CSE 4553
Machine Learning
Lecture 8: Convolutional Neural Network (CNN)

Winter 2022

Hasan Mahmud | hasan@iut-dhaka.edu

Deep Learning

- Deep learning is supervised learning technique that contains a stack of hidden layers in addition to input and output layers of ANN.
- Deep Neural Network are capable of extracting deep features out of the data; hence the name deep learning.
- Some deep learning algorithms are:
  - Convolution Neural Network (CNN)
  - Recurrent Neural Network (RNN)
  - Long-Short Term Memory (LSTM)

ANN in handwritten digit recognition

- ANN can handle these variations of input of the hand written digit ‘9’

ANN in handwritten digit recognition

0	0	0	0	0	0	0
0	87	240	210	24	0	0
0	13	0	101	195	0	0
0	35	167	99	210	0	0
0	145	230	240	201	189	140
0	0	102	67	17	13	0
0	0	0	0	0	0	0

7 by 7 grid

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 87 \\ 240 \\ 210 \\ 24 \\ 0 \\ \\ \cdots \\ \\ \\ 0 \end{bmatrix}$$

49*49 = 2401 49*10 = 490

Problems with ANN

Image size = 1920 x 1080 X 3

First layer neurons = 1920 x 1080 X 3 ~ 6 million

Hidden layer neurons = Let's say you keep it ~ 4 million

Weights between input and hidden layer = 6 mil * 4 mil
= 24 million

- Too much computation
- Treats local pixels same as pixels far apart
- Sensitive to location of an object in an image

Introduction to CNN

- In 2012, neural nets grew to prominence as Alex Krizhevsky used them to win that year's ImageNet competition (basically, the annual Olympics of computer vision), dropping the classification error record from 26% to 15%, an astounding improvement at the time.
- Deep learning-based applications:
  - Facebook uses neural nets for their automatic tagging algorithms
  - Google for their photo search
  - Amazon for their product recommendations
  - Pinterest for their home feed personalization
  - Instagram for their search infrastructure

Introduction to CNN…

- When we see an image or just when we look at the world around us, most of the time we are able to immediately characterize the scene and give each object a label, all without even consciously noticing.

- These skills of being able to quickly recognize patterns, generalize from prior knowledge, and adapt to different image environments are ones that we do not share with our fellow machines.

Introduction to CNN…

- Image classification is the task of taking an input image and outputting a class (a cat, dog, etc) or a probability of classes that best describes the image.
- The way that we see images around us is different than the way machine sees it.

What we see

0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

What a computer sees

Convolution Neural Network (CNN)

- It is a multilayered feed-forward neural network with a special architecture to detect complex features in data.
- CNNs are able to classify images by detecting features, similar to how the human brain detects features to identify objects.
- Images are high-dimensional vectors. It would take a huge amount of parameters to characterize the network. To address this problem, bionic convolutional neural networks are proposed to reduced the number of parameters and adapt the network architecture specifically to vision tasks.

Conceptual understanding of CNN

Conceptual understanding of CNN

Conceptual understanding of CNN

CNN

- A CNN unit contains the following layers:
  1. Convolutional layer containing a set of filters
  2. Pooling layer
  3. Non-linearity
- Deep CNN: a stack of multiple CNN units
  - Inspired by the human visual system (V1, V2, V3 ….)

Hubel and Weisel, 1968
Structure of V1
“simple” cells: edge detectors

Typical steps of CNN

- **Step 1: Convolution**
- **Step 1b: ReLU Layer**
- **Step 2: Pooling**
- **Step 3: Flattening**
- **Step 4: Full Connection**

Convolution

- Convolution operation

$$s[t]=(x\star w)[t]=\sum_{a=-\infty}^{a=\infty} x[a]w[a+t]$$

Feature map Input kernel

Convolution…

$$-1+1+1-1-1-1-1+1+1 = -1 \rightarrow -1/9 = -0.11$$

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11		

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	**1**	

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	1	-0.11

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	1	-0.11
-0.55		

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	1	-0.11
-0.55	0.11	

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	1	-0.11
-0.55	0.11	-0.33

Convolution…

-1	**1**	**1**	**1**	-1
-1	**1**	-1	**1**	-1
-1	**1**	**1**	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	-1	**1**	-1
-1	-1	**1**	-1	-1
-1	**1**	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	**1**	-0.11
-0.55	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33

Feature Map
OR Activation Map

Convolution…

Convolution…

Location invariant: It can detect eyes in any location of the image

Convolution…

Convolution…

Convolution…

Convolution…

Convolution…

- Kernel: A kernel is a small matrix whose contents are based upon the operations to be performed. The kernel is the filter which is convolved with the original image to detect important patterns.

Original	*Gaussian Blur*	*Sharpen*	*Edge Detection*
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$

- Stride: Stride defines by what step does to kernel move, for example stride of 1 makes kernel slide by one row/column at a time and stride of 2 moves kernel by 2 rows/columns.

Convolution…

- let's take an image of 5 by 5 matrix with 3 channels(RGB) , a feature detector of 3 by 3 with 3 channels (RGB) and scan the feature detector over the image by 1 stride.
- **Dimension of the feature map as a function of the input image size(W), feature detector size(F), Stride(S) and Zero Padding on image(P)** is: $(W - F + 2P)/S + 1$

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

Image

1	0	1
0	1	0
1	1	0

Feature detector

- Input image size W in our case is 5.
- Feature detector or receptive field size is F, which in our case is 3
- Stride (S) is 1, and the amount of zero padding used (P) on the image is 0.
- so, our feature map dimension will (5–3 +0)/1 + 1=3.
- **so feature map will a 3*3 matrix with three channels(RGB).**

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

(1*1+0*1+1*0+0*0+1*0+1*0+1*0+1*1+0*1) =2		

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

2	(1*1+0*0+1*0+1*0+1*0+1*0+1*1+1*1+0*1) =3	

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

2	3	(1*0+0*0+1*1+0*0+1*0+0*1+1*1+1*1+0*1) =3

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

2	3	3
(1*0+0*0+1*0+0*0+1*1+0*1+1*0+1*0+1*1)=2		

1	1	0	0	1
0	0	0	0	1
0	1	1	1	1
0	0	1	0	1
1	0	1	1	1

2	3	3
2	(1*0+0*0+1*0+0*1+1*1+0*1+1*0+1*1+0*0)=2	

Continuing to scan through input matrix and the final feature map will be

2	3	3
2	2	3
2	4	4

Padding

0	0	0	0	0	0	0
0	1	1	0	0	1	0
0	0	0	0	0	1	0
0	0	1	1	1	1	0
0	0	0	1	0	1	0
0	1	0	1	1	1	0
0	0	0	0	0	0	0

Rectified Linear Unit (ReLU)

$R(z) = \max(0, z)$

*

Loopy pattern filter

1	1	1
1	-1	1
1	1	1

→

-0.11	1	-0.11
-0.55	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33

→

0	1	0
0	0.11	0
0	0.33	0
0	0	0
0	0	0

Pooling

- Reduces dimensions and computations
- Reduce overfitting as there are less parameters
- Model is tolerant towards spatial/translations, distortion
- Types of pooling
  - Mean pooling
  - Max pooling
  - Average Pooling
  - Sum pooling

Pooling

1920 x 1080
eye
1920 x 1080
nose
1920 x 1080
ears
1920 x 1080
hands
1920 x 1080
legs
head
1920 x 1080
flatten
body
1920 x 1080
flatten
Is this Koala?

Pooling…

5	1	3	4
8	2	9	2
1	3	0	1
2	2	2	0

8	**9**
3	2

2 by 2 filter with stride = 2

Pooling…

0	**1**	0
0	0.11	0
0	0.33	0
0	0	0
0	0	0

1	**1**
0.33	0.33
0.33	0.33
0	0

2 by 2 filter with stride = 1

Position invariant property using pooling

-1	1	1	1	-1
-1	1	-1	1	-1
-1	1	1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1

*

Loopy pattern filter

1	1	1
1	-1	1
1	1	1

→

-0.11	1	-0.11
-0.55	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33

→

0	1	0
0	0.11	0
0	0.33	0
0	0	0
0	0	0

Max pooling

→

1	1
0.33	0.33
0.33	0.33
0	0

Shifted 9 at different position

1	1	1	-1	-1
1	-1	1	-1	-1
1	1	1	-1	-1
-1	-1	1	-1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1
1	-1	-1	-1	-1

*

Loopy pattern filter

1	1	1
1	-1	1
1	1	1

→

1	-0.11	-0.11
0.11	-0.33	0.33
0.33	-0.33	-0.33
-0.11	-0.55	-0.33
-0.55	-0.33	-0.55

→

1	0	0
0.11	0	0.33
0.33	0	0
0	0	0
0	0	0

Max pooling

→

1	0.33
0.33	0.33
0.33	0
0	0

2. Pooling

- enables CNN to detect features diff. lighting and angles orientation of the images.
- Spatial Invariant concept
- applies Non-Linearity down sampling on the activation map.

Max pooling

Avg - Pooling

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	61

2x2

100	184
12	45

2x2

36	80
12	15

Spatial invariant

Flattening and fully connected layer

Pooled Feature Map

Flattening

Pooling Layer

Flattening

Input layer of a future ANN

3) Flattening

Deep Fake

1	1	0
4	2	1
0	2	1

Flattening →

1	1	0	4	2	1	0	2	1

gradient descent

method

Parameter fitting

4) Fully connected

$$\sigma(z) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

ARIMA

0,1

Soft Max function

Complete structure of CNN

Calculations of the CNN parameters

CONV 1

Input Size (W_1 x H_1 x D_1) = 28 x 28 x 1

- Requires four hyperparameter:
  - Number of kernels, k = 16
  - Spatial extend of each one, F = 5
  - Stride Size, S = 1
  - Amount of zero padding, P = 2
- Outputting volume of W_2 x H_2 x D_2
  - W_2 = (28 – 5 + 2 (2)) / 1 + 1 = 28
  - H_2 = (28 – 5 + 2 (2)) / 1 + 1 = 28
  - D_2 = k

Output of Conv 1 (W_2 x H_2 x D_2) = 28 x 28 x 16

POOL 1

Input Size (W_2 x H_2 x D_2) = 28 x 28 x 16

- Requires two hyperparameter:
  - Spatial extend of each one, F = 2
  - Stride Size, S = 2
- Outputting volume of W_3 x H_3 x D_2
  - W_3 = (28 – 2) / 2 + 1 = 14
  - H_3 = (28 – 2) / 2 + 1 = 14

Output of Pool 1 (W_3 x H_3 x D_2) = 14 x 14 x 16

CONV 2

Input Size (W_3 x H_3 x D_2) = 14 x 14 x 16

- Requires four hyperparameter:
  - Number of kernels, k = 32
  - Spatial extend of each one, F = 5
  - Stride Size, S = 1
  - Amount of zero padding, P = 2
- Outputting volume of W_4 x H_4 x D_3
  - W_4 = (14 – 5 + 2 (2)) / 1 + 1 = 14
  - H_4 = (14 – 5 + 2 (2)) / 1 + 1 = 14
  - D_3 = k

Output of Conv 2 (W_4 x H_4 x D_3) = 14 x 14 x 32

POOL 2

Input Size (W_4 x H_4 x D_3) = 14 x 14 x 32

- Requires two hyperparameter:
  - Spatial extend of each one, F = 2
  - Stride Size, S = 2
- Outputting volume of W_5 x H_5 x D_3
  - W_5 = (14 – 2) / 2 + 1 = 7
  - H_5 = (14 – 2) / 2 + 1 = 7

Output of Pool 2 (W_5 x H_5 x D_3) = 7 x 7 x 32

FC Layer

Input Size (W_5 x H_5 x D_3) = 7 x 7 x 32

Output Size (Number of Classes) = 10

Convolution

- Connections sparsity reduces overfitting
- Conv + Pooling gives location invariant feature detection
- Parameter sharing

ReLU

- Introduces nonlinearity
- Speeds up training, faster to compute

Pooling

- Reduces dimensions and computation
- Reduces overfitting
- Makes the model tolerant towards small distortion and variations