Introduction to Artificial Intelligence

Lecture 5: Neural networks II

September 18, 2025

Recap: What are neural networks?

Input (A)	Output (B)	Application
Email	Spam? (0/1)	Spam filtering
Audio	Text transcripts	Speech recognition
English	Chinese	Machine translation
Ad, user info	Click? (0/1)	Online advertising
Image, radar info	Position of other cars	Self-driving cars
Image of phone	Defect? (0/1)	Visual inspection
Sequence of words	The next word	Chatbot

How large language models (LLMs) work

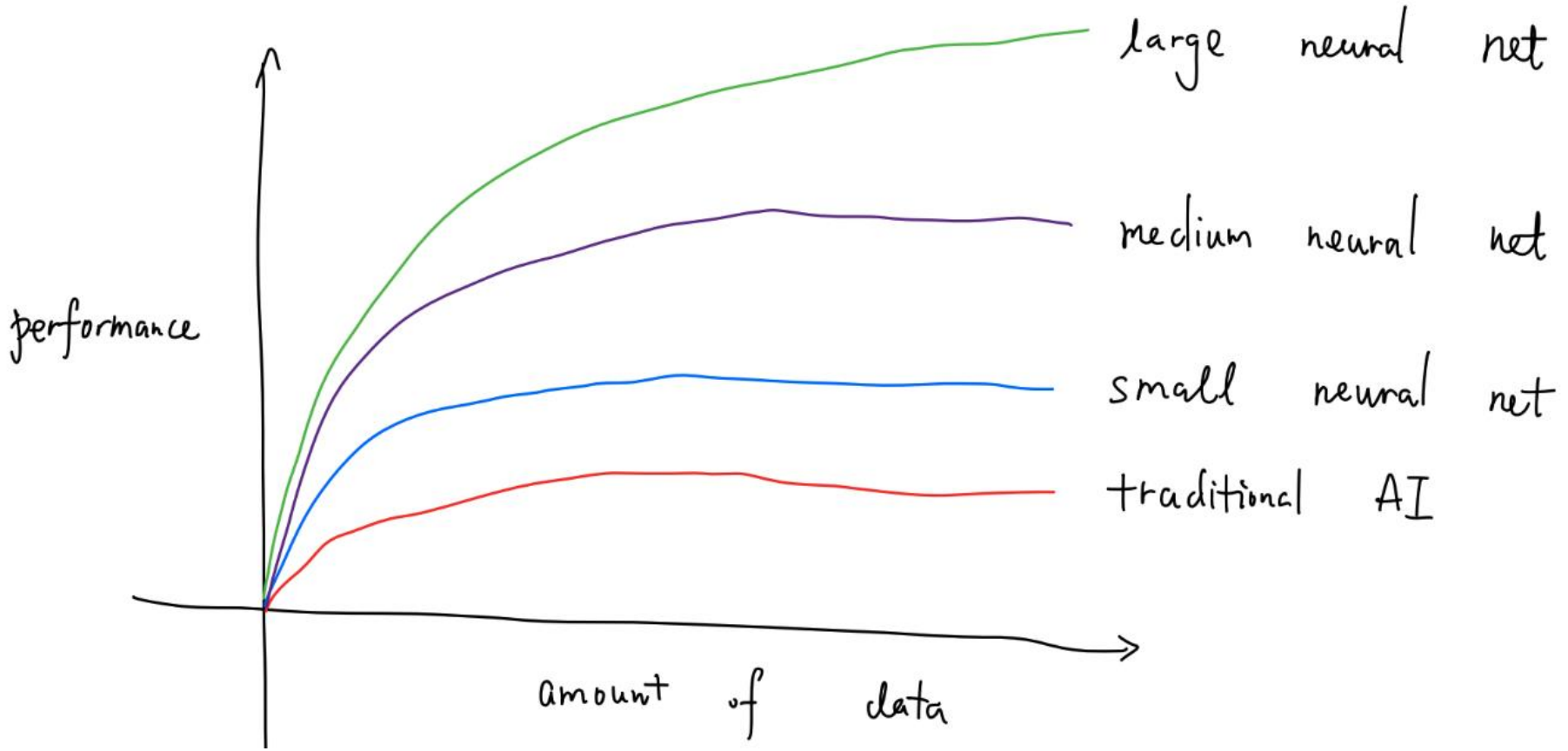
- LLMs are built by using supervised learning (A -> B) to repeatedly predict the next word

CS4100 is a class about artificial intelligence

Input (A)	Output (B)
CS4100	is
CS 4100 is	a
CS 4100 is a	class
CS 4100 is a class	about
CS 4100 is a class about	artificial
CS 4100 is a class about artificial	intelligence

- When we train a very large AI system on a lot of data (hundreds of billions of words), we get a large language model like ChatGPT

From neural networks to AI



- It helps to have more data, plus more compute

Example of a table of data (dataset)

Size of house (square feet)	# of bedrooms	Price (1000$)
523	1	305
645	1	384
726	2	540
1088	2	653

A

B

- Regression models: map A to B
- Neural networks

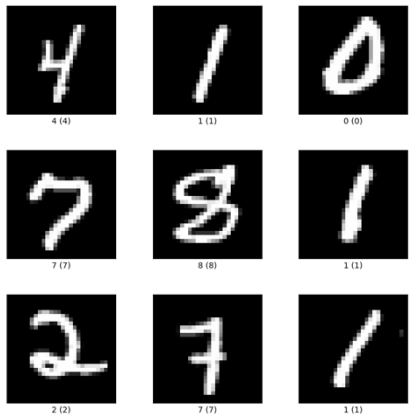
image	label
	cat
	not cat
	cat
	not cat

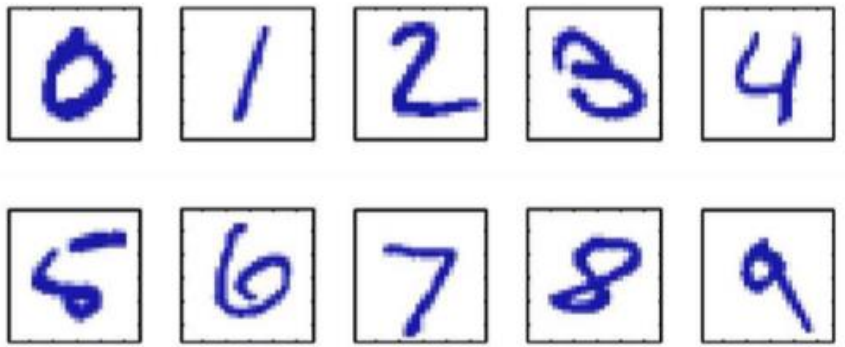
Lecture plan

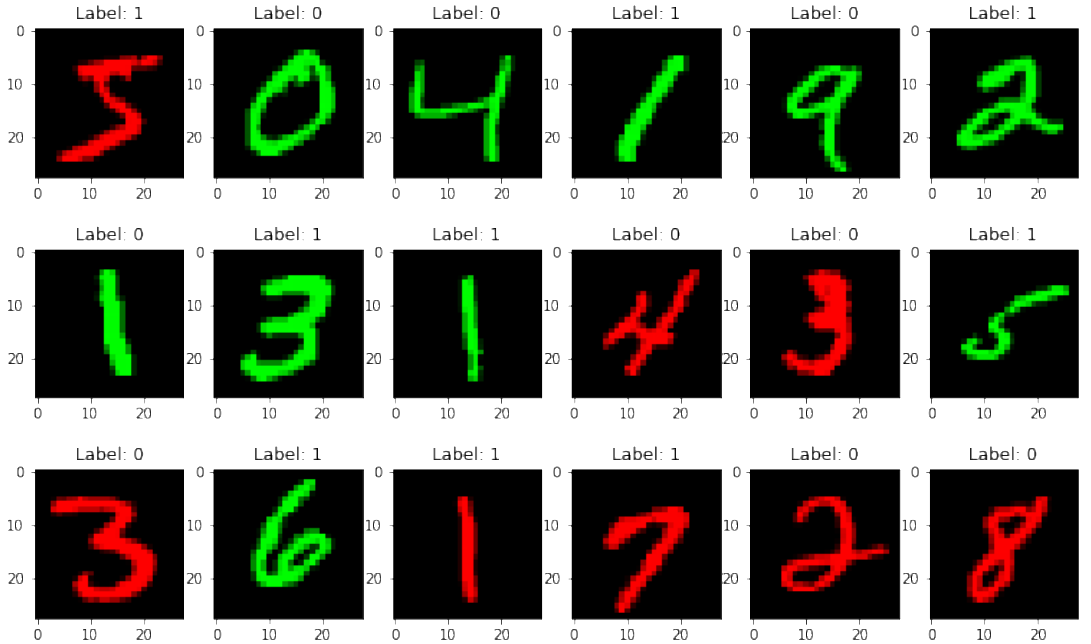
- Introduction to convolutional neural networks
  - What is a convolutional layer?

Application I: Handwritten digit classification

- Classifying handwritten digits

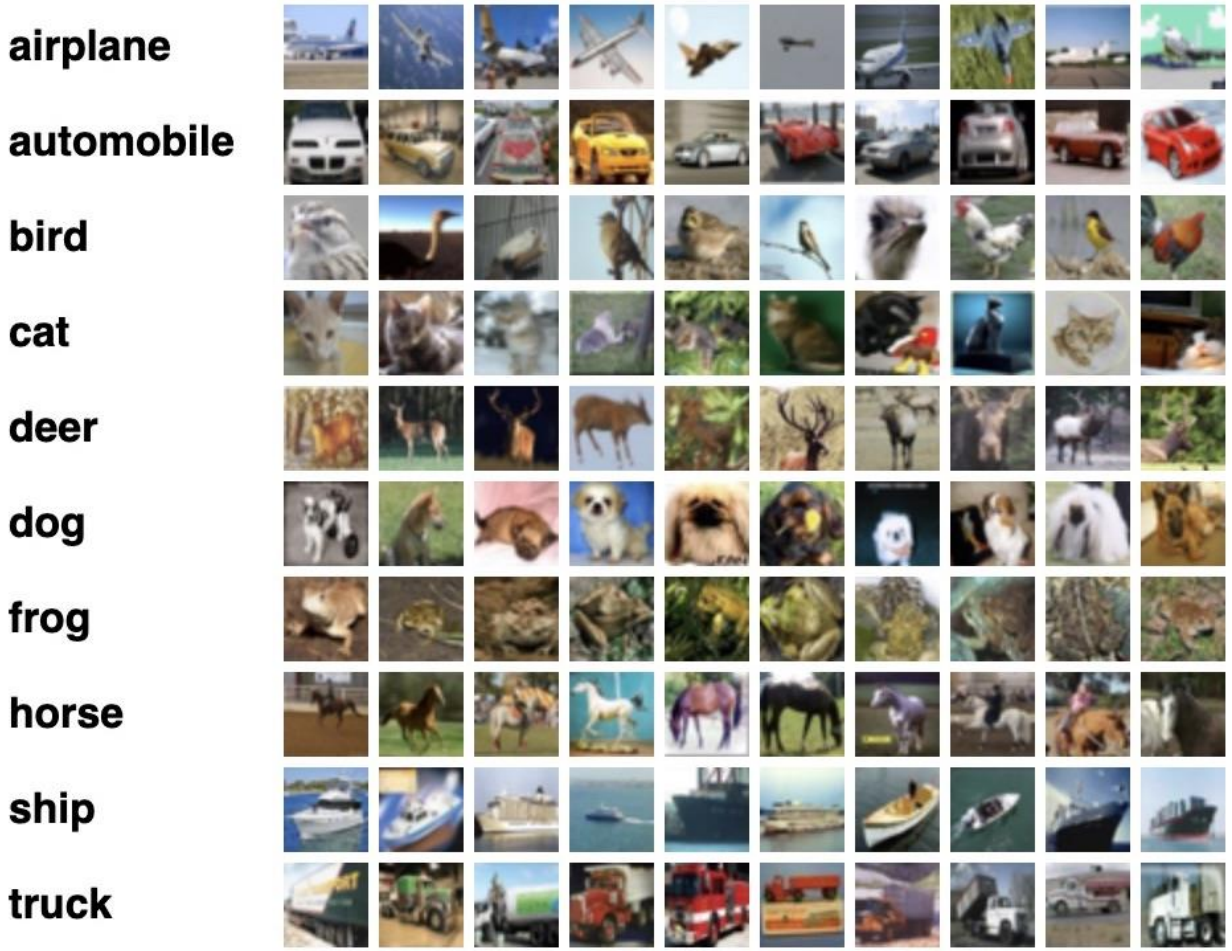






Application II: Object recognition

- CIFAR-10



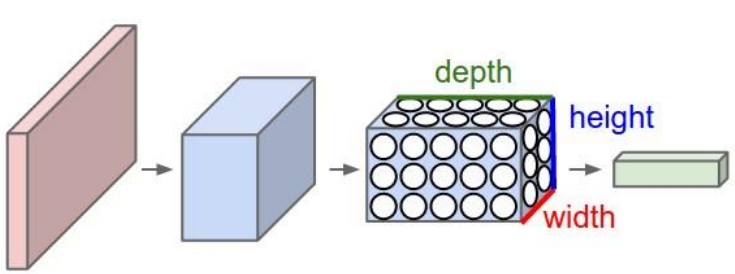
- ImageNet (1000 classes)

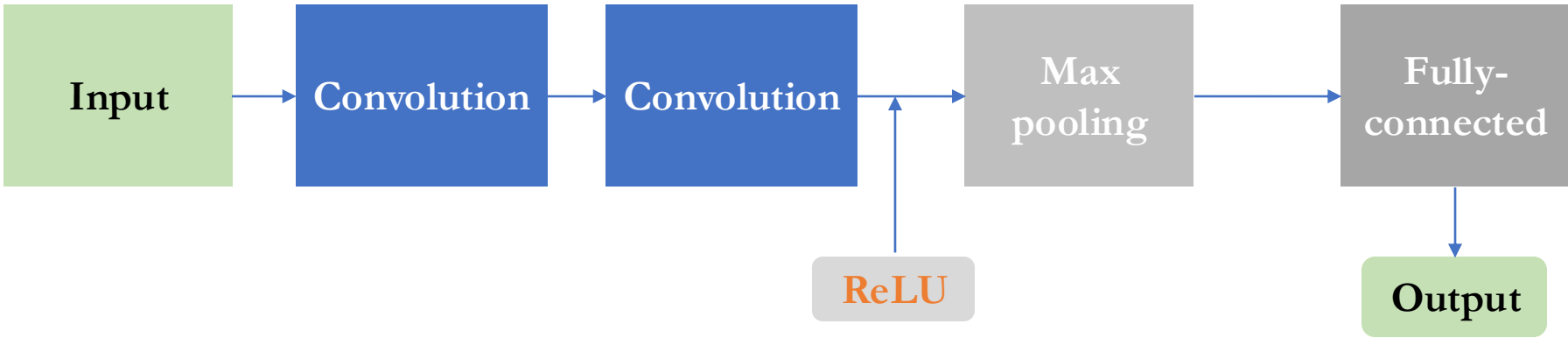
Why not using feedforward neural networks?

- Feedforward neural networks use fully-connected layers to transform the input
- **Fully-connected layers do not scale to large images**
  - A black-and-white digit in MNIST has size 28 by 28. A colored image in CIFAR-10 has size 32 by 32 by 3
  - For MNIST, a fully-connected neuron needs $28 \times 28 = 784$ weights
  - For CIFAR-10, a fully-connected neuron needs $32 \times 32 \times 3 = 3{,}072$ weights
  - Processing larger images requires more parameters

What is a convolutional neural network?

- In convolutional neural networks (CNN), a neuron only connects to a small local region of the image
  - Example: A colored (2D) image is specified by width, height, and depth

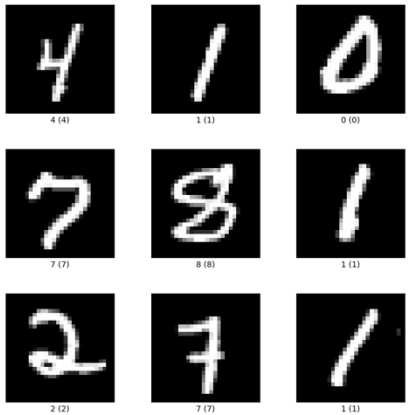




- A CNN involves a combination of the following types of layers
  - **Input layer:** Raw pixel values of the image
  - **Convolution layer:** Combine pixel values in a local region
  - **Pooling layer:** Down sample pixels
  - **Fully-connected layers:** Classification/prediction

Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **First row, first patch**



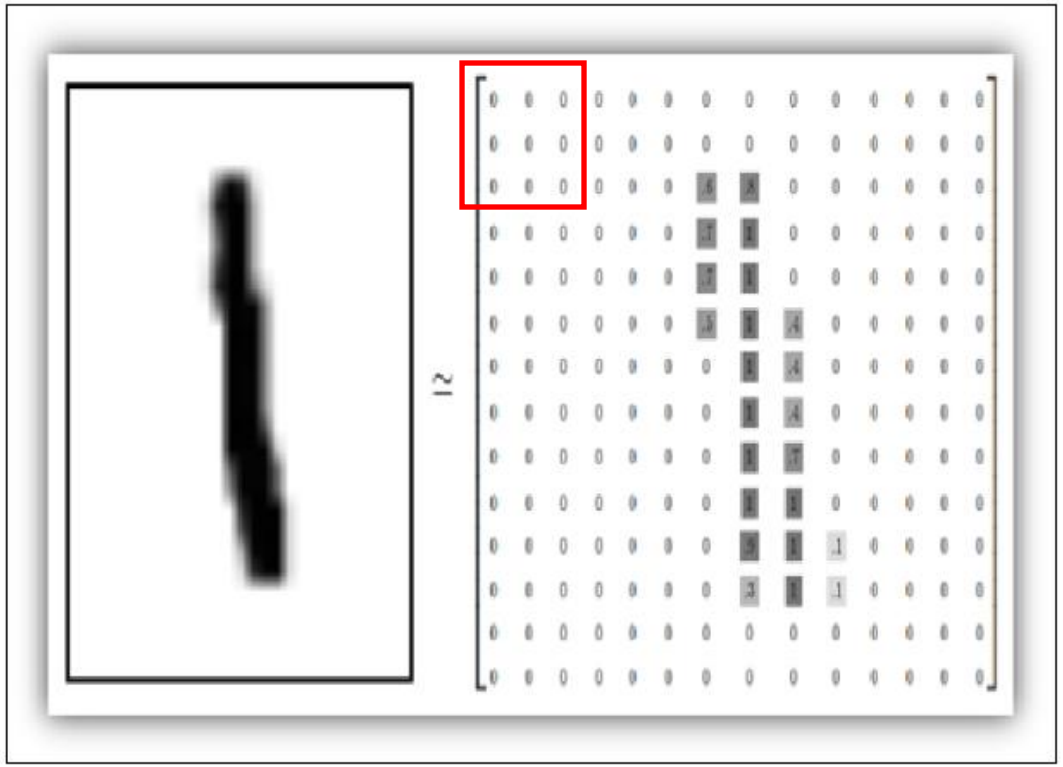


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **First row, second patch**

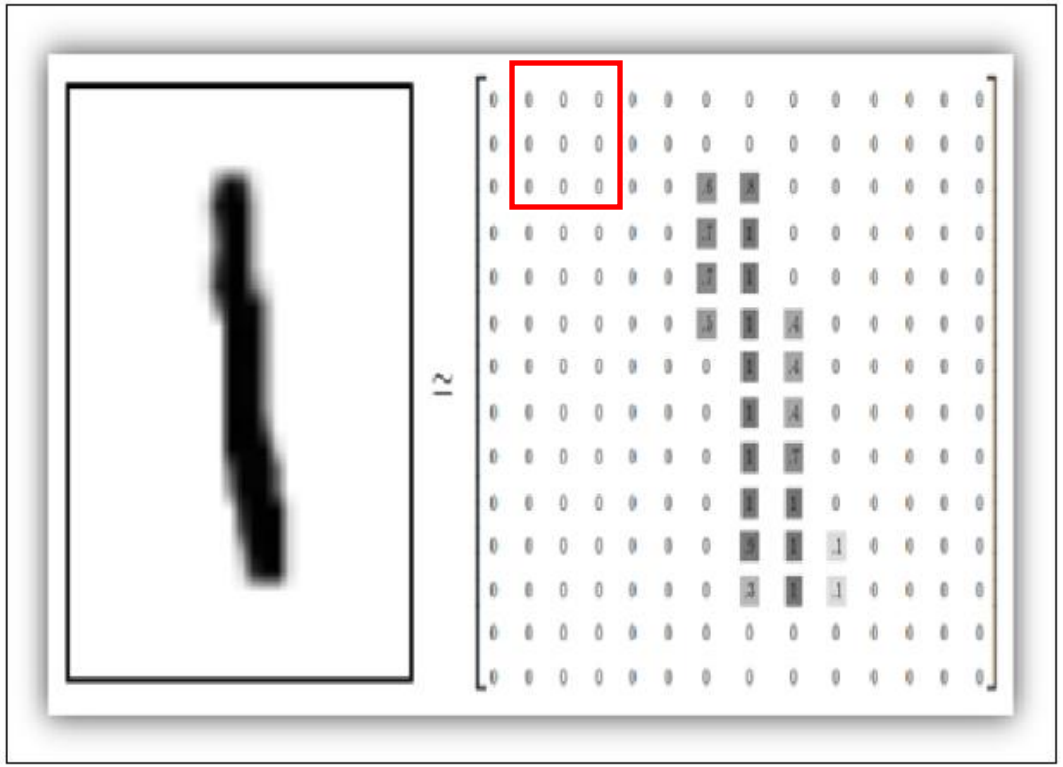


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **First row, third patch**

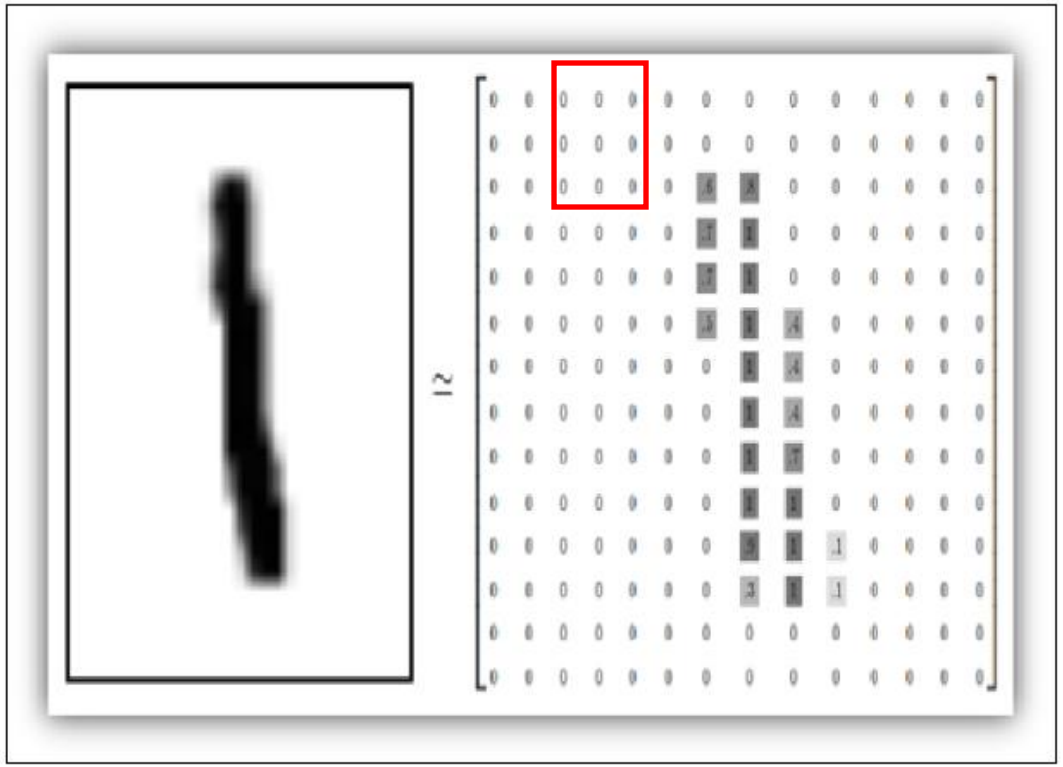


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **First row, last patch**

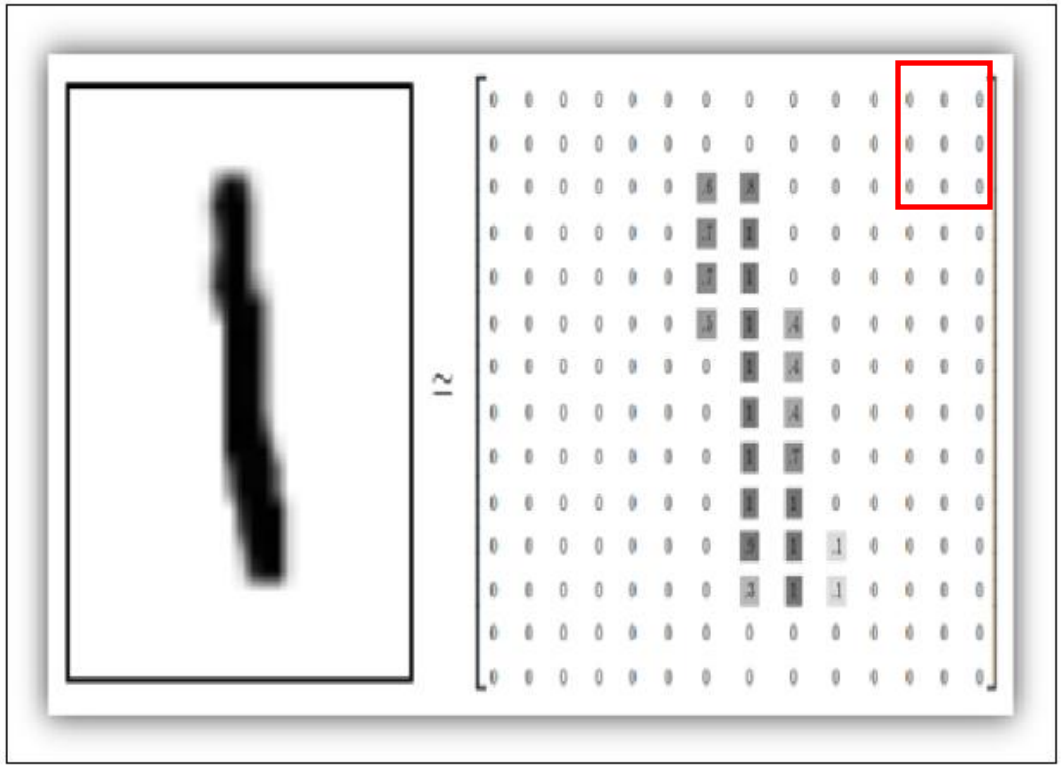


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 1
  - **Second row, first patch**

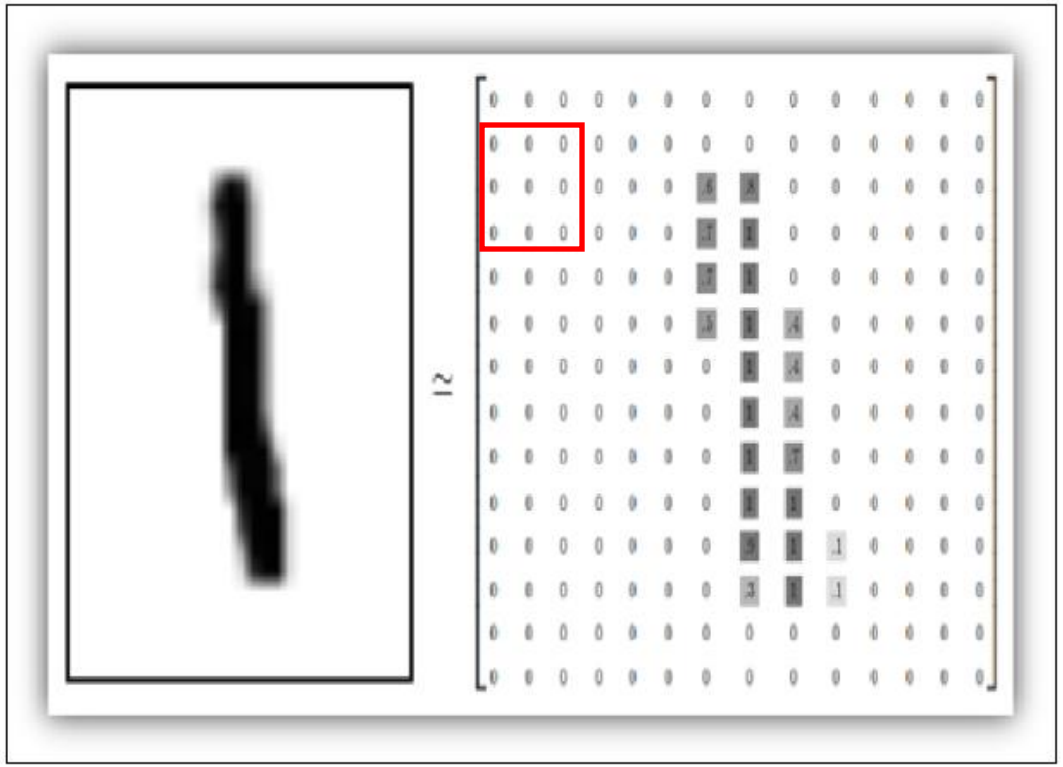


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **Second row, second patch**

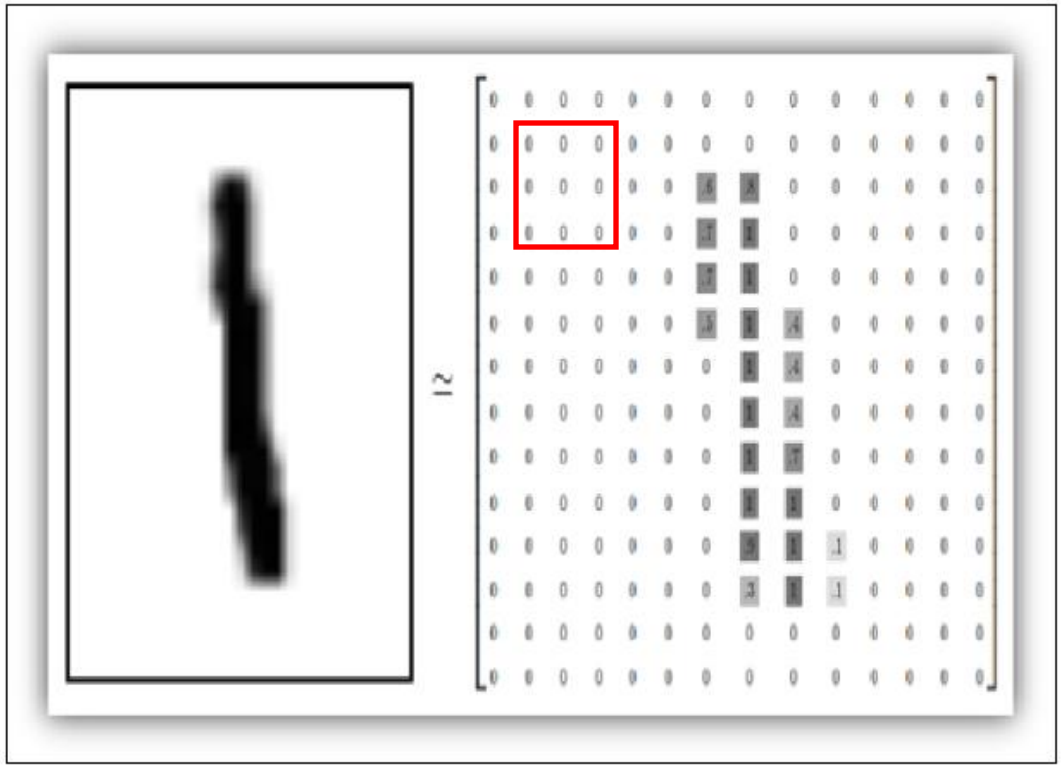
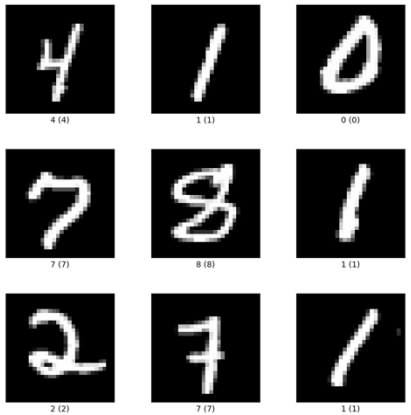
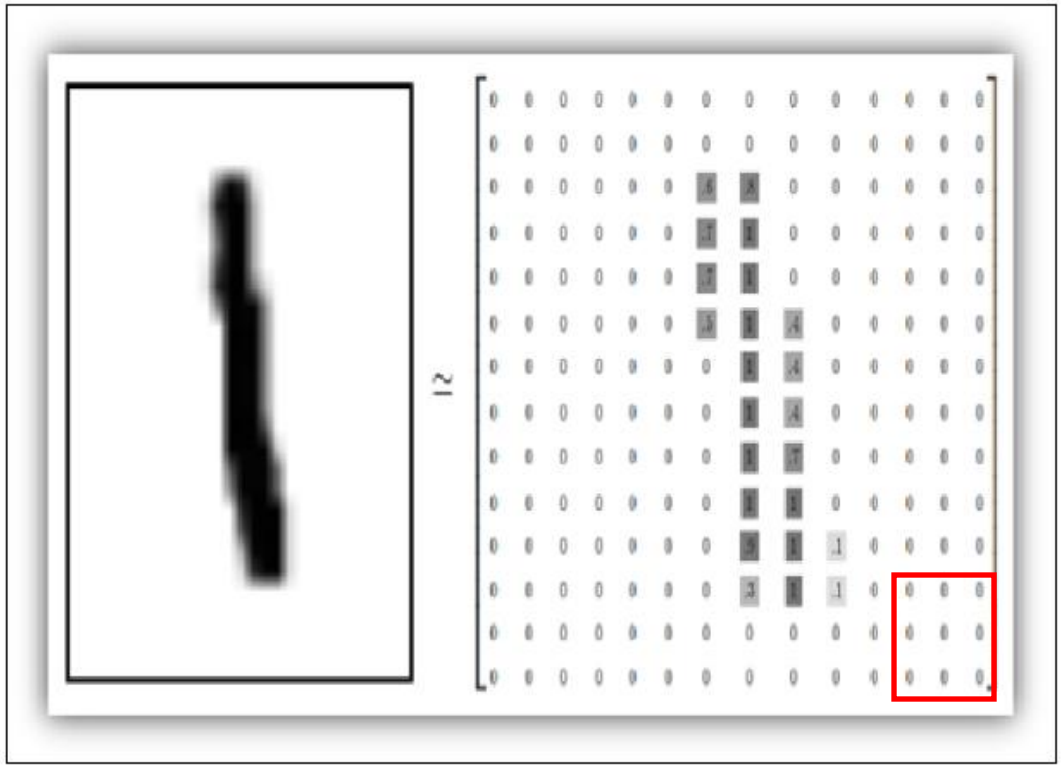


Illustration of a convolution layer

- **Example** (MNIST)
  - **Input** size: 28 by 28
  - **Convolutional layer:**
    - Filter size: (3, 3)
    - Stride: (1, 1)
    - Zero padding size: 0
  - **Last row, last patch**





- **Question:** What is the final output size?

Convolution layer

- **Filter (depth times width):** Larger filter captures coarser spatial patterns, while smaller filters capture finer spatial patterns
- **Stride (depth times width):** How often do we slide the filter? For example, when the stride is 1, we slide the filter one pixel at a time
- **Zero padding:** Pad the input with zeros around the border
- MNIST example: filter size (3, 3), stride size (1, 1), zero padding size 0

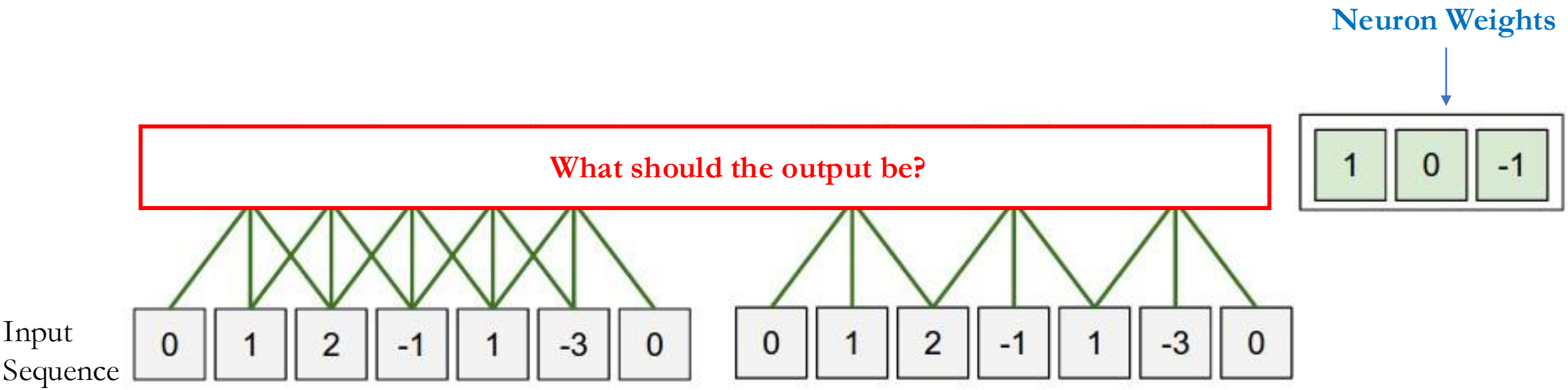
Lecture plan

- Introduction to convolutional neural networks
  - What is a convolutional layer?
  - **Illustrative examples**

Illustration

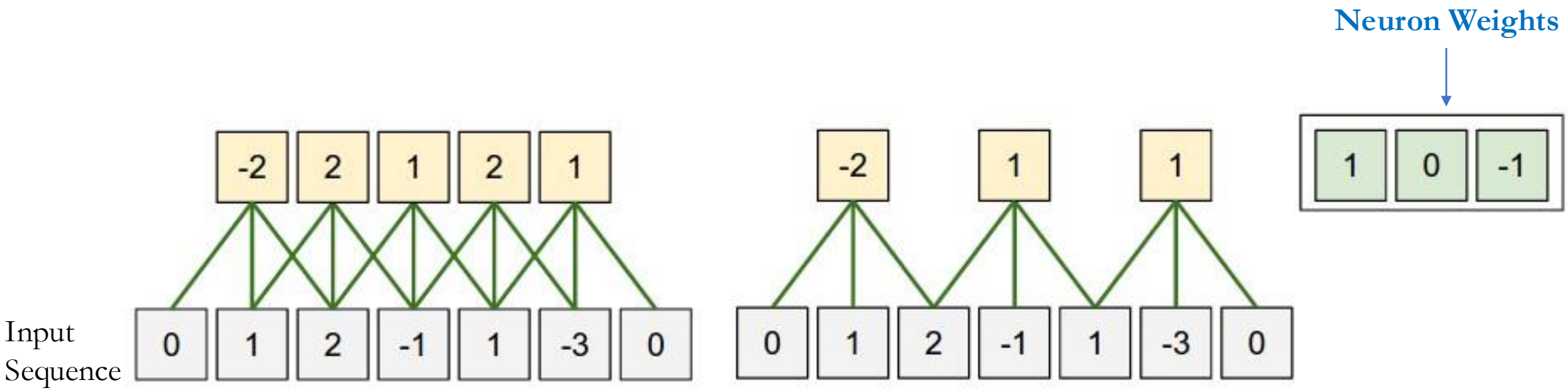
- Input dimension is one, filter size is (3), stride is (1)

- Multiply the input with the neuron weights pixel-by-pixel



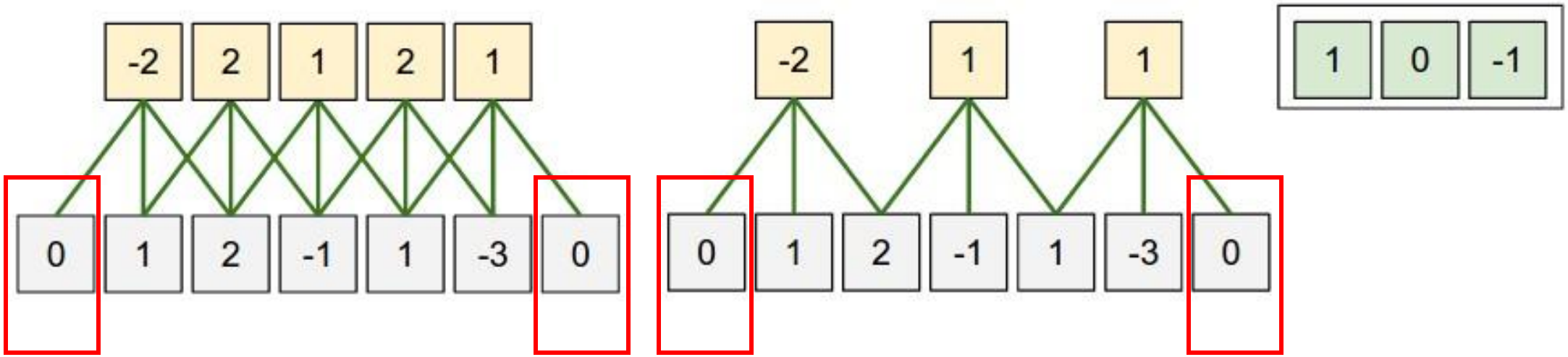
Illustration

- Illustration of spatial arrangement with a simplified example
  - Filter size is (3)
  - Stride is (1)



Explaining zero padding size

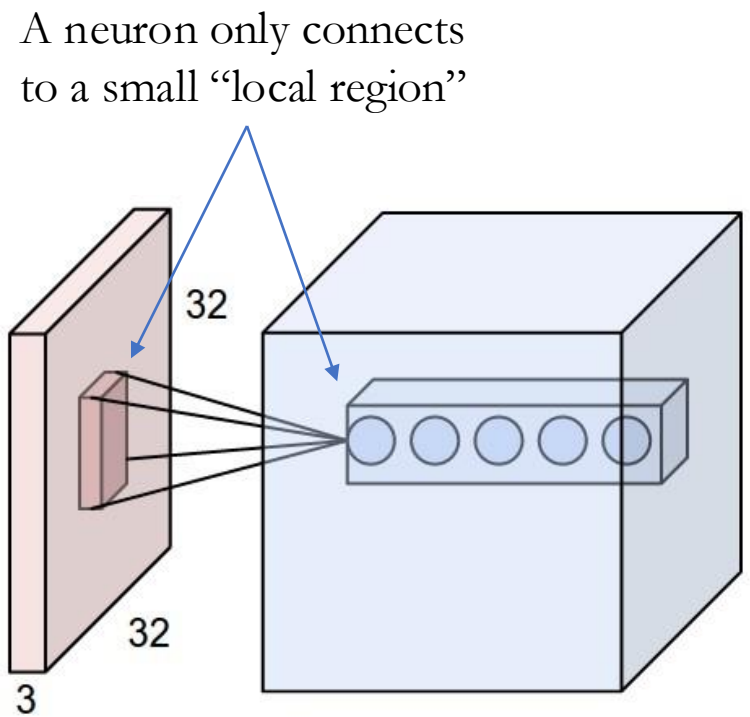
- This example uses **a single zero padding** on **both left and right**



- We can use zero padding to adjust the output dimension, e.g., in sentence classification, use zero padding for fixed (max) length sentences

Stride size

- Filter size and stride size must satisfy that: (image width – filter size) should be divisible by (stride size); Otherwise, add zero padding
- Illustrating the convolution operation for an image of size $(32, 32, 3)$



Summary

- Input: A 3D image of size (W_1, H_1, D_1)
- Convolution layer:
  - Number of filters K
  - Filter size F $(F \times F \times D_1)$
  - Stride size S
  - Zero padding size P
- Produces an output of size (W_2, H_2, D_2). What is it?
  - $W_2 = \frac{W_1 - F + 2P}{S} + 1$
  - $H_2 = \frac{H_1 - F + 2P}{S} + 1$
  - $D_2 = K$
- With parameter sharing, $F \times F \times D_1$ weights per filter, for a total of $(F^2 \times D_1) \times K$ weights

Lecture plan

- Introduction to convolutional neural networks
  - What is a convolutional layer?
  - Illustrative examples
  - **Numpy examples**

Numpy example

- **Input: numpy array *X***
  - $X.shape = (11,11,4)$
- **Convolution layer**
  - Number of filters: $K = 2$
  - Filter size: $5 \times 5 \times 4$
  - Stride size: 2×2
  - Zero padding size: 0
- **Output:** Denote as V
  - Output width and height: $\frac{11-5}{2} + 1 = 4$
  - Depth: 2

Numpy example

- First depth slice, along the first column: Filter parameters W_0, Bias b_0. $W_0.shape = (5, 5, 4)$
  - $V[0,0,0] = np.sum(X[:5, :5, :] * W_0) + b_0$
  - $V[1,0,0] = np.sum(X[2:7, :5, :] * W_0) + b_0$
  - $V[2,0,0] = np.sum(X[4:9, :5, :] * W_0) + b_0$
  - $V[3,0,0] = np.sum(X[6:11, :5, :] * W_0) + b_0$

Numpy example

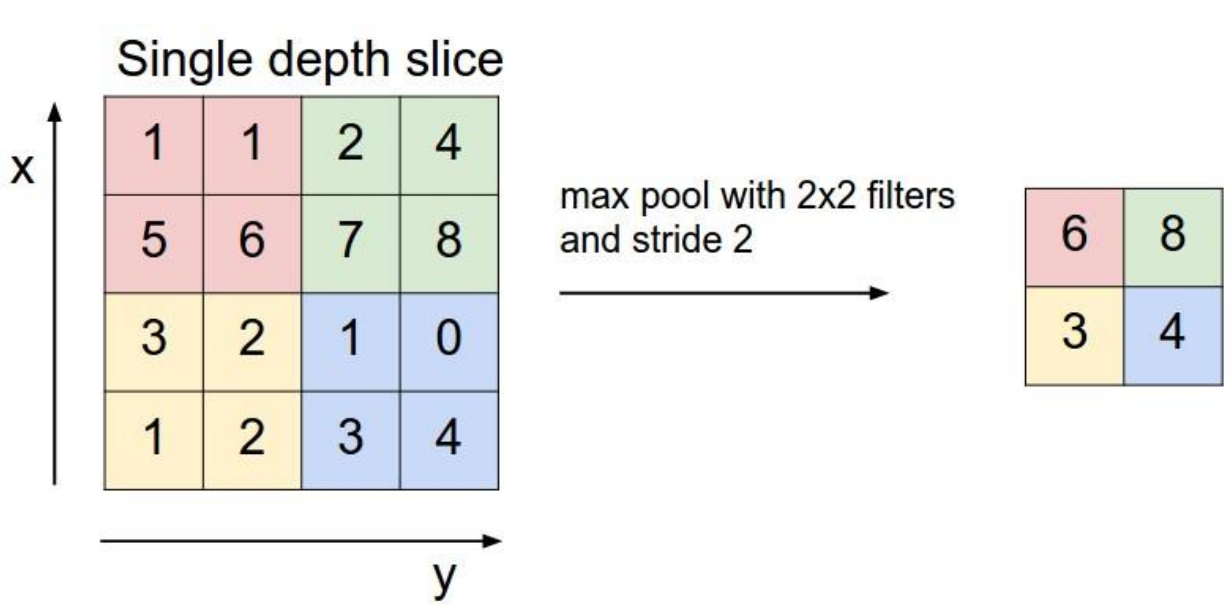
- For a different neuron: Filter parameters W_1, bias b_1
  - $V[0,0,1] = np.sum(X[:5,:5,:] * W_1) + b_1$
  - $V[1,0,1] = np.sum(X[2:7,:5,:] * W_1) + b_1$
  - $V[2,0,1] = np.sum(X[4:9,:5,:] * W_1) + b_1$
  - $V[3,0,1] = np.sum(X[6:11,:5,:] * W_1) + b_1$
  - Question: how do we calculate $V[0,1,1]$ and $V[2,3,1]$?

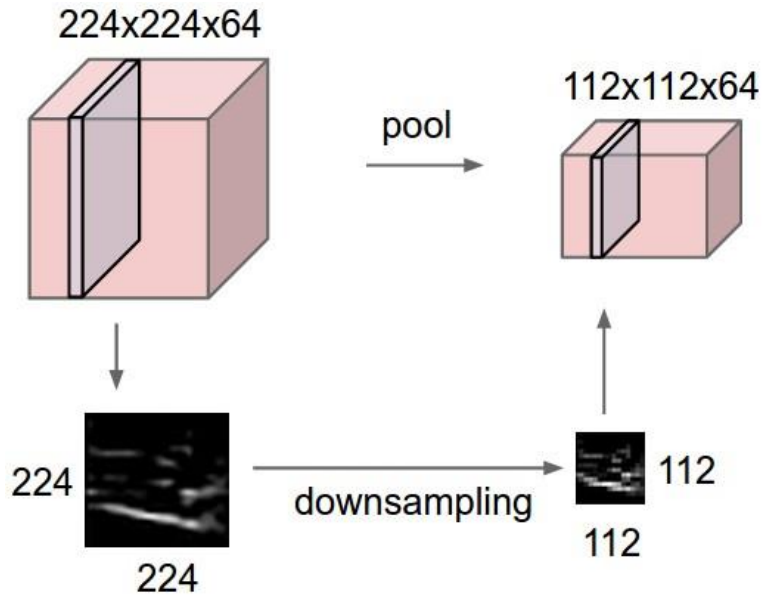
Lecture plan

- Introduction to convolutional neural networks
  - **Pooling and downsampling**

Pooling layer

- **Pooling** reduces the spatial size of the input: Insert a pooling layer between convolution layers

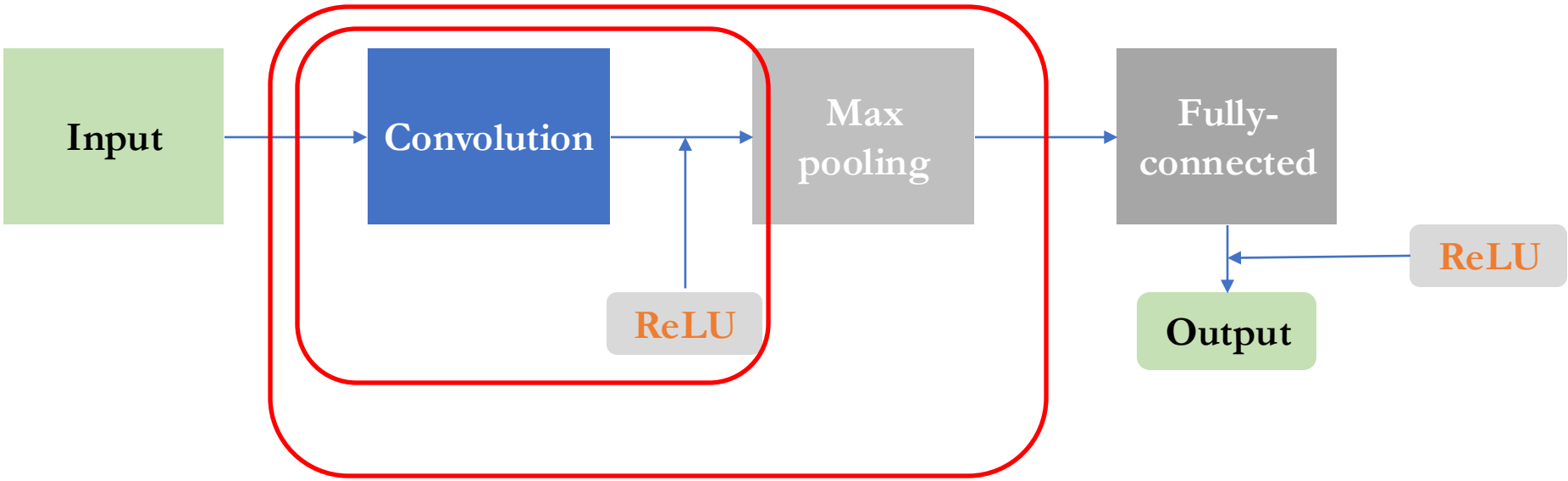




Pooling layer

- Input: An image of size (W_1, H_1, D_1)
- Pooling layer
  - Filter size F
  - Stride size S
- Output size: (W_2, H_2, D_2)
  - $W_2 = \frac{W_1 - F}{S} + 1$
  - $H_2 = \frac{H_1 - F}{S} + 1$
  - $D_2 = D_1$
- **Previous example: $F = 2$ and $S = 2$**

Summary



- A deep CNN architecture involves multiple convolution and pooling layers

Residual networks

- Residual networks are a popular and successful approach to building very deep networks
- Main idea: Let's say we map the information from layer $i-1$ to layer i

$$z^{(i)} = f\left(z^{(i)}\right) = g\left(W^{(i)} z^{(i-1)}\right)$$

- The key idea of residual networks is that a layer should perturb the representation from the previous layer rather than replace it entirely

$$z^{(i)} = g\left(z^{(i-1)} + f\left(z^{(i)}\right)\right)$$

Applications

- Convolutional neural networks are most often used for computer vision
  - The AlexNet deep learning system in the 2012 ImageNet competition revolutionized the field
  - The ImageNet competition was a supervised learning task with 1,200,000 images in 1,000 different categories, and systems were evaluated on the "top-5" score—how often the correct category appears in the top five predictions
- Convolutional neural networks are also useful for text classification
  - Given a sentence, predict a certain label based on extracting meaning from the sentence