Large Scale Data Analysis Using Deep Learning

Convolutional Networks

U Kang
Seoul National University

In This Lecture

- Convolutional Neural Network
  - Main idea
  - Efficiency
  - Parameter learning
  - Major architectures

Convolutional Networks

- Scale up neural networks to process very large images/video sequences
  - Sparse connections
  - Parameter sharing
- Automatically generalize across spatial translations of inputs
- Applicable to any input that is laid out on a grid (1-D, 2-D, 3-D, …)

Key Idea

- Replace matrix multiplication in neural nets with convolution
- Everything else stays the same (with minor changes)
  - Maximum likelihood
  - Back-propagation
  - Etc.

Convolution

- Suppose we are tracking the location of a spaceship with a laser sensor which provides a single output x(t), the position of the spaceship at time t
- Suppose the sensor is noisy, and we want to average together several measurements with a weighting function w(a) where a is the age of a measurement
  - $s(t) = \int x(a)w(t-a)da$
  - w needs to be a valid probability density function
  - Convolution operation is denoted with an asterisk: s(t) = (x*w)(t)
  - x is called input, w is called kernel, and the output is called feature map

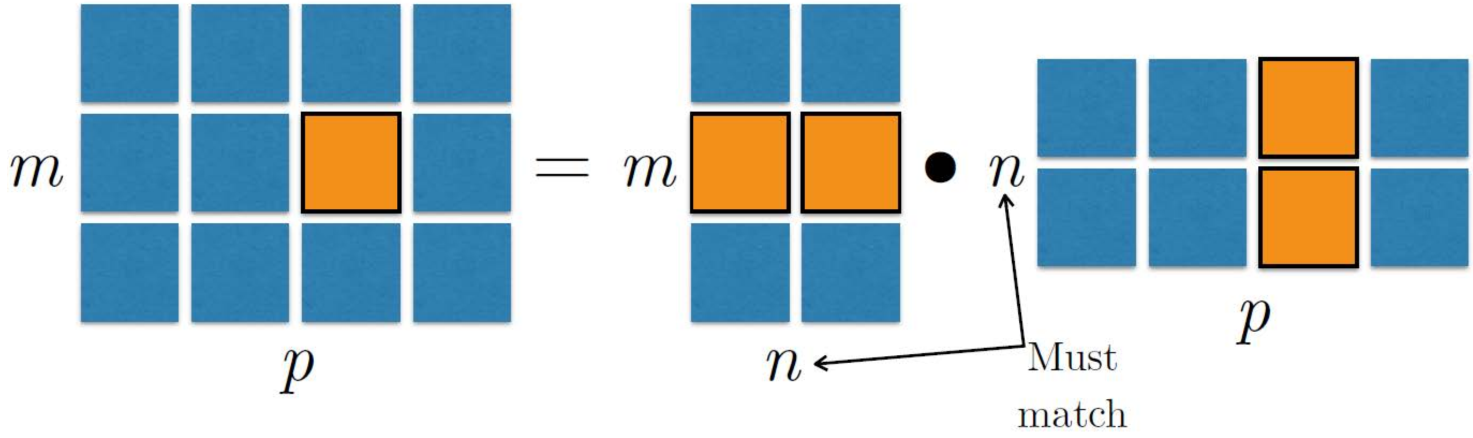
Convolution

- Discrete convolution (x and w defined only on integers)
  - $s(t) = (x * w)(t) = \sum_{-\infty}^{\infty} x(a)w(t-a)$
  - In ML applications the kernel contains a finite number of array elements
- 2-D convolution (I: 2-D image, K: 2-D kernel)
  - $S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$
- Convolution is commutative
  - $S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n)$
- Many neural network libraries implement a related function called cross-correlation
  - $S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n)$

Matrix Product

- C = AB

$$C_{i,j} = \sum_k A_{i,k} B_{k,j}$$



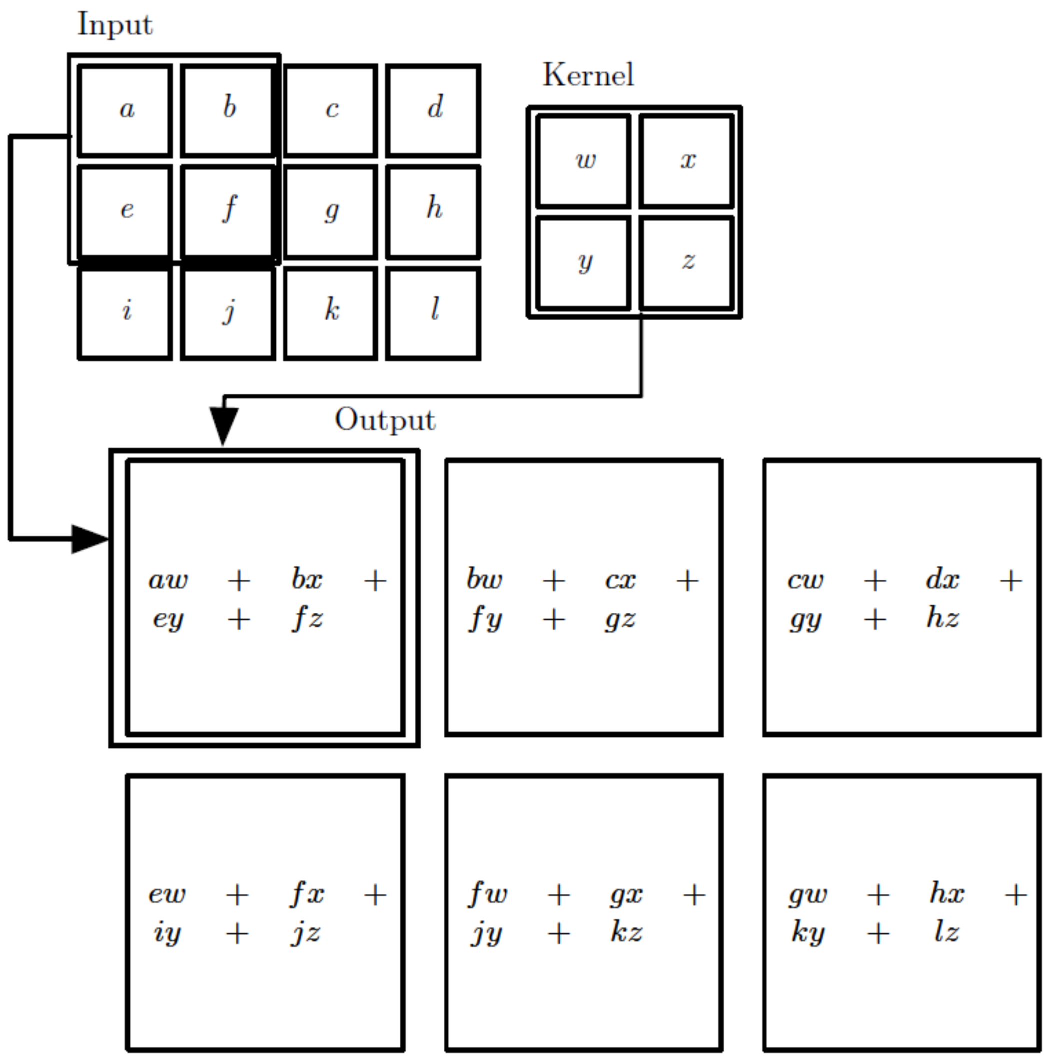
Matrix Transpose

- $(A^T)_{i,j} = A_{j,I}$

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} \Rightarrow \boldsymbol{A}^\top = \begin{bmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \end{bmatrix}$$

- $(AB)^T = B^T A^T$

2D Convolution



Main Ideas in Convolution

- Three main ideas in convolution
  - Sparse interactions
  - Parameter sharing
  - Equivariant representation

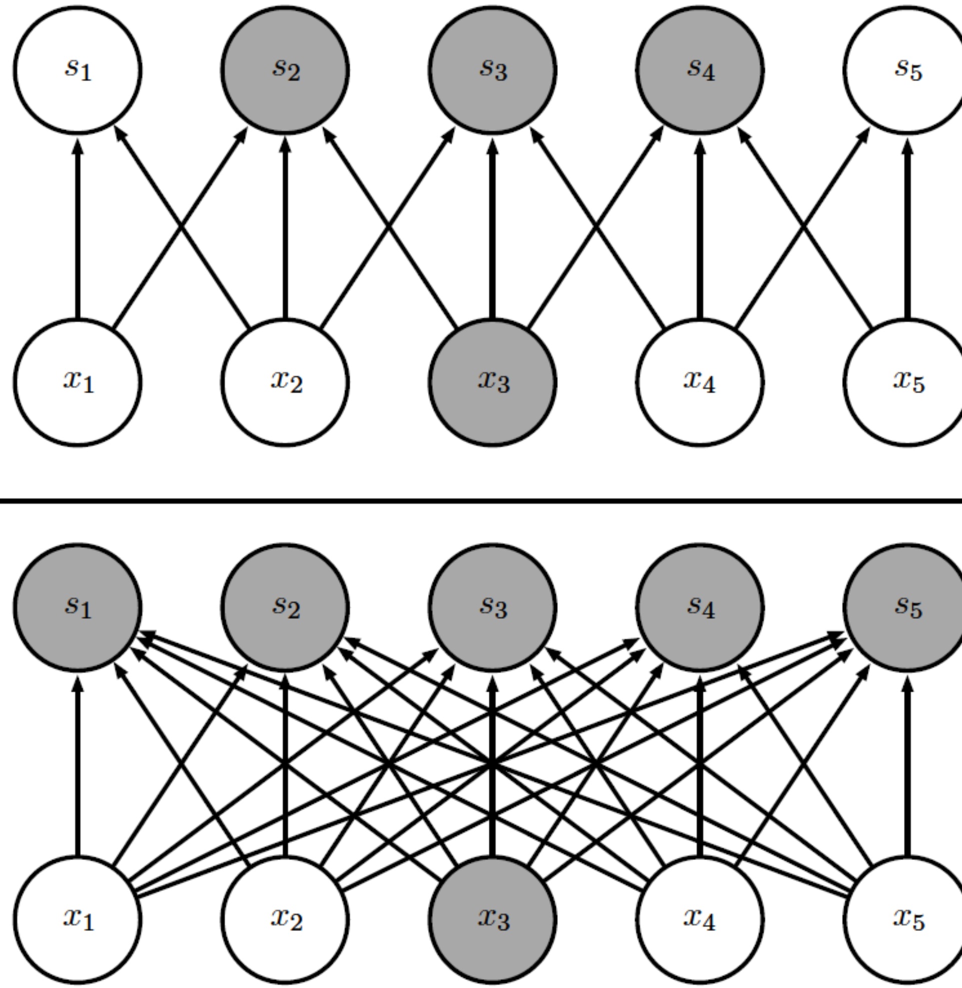
Sparse Interactions

- Also called sparse connectivity or sparse weights
- In a typical neural network, every output unit interacts with every input unit
- Convolutional networks have sparse interactions, by making the kernel smaller than the input
- Efficiency of sparse interactions
  - Typical layer of neural network with m inputs and n outputs: mn parameters and O(mn) running time
  - Limiting the number of connections for each output to k: kn parameters and O(kn) running time

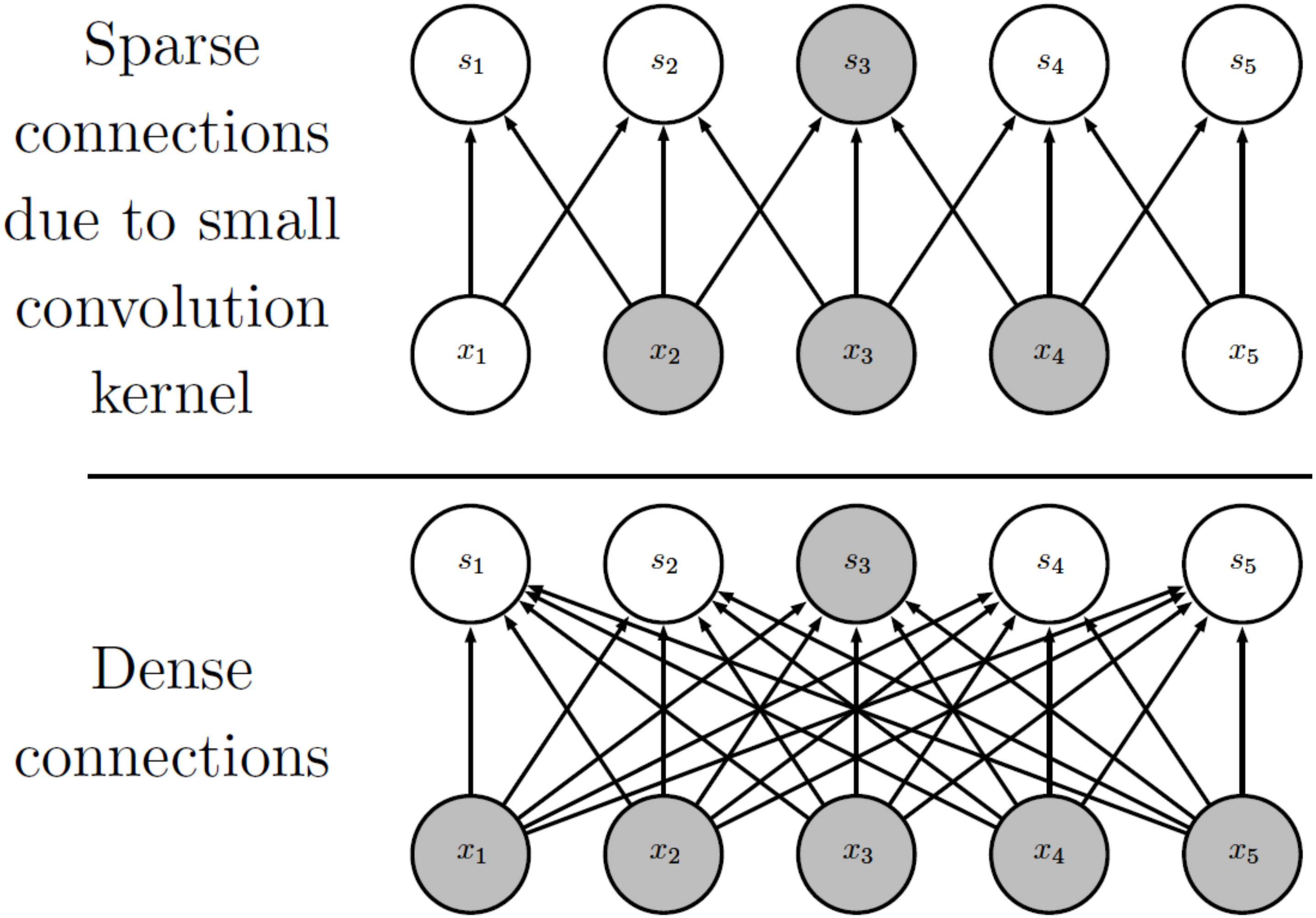
Sparse Connectivity

Sparse connections due to small convolution kernel

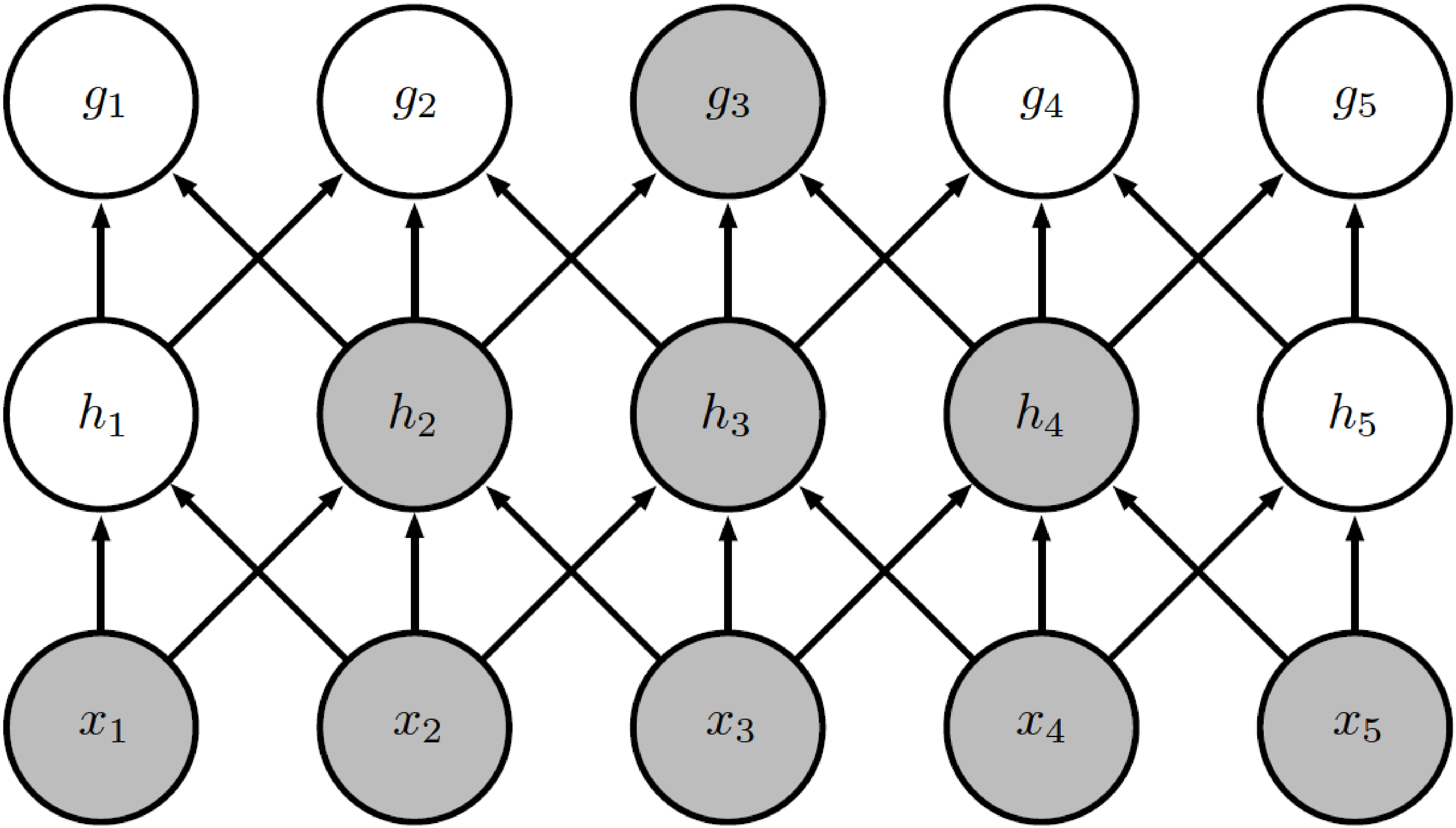
Dense connections



Sparse Connectivity

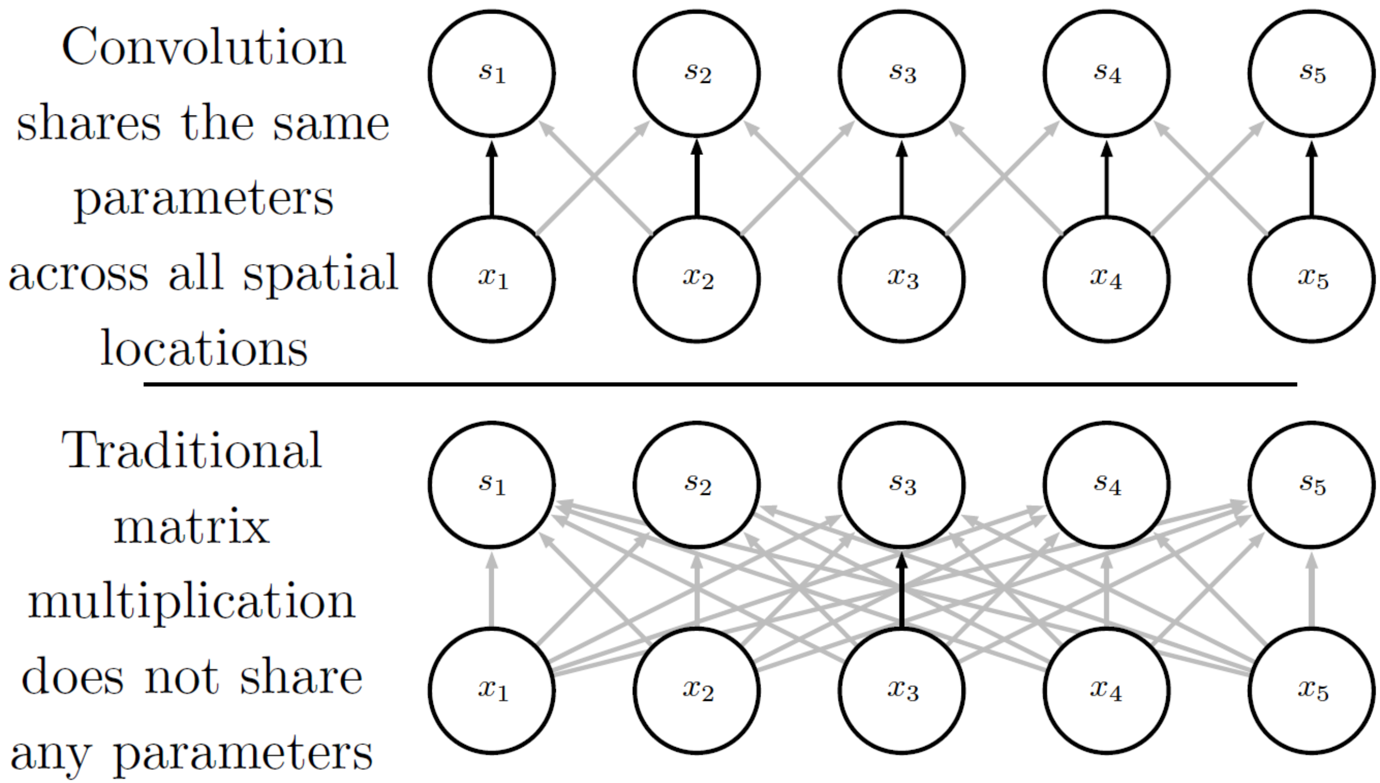


Growing Receptive Fields



Parameter Sharing

- Use the same parameter for more than one function in a model
- Parameter sharing = tied weights
- Requires only O(k) parameters although the running time is O(kn)

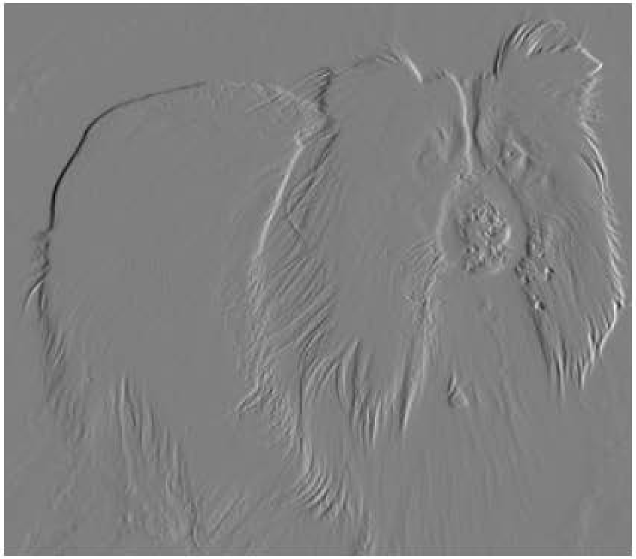


Edge Detection by Convolution

Input

1	-1

Kernel



Output

Efficiency of Convolution

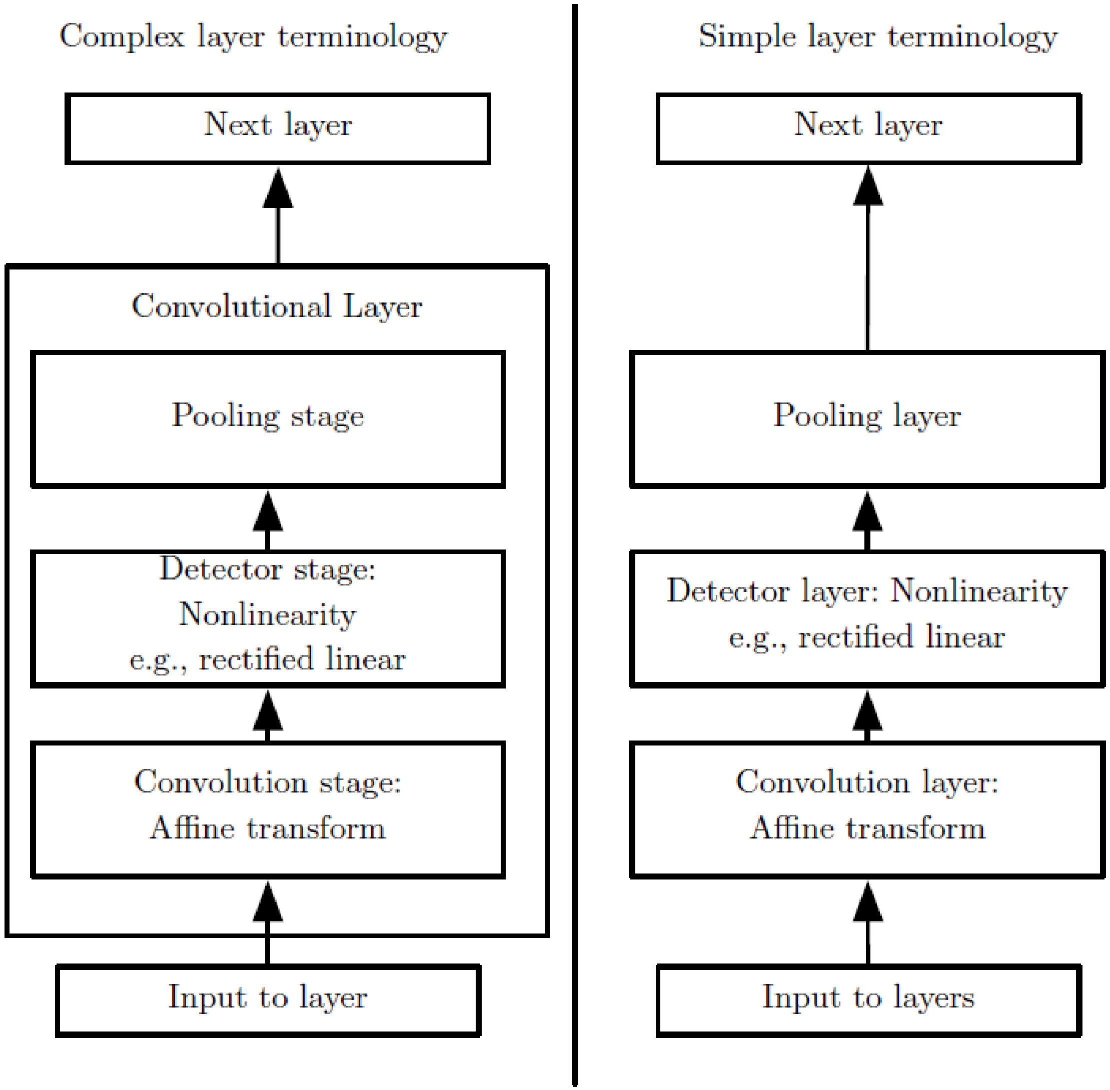
- Input size: 320 by 280
- Kernel size: 2 by 1
- Output size: 319 by 280

	Convolution	Dense matrix	Sparse matrix
Stored floats	2	319*280*320*280 $>$ 8e9	2*319*280 $=$ 178,640
Float muls or adds	319*280*3 $=$ 267,960	$>$ 16e9	Same as convolution (267,960)

Equivariant Representation

- Convolution function is equivariant to translation
  - This means that shifting the input and applying convolution is equivalent to applying convolution to the input and shifting it
  - If we move the object in the input, its representation will move the same amount in the output

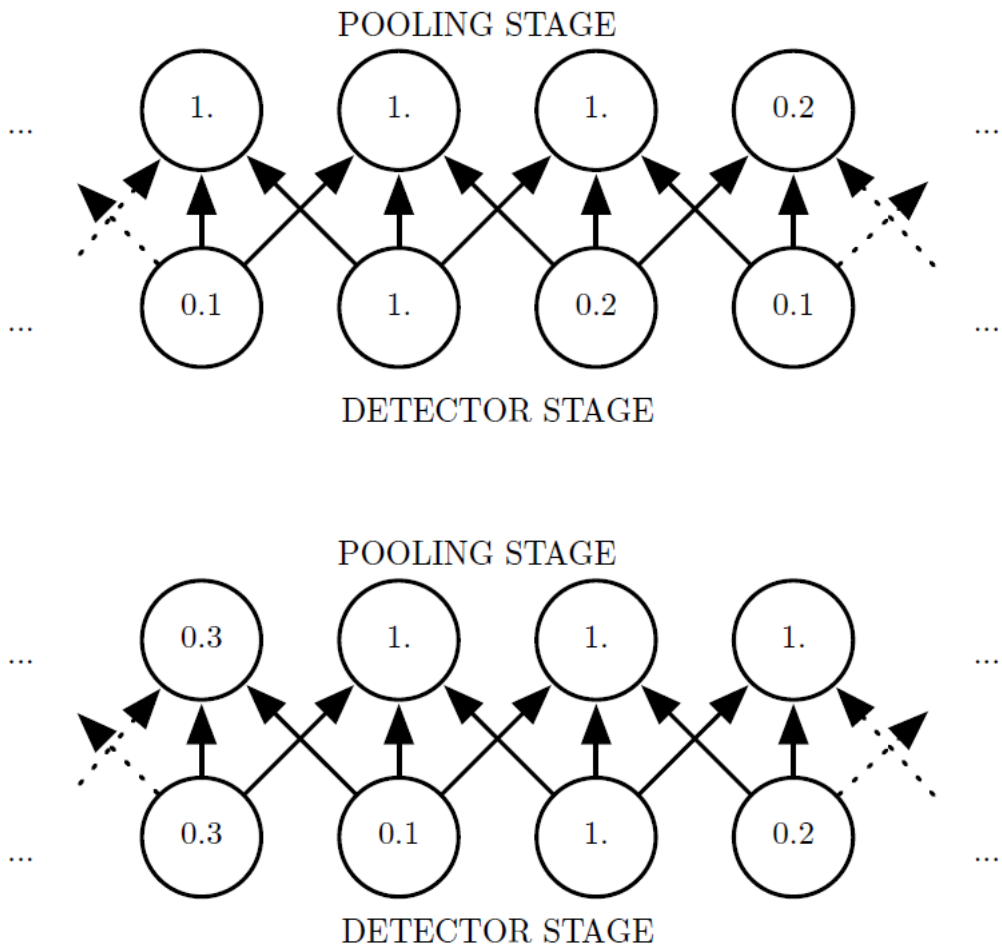
Convolutional Network Components



Pooling

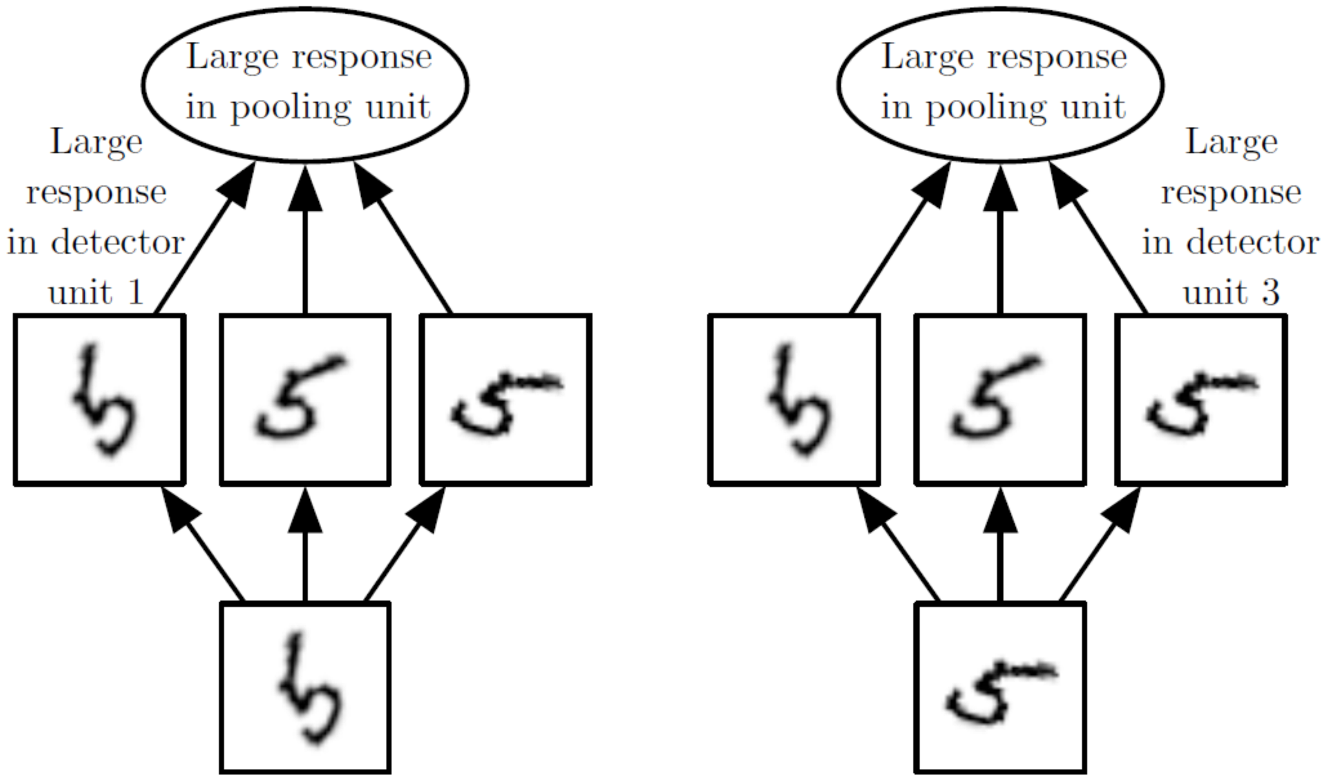
- A pooling function replaces the output of the net at a certain location with a summary statistic of the nearby outputs
  - Max pooling: reports the maximum output within a rectangular neighborhood
  - Average pooling
  - L^2 norm of a rectangular neighborhood
  - Weighted average on the distance from the central pixel
- Pooling helps make the representation become approximately invariant to small translation of the input
  - This can be useful if we care more about whether some feature is present than exactly where it is

Max Pooling and Invariance to Translation



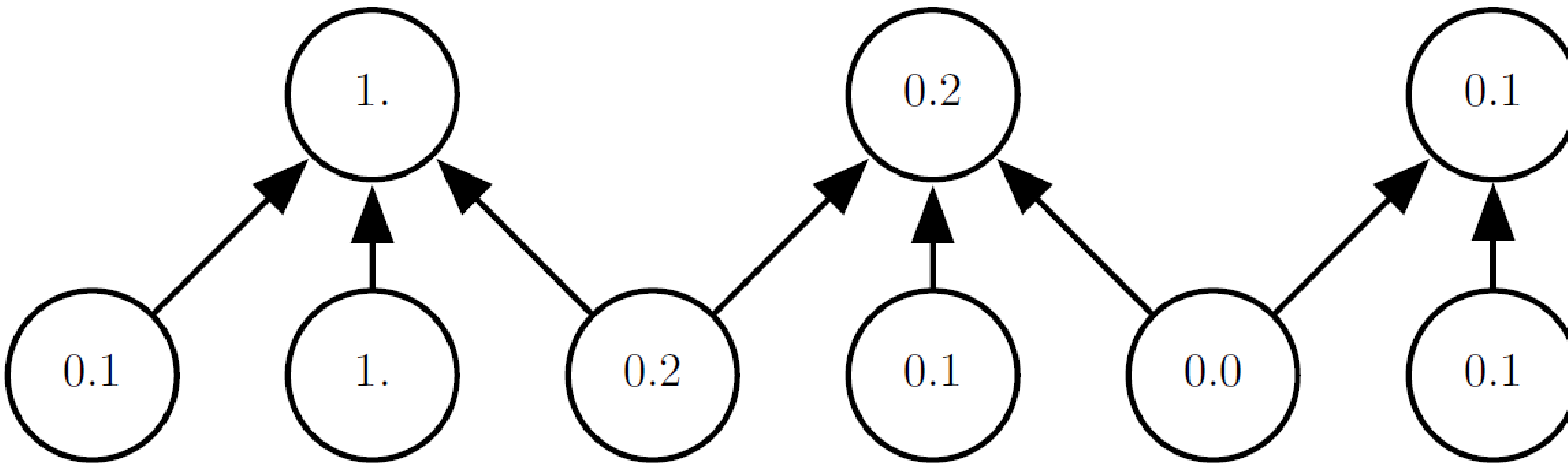
Cross-Channel Pooling and Invariance to Learned Transformations

- A pooling unit that pools over multiple features that are learned with separate parameters can learn to be invariant to transformations of the input
  - E.g., invariant to rotation

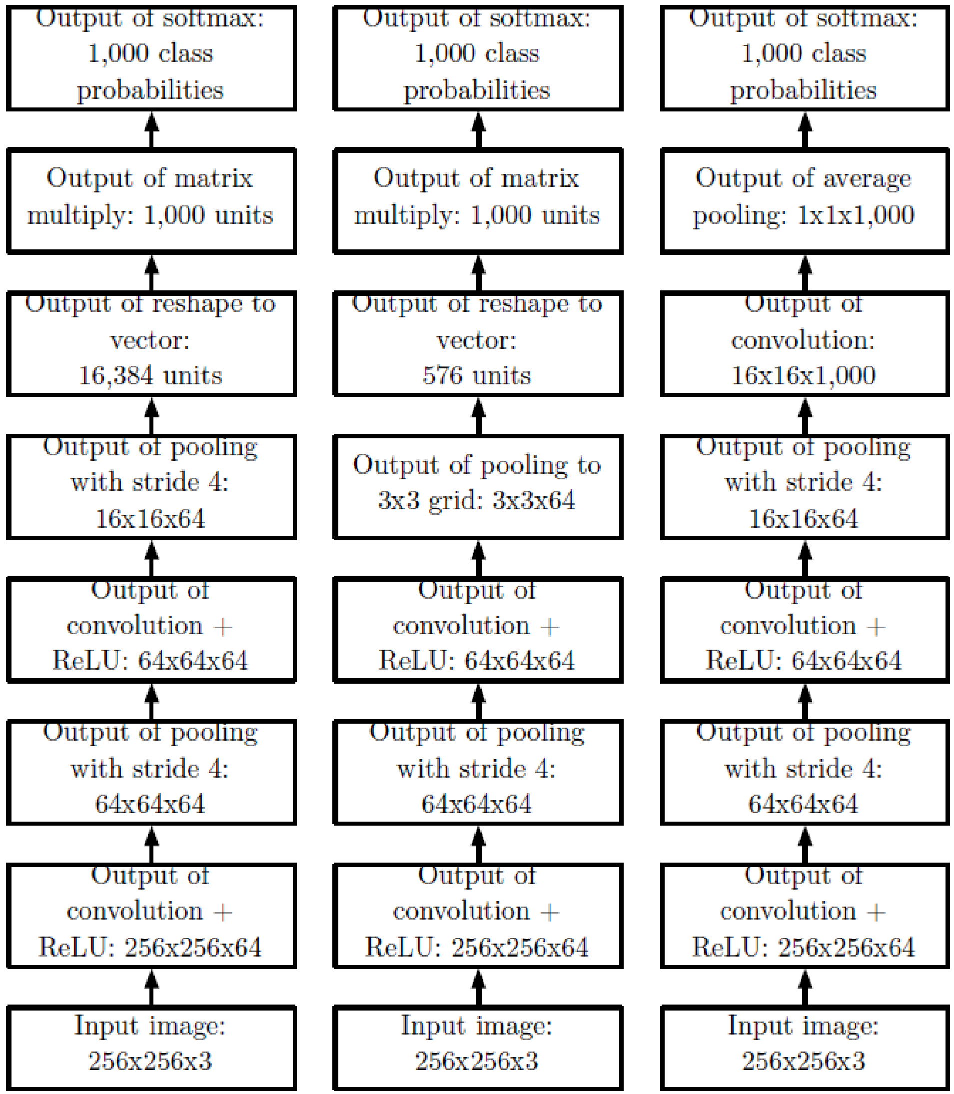


Pooling with Downsampling

- Use fewer pooling units than detector units



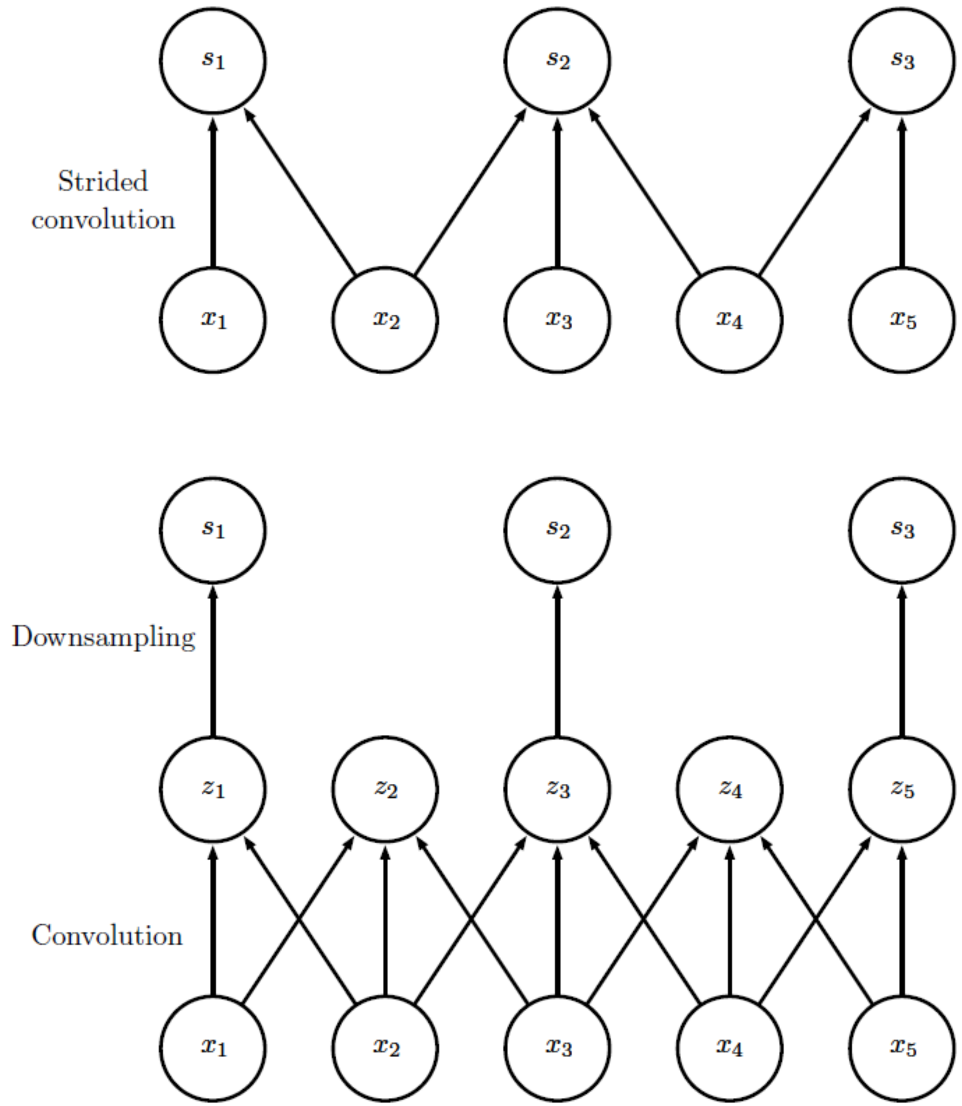
Example Classification Architectures



Convolution and Pooling as an Infinitely Strong Prior

- Prior probability distribution on parameters: encodes our beliefs about what models are reasonable, before we have seen any data
- Priors can be considered weak or strong
  - Weak prior: a prior with high entropy
  - Strong prior: a prior with low entropy
- Convolutional net can be viewed as a fully connected net but with an infinitely strong prior over its weights
  - The weights for one hidden unit must be identical to the weights of its neighbor, but shifted in space
  - The weights must be zero, except for in the small, spatially contiguous receptive field assigned to that hidden unit
- Pooling is an infinitely strong prior that each unit should be invariant to small translations

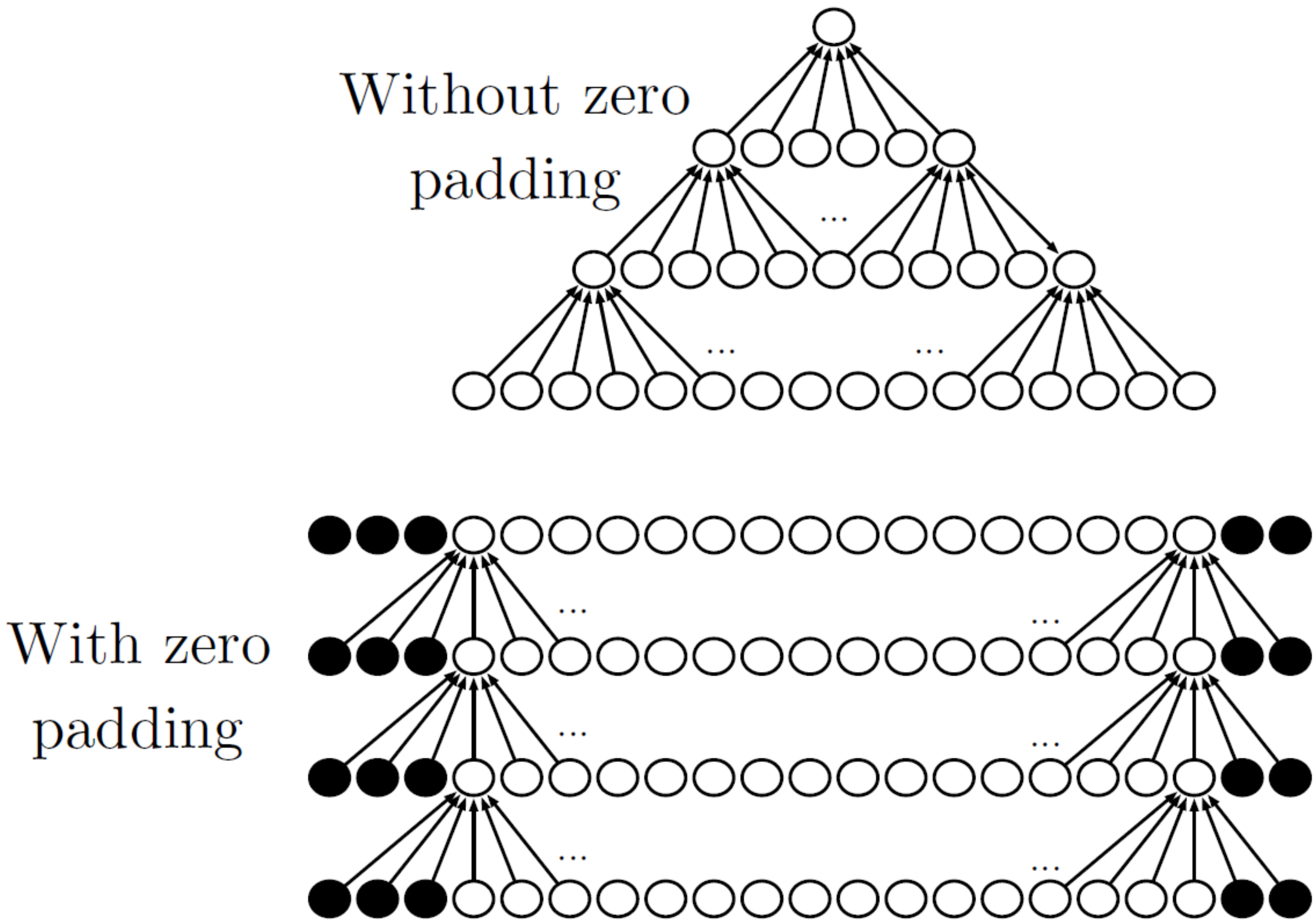
Convolution with Stride



Stride of two

Zero Padding Controls Size

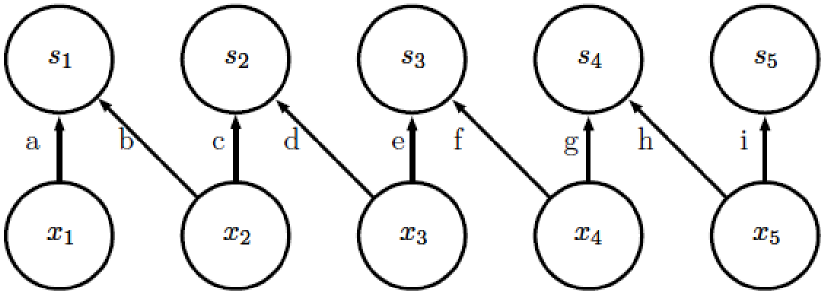
- Zero padding allows us to make an arbitrary deep convolutional network



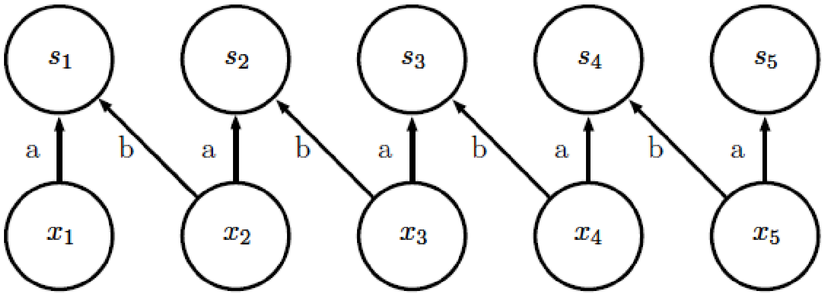
Locally Connected Layer

- Similar to convolution, but every connection has its own weight
- Also called unshared convolution
- Useful when we know that each feature should be a function of a small part of space, but there is no reason to think that the same feature should occur across all of space
  - E.g., if we want to tell if an image is a picture of a face, we only need to look for the mouth in the bottom half of the image

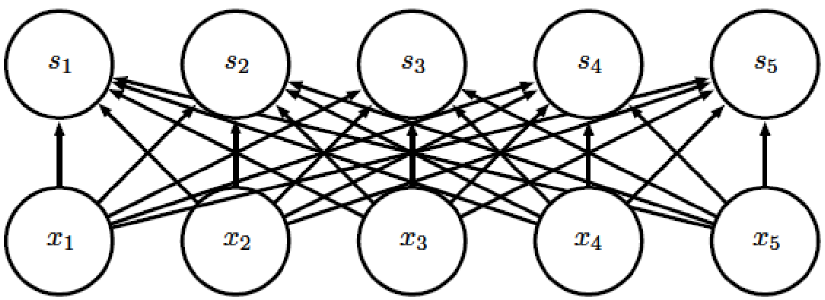
Kinds of Connectivity



Local connection: like convolution, but no sharing



Convolution

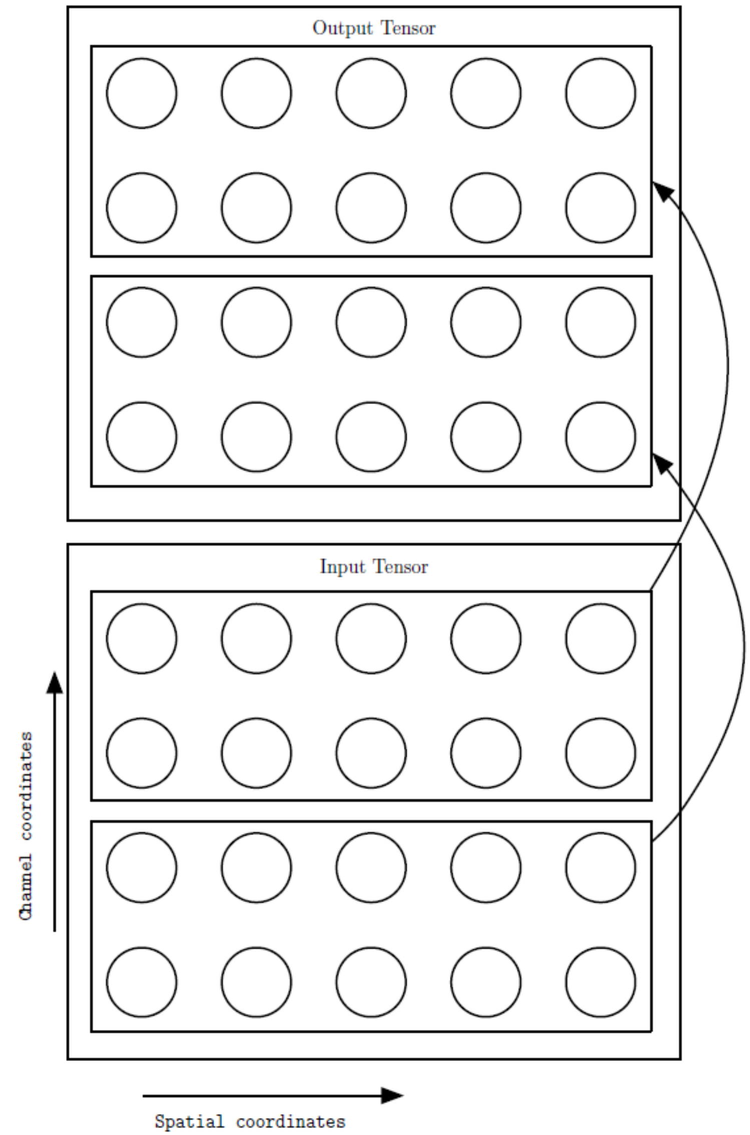


Fully connected

Partial Connectivity Between Channels

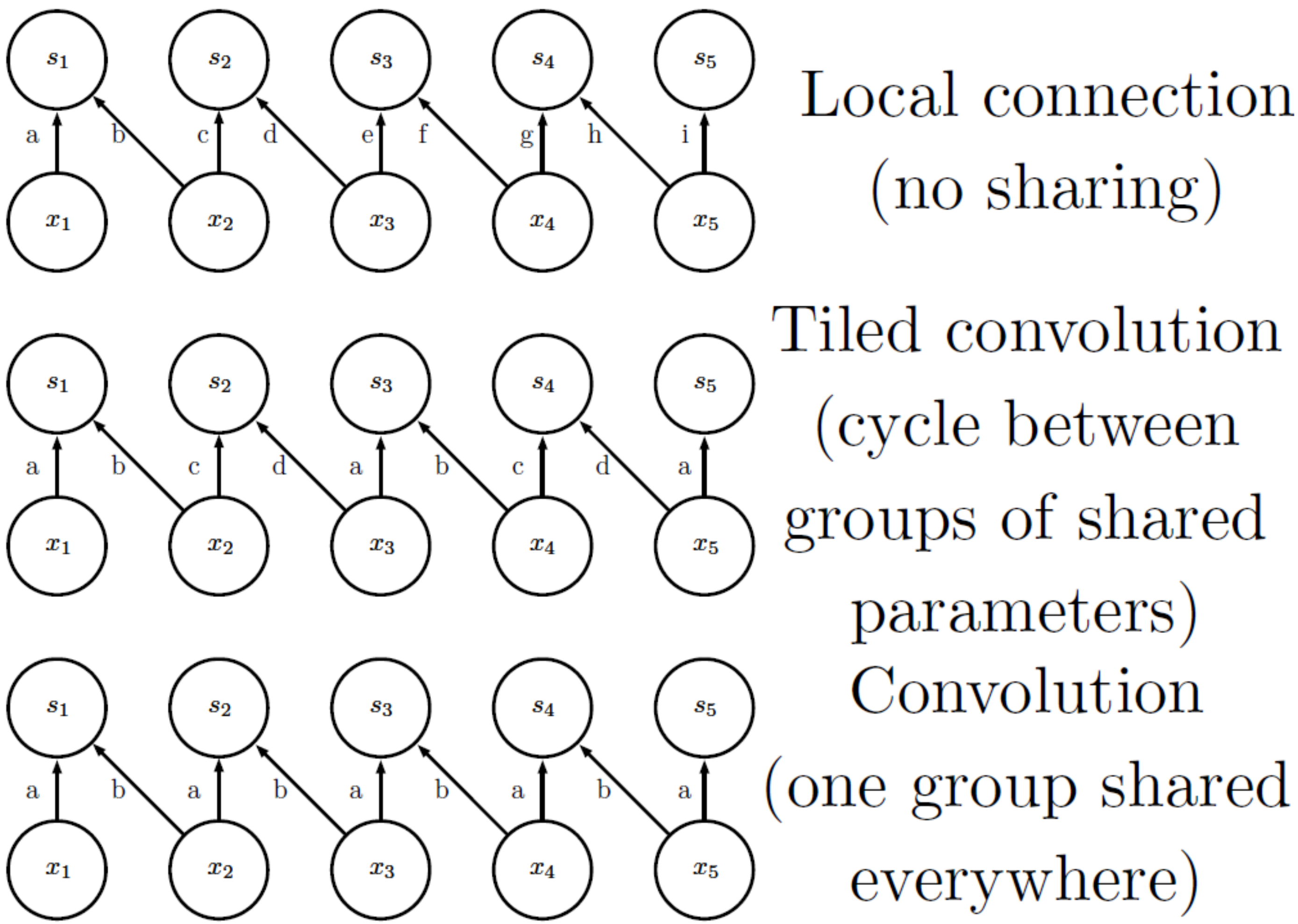
- Further restrict connectivity
- E.g., constrain each output channel i to be a function of only a subset of the input channels l

- This allows the network to have fewer parameters in order to reduce memory consumption and increase statistical efficiency
- This also reduces the amount of computation needed to perform forward and back-propagation



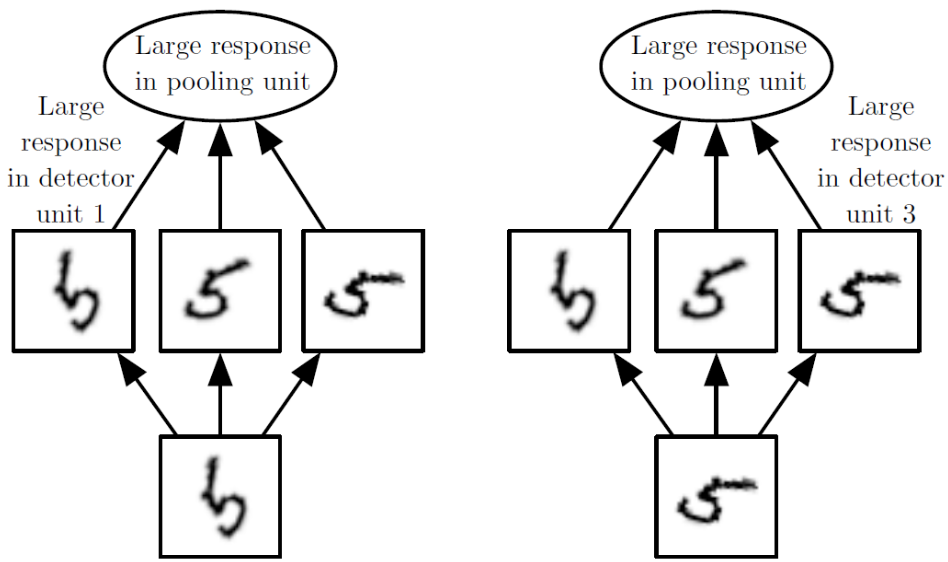
Tiled Convolution

- A compromise between a convolutional layer and a locally connected layer



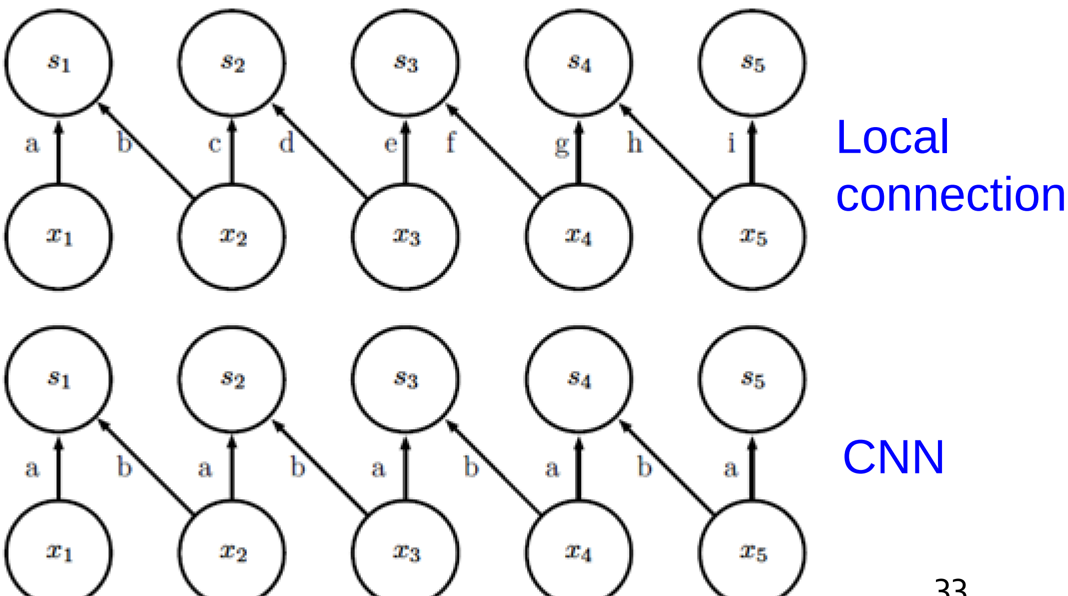
Interaction of Convolution and Max-Pooling

- Both locally connected layers and tiled convolutional layers have an interesting interaction with max-pooling: the detector units of these layers are driven by different filters
- If these filters learn to detect different transformed versions of the same underlying features, then the max-pooled units become invariant to the learned transformation
  - E.g., rotation
- Standard convolutional layers are hard-coded to be invariant specifically to translation



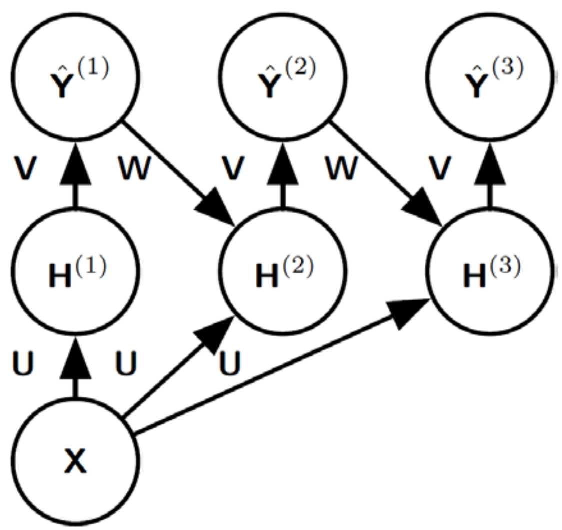
Backpropagation in CNN

- Backpropagation in CNN is similar to that of typical neural network; the only difference comes from the parameter sharing
- Let $g_i = \frac{\partial J}{\partial s_i}$
- No parameter sharing (e.g. local connection)
  - $\frac{\partial J}{\partial a} = x_1 g_1, \frac{\partial J}{\partial b} = x_2 g_1, \ldots, \frac{\partial J}{\partial i} = x_5 g_5$
- Parameter sharing (e.g. CNN)
  - $\frac{\partial J}{\partial a} = \sum_i x_i g_i, \frac{\partial J}{\partial b} = \sum_i x_{i+1} g_i$



Recurrent Pixel Labeling

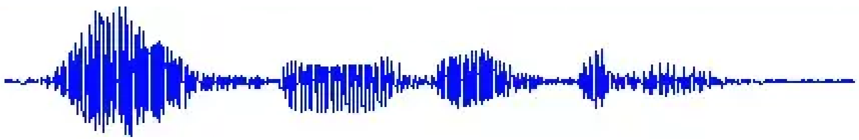
- Convolutional networks can be used to output a high-dimensional, structured object
- E.g., pixel-wise labeling of images
  - Output a tensor S where $S_{i,j,k}$ is the probability that pixel (j,k) belongs to class i
  - One strategy is to produce an initial guess of the image labels, then refine this initial guess using the interactions between neighboring pixels



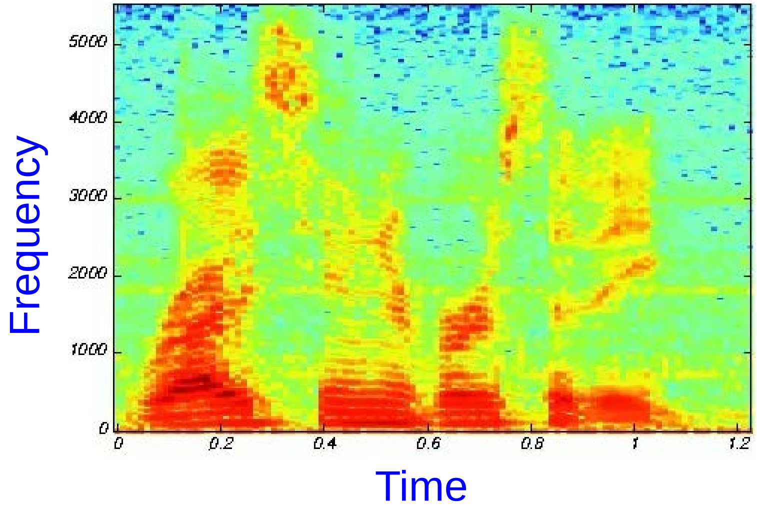
Recurrent convolutional network for pixel labeling

Data Types for CNN

- Single channel
  - 1-D: audio waveform: amplitude of the waveform over time



  - 2-D (spectrogram): audio data preprocessed with a Fourier transform
    - Different rows corresponding to different frequencies
    - Different columns corresponding to different points in time



  - 3-D: volumetric data: CT scan image

Data Types for CNN

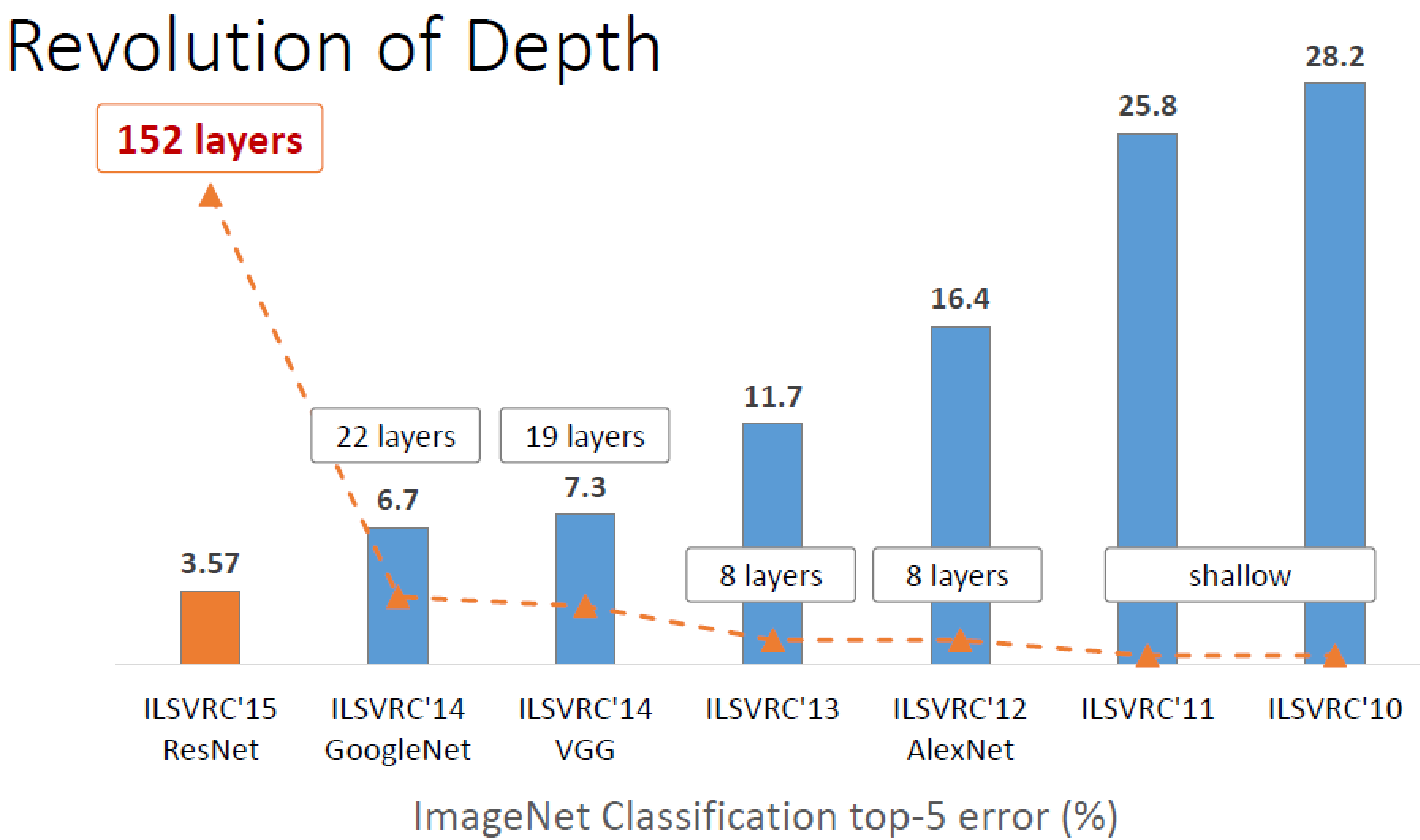
- Multi-channel
  - 1-D: skeleton animation data
    - At each point in time, the pose of the character is described by a specification of the angles of each of the joints in the character's skeleton. Each channel in the data represents the angle about one axis of one joint
  - 2-D: color image data
  - 3-D: color video data

Major Architectures

- Spatial Transducer Net: input size scales with output size, all layers are convolutional

- All Convolutional Net: no pooling layers, just use strided convolution to shrink representation size

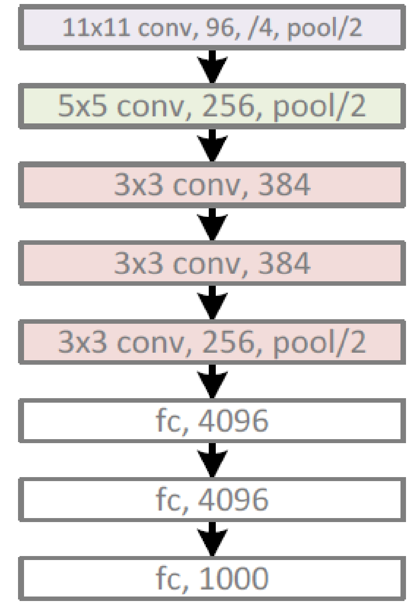
Major Architectures

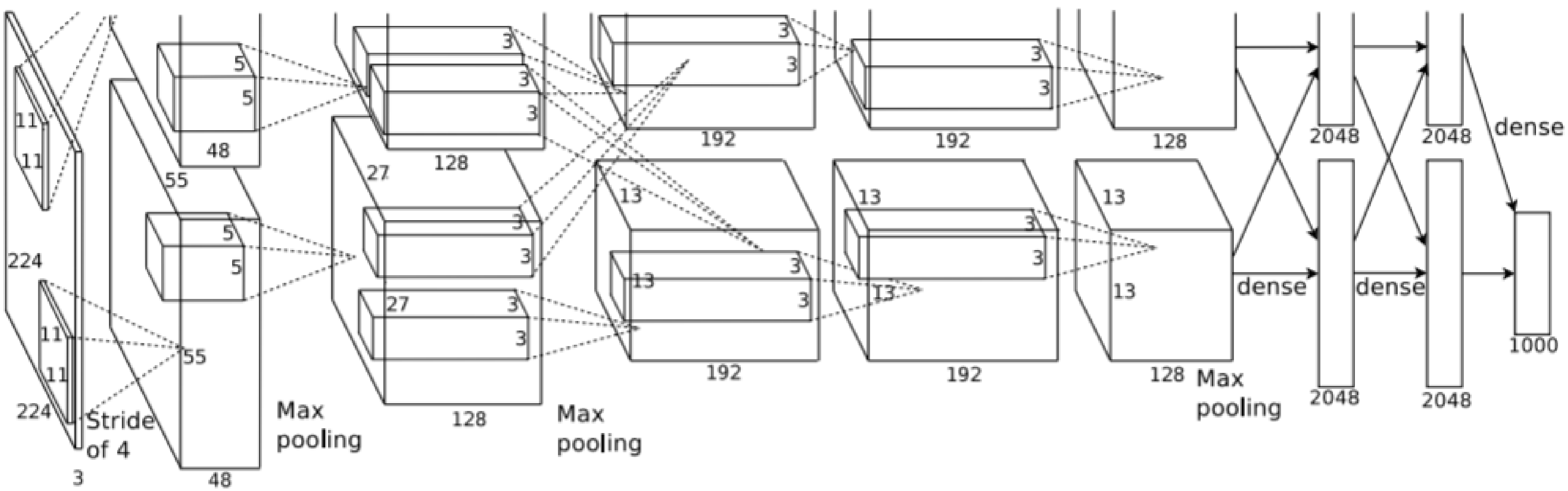


Slide: Kaiming He

Alexnet

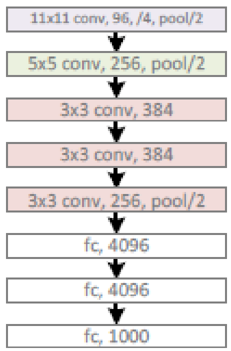
- 8 layers
  - 1st layer: filters 224 x 224 x 3 input image with 96 kernels of size 11 x 11 x 3 with a stride of 4 pixels (+max pooling)
  - 2nd layer: filters the input with 256 kernels of 5 x 5 x 48 (+max pooling)
  - 3rd layer: filters the input with 384 kernels of size 3 x 3 x 256
  - …
  - 6, 7, 8th layers: fully connected layers



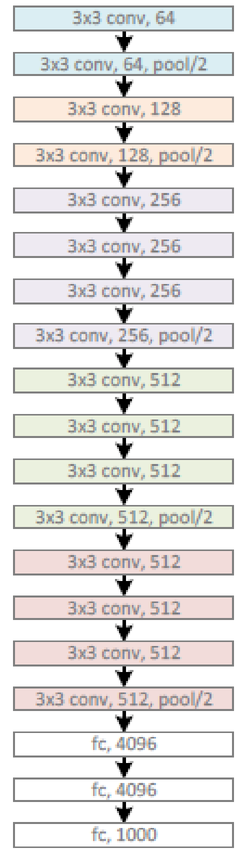


Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)



VGG, 19 layers
(ILSVRC 2014)



GoogleNet, 22 layers
(ILSVRC 2014)

Slide: Kaiming He

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)

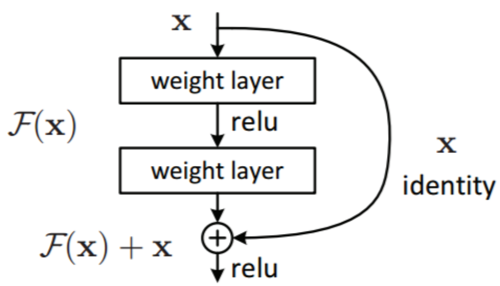
VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

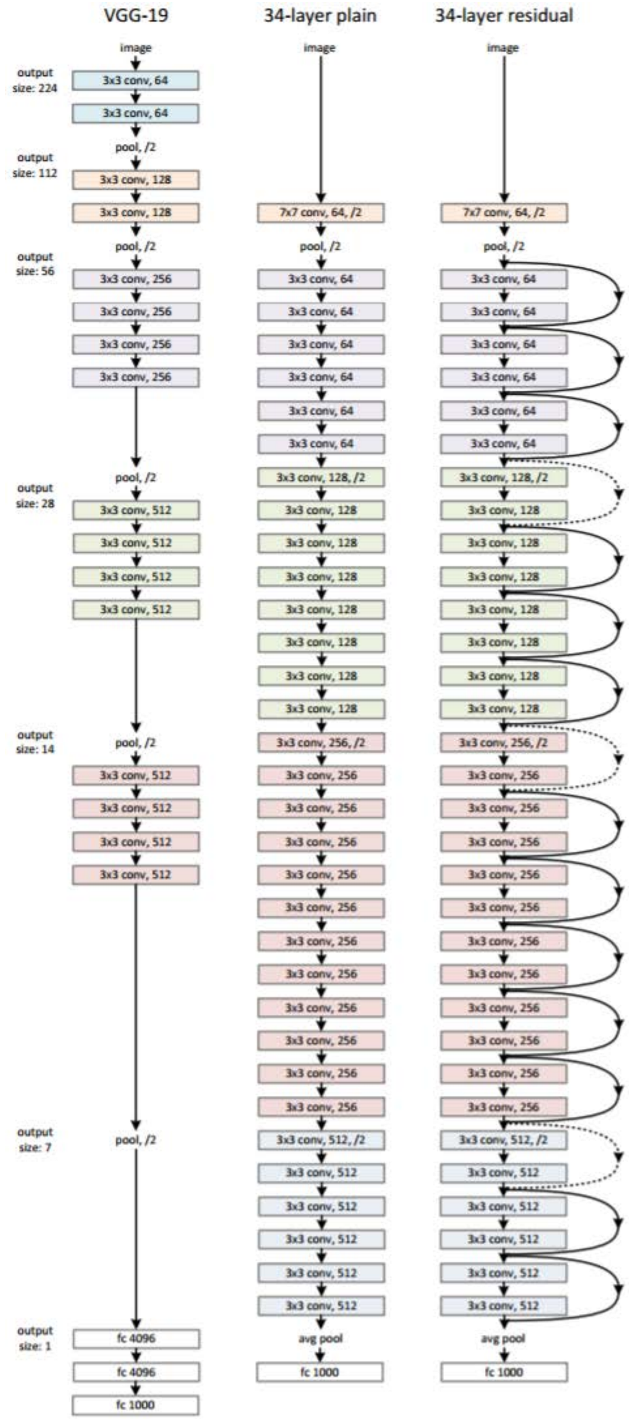
Slide: Kaiming He

ResNet

- Shortcut connection



Residual learning: a building block.



What you need to know

- Convolutional Neural Network
  - Main idea:
    - Replace matrix multiplication in neural nets with convolution
    - Pooling
  - Efficiency: from sparse interaction and parameter sharing
  - Major architectures
    - AlexNet, GoogleNet, ResNet

Questions?