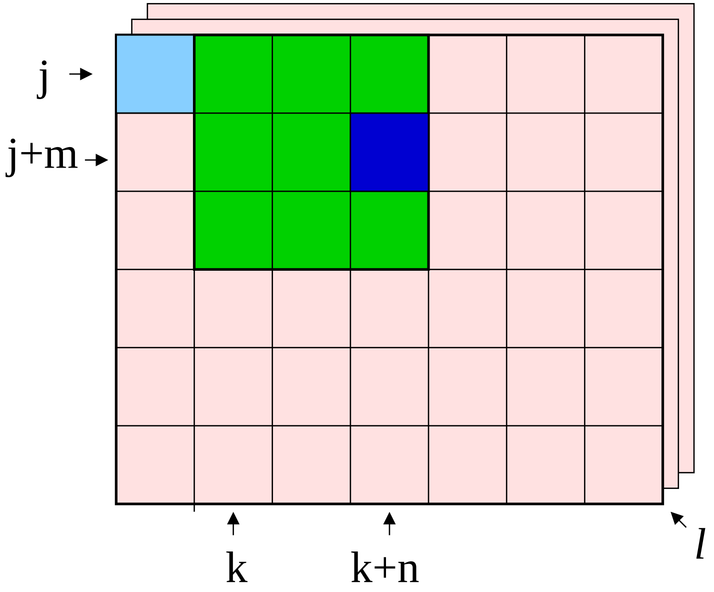
Convolutional Neural Networks



Assume the original image is $J \times K$, with L channels.

We apply an $M \times N$ "filter" to these inputs to compute one hidden unit in the convolution layer. In this example $J = 6, K = 7, L = 3, M = 3, N = 3$.

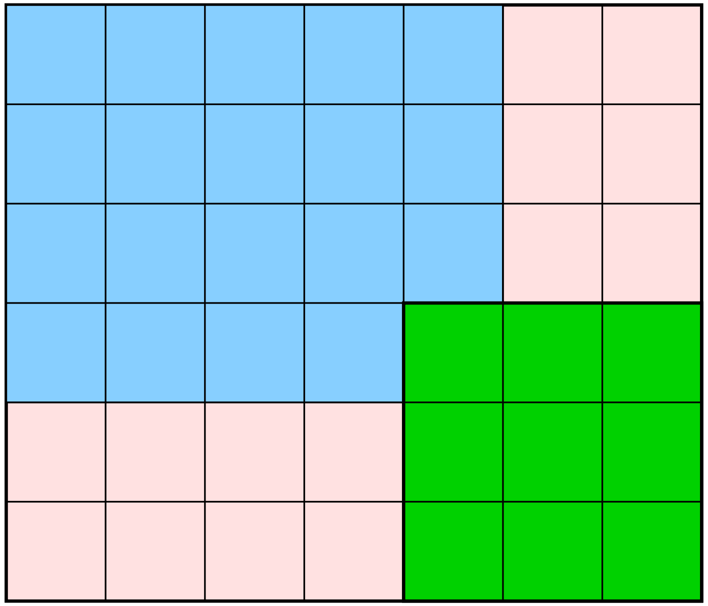
Convolutional Neural Networks



$$Z^i_{j,k} = g\big(b^i + \sum_l \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} K^i_{l,m,n} V^l_{j+m,k+n}\big)$$

The same weights are applied to the next $M \times N$ block of inputs, to compute the next hidden unit in the convolution layer ("weight sharing").

Convolutional Neural Networks



If the original image size is $J \times K$ and the filter is size $M \times N$, the convolution layer will be $(J+1-M) \times (K+1-N)$

Example: LeNet

For example, in the first convolutional layer of LeNet, $J = K = 32$, $M = N = 5$.

The width of the next layer is

$$J + 1 - M = 32 + 1 - 5 = 28$$

Question: If there are 6 filters in this layer, compute the number of:

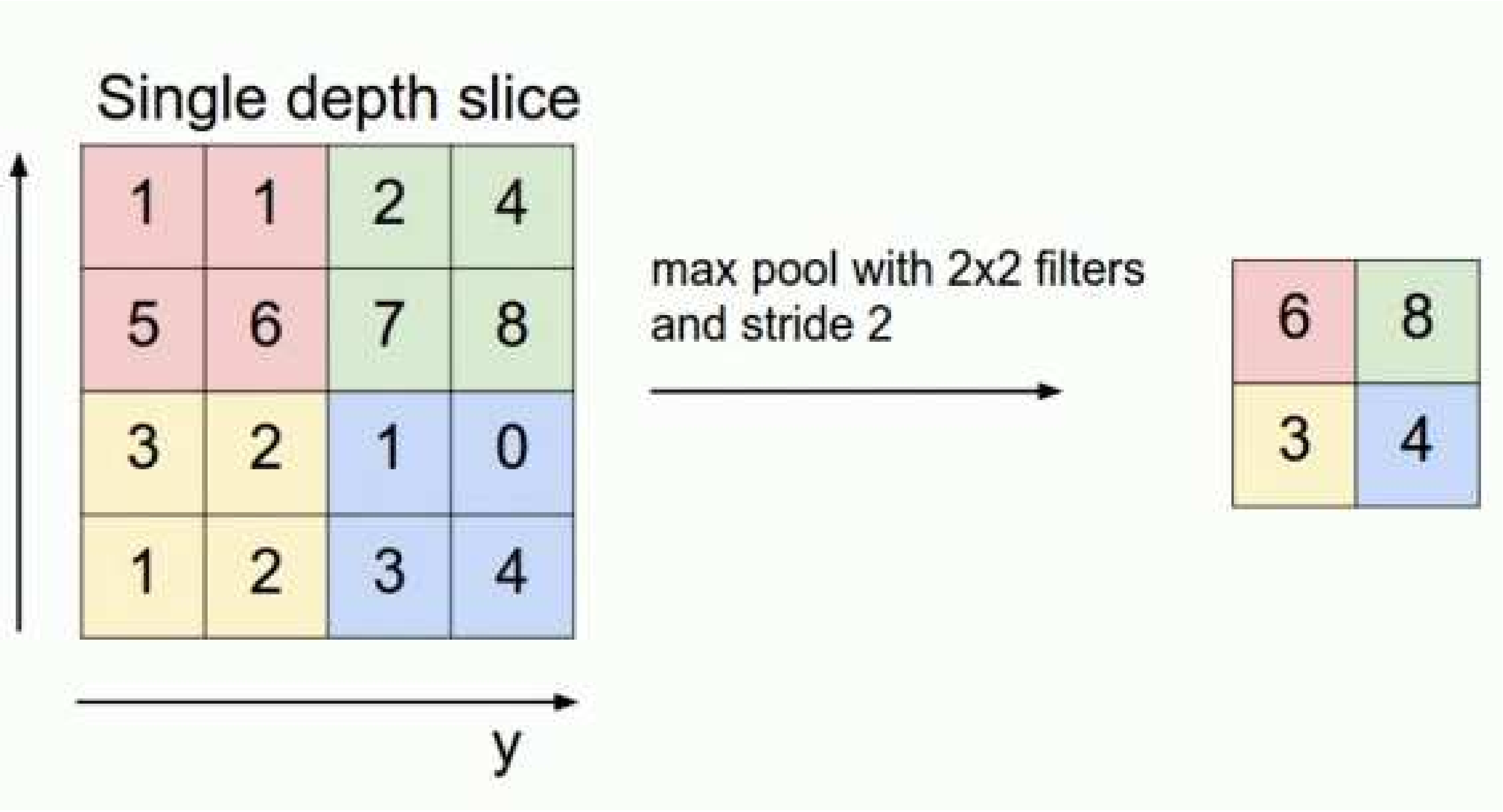
weights per neuron? $1 + 5 \times 5 \times 3 = 76$

neurons? $28 \times 28 \times 6 = 4704$

connections? $28 \times 28 \times 6 \times 76 = 357{,}504$ = neurons · weights/neuron

independent parameters? $6 \times 76 = 456$

Max Pooling



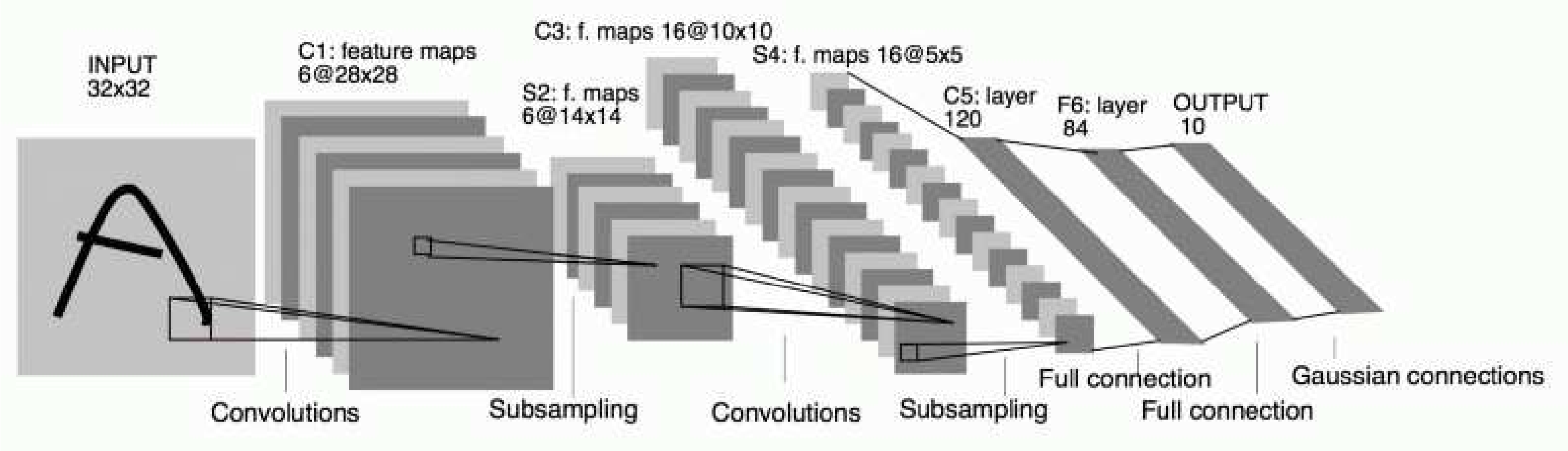
Single depth slice

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

max pool with 2x2 filters and stride 2

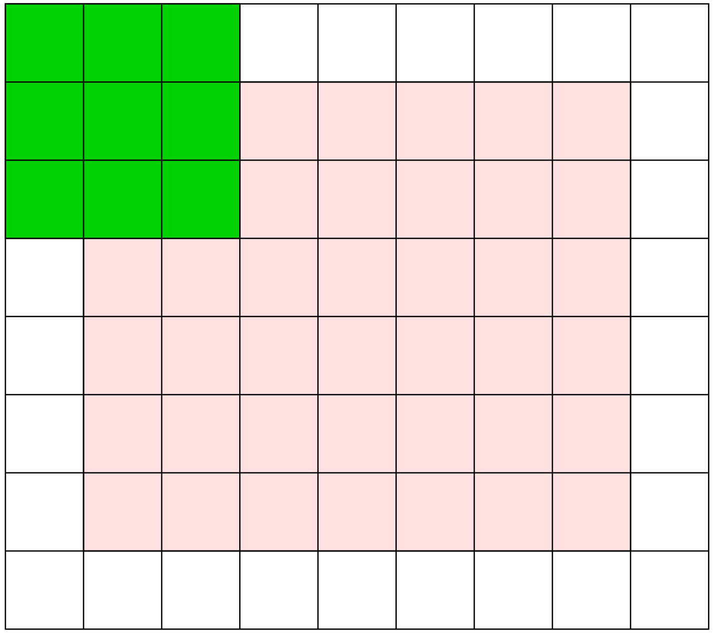
6	8
3	4

Example: LeNet trained on MNIST



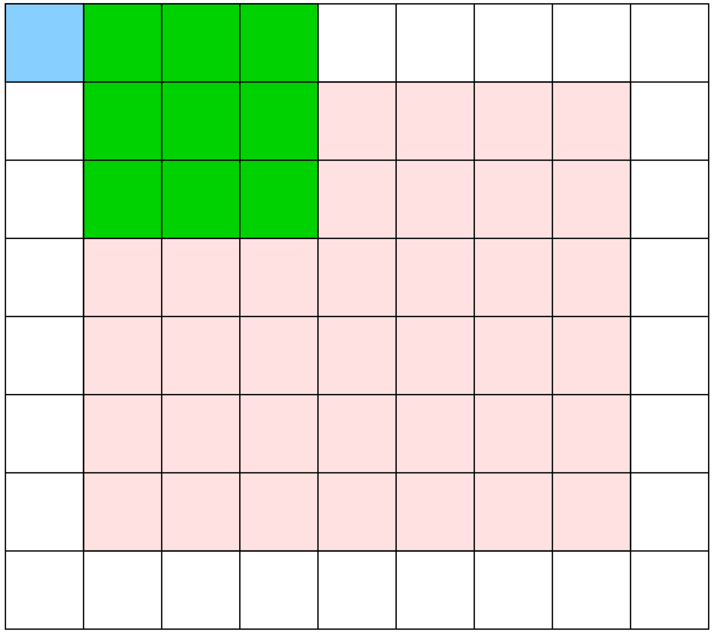
The 5×5 window of the first convolution layer extracts from the original 32×32 image a 28×28 array of features. Subsampling then halves this size to 14×14. The second Convolution layer uses another 5×5 window to extract a 10×10 array of features, which the second subsampling layer reduces to 5×5. These activations then pass through two fully connected layers into the 10 output units corresponding to the digits ’0’ to ’9’.

Convolution with Zero Padding



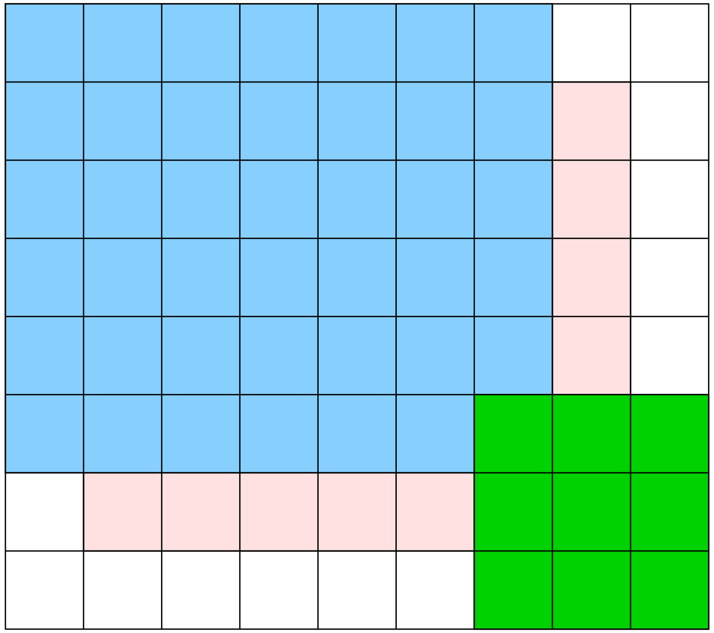
Sometimes, we treat the off-edge inputs as zero (or some other value).

Convolution with Zero Padding



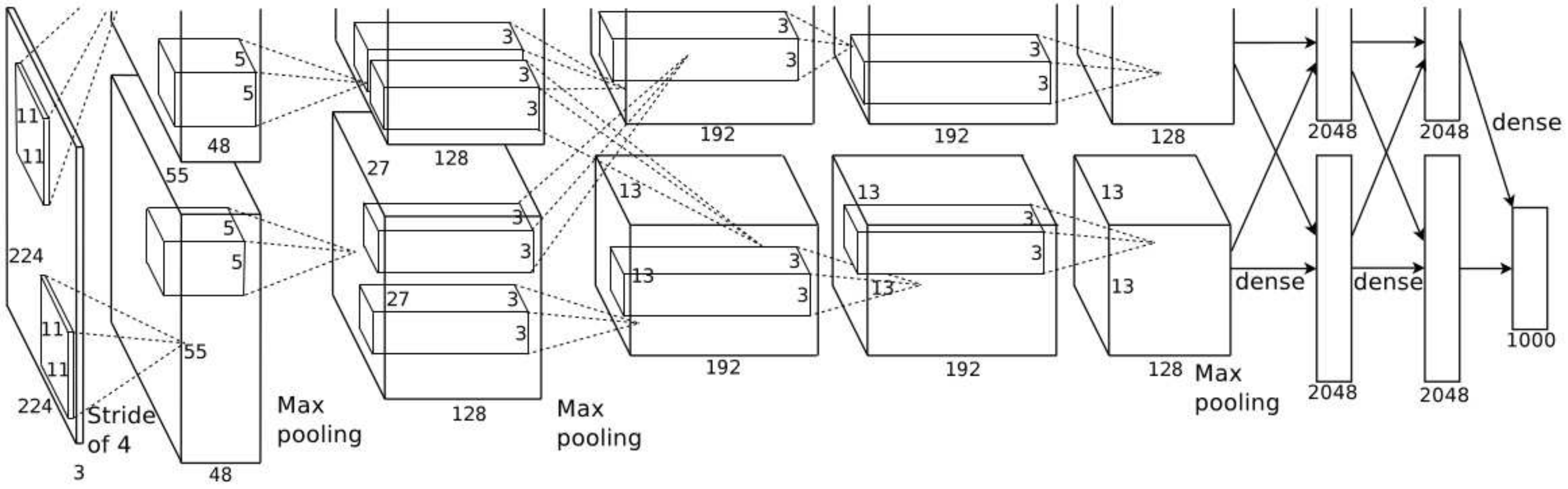
This is known as “Zero-Padding”.

Convolution with Zero Padding



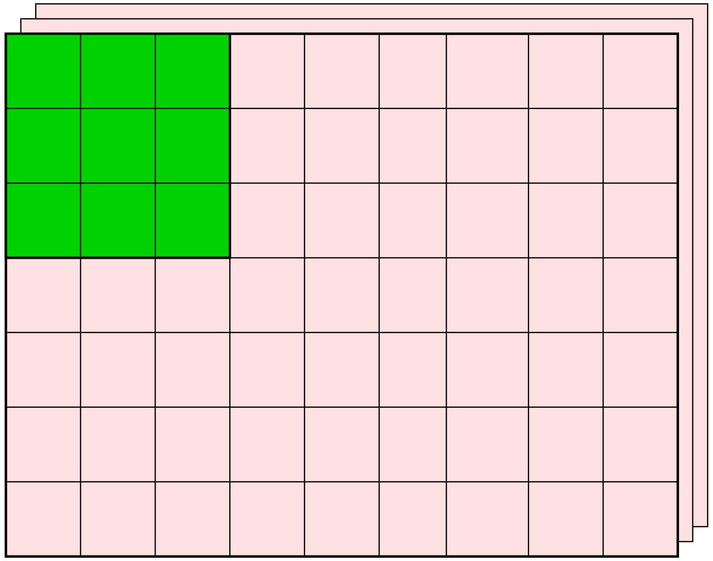
With Zero Padding, the convolution layer is the same size as the original image (or the previous layer).

Example: AlexNet (2012)



- 5 convolutional layers + 3 fully connected layers
- max pooling with overlapping stride
- softmax with 1000 classes
- 2 parallel GPUs which interact only at certain layers

Stride

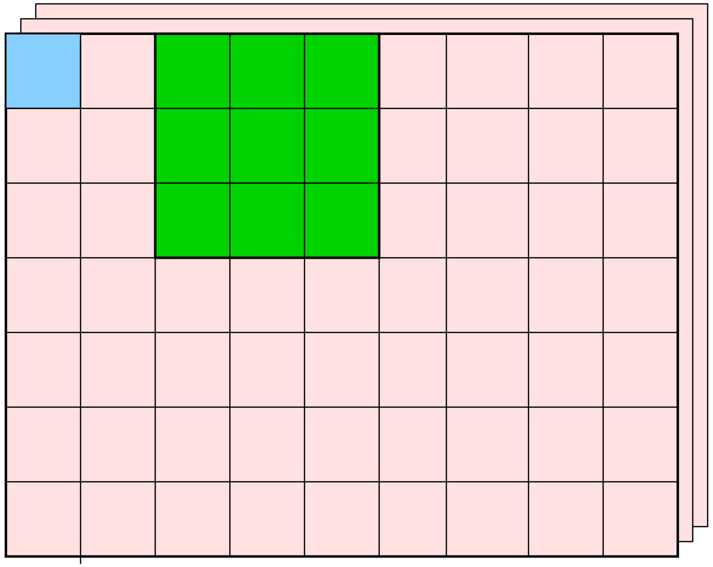


Assume the original image is $J \times K$, with L channels.

We again apply an $M \times N$ filter, but this time with a "stride" of $s > 1$.

In this example $J = 7, K = 9, L = 3, M = 3, N = 3, s = 2$.

Stride

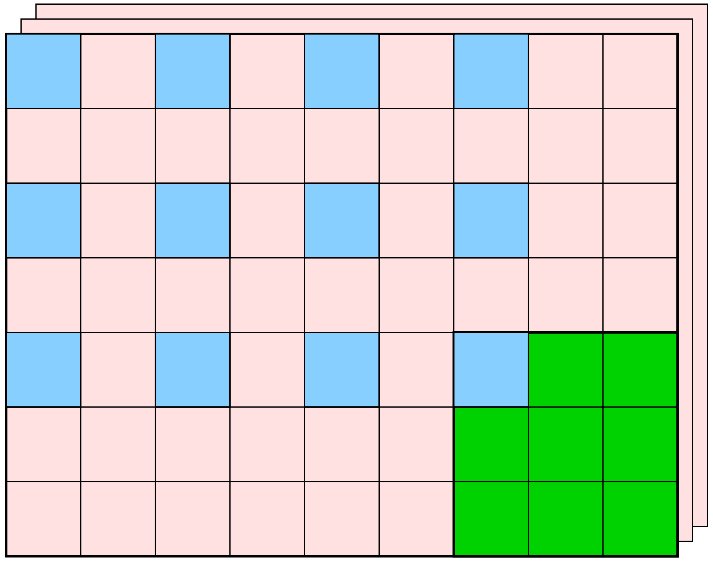


$$Z^i_{j,k} = g\big(b^i + \sum_l \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} K^i_{l,m,n} V^l_{j+m,k+n}\big)$$

The same formula is used, but j and k are now incremented by s each time.

The number of free parameters is $1 + L \times M \times N$

Stride Dimensions

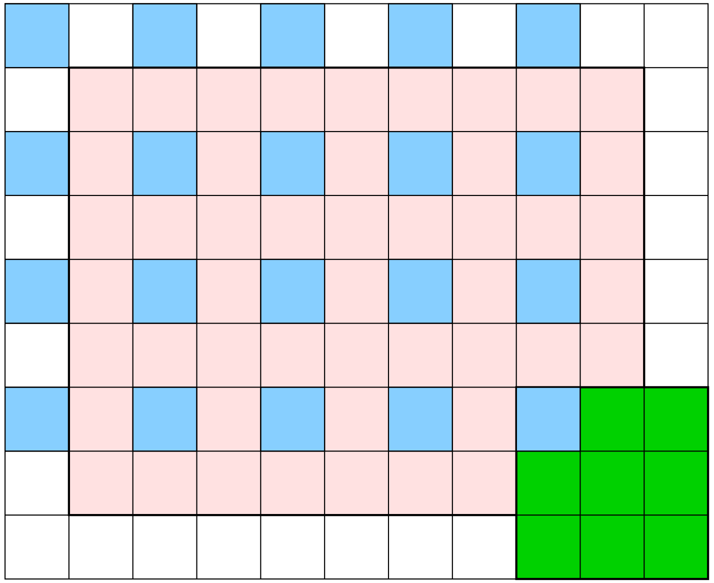


j takes on the values $0, s, 2s, \ldots, (J-M)$

k takes on the values $0, s, 2s, \ldots, (K-N)$

The next layer is $(1+(J-M)/s)$ by $(1+(K-N)/s)$

Stride with Zero Padding



When combined with zero padding of width P,

j takes on the values $0, s, 2s, \ldots, (J+2P-M)$

k takes on the values $0, s, 2s, \ldots, (K+2P-N)$

The next layer is $(1+(J+2P-M)/s)$ by $(1+(K+2P-N)/s)$

Example: AlexNet Conv Layer 1

For example, in the first convolutional layer of AlexNet,
$J = K = 224$, $P = 2$, $M = N = 11$, $s = 4$.

The width of the next layer is

$$1 + (J + 2P - M)/s = 1 + (224 + 2 \times 2 - 11)/4 = 55$$

Question: If there are 96 filters in this layer, compute the number of:

weights per neuron? $1 + 11 \cdot 11 \cdot 3 = 364$

neurons? $55 \cdot 55 \cdot 96 = 290{,}400$

connections? $55 \cdot 55 \cdot 96 \cdot 364 = 105{,}705{,}600$

independent parameters? $96 \cdot 364 = 36036$

Overlapping Pooling

If the previous layer is $J \times K$, and max pooling is applied with width F and stride s, the size of the next layer will be

$$(1+(J-F)/s)\times(1+(K-F)/s)$$

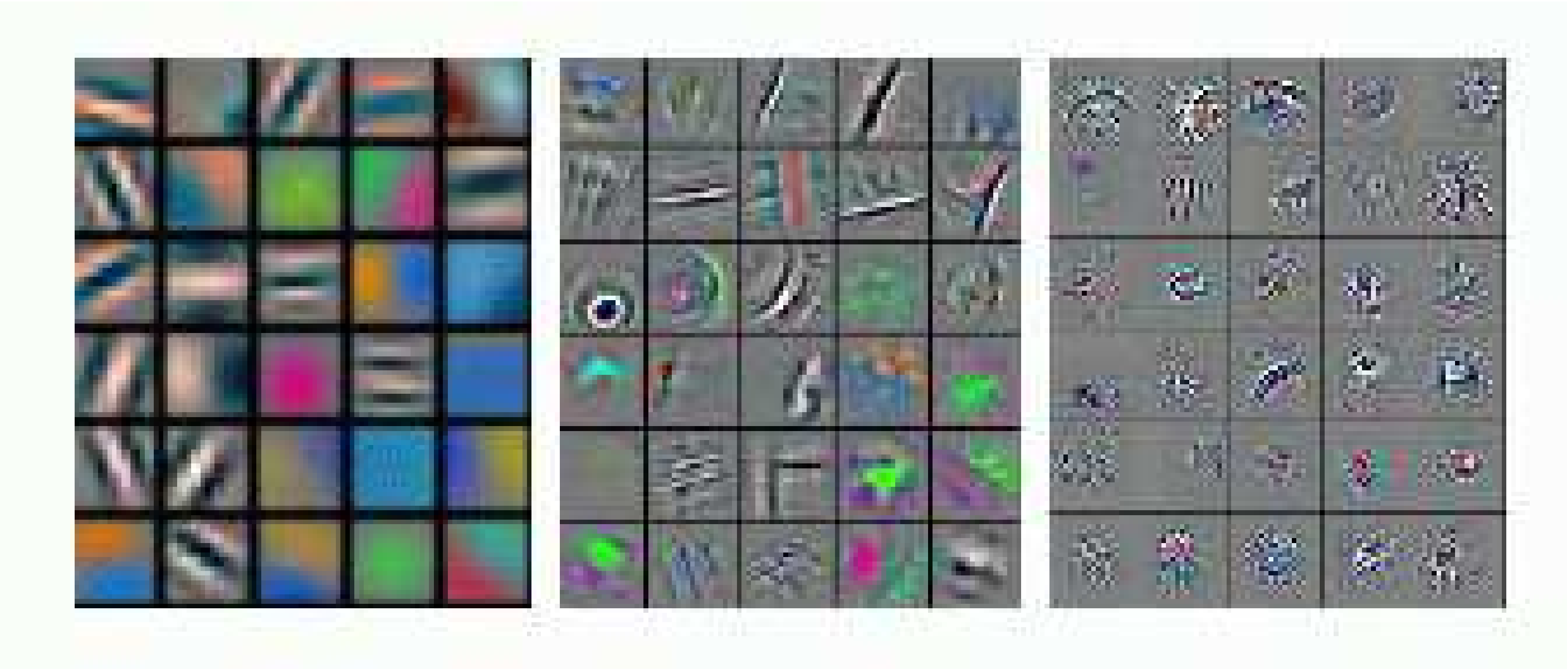
Question: If max pooling with width 3 and stride 2 is applied to the features of size 55 × 55 in the first convolutional layer of AlexNet, what is the size of the next layer?

Answer: $(55-3)/2 \ +1 \ = 27$

Question: How many independent parameters does this add to the model?

Answer: None! (No weight to be learned, just computing max)

Convolutional Filters



First Layer Second Layer Third Layer