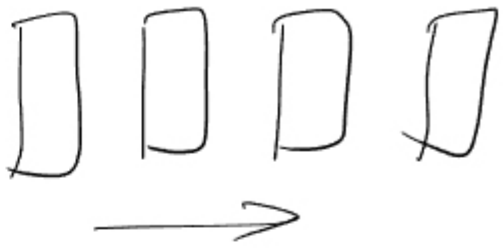
Introduction

05 July 2022 09:57

Performance of NN

↳ how the speed the training



Optimizers ✓

→ Weigh init

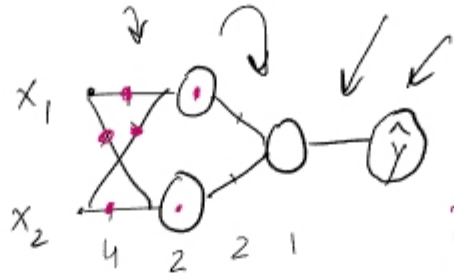
→ Batch Norm

→ Activation function

Role of Optimizer

05 July 2022 10:01

cgpa	iq	placed
8	80	1
9	90	1
7	70	0



9 parameters

2 weights

Optimization

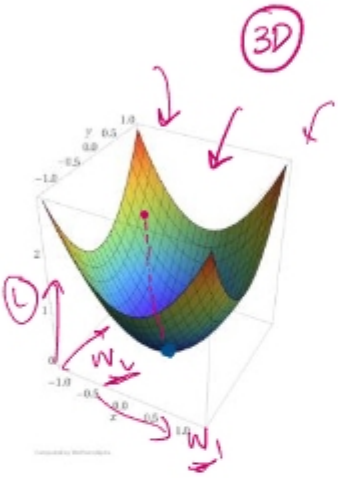
2 min

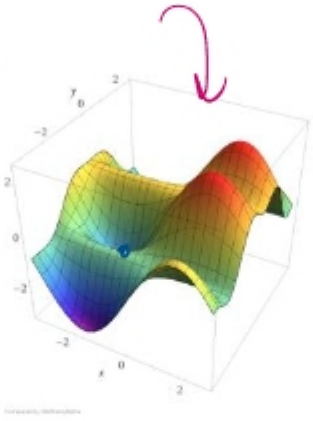
y $\hat{y}$ (w, b)

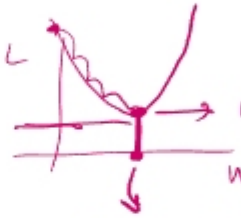
C/L()

9 para

(L) 3D L(-,-,-,-,-) 10D







Gradient Descent

$$w_n = w_0 - \eta \frac{\partial L}{\partial w}$$

Types of Optimizers

05 July 2022 10:02

- Batch GD
- Stochastic GD
- Mini batch GD

(10) ↔ (10)

epochs = 10

$$W_n = W_0 - \eta \frac{\partial L}{\partial W}$$

500 rows → pred

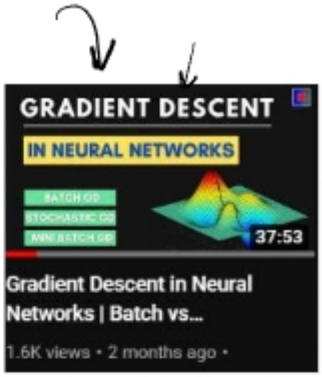
↓

loss → weight update

10 × 500 → 5000

batch size = 100

5 batch → 10 × 5 times



Gradient Descent in Neural Networks | Batch vs...

1.6K views • 2 months ago •

Challenges

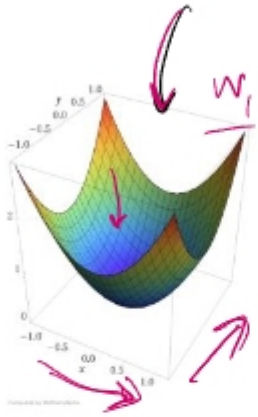
05 July 2022 10:02

1) learning rate

$$w_n = w_0 - \eta \frac{\partial L}{\partial w_0}$$

2) learning rate scheduling → pre define

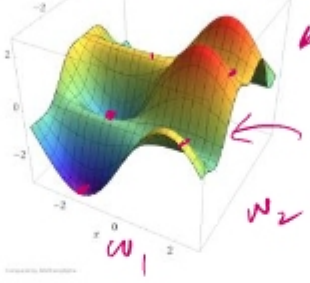
3)



(1) same both the directions w_1 w_2

(9) weights 9 sep eqns and lies

10-D → 10 directions

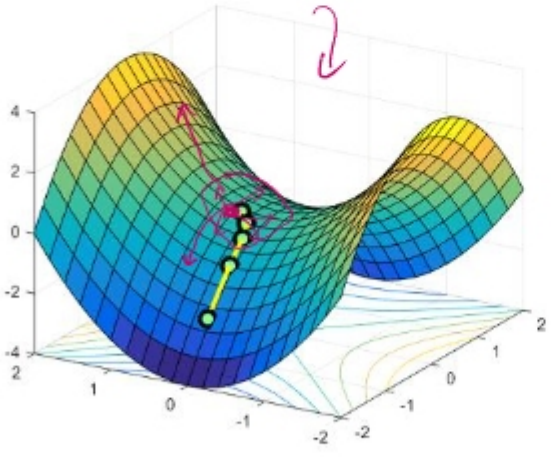


complex minima

sub optimum

4) local minima

5) saddle point →

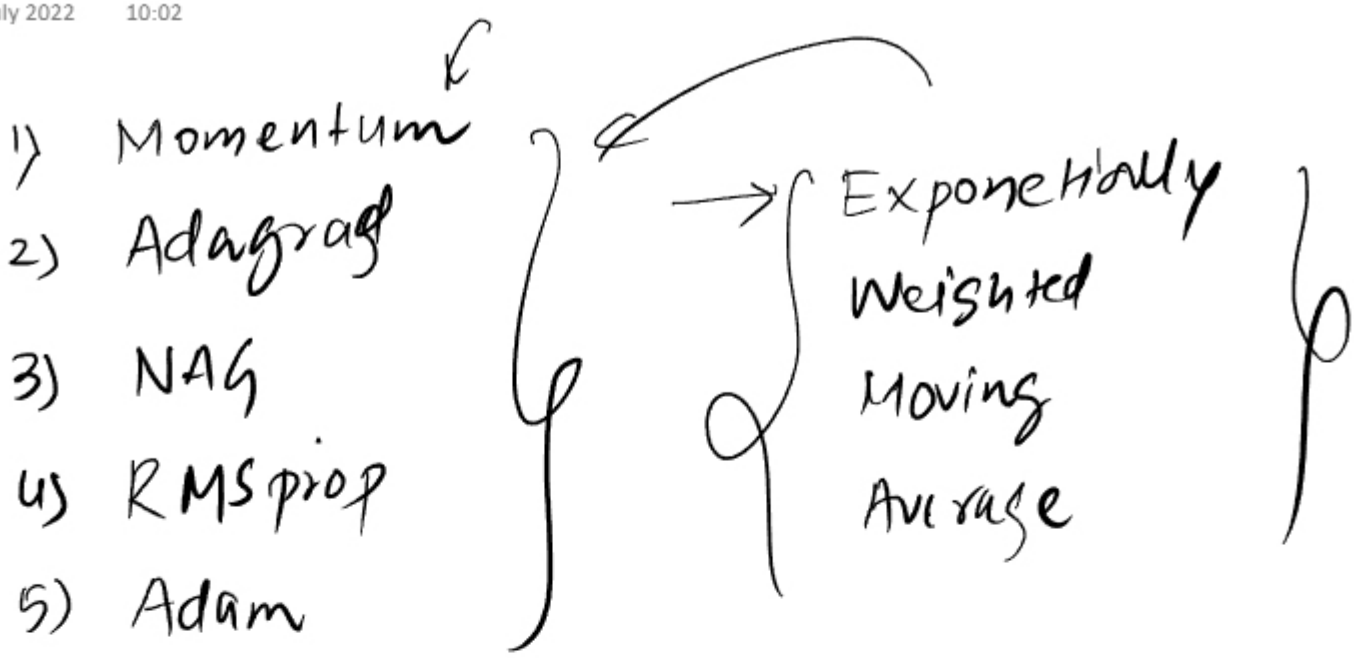


$$\frac{\partial L}{\partial w} = 0$$

$$w_n = w_0 - \eta \frac{\partial L}{\partial w}$$

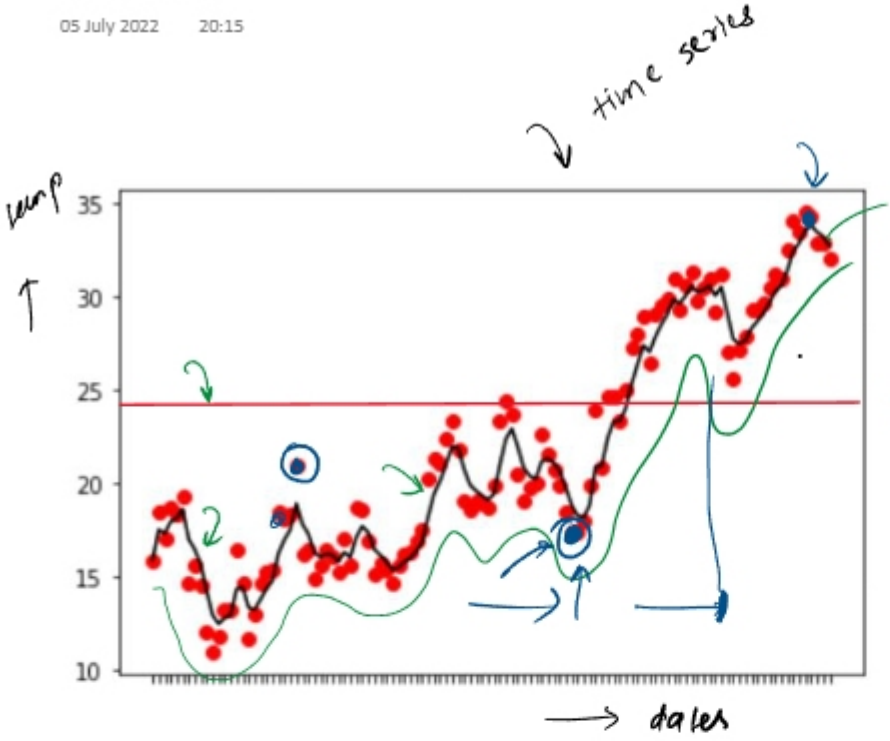
What next?

05 July 2022 10:02



The What

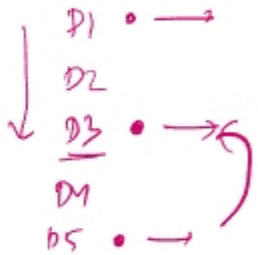
05 July 2022 20:15



→ time series
→ financial
→ signal process
→ Deep learning
optimizers

average

EWMA



Mathematical Formulation

05 July 2022 20:16

$$V_t = \beta V_{t-1} + (1-\beta)\theta_t$$

$\beta \rightarrow 0 < \beta < 1$

$V_0 = 0$ $\quad V_0 = \theta_0$

$V_1 = 0.9 \times V_0 + 0.1 \times 13$

$= 1.3$

$V_2 = 0.9 \times 1.3 + 0.1 \times 17 =$

$\beta = 0.9$

index	temp (θ)
D1	25
D2	13
D3	17
D4	31
→ D5	43

$\beta = 0.5$

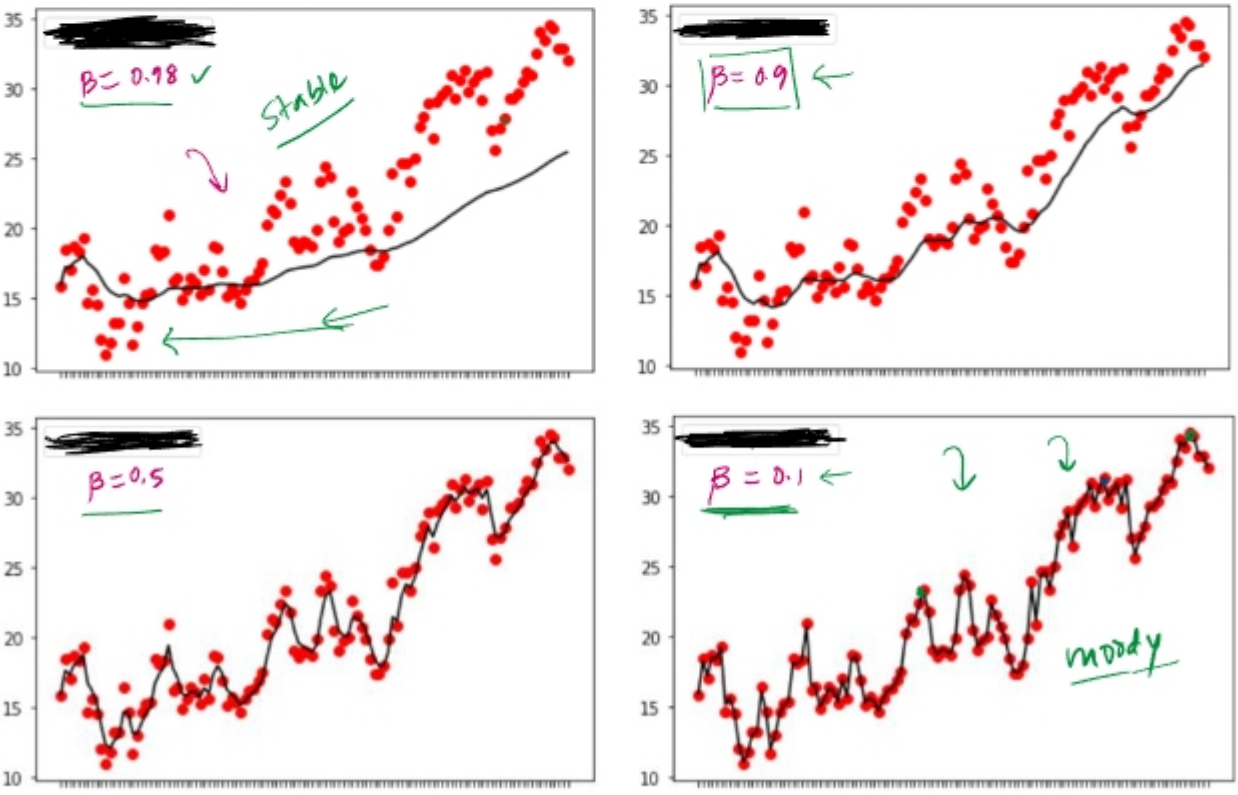
$\frac{1}{1-0.5} = \frac{1}{0.5}$

2 days

$\beta \rightarrow \boxed{\frac{1}{(1-\beta)}} \rightarrow$ $\beta = 0.9$

$\frac{1}{1-0.9} = \frac{1}{0.1} = 10$

Effect of β



Mathematical Intuition

$$V_t = \beta V_{t-1} + (1-\beta)\theta_t$$

$$V_4 = \beta V_3 + (1-\beta)\theta_4$$

$$V_4 = \beta^3(1-\beta)\theta_1 + \beta^2(1-\beta)\theta_2 + \beta(1-\beta)\theta_3 + (1-\beta)\theta_4$$

$$V_t = \beta V_{t-1} + (1-\beta)\theta_t$$

$V_0 = 0$

$V_1 = (1-\beta)\theta_1$

$V_2 = \beta V_1 + (1-\beta)\theta_2$

$= \beta(1-\beta)\theta_1 + (1-\beta)\theta_2$

$V_3 = \beta V_2 + (1-\beta)\theta_3$

$= \beta^2(1-\beta)\theta_1 + \beta(1-\beta)\theta_2 + (1-\beta)\theta_3$

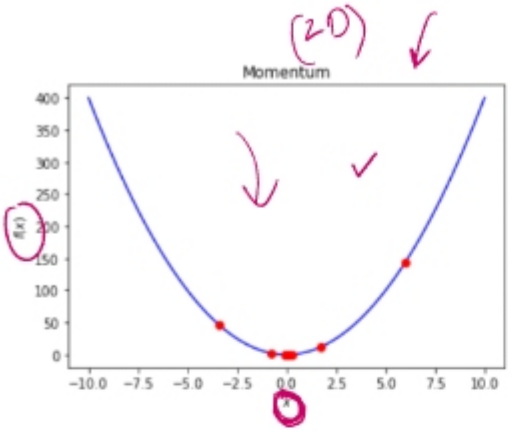
$V_4 = \beta^3(1-\beta)\theta_1 + \beta^2(1-\beta)\theta_2 + \beta(1-\beta)\theta_3 + (1-\beta)\theta_4$

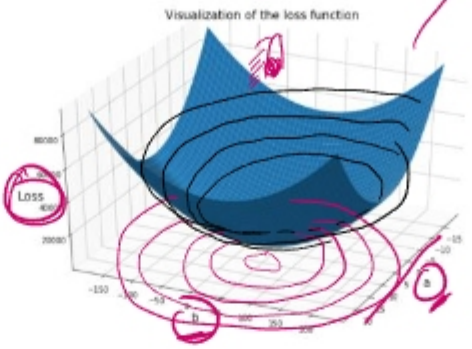
$= (1-\beta)[\beta^3\theta_1 + \beta^2\theta_2 + \beta\theta_3 + \theta_4]$

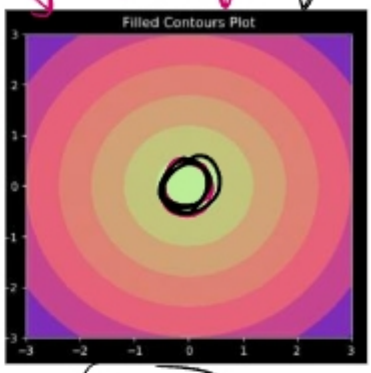
$\beta^3 < \beta^2 < \beta$ $\quad 0 < \beta < 1$

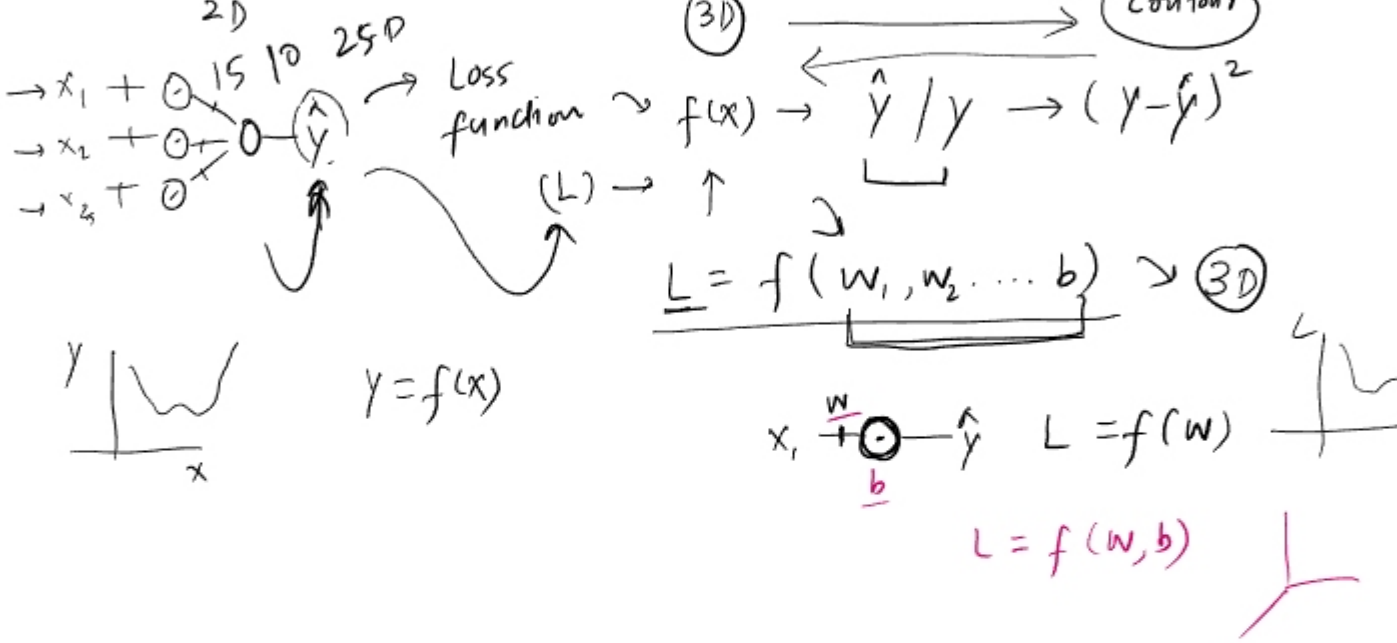
Understanding Graphs

20 July 2022 12:03



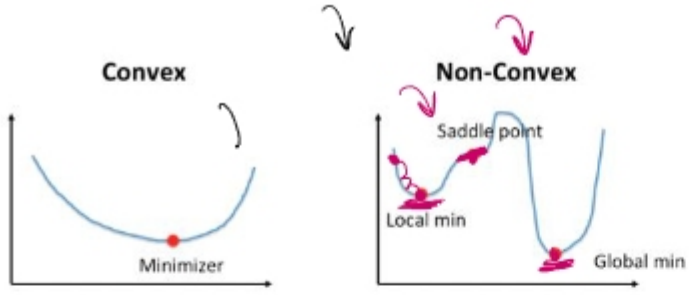




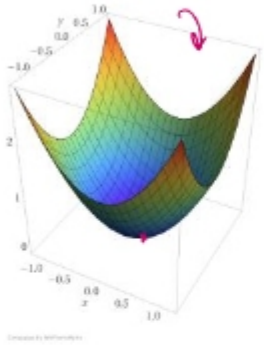


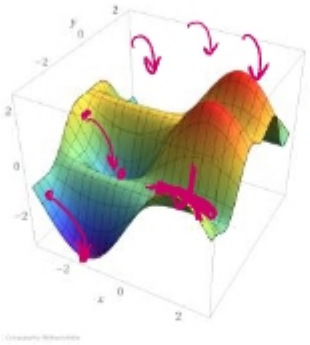
Convex Vs Non-Convex Optimization

20 July 2022 13:06

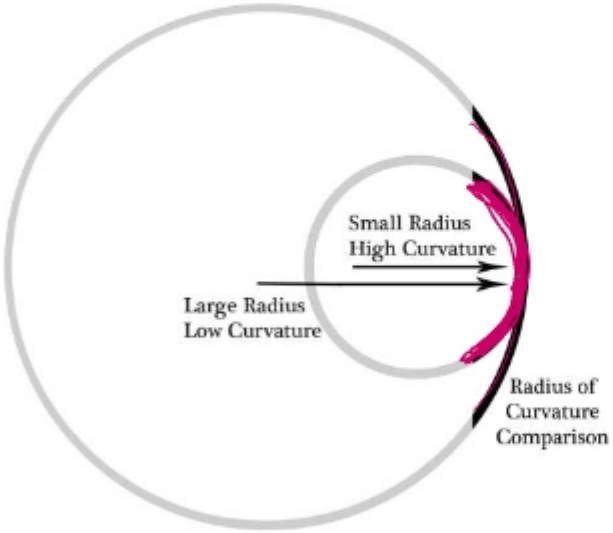


L (w)



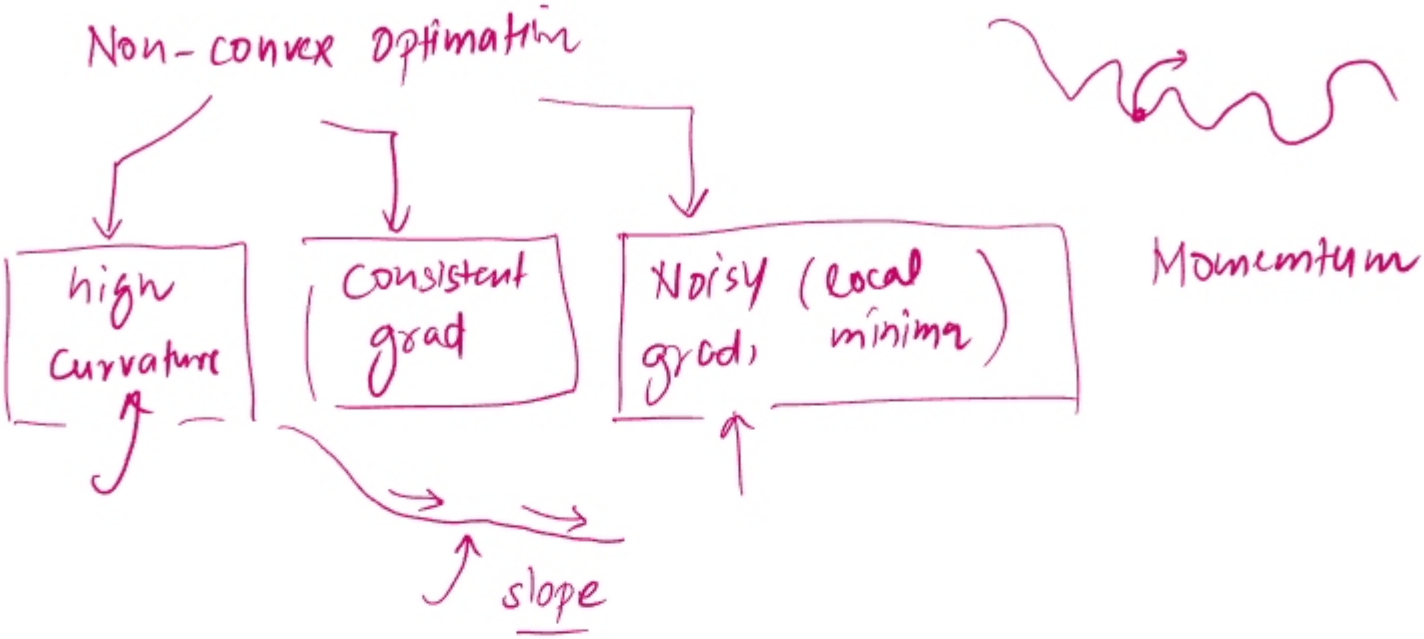


1) local minima
2) Saddle point
3) High curvature



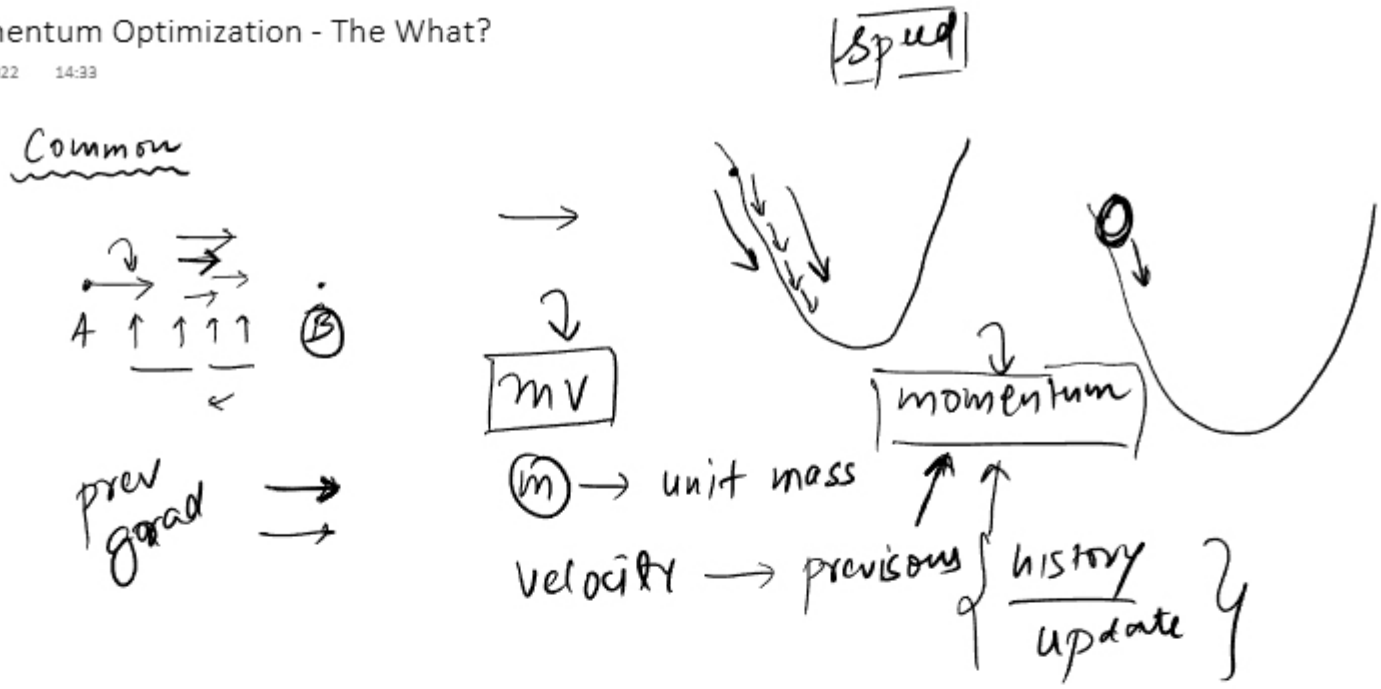
Momentum Optimization - The Why?

20 July 2022 14:28



Momentum Optimization - The What?

20 July 2022 14:33



Momentum Optimization - Mathematics(The How?)

20 July 2022 14:56

$$w_{t+1} = w_t - \eta \nabla w_t$$

v → velocity

$$w_{t+1} = w_t - v_t$$

$$v_t = \beta * v_{t-1} + \eta \nabla w_t$$

acceleration

momentum

0.9

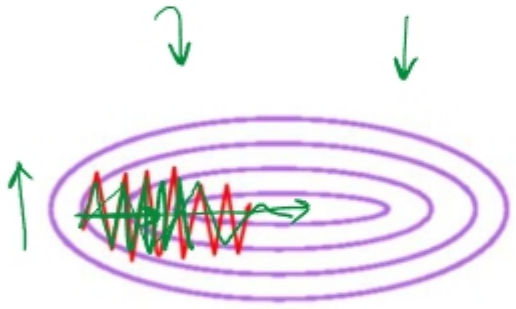
$0 < \beta < 1$

history of past velo- use

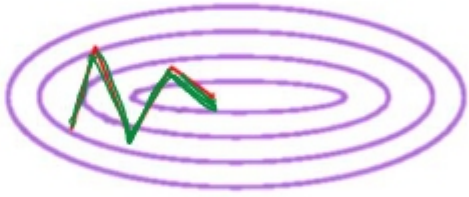
prev velocity

v_{t-2}

v_{t-3}



SGD



SGD momentum

faster convergence

Effect of beta

20 July 2022 15:09

$$w_{t+1} = w_t - \eta \nabla w_t$$

$$w_{t+1} = w_t - v_t$$

$$v_t = \beta v_{t-1} + \eta \nabla w_t$$

SGD

$\beta = 0$ momk → SGD

$\beta = 0$

decaying factor

$\frac{1}{1-0.9} = \frac{1}{0.1} = 10$

$\frac{1}{1-\beta}$

0.9

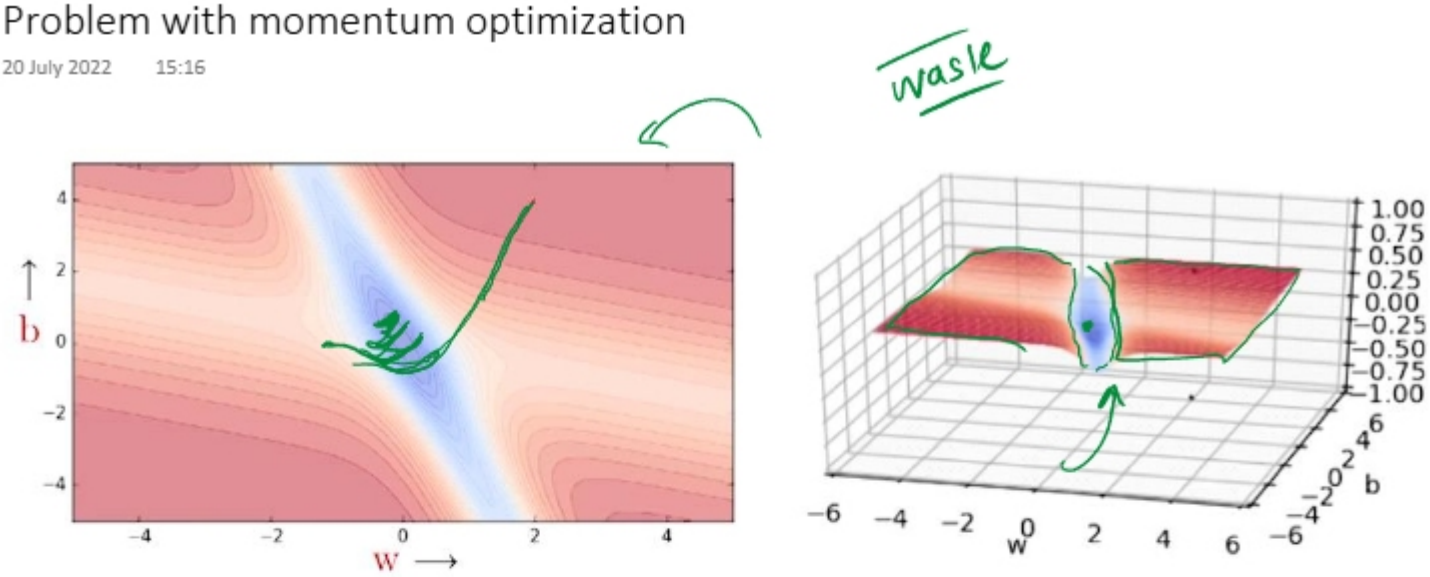
current avg past 10 veloci

up1 up2 . - up3

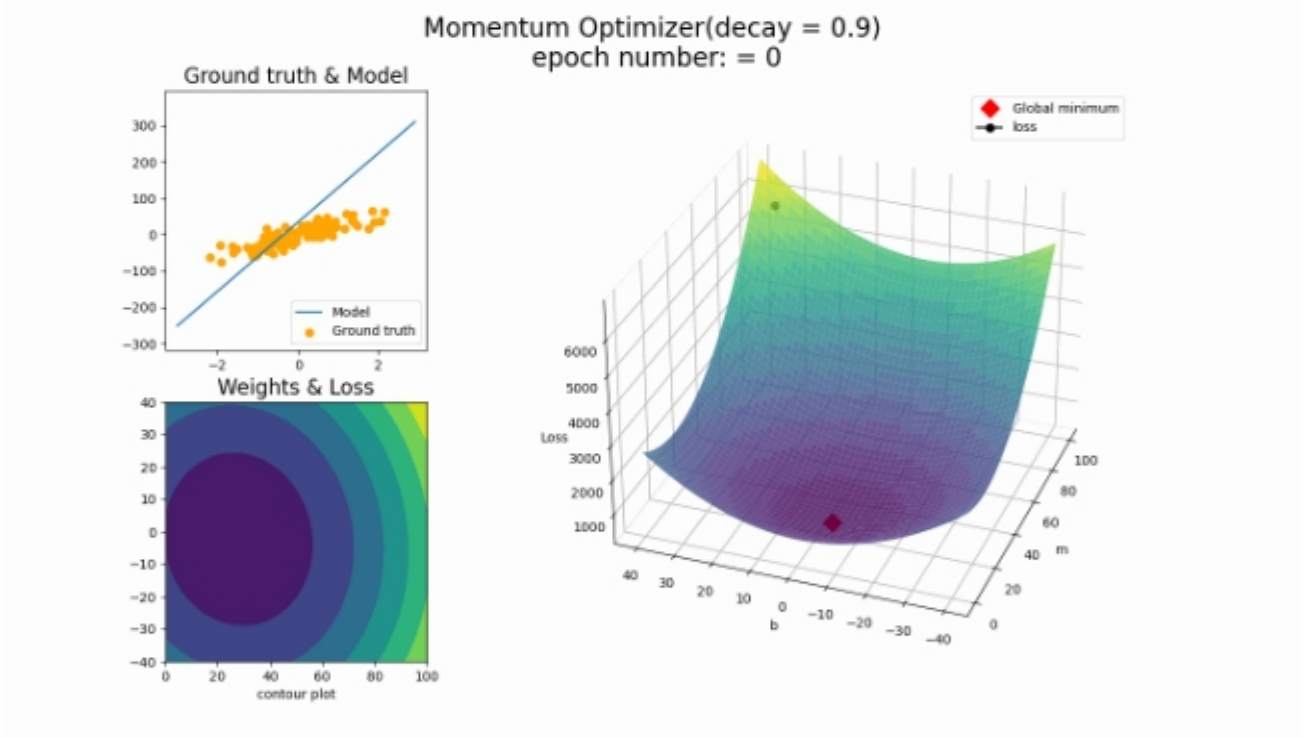
$\beta = 1$

Problem with momentum optimization

20 July 2022 15:16



decay 0.9



Demo

24 July 2022 13:17

Mathematical Intuition

24 July 2022 13:17

Momentum

$$w_{t+1} = w_t - v_t$$

where

$$v_t = \beta v_{t-1} + \eta \nabla w_t$$

$\beta \rightarrow$ decay factor

$$w_{t+1} = w_t - (\beta v_{t-1} + \eta \nabla w_t)$$

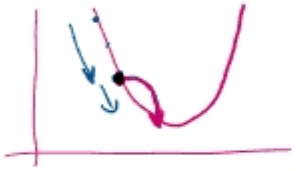
past velocity; gradient at that point

$$w_{t+1} = w_t - \eta \nabla w_t$$

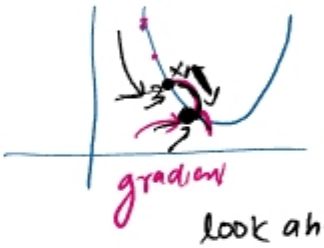
NAG

Nesterov Accelarated Gradient

better



history of velocity + gradient at that point



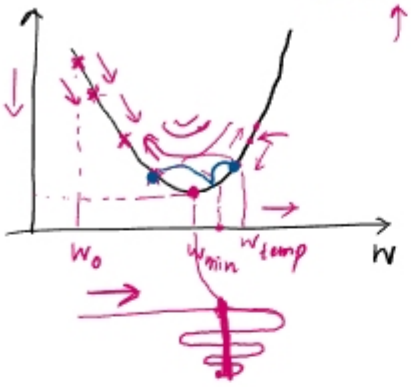
look ahead

$$\begin{cases} w_{la} = w_t - \beta v_{t-1} \\ v_t = \beta v_{t-1} + \eta \nabla w_{la} \\ w_{t+1} = w_t - v_t \end{cases} \rightarrow v_t = (w_t - w_{la}) + \eta \nabla w_{la}$$

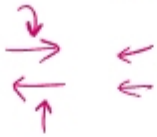
Geometric Intuition

24 July 2022 13:17

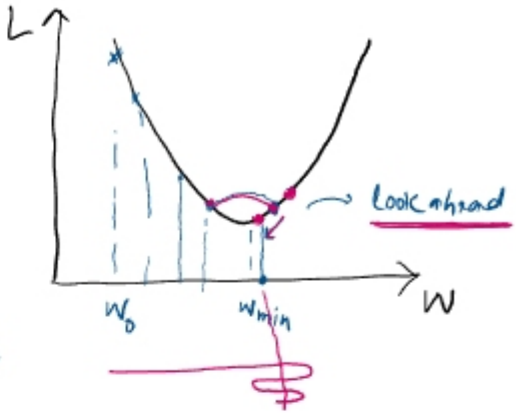
momentum ↑ gradient



-ve of grad / slope

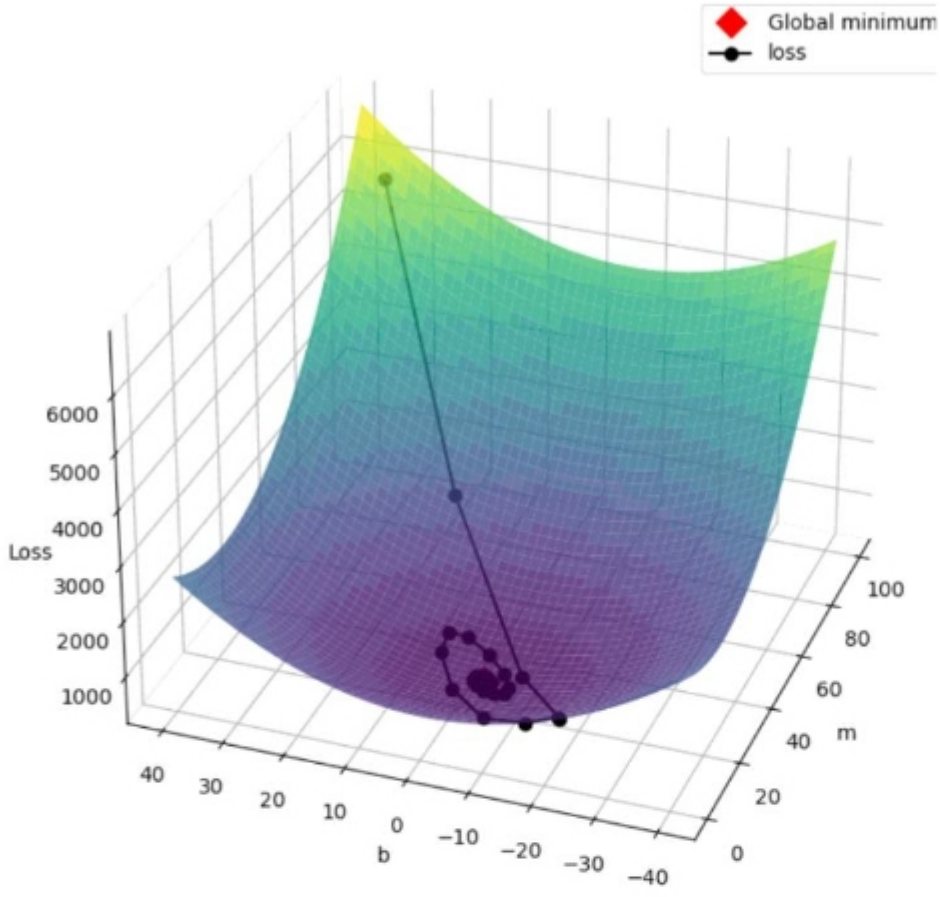


moment



Disadvantage

24 July 2022 13:18



Disadvantage

NAG → oscillation → dampen

local minima

Keras Code

24 July 2022 13:18

```
tf.keras.optimizers.SGD(
    learning_rate=0.01, momentum=0.0, nesterov=False, name="SGD", **kwargs
)
```

SGD ✓

Momentum ✓
momentum = 0.9
nesterov = False

NAG ✓
momentum = 0.9
nesterov = True

AdaGrad Intro

02 August 2022 18:45

Adaptive Gradient → learning rate fix X
change
adapt →

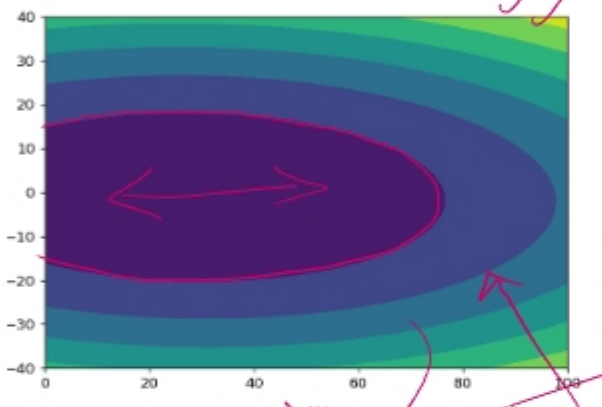
cgpa | salary | bank loan
0-10 ↓ 0 – Lakh
↓
Normalize

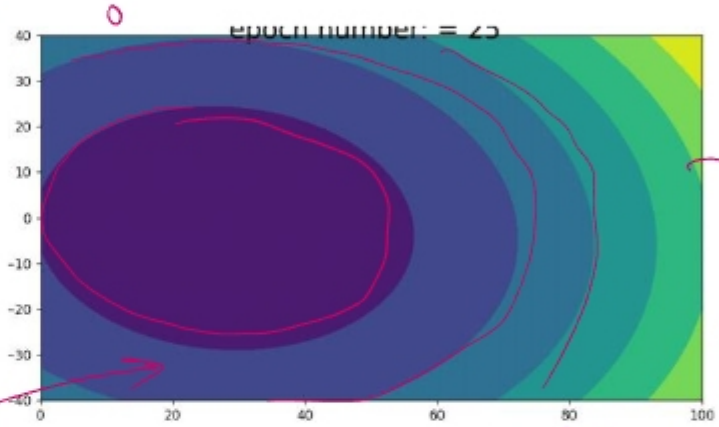
→ input features → scale
↳ different

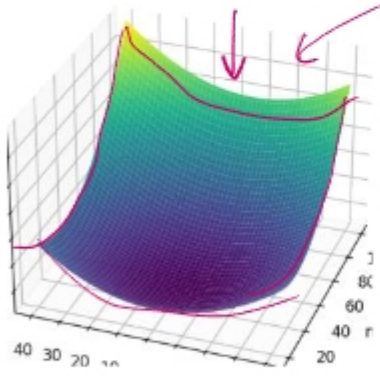
⇒ features → sparse → 0 (most)
↳ Adagrad

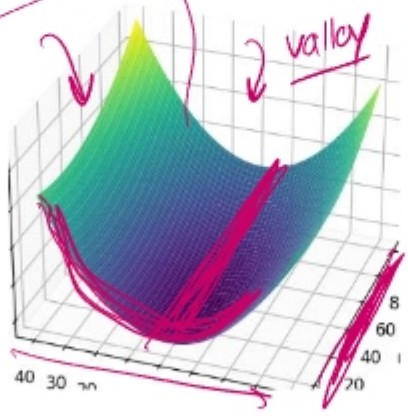
iq | cgpa | from iit | package
0 → 0,0,0,
1 →

Elongated bowl problem









valley

Loss → w,b

Bgd → momentum → NAG

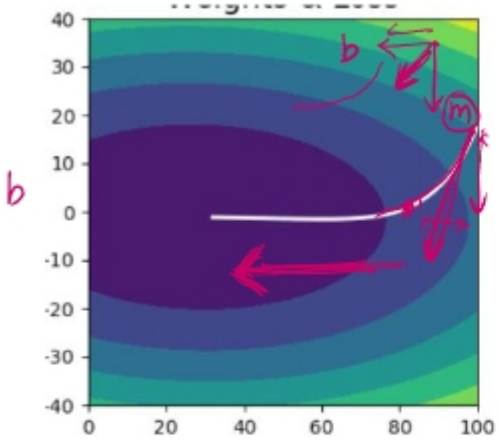
Adagrad

elongated

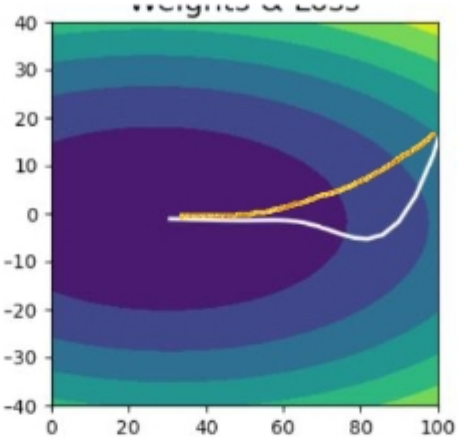
slope → laxis – along
2nd axis constant

How optimizers behave(Why?)

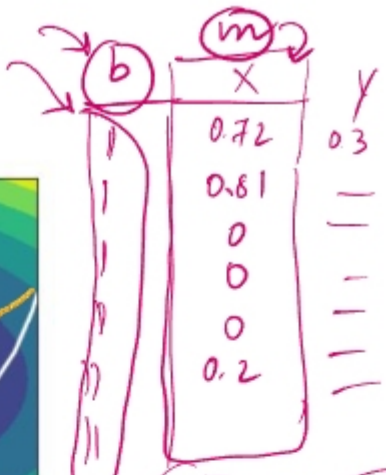
02 August 2022 18:44



Batch GD

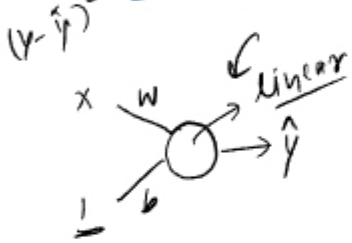


Momentum



less movement

normal movement



BGD → sparse

for i in epochs:

$$w = w - \eta \frac{\partial L}{\partial w}$$

$$b = b - \eta \frac{\partial L}{\partial b}$$

→ big update in every epoch

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w}$$

$$= -2(y-\hat{y})\,x \quad \text{add}$$

$$\frac{\partial L}{\partial b} = -2(y-\hat{y}) * 1 \quad \text{bias num}$$

100 rows — 0 — X sparse

small number

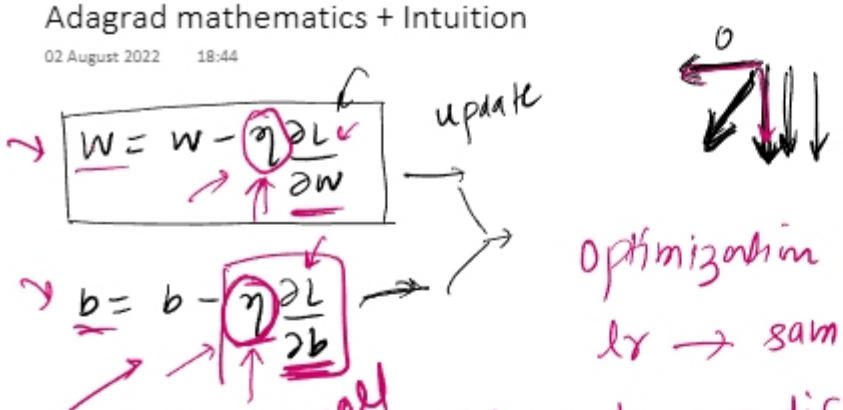
small in every epoch

sparse col → small

non sparse → normal / big

Adagrad mathematics + Intuition

02 August 2022 18:44



Adagrad Demo

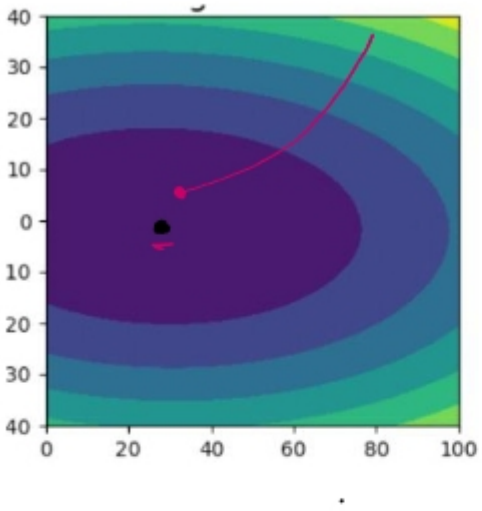
02 August 2022 18:44

Keras Implementation

02 August 2022 18:44

Disadvantage

02 August 2022 18:43



NN → use X ↘ linear regress

η → small + decrease

$\sqrt{v_t}$

v_t

past updates ↳ gradient epochs

update small

global minima ↘ RMSprop ↘ Adam

The Why?

03 August 2022 14:03

RMSprop → Root Mean square Prop

↑ ↳ improvement

↓

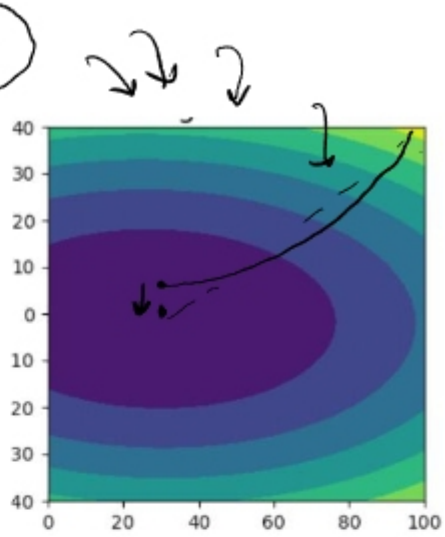
Adagrad ← sparse data

BGD → momentum

Disadvantage → learning rate ↓

x_1	x_2	y
0		
0		
0		
1		
0		

Update → small



Mathematical Formulation

03 August 2022 14:04

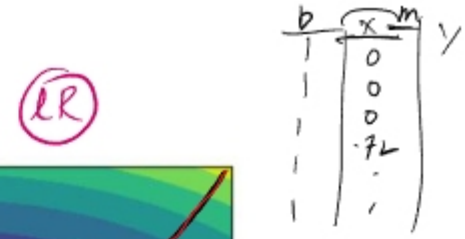
epochs

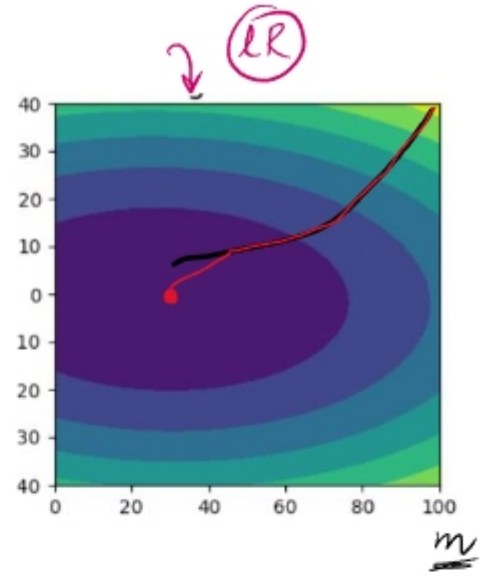
Adagrad

$v_t = v_{t-1} + (\nabla w_t)^2$ — gradients

$w_{t+1} = w_t - \frac{\eta}{\sqrt{v_t + \epsilon}} \nabla w_t$

(b) small





↓ b move

non-sphere

RMSprop

$v_t = \beta v_{t-1} + (1-\beta)(\nabla w_t)^2$ — exp decaying avg

$\beta = 0.95$

old epoch

$w_{t+1} = w_t - \frac{\eta}{\sqrt{v_t + \epsilon}} \nabla w_t$

→ $v_0 = 0$

→ $v_1 = 0.95 \times 0 + 0.05 (\nabla w_1)^2$

→ $v_2 = 0.95 \times 0.05 (\nabla w_1)^2 + 0.05 (\nabla w_2)^2$

→ $v_3 = 0.95 \times 0.95 \times 0.05 (\nabla w_1)^2 + 0.95 \times 0.05 (\nabla w_2)^2 + 0.05 (\nabla w_3)^2$

smaller ← epoch 1 | epoch 2 | epoch 3 ← largest

v_t shoot → η small

Adagrad → NN → non-convex optimization

↳ convex optimiz → linear reg

↘ convex optimiz → xmax / reg

global minima

RMSprop

Disadvantage

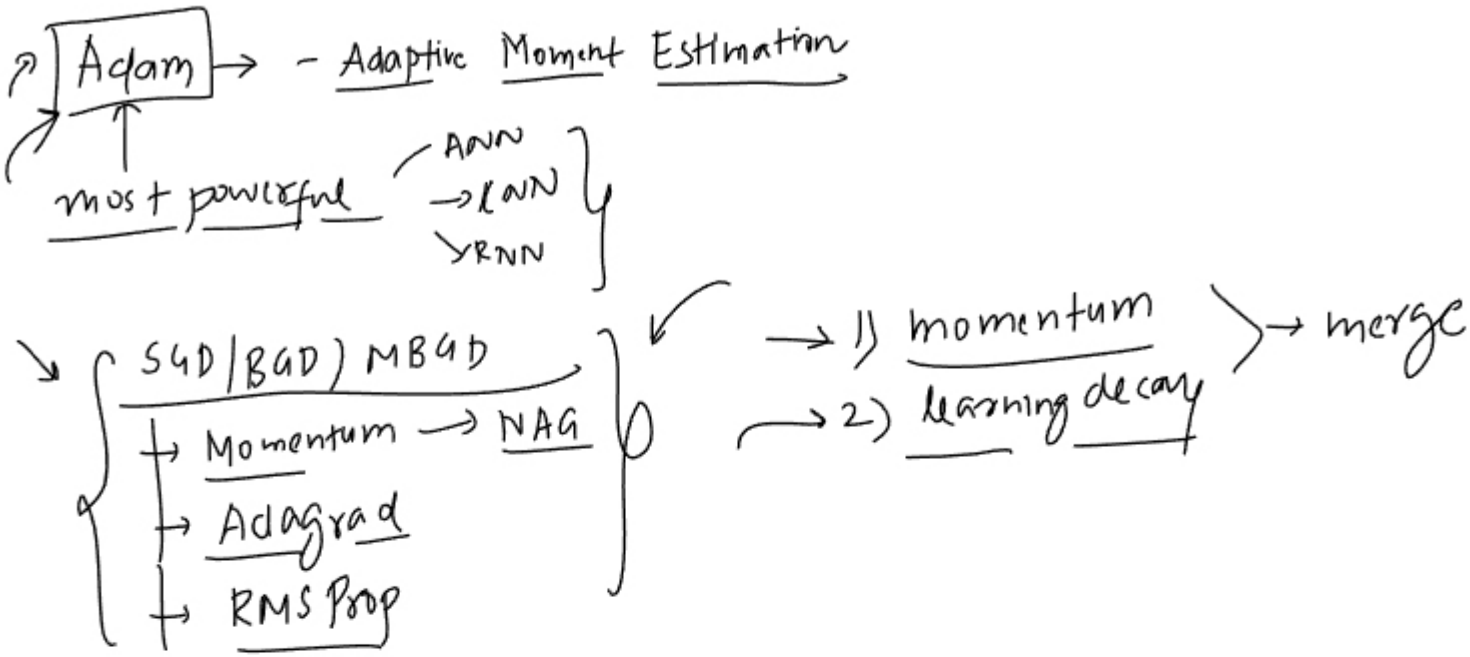
03 August 2022 14:54

No,
Optimization

↓
Adam

Introduction

04 August 2022 10:45



Mathematical Formulation

04 August 2022 10:45

Keras

Bias correction

epoch #

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{v_t + \epsilon}} * m_t \rightarrow \hat{m}_t = \frac{m_t}{1-\beta_1^t} \rightarrow \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$\beta_1 = 0.9$

$\beta_2 = 0.99$ → Keras

where

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla w_t$$ → momentum

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla w_t)^2$$ → Adagrad

$\eta = 0.1$

0.01

$m_0 = 0$ $m_t = 0$ $v_t = 0$

$v_0 = 0$

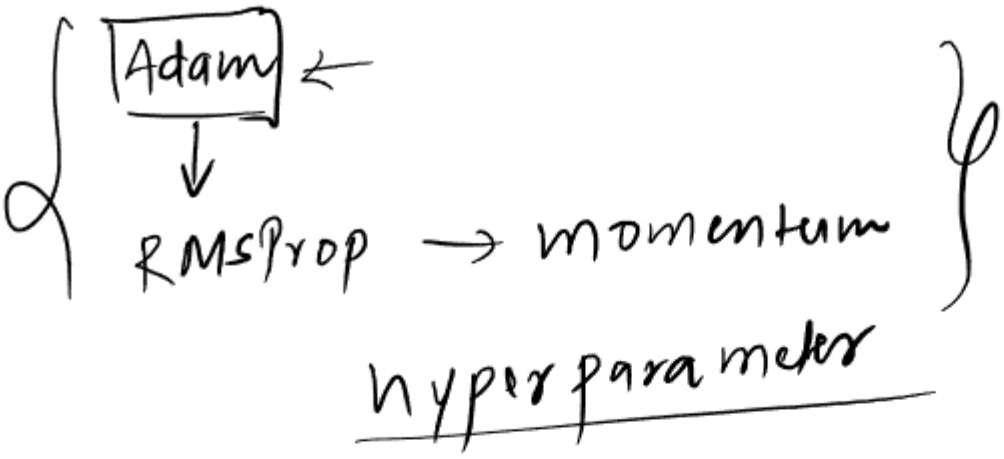
$0 \rightarrow$

Demo

04 August 2022 10:45

Verdict

04 August 2022 10:45

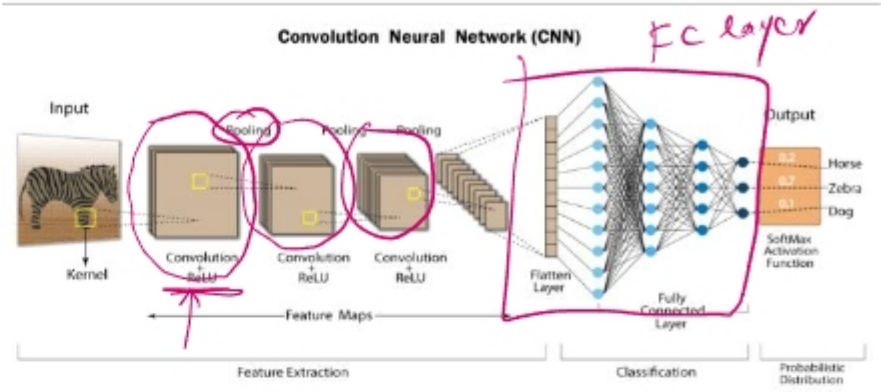


What is a CNN?

17 August 2022 06:47

Convolutional neural networks, also known as convnet, or CNNs, are a special kind of neural network for processing data that has a known grid-like topology like time series data(1D) or images(2D).

→ Convolution
→ Pooling
→ FC layer - ANN



→ ANN → Convolution → CNN

ANN → matrix mult.

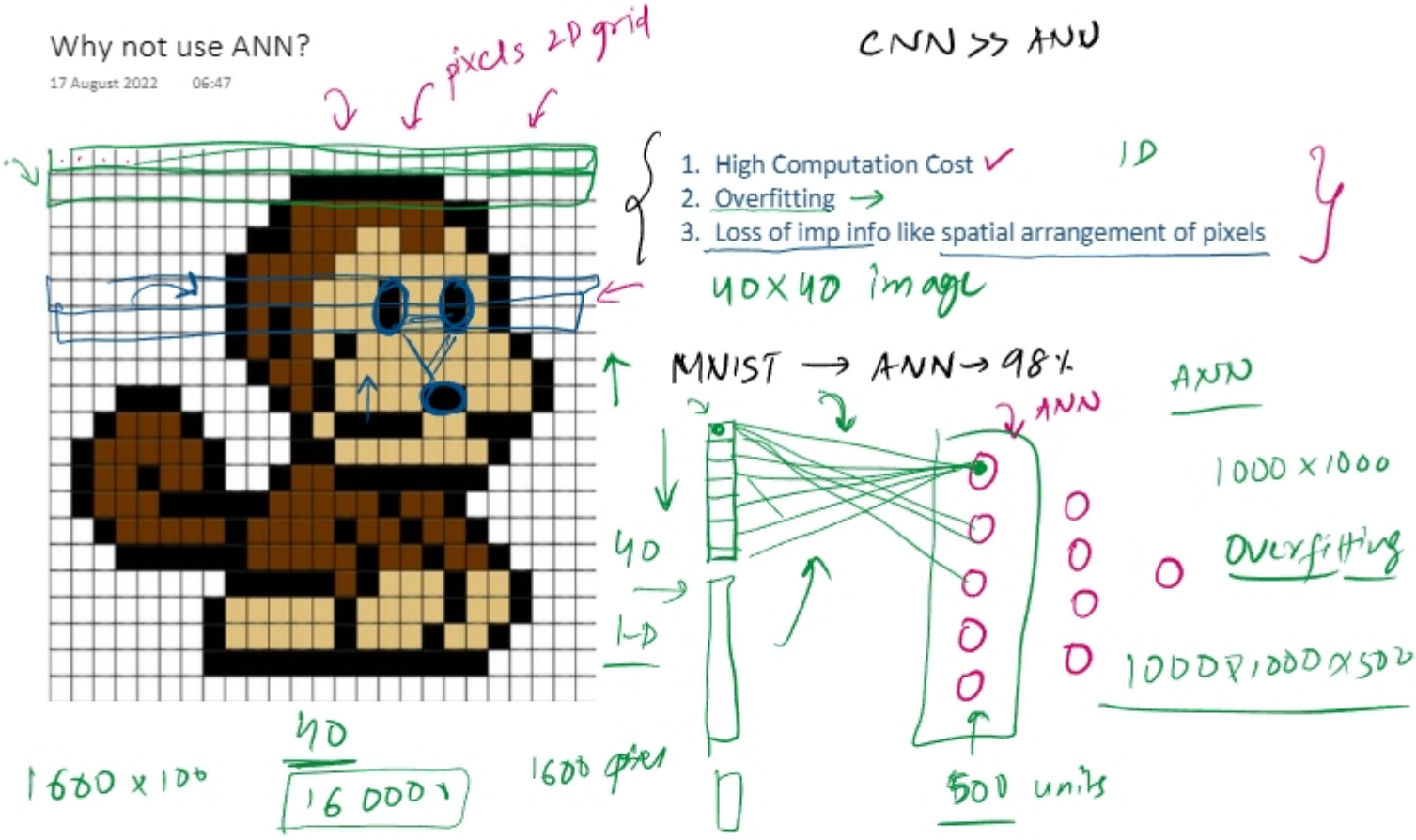
Inspirations
↳ visual cortex

1998 Yann Lecun → AT&T →
↳ Microsoft → OCR, hand writing reog } CNN

{ facial reog, self driving } CNN RNN

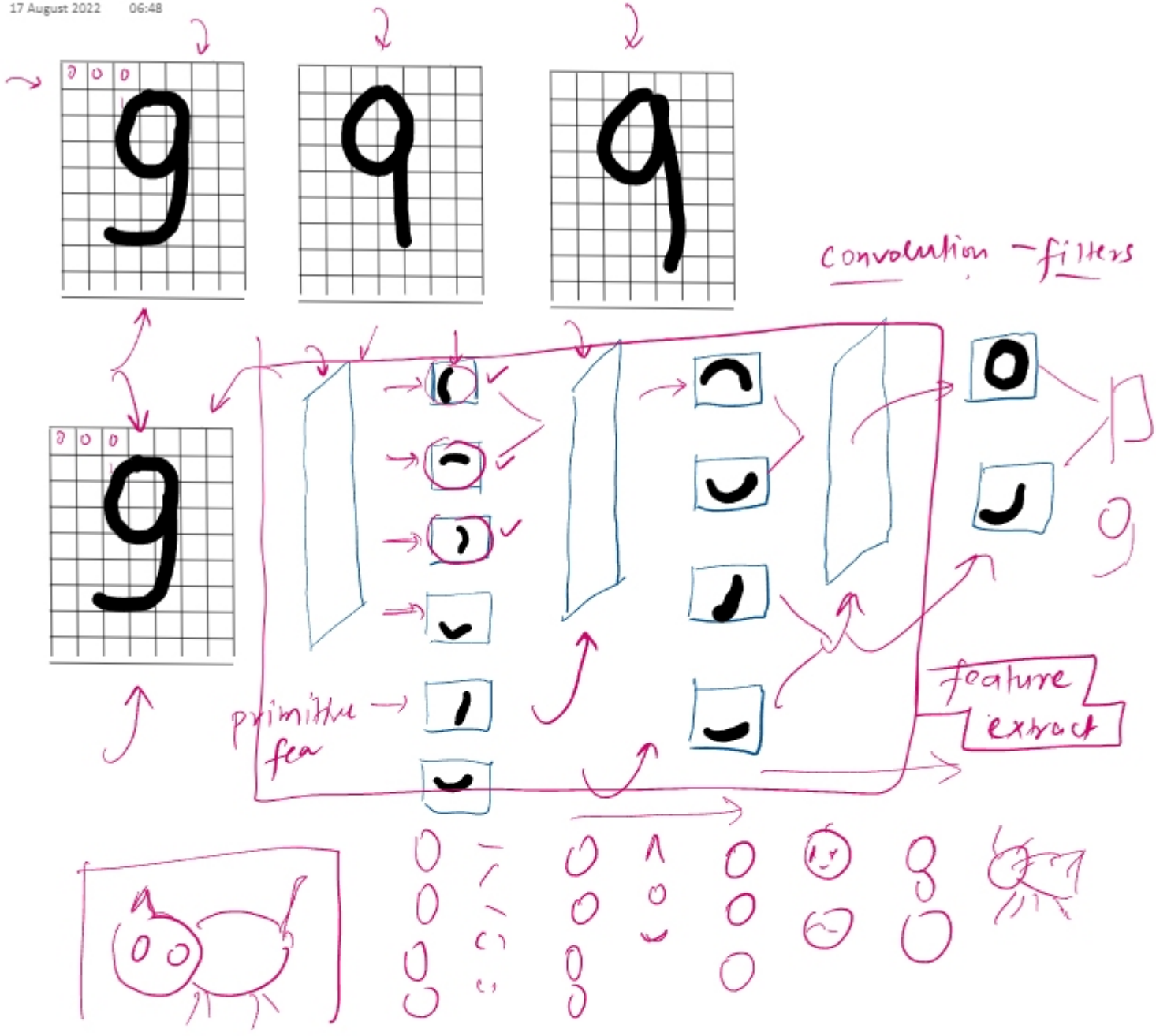
Why not use ANN?

17 August 2022 06:47



CNN Intuition

17 August 2022 06:48

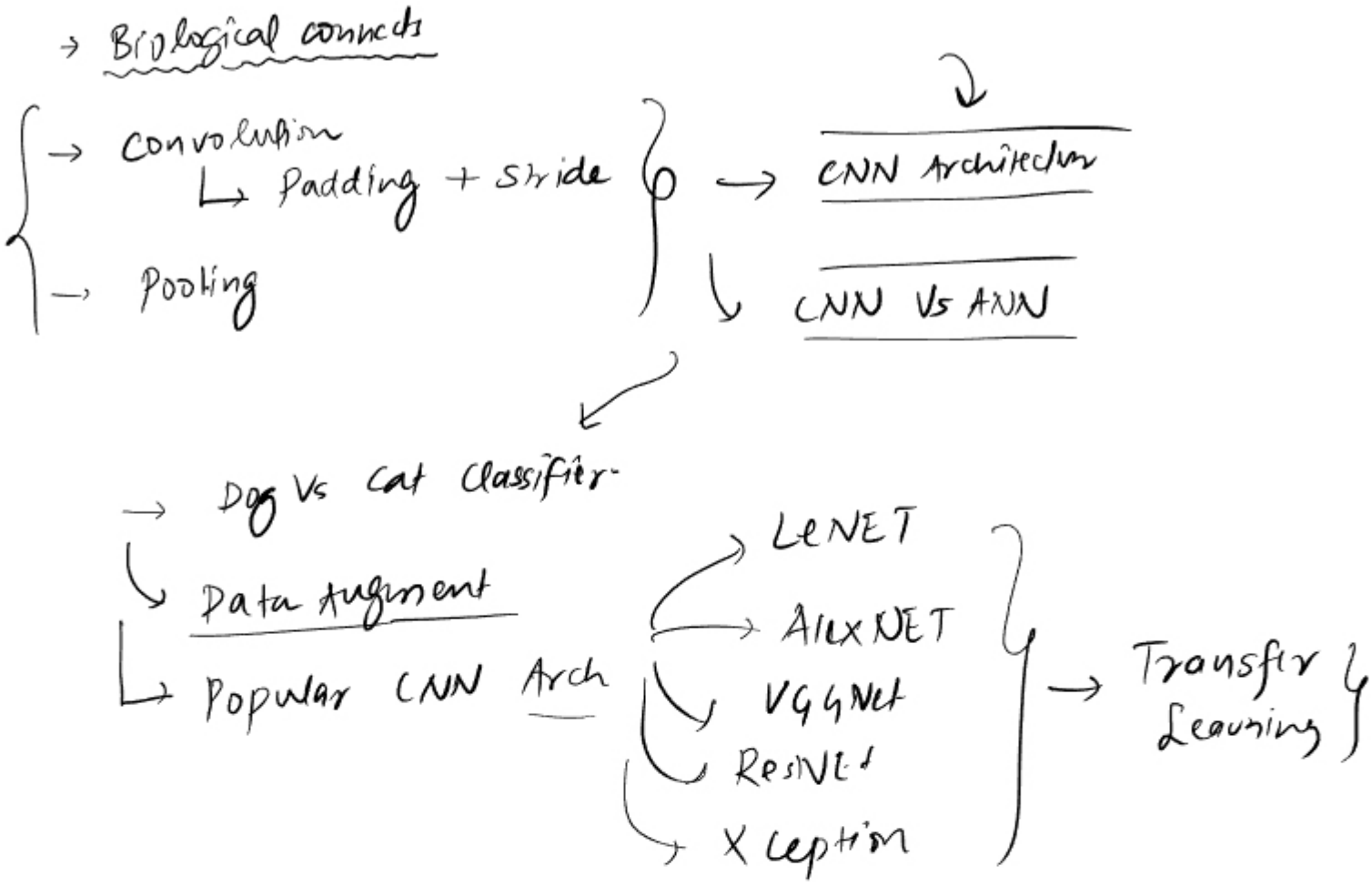


CNN Applications

17 August 2022 06:48

Roadmap

17 August 2022 06:48



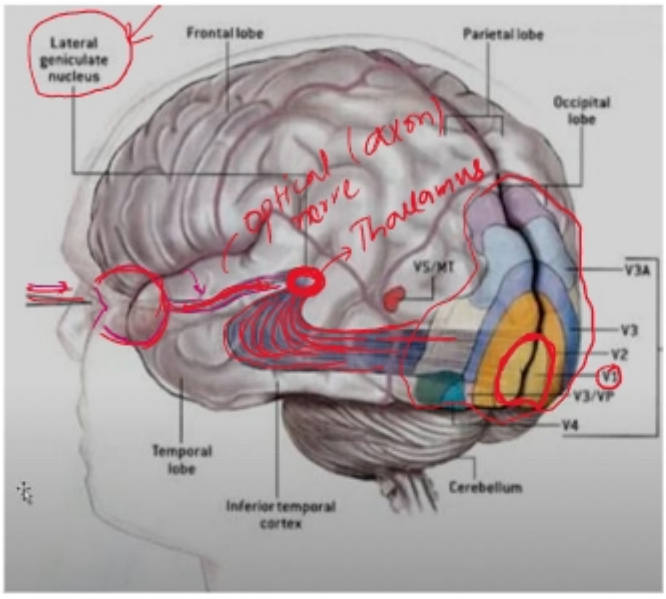
Human Visual Cortex

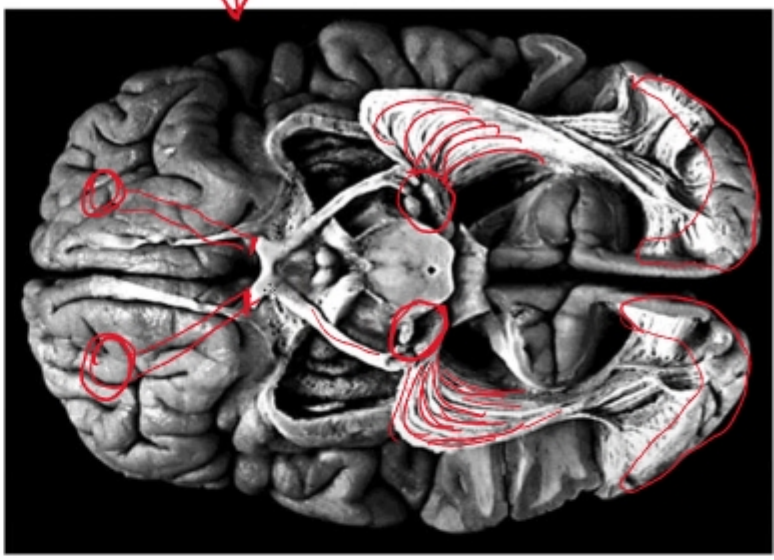
18 August 2022 16:05

2D sheet

primary V cortex

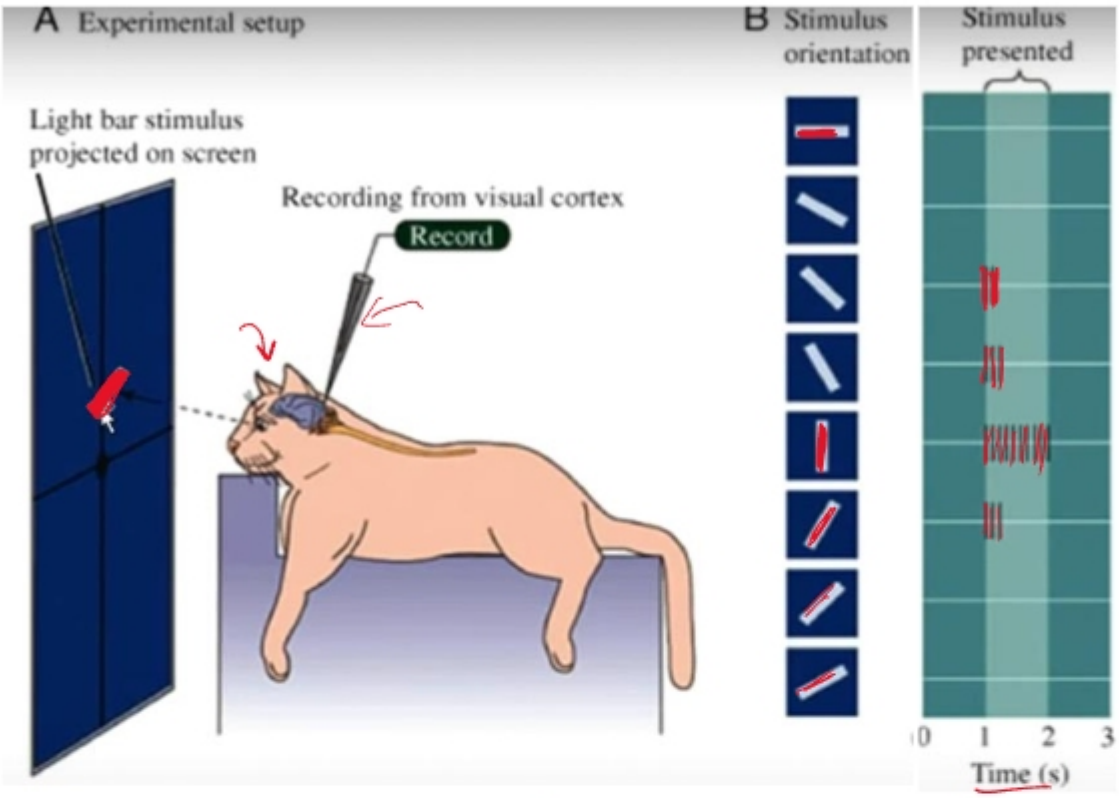
actual brain

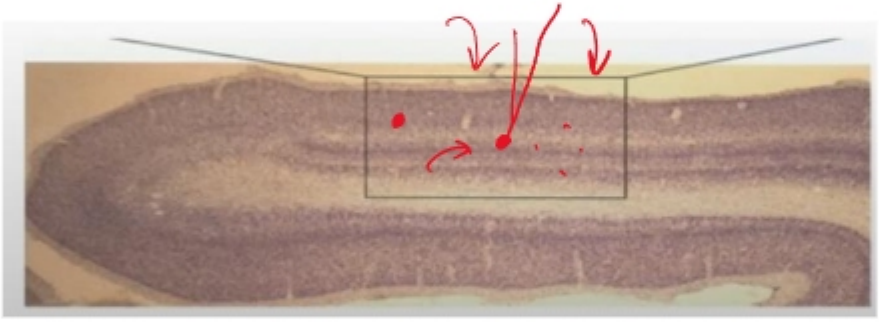




The Experiment

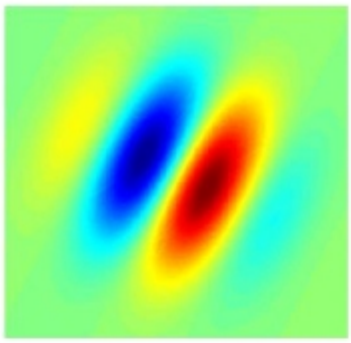
18 August 2022 16:09





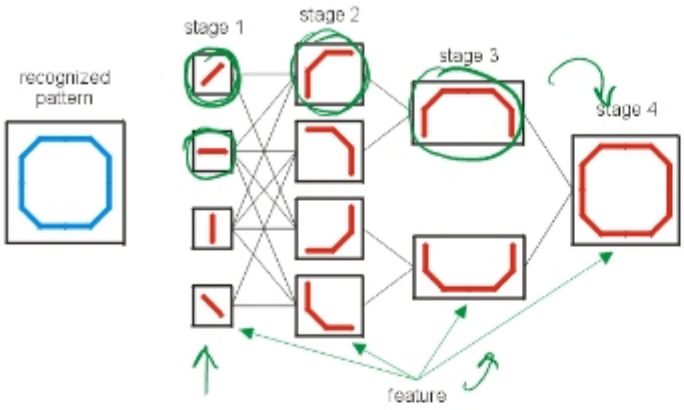
Conclusion

18 August 2022 17:57

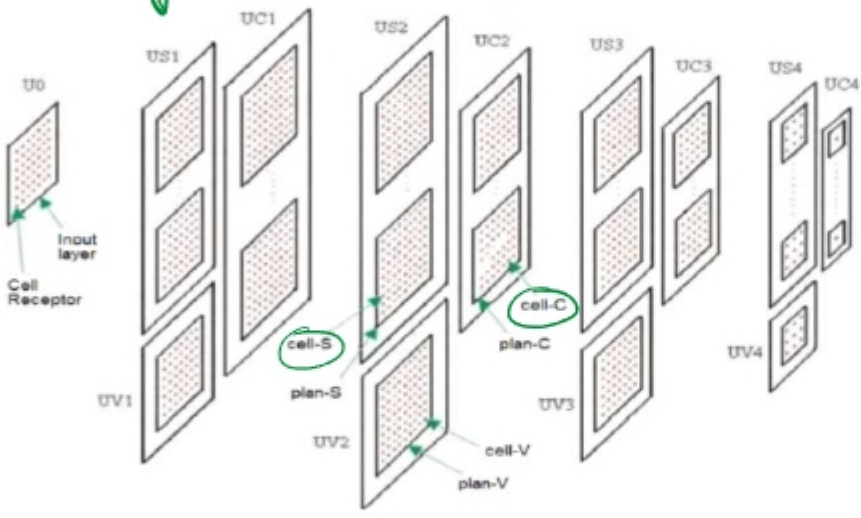


Development

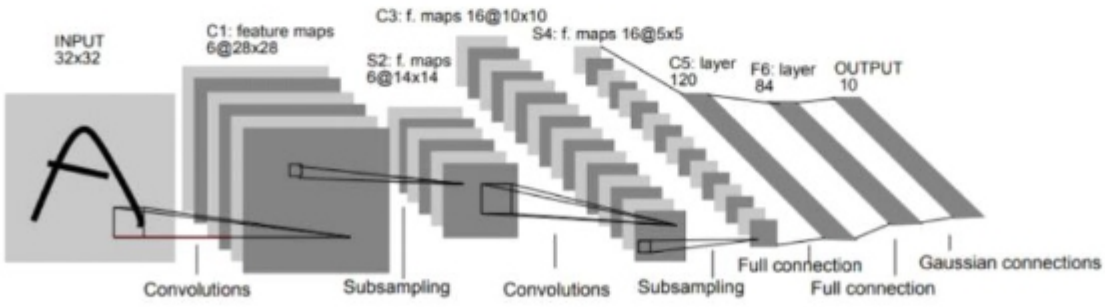
18 August 2022 16:15



Neocognitron → Fukushima



Yann LeCun → CNN → Backprop convolution

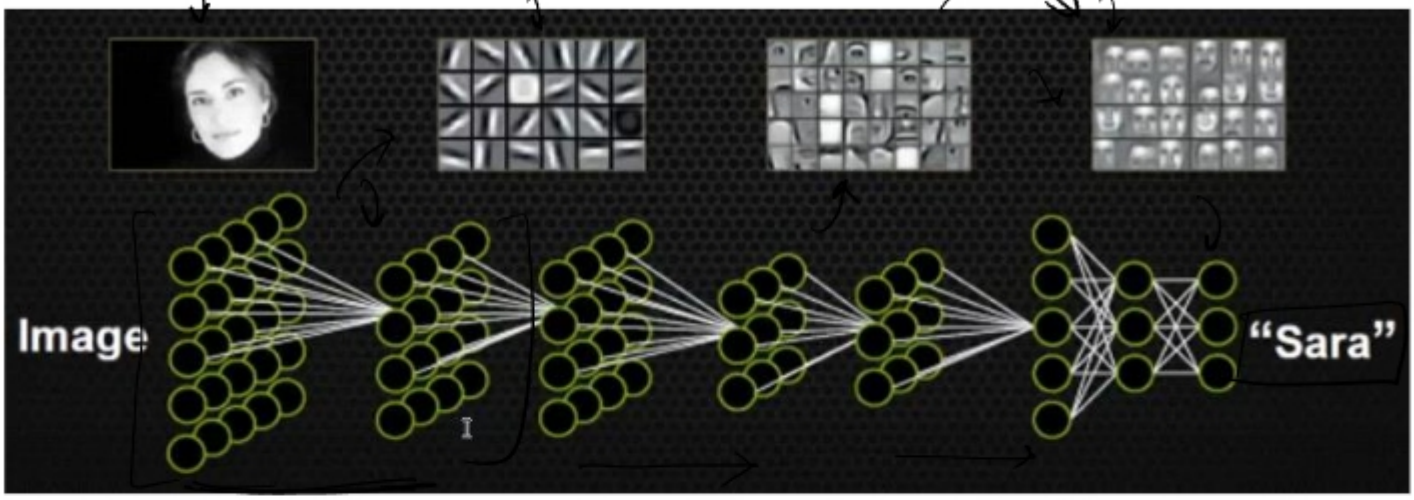


early CNN

2012 → AlexNET → ImageNET → CNNs

Introduction

19 August 2022 16:45



CNN → NN special → Convolution, Pooling, FC

CNN

Basics of Images

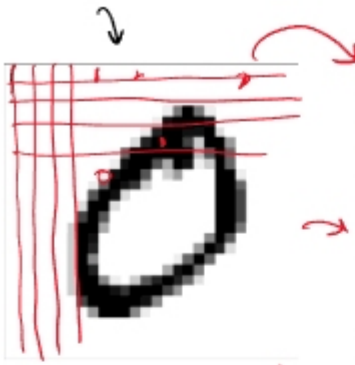
19 August 2022 16:53

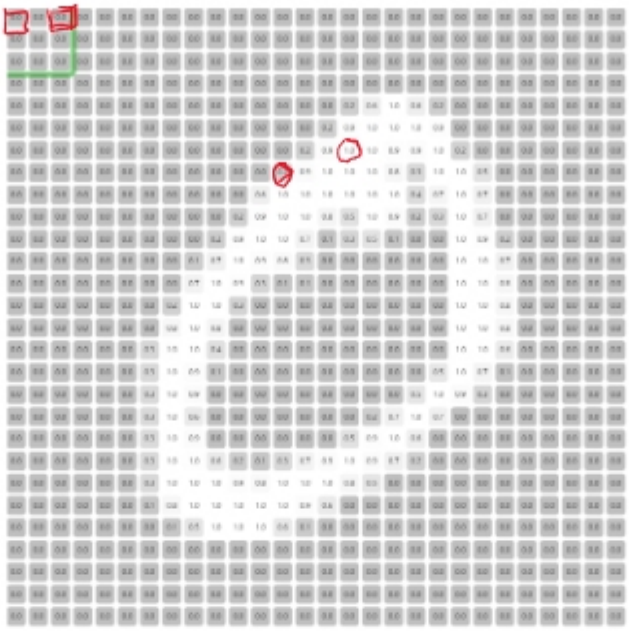
0-255 → 0-1

B/W

1 channel – Greyscale → B/W

RGB → colored images





28 x 28 pixels

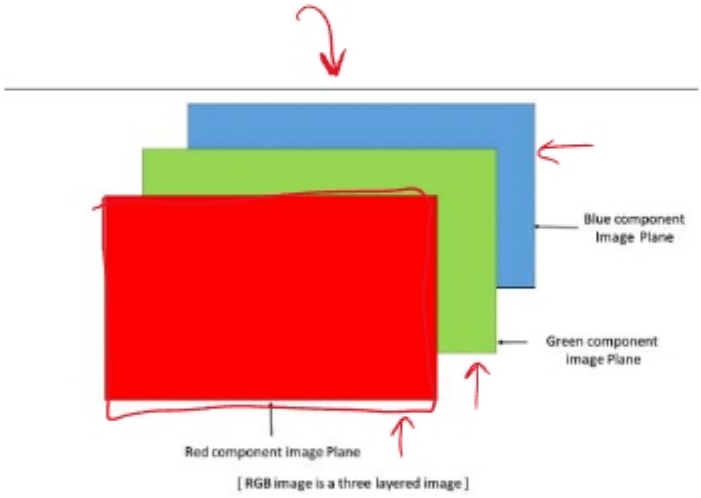
2D arrays →

2D numpy arrays

mnist → 0

(28 x 28)

3 channels → Red Green Blue (primary)

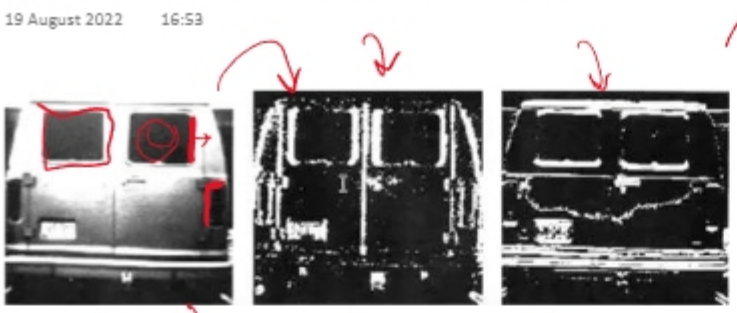


228 * 228 * 3

3 channels

Edge Detection (Convolution Operation)

19 August 2022 16:53



0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

image

6×6

*

-1	-1	-1
0	0	0
1	1	1

=

ANN weight

filter / kernel

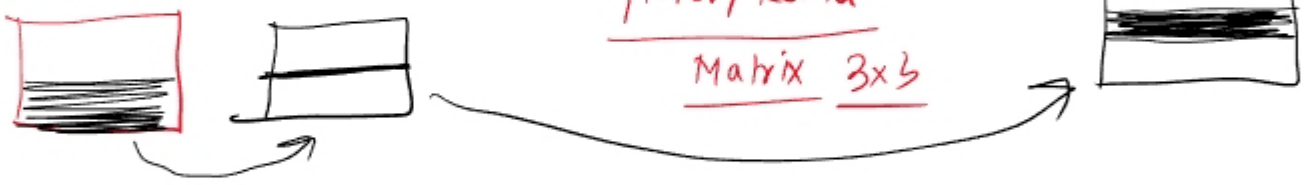
Matrix 3×3

255×1 + 255×1 + ... horizontal edge detection film

-1×0 + -1×0 + -1×0 ...

0	0	0	0
255	255	255	255
255	255	255	255
0	0	0	0

→ feature map



0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

(6 x 6)

*

-1	-1	-1
0	0	0
1	1	1

(3x3)

=

(4x4)

(28 x 28) → (26 x 26)

n x n

(3x3) → ? (26 x 26)

m x m → (n - m + 1) x (n - m + 1)

(64 x 64) (3x3) → (62 x 62)

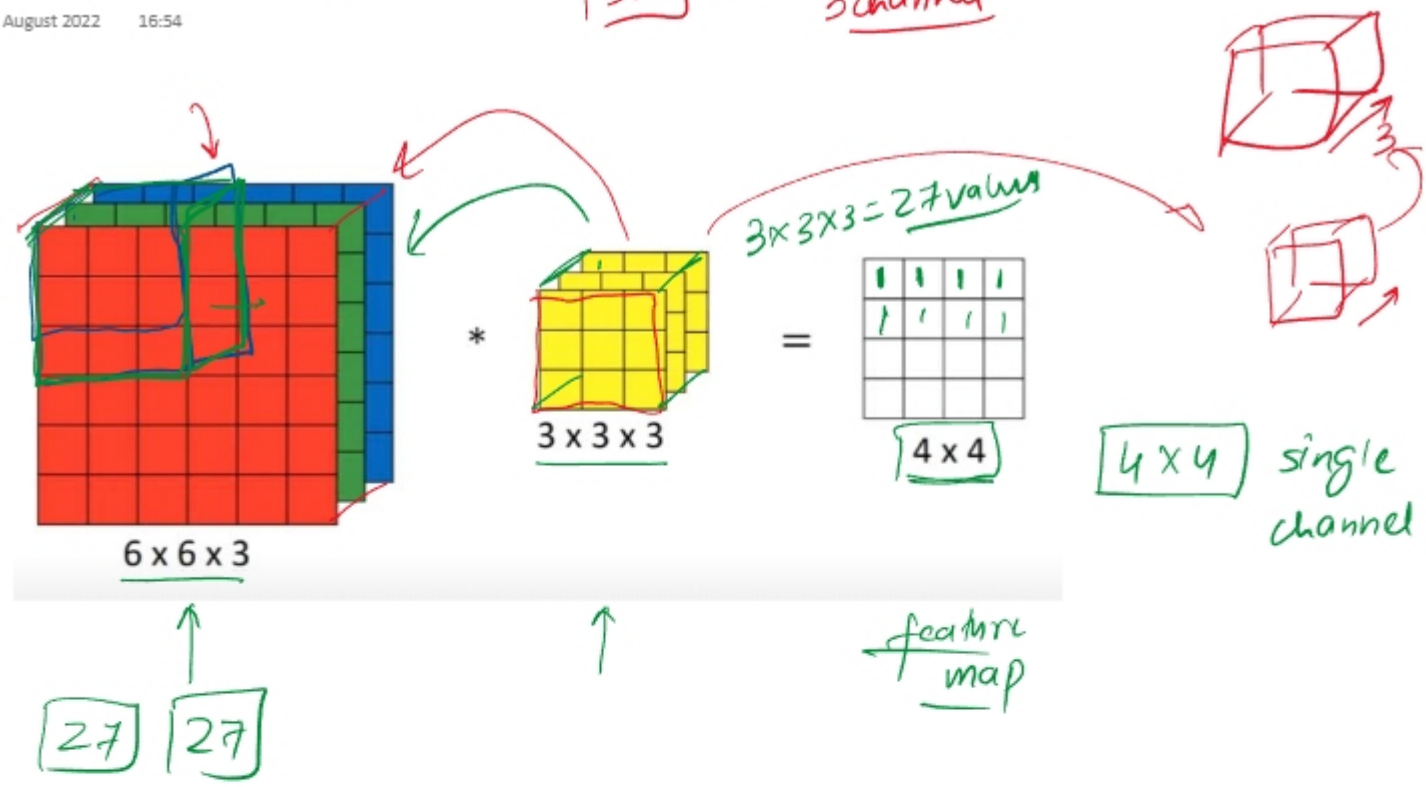
Demo

19 August 2022 16:54

Working with RGB Images

19 August 2022 16:54

3x3 x3 — 3 channel



$m \times m \times c$ $n \times n \times c$ $\longrightarrow$ $(m-n+1)(m-n+1)$ single channel

Multiple Filters

23 August 2022 08:24

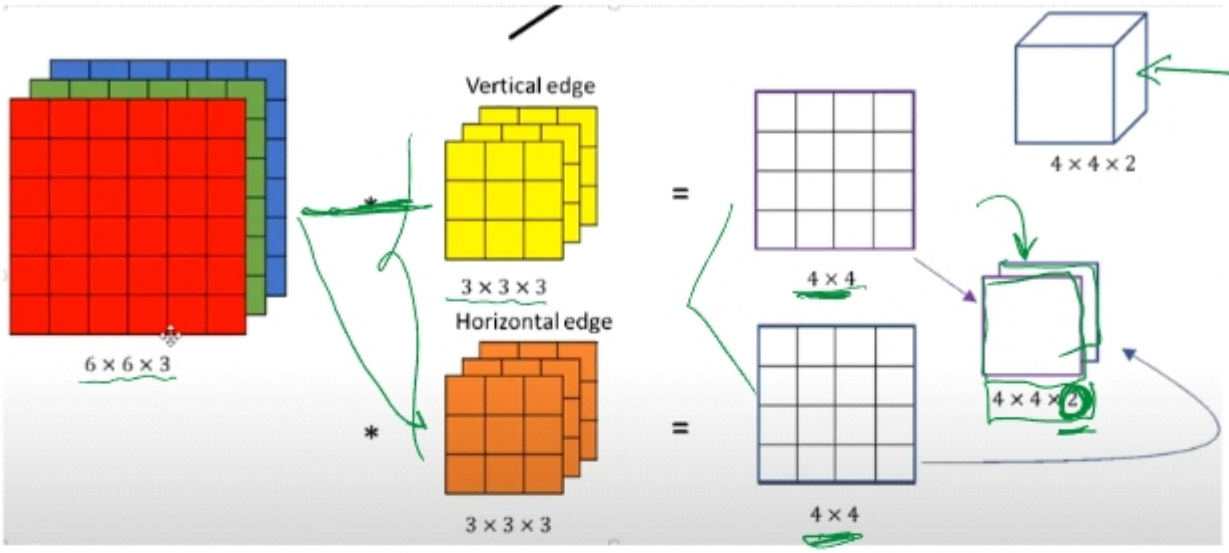


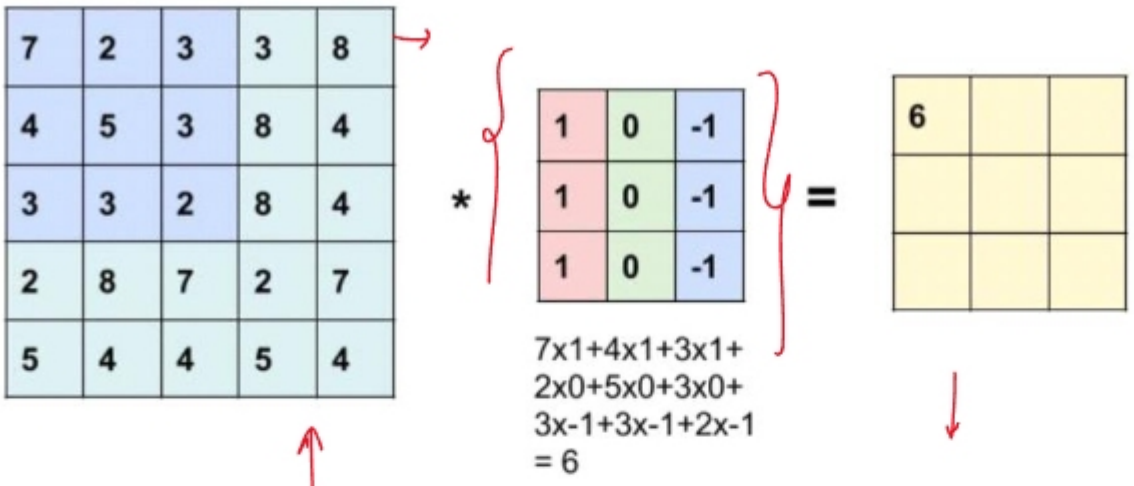
Image taken from Andrew NG's lecture

Problem with Convolution

26 August 2022 14:25

What is Padding?

26 August 2022 14:26



$n-f+1=n$

$n \times n$

5×5

7×7

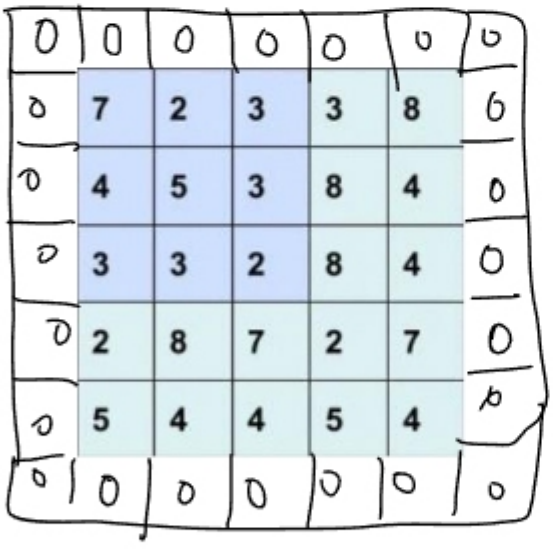
$f \times f$ → $(n-f+1)(n-f+1)$

3×3 $(5-3+1) = 3 \times 3$

$n-f+1 = 5$

$n-3+1=5 \Rightarrow n = 8-1$

$n=7$



7×7

convolution

Zero padding

$5 \times 5 \longrightarrow 3 \times 3$

$5 \times 5 \longrightarrow 3 \times 3$

$(n - f + 1)$

$\downarrow$

$(n + 2p - f + 1)$

$5 + 2(1) - 3 + 1$

$= 7 - 3 + 1 = 5$

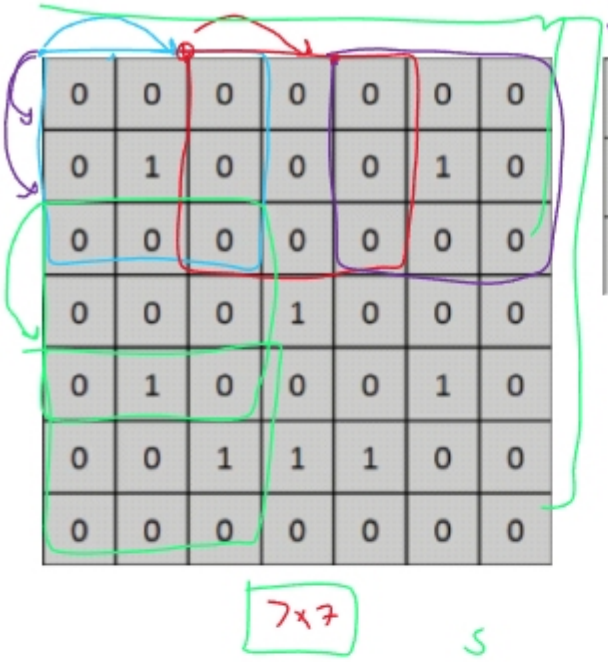
Keras

Valid

Same

Strides

27 August 2022 08:23



Stride = 1

(1,1)

3x3

right

bottom

→ result feature

Stride = (2,2)

Stride = 2 →

$\frac{7-3}{2} + 1$

$2+1 = 3$

$(n-f+1) \longrightarrow \left[\frac{n-f}{s} + 1\right]$ ↘ $p = p$

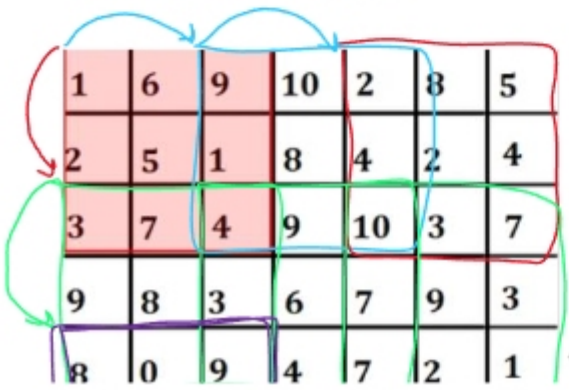
$\left[\frac{n+2p-f}{2} + 1\right]$ → strided convolution

$\frac{7+2-3}{2} + 1 = [4 \times 4]$

$\frac{n-f}{2}$ $\frac{6-3}{2}$ $1.5 = 1 + 1 = 2$

Special Case

Stride = 2



2x3

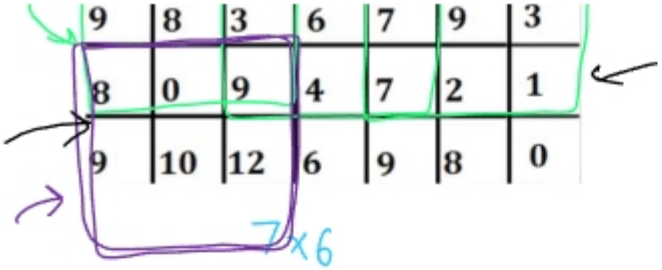
$\frac{6 \times 7}{3 \times 3}$

$\left[\frac{n-f}{s} + 1\right]$

19 → 1

1.1 → 1

floor



$$\left[\frac{\cdot}{s}+1\right]$$

floor

$$\left[2 \quad \frac{6-3}{2}+1 = 1.5+1\right]$$

$1+1=2$

$$\frac{7-3}{2}+1 = 2+1=3$$

Why Strides are required?

27 August 2022 08:24

1) High level features

2) Computing →

Keras → stride

$$\left[\frac{n+2p-f}{s}+1\right]$$

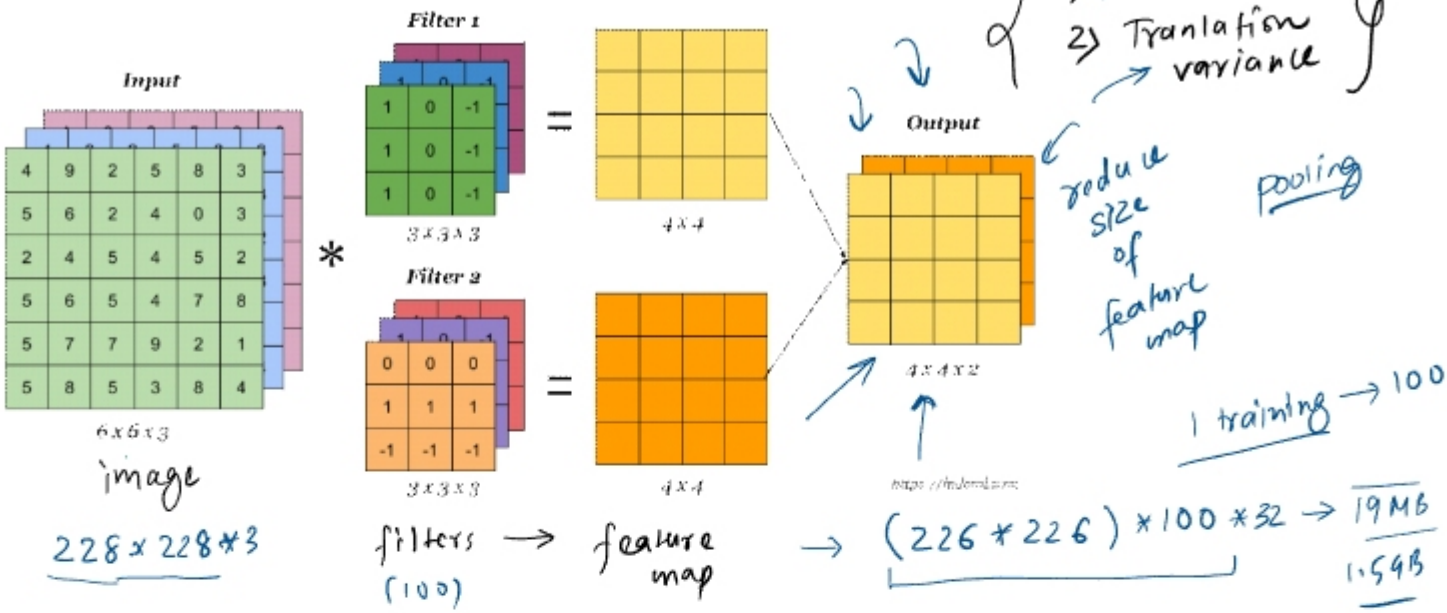
$$\frac{28+2-3}{2}+1$$

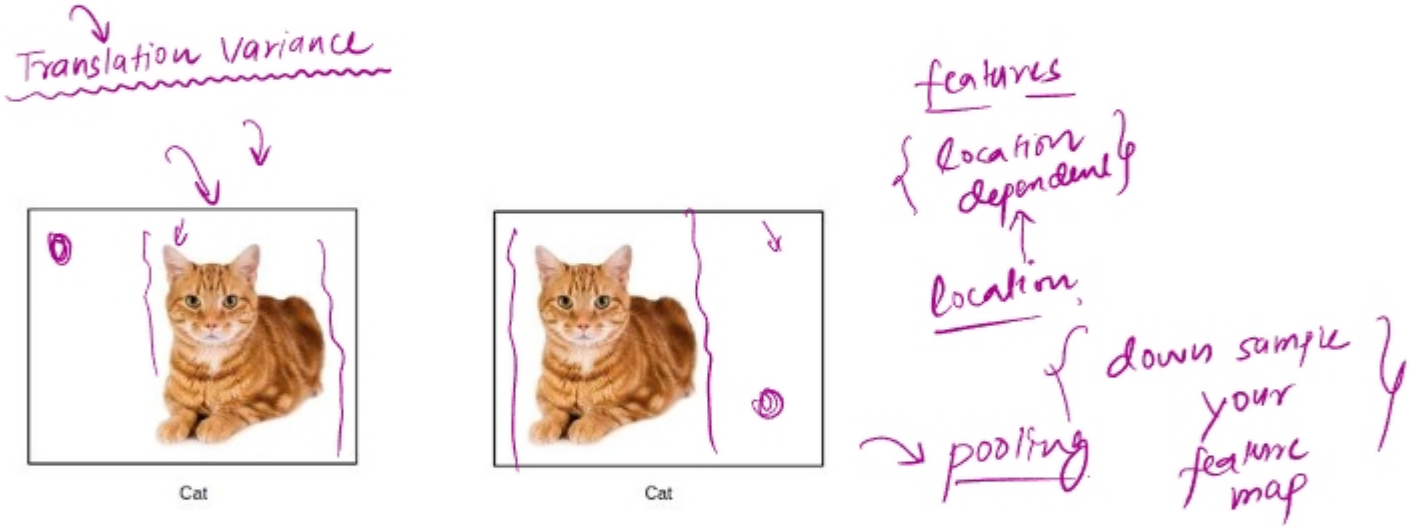
$$\frac{13.5}{}+1$$

$$13+1=14$$

The Problem with Convolution

01 September 2022 09:55





Pooling

01 September 2022 09:55

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

*

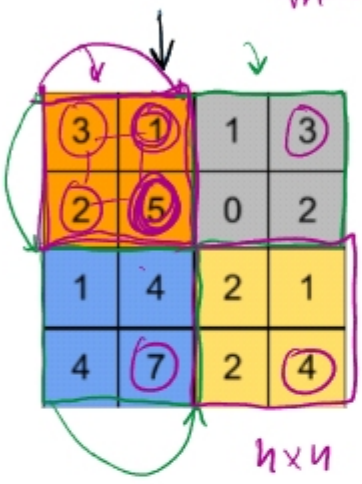
-1	-1	-1
0	0	0
1	1	1

= relu((4x4))

Max pooling
Min pooling
Avg pooling
L2 pooling
Global pooling

feature map (non-linear)

max pooling



4x4

(size) → (2,2)
stride → (2)
type → Max

receptive field

5	3
7	4

2x2

feature map

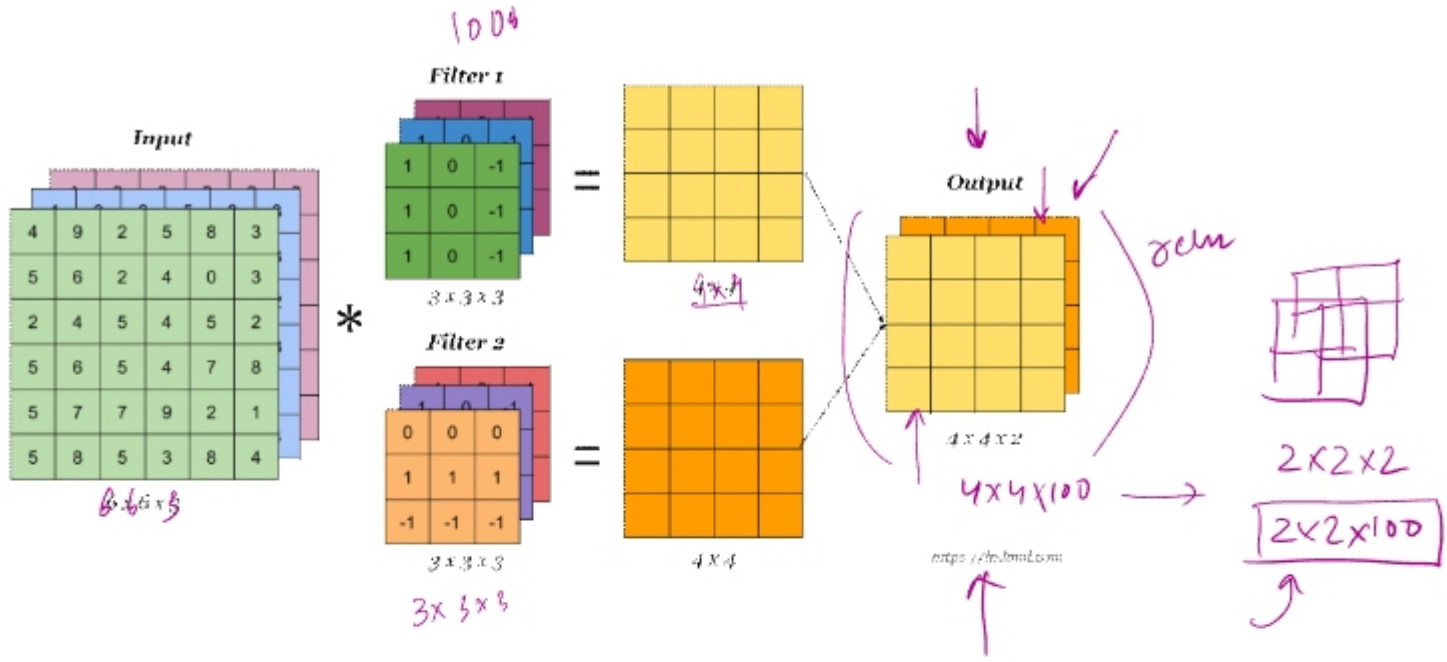
low level details eliminat

Demo

01 September 2022 09:57

Pooling on Volumes

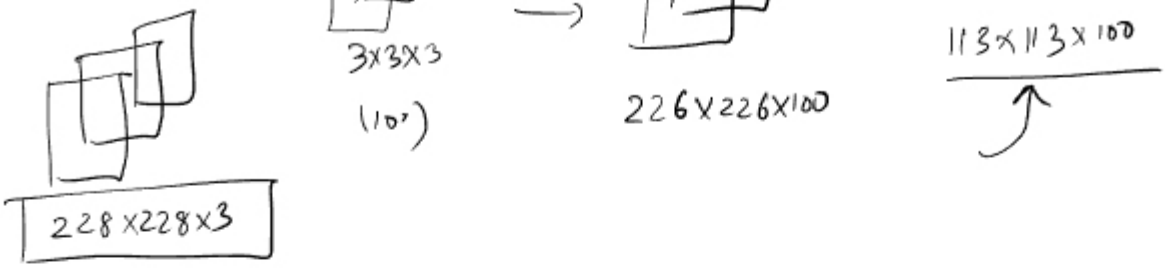
01 September 2022 09:56



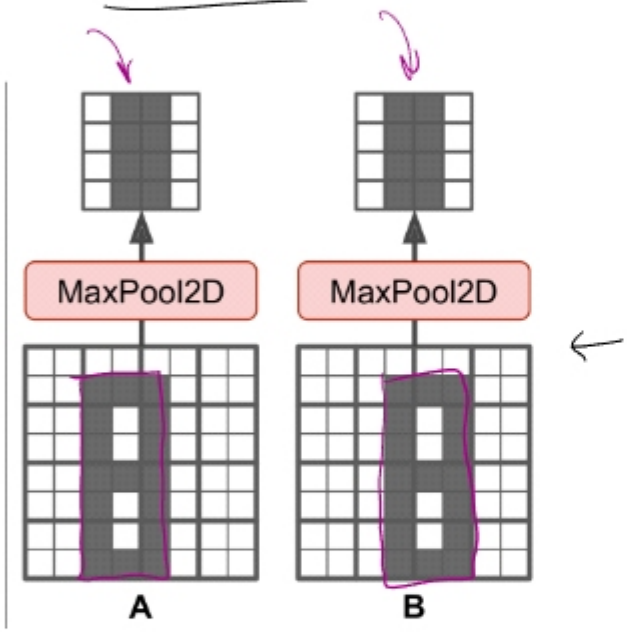
Advantages of Pooling

01 September 2022 09:56

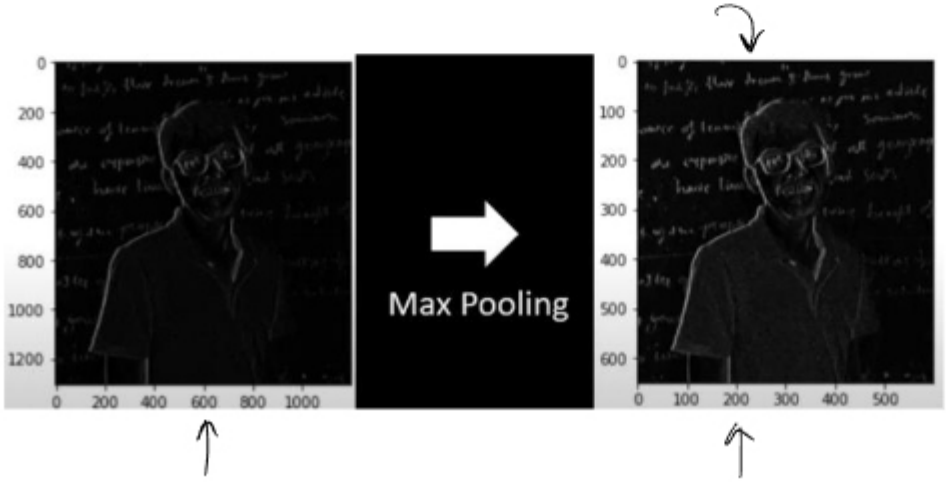
1) reduced size



2) Translation invariance



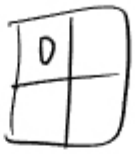
3) Enhanced features (only in case of Max pooling)



4) No need of training min faster

max pooling → aggregate

avg pooling



→ (2x2)

→ 2

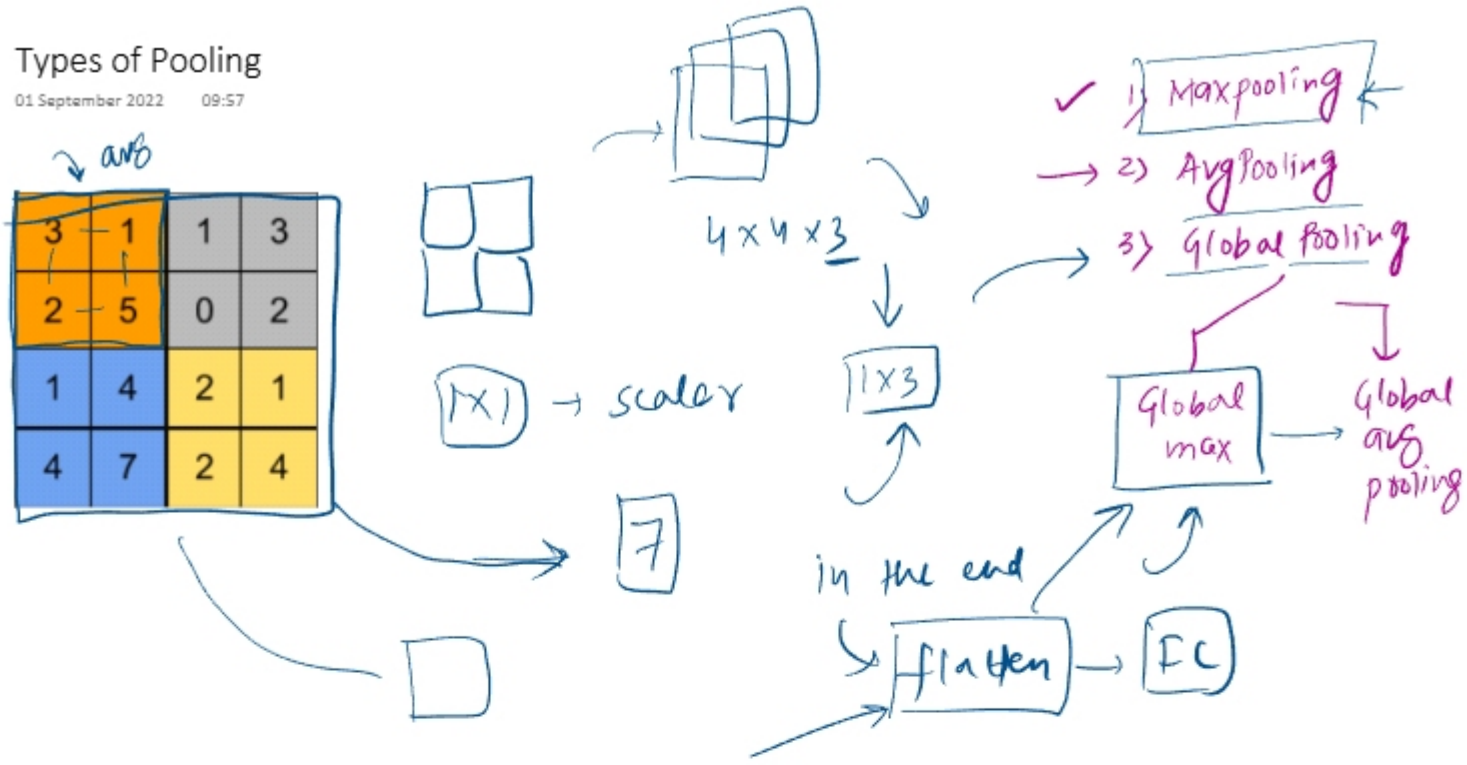
→ type

Keras Code

01 September 2022 09:56

Types of Pooling

01 September 2022 09:57



Disadvantages of Pooling

01 September 2022 09:57

lose a lot of info

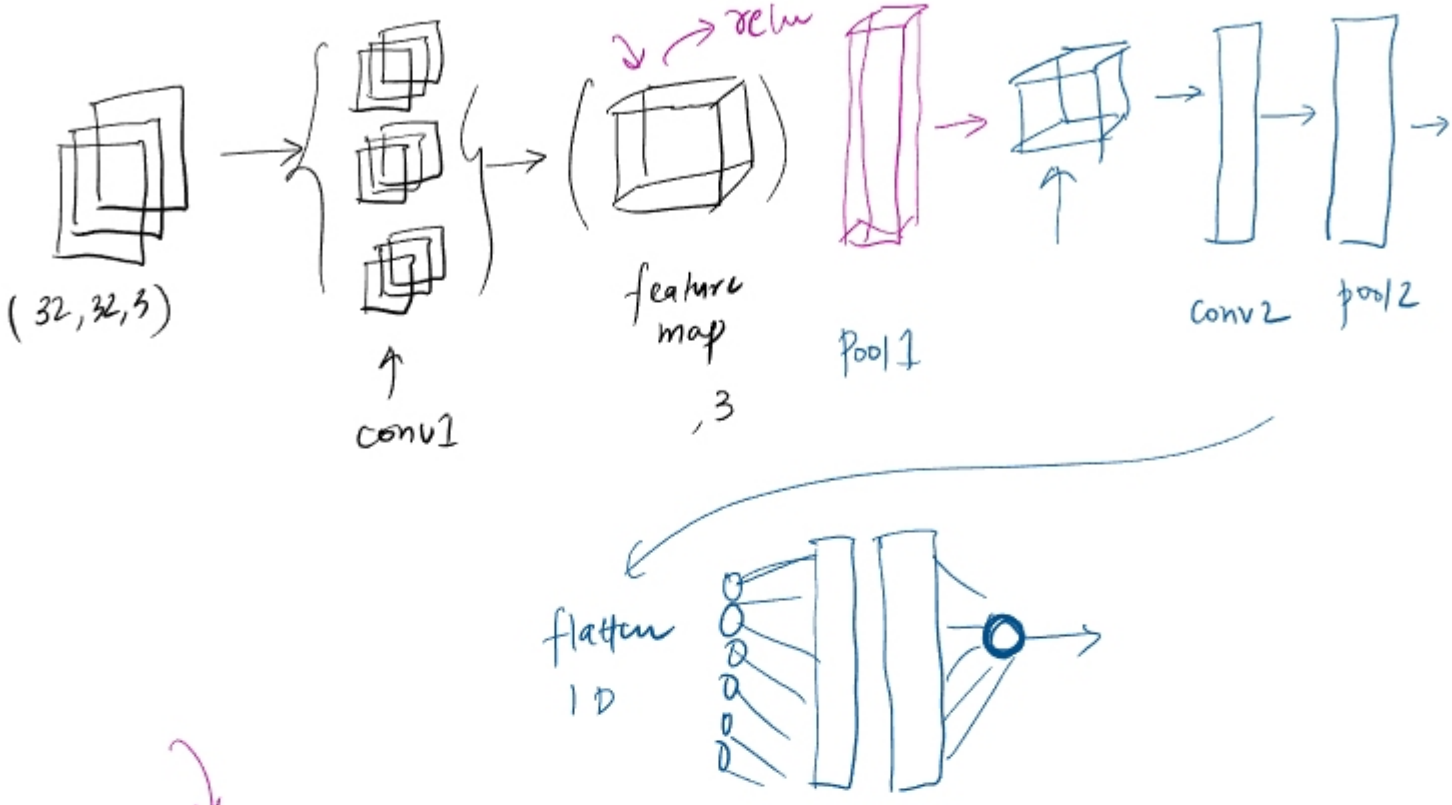
4x4 → 16

2x2 → 4

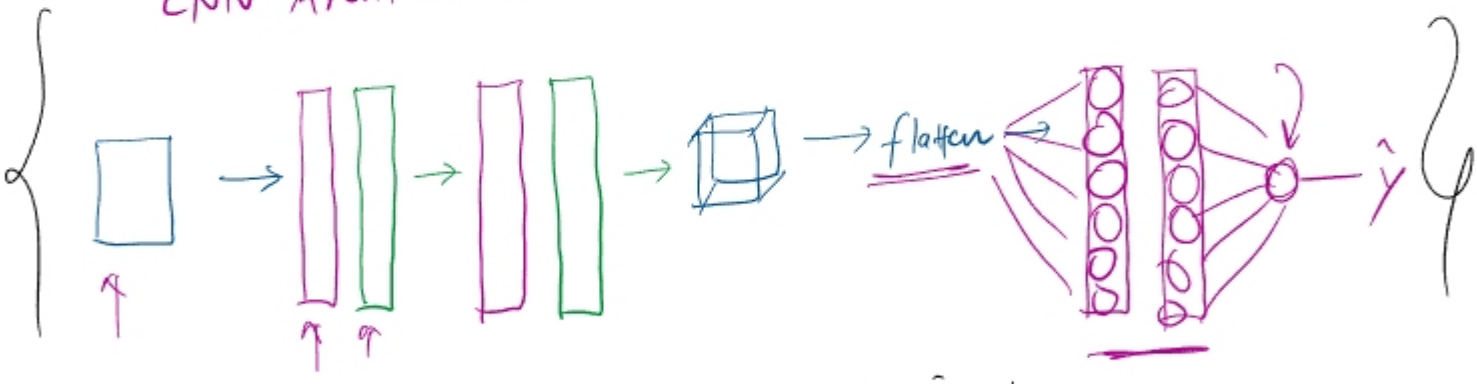
CNN Architecture

02 September 2022 10:55

1) Convolution 2) Padding / stride 3) Pooling



CNN Architecture



# conv # filters	stride padding	FC nodes # FC layer	activation dropout	Batch Norm

ImageNET

1) [LeNET] → Yann LeCunn
2) AlexNET
3) GoogLeNET
4) VggNET
5) ResNTE
6) Inception

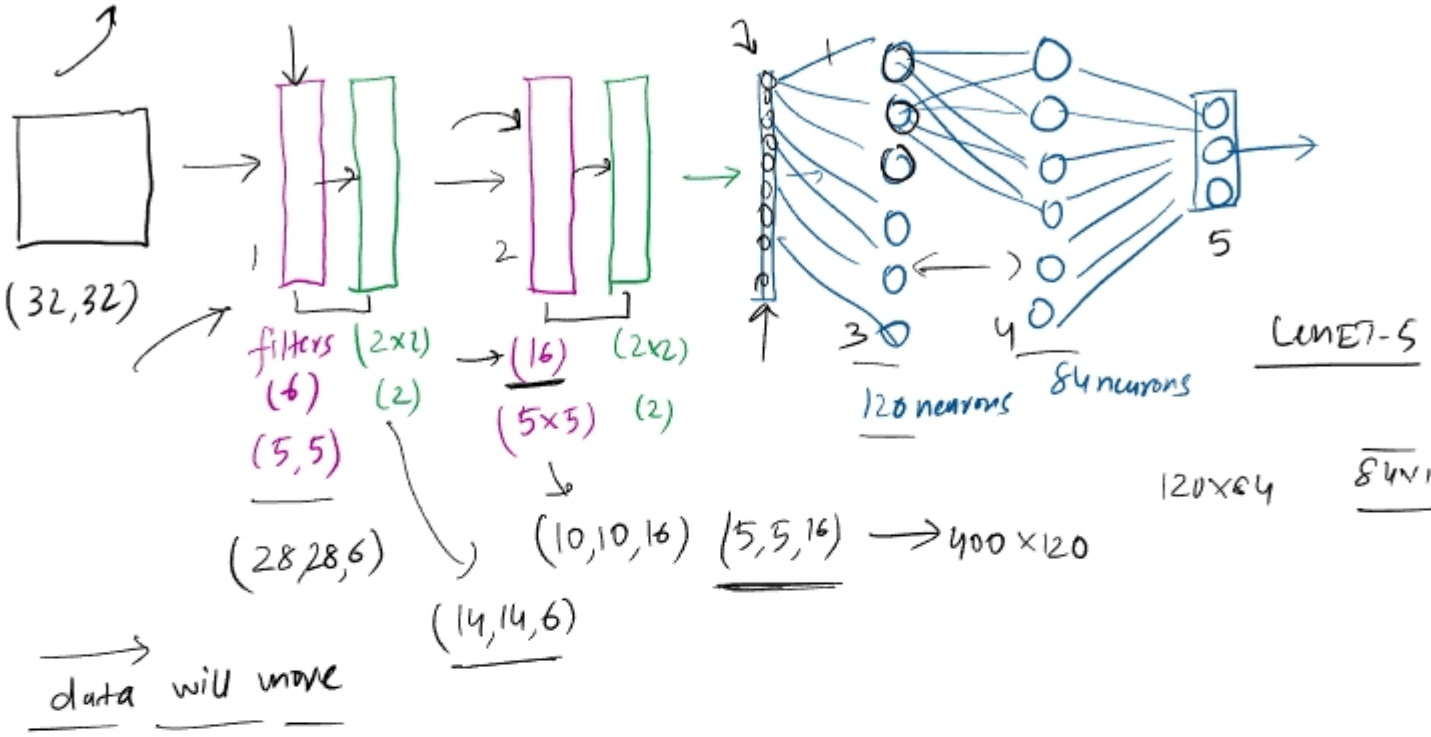
3) GoogLeNET

LeNet

02 September 2022 10:55

Yann LeCun → 1989 → 1998 → LeNET → CNN → US Navy Postal Service

LeNET-5



→ data will move

- AlexNET
- VggNET
- ResNET
- Inception

Guidelines

02 September 2022 10:58

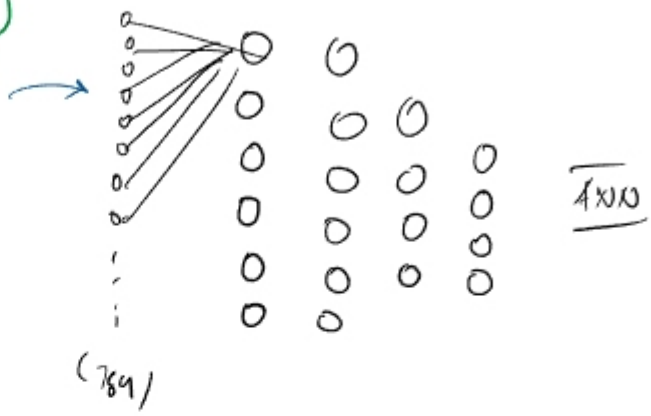
Keras Code

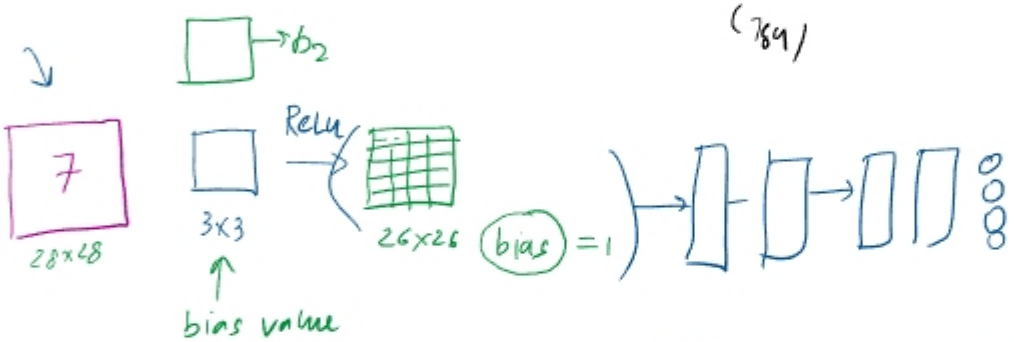
02 September 2022 14:58

CNN Vs ANN

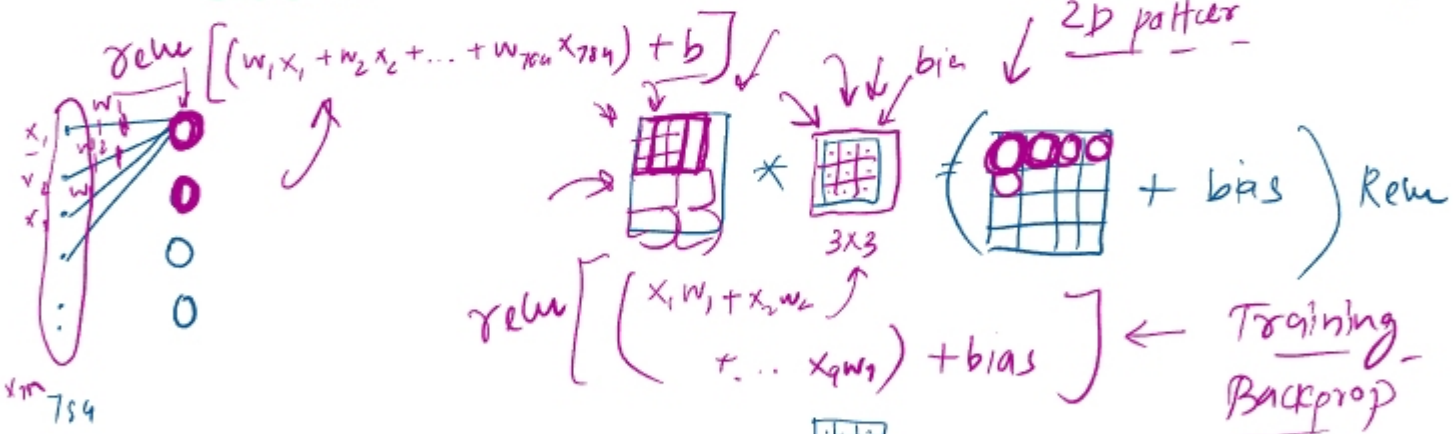
06 September 2022 10:00

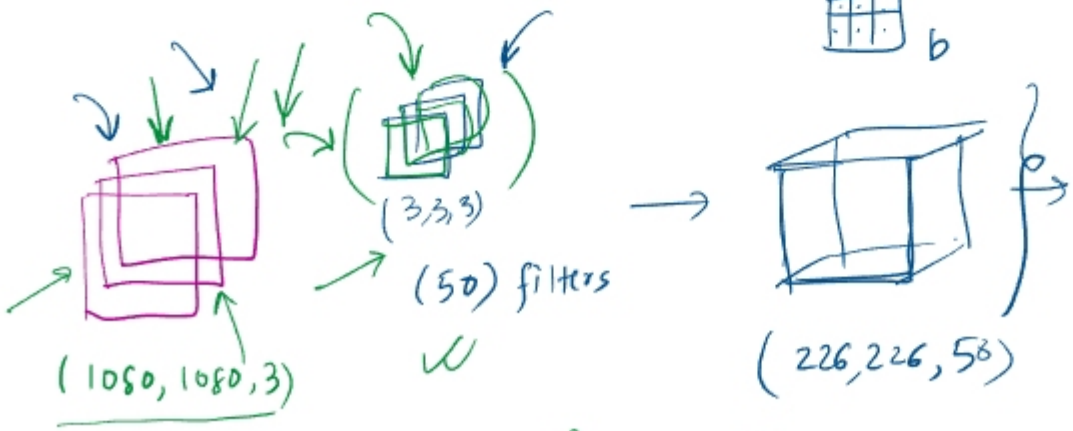
1) Computation Cost → ↓↓
2) Overfitting →
3) Loss of imp features like spatial arrangement of pixels

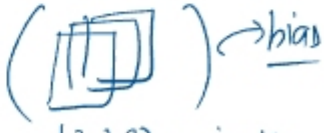




similarity ANN CNN

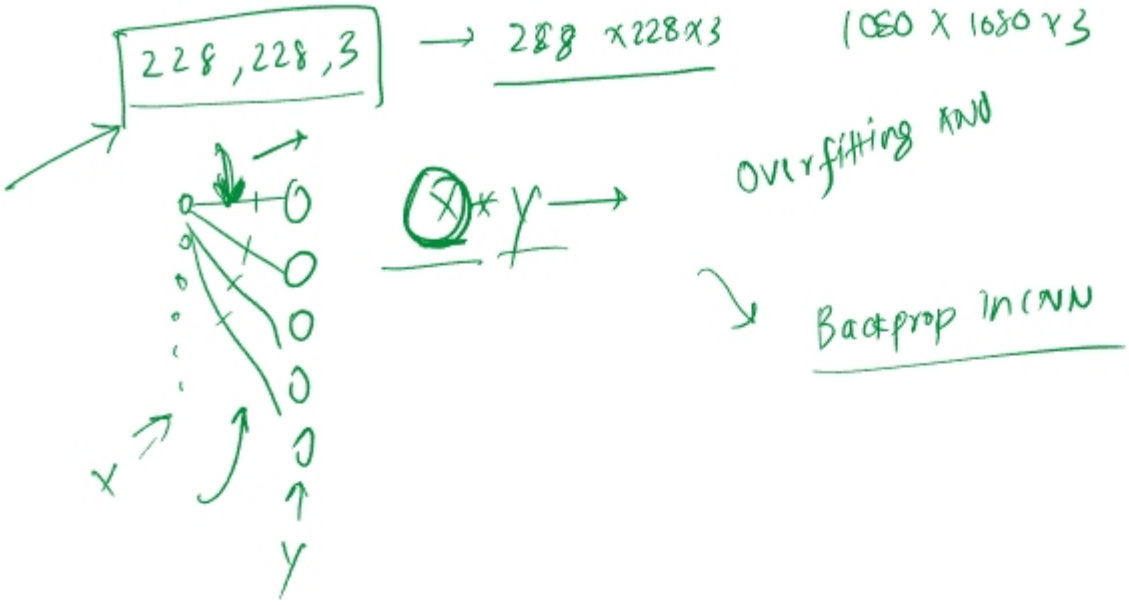






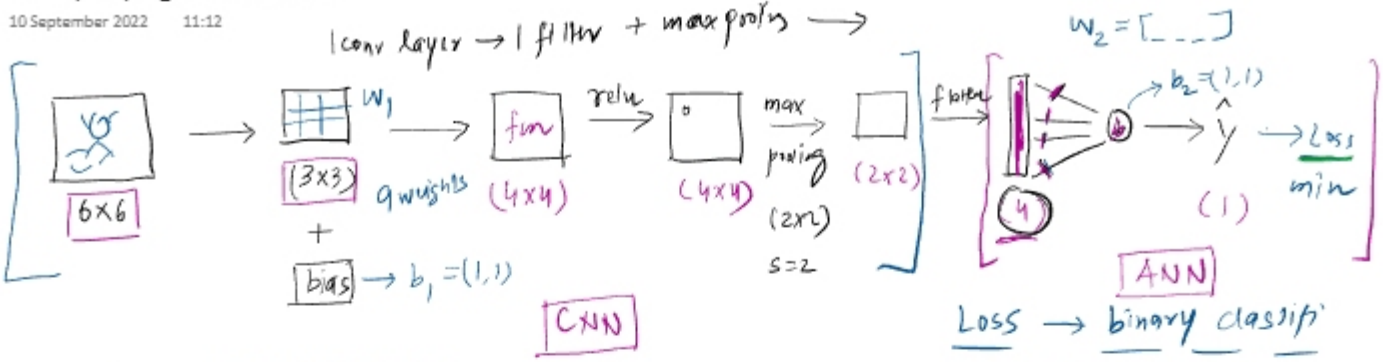
learnable parameters 1400

1350 wts
+50
1400



Backpropagation in CNN

10 September 2022 11:12



Loss → binary classification

$$L = -y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)$$

$A_2 = \hat{y}$

Trainable Parameters

$W_1 = (3,3)$ $W_2 = (1,4)$

$b_1 = (1,1)$ $b_2 = (1,1)$

$= 15$ trainable parameters

Logical Flow

Forward Prop

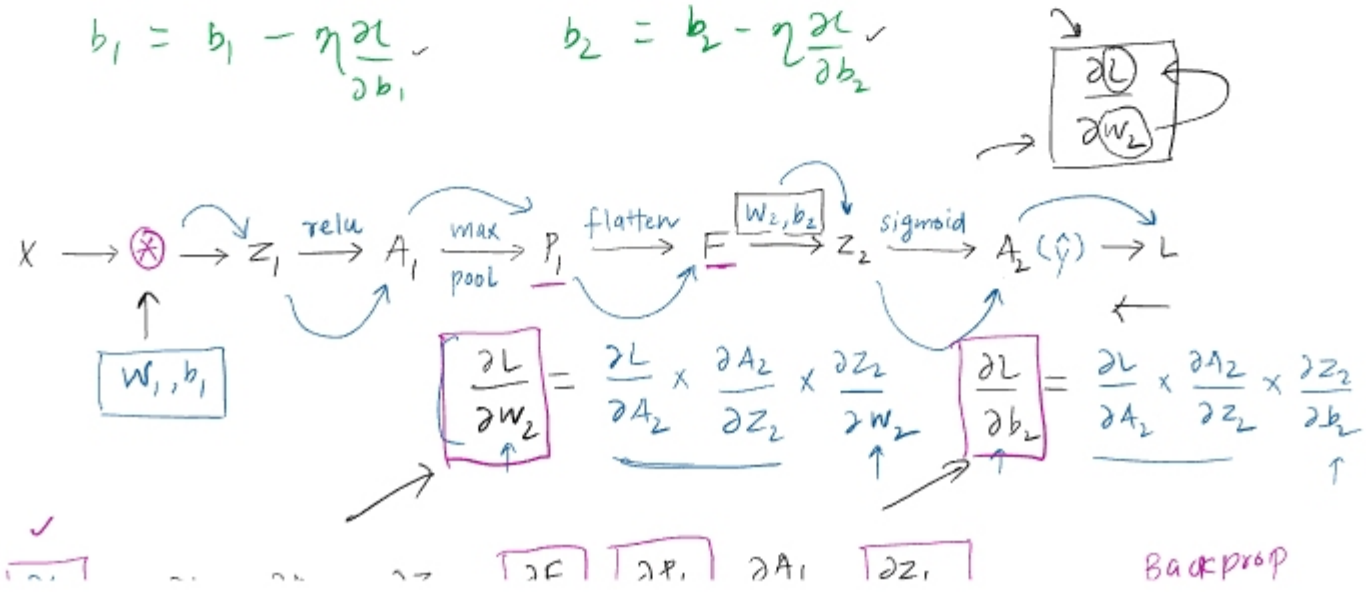
$$Z_1 = Conv(X, W_1) + b_1$$
$$A_1 = relu(Z_1)$$
$$P_1 = maxpool(A_1)$$
$$F = flatten(P_1)$$
$$Z_2 = W_2 F + b_2$$
$$A_2 = \sigma(Z_2)$$

Gradient Descent

$$W_1 = W_1 - \eta \frac{\partial L}{\partial W_1} \quad W_2 = W_2 - \eta \frac{\partial L}{\partial W_2}$$

$$b_1 = b_1 - \eta \frac{\partial L}{\partial b_1} \quad b_2 = b_2 - \eta \frac{\partial L}{\partial b_2}$$

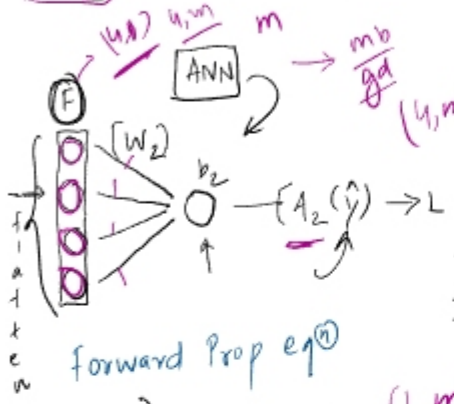
Loss is minimized



$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial A_2} \times \frac{\partial A_2}{\partial Z_2} \times \frac{\partial Z_2}{\partial F} \times \frac{\partial F}{\partial P_1} \times \frac{\partial P_1}{\partial A_1} \times \frac{\partial A_1}{\partial Z_1} \times \frac{\partial Z_1}{\partial w_1}$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial A_2} \times \frac{\partial A_2}{\partial Z_2} \times \frac{\partial Z_2}{\partial F} \times \frac{\partial F}{\partial P_1} \times \frac{\partial P_1}{\partial A_1} \times \frac{\partial A_1}{\partial Z_1} \times \frac{\partial Z_1}{\partial b_1}$$

Backprop
- → Convolution
- → Flatten
- → Max pooling



$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial A_2} \frac{\partial A_2}{\partial Z_2} \frac{\partial Z_2}{\partial w_2} \qquad \frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial A_2} \frac{\partial A_2}{\partial Z_2} \frac{\partial Z_2}{\partial b_2}$$

Forward Prop eqn

$$\begin{cases} Z_2 = W_2 F + b_2 \\ A_2 = \sigma(Z_2) \end{cases}$$

$$\frac{\partial L}{\partial a_2} = \frac{\partial}{\partial a_2}\left[-y_i \log(a_2) - (1-y_i)\log(1-a_2)\right]$$

$$= -\frac{y_i}{a_2} + \frac{(1-y_i)}{(1-a_2)} = \frac{-y_i(1-a_2) + a_2(1-y_i)}{a_2(1-a_2)}$$

(1,m) (1,m)

$$\frac{\partial L}{\partial a_2} = \frac{-y_i + y_i a_2 + a_2 - a_2 y_i}{a_2(1-a_2)} = \frac{(a_2 - y_i)}{a_2(1-a_2)}$$

$$\left[\frac{\partial A_2}{\partial Z_2} = \sigma(Z_2)[1-\sigma(Z_2)] = a_2[1-a_2]\right]$$

$$\boxed{\frac{\partial Z_2}{\partial W_2} = F} \qquad \frac{\partial Z_2}{\partial b_2} = 1$$

W_2 update

Shape =

$$\boxed{\frac{\partial L}{\partial W_2}} = \frac{(a_2 - y_i)}{a_2(1-a_2)} \times a_2(1-a_2) \times F = (a_2 - y_i) F = (A_2 - Y) F^T$$

(1,4) (4,1)

(1,1) (1,1)

(1,1) (1,4)

(1,4)

$$\frac{\partial L}{\partial b_2} = \frac{(a_2 - y_i)}{a_2(1-a_2)} a_2(1-a_2) \times 1 = (A_2 - Y)$$

m images

$$\boxed{\frac{\partial L}{\partial W_2} = (A_2 - Y) F^T} \qquad \boxed{\frac{\partial L}{\partial b_2} = (A_2 - Y)}$$

$\frac{\partial L}{\partial w_2} = (A_2 - Y) A_1^T$ | $\frac{\partial L}{\partial b_2} =$

$(1,m) - (1,m)$

$(1,m)\ (m,n) \rightarrow (1,n)$

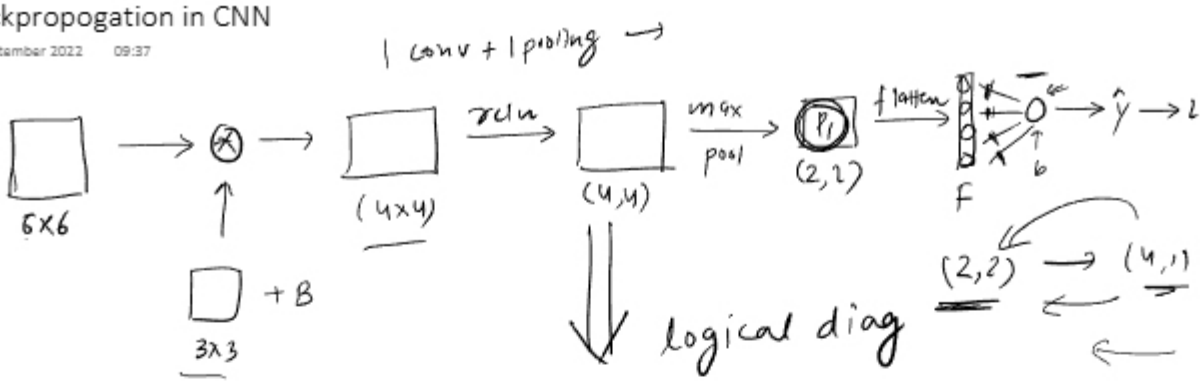
1 image

batch of image

$w_2 \rightarrow (1,n)$

Backpropogation in CNN

15 September 2022 09:37



Forward Prop

$Z_1 = conv(X, W_1) + b_1$

$A_1 = relu(Z_1)$

$P_1 = maxpool(A_1)$

$F = flatten(P_1)$

flatten

$Z_2 = W_2 F + b_2$

$A_2 = \sigma(Z_2)$

$L = \frac{1}{m} \sum_{i=1}^{m} [-y_i \log(A_2) - (1-y_i)\log(1-A_2)]$

$(A_2 - y)$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial A_2} \times \frac{\partial A_2}{\partial Z_2} \times \frac{\partial Z_2}{\partial F} \times \frac{\partial F}{\partial P_1} \times \frac{\partial P_1}{\partial A_1} \times \frac{\partial A_1}{\partial Z_1} \times \frac{\partial Z_1}{\partial W_1}$$

5

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial A_2} \times \frac{\partial A_2}{\partial Z_2} \times \frac{\partial Z_2}{\partial F} \times \frac{\partial F}{\partial P_1} \times \frac{\partial P_1}{\partial A_1} \times \frac{\partial A_1}{\partial Z_1} \times \frac{\partial Z_1}{\partial b_1}$$

1 2 3 4 6

reshape(P_1.shape)

6 derivatives

$\left[\frac{\partial Z_2}{\partial F}\right] = W_2 \rightarrow$

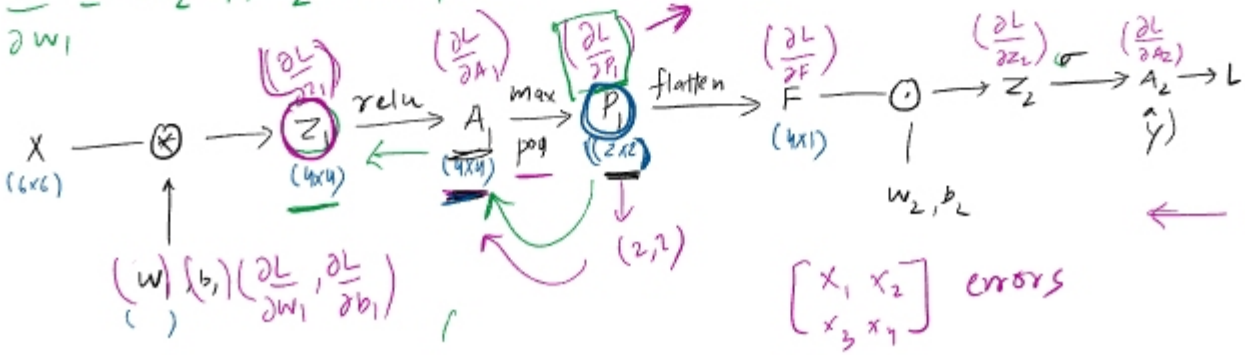
Shape ? $\rightarrow$ (F)

$\frac{\partial F}{\partial P_1}$ no trainable parameters

$conv(X, \frac{\partial L}{\partial Z_1})$

$\frac{\partial L}{\partial W_1} = (A_2 - y) W_2 .\text{reshape}(P_1.\text{shape})$

$\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$

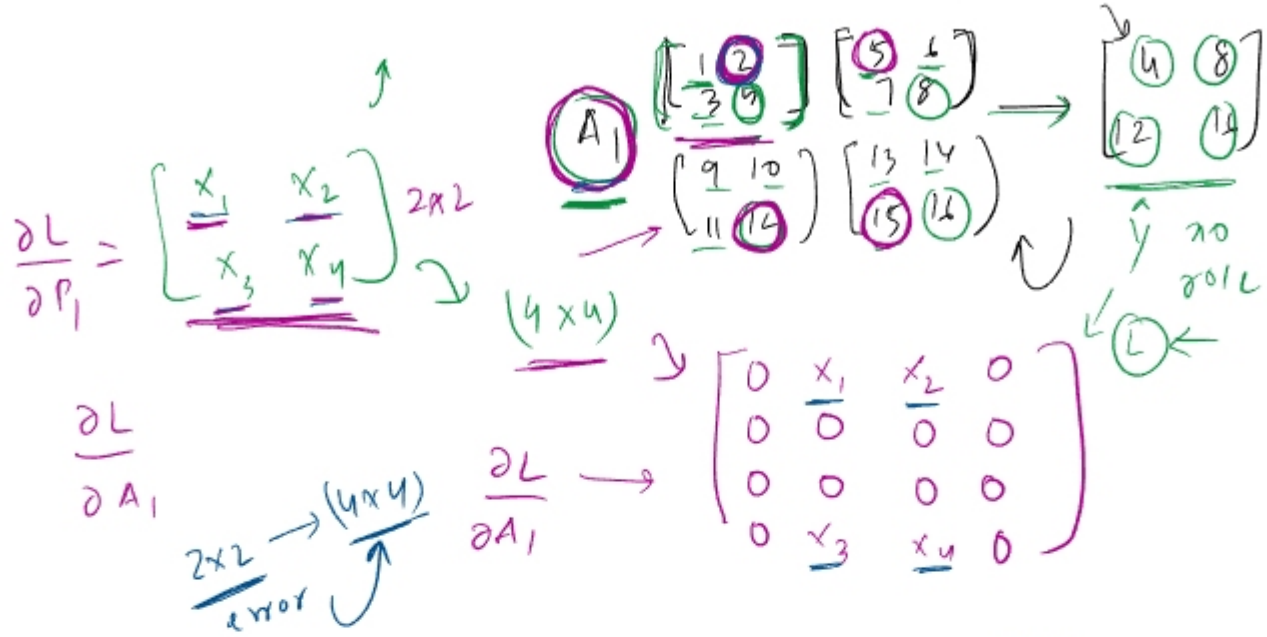


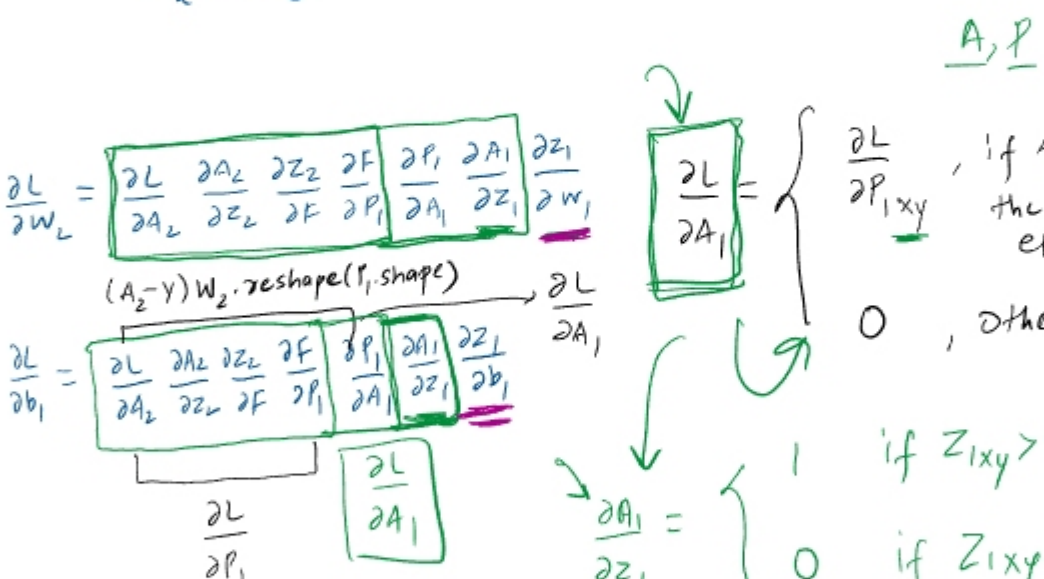
$\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$ errors

$\frac{\partial L}{\partial A_1} = (4,4)$

$\frac{\partial L}{\partial A_1} = (4,4)$

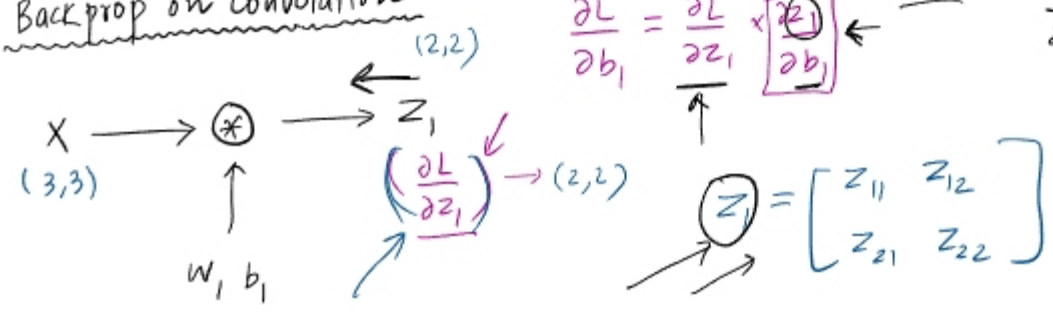
Flatten → no trainable parameters





Convolution → Backprop → max pooling, Flatten

Backprop on Convolution



$$X = \begin{bmatrix} \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} & \begin{matrix} x_{13} \\ x_{23} \end{matrix} \\ \begin{matrix} x_{31} & x_{32} \end{matrix} & x_{33} \end{bmatrix} \circledast W_1 = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$$

b_1

$$\boxed{\frac{\partial L}{\partial b_1}} = \frac{\partial L}{\partial z_1} \times \frac{\partial z_1}{\partial b_1} = \boxed{\frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial b_1} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial b_1} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial b_1} + \frac{\partial L}{\partial z_{22}} \times \frac{\partial z_{22}}{\partial b_1}}$$

$$= \left(\frac{\partial L}{\partial z_{11}} + \frac{\partial L}{\partial z_{12}} + \frac{\partial L}{\partial z_{21}} + \frac{\partial L}{\partial z_{22}} \right) = \text{sum}\left(\frac{\partial L}{\partial z_1} \right)$$

$$\boxed{\frac{\partial L}{\partial b_1}} = \text{sum}\left(\frac{\partial L}{\partial z_1} \right) \rightarrow \text{scalar}$$

bias

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \quad W_1 = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} + b$$

$X \rightarrow (*) \rightarrow Z_1$

(3×3)

$W_1\ b_1$

(2×2)

$\left(\frac{\partial L}{\partial z_1}\right)$

$$\frac{\partial L}{\partial W_1} = \begin{bmatrix} \frac{\partial L}{\partial w_{11}} & \frac{\partial L}{\partial w_{12}} \\ \frac{\partial L}{\partial w_{21}} & \frac{\partial L}{\partial w_{22}} \end{bmatrix} \qquad \frac{\partial L}{\partial z_1} = \begin{bmatrix} \frac{\partial L}{\partial z_{11}} & \frac{\partial L}{\partial z_{12}} \\ \frac{\partial L}{\partial z_{21}} & \frac{\partial L}{\partial z_{22}} \end{bmatrix}$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial z_1} \times \frac{\partial z_1}{\partial W_1}$$

$$\frac{\partial L}{\partial w_{11}} = \frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial w_{11}} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial w_{11}} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial w_{11}} + \frac{\partial L}{\partial z_{22}} \times \frac{\partial z_{22}}{\partial w_{11}}$$

$$\frac{\partial L}{\partial w_{12}} = \frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial w_{12}} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial w_{12}} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial w_{12}} + \frac{\partial L}{\partial z_{22}} \times \frac{\partial z_{22}}{\partial w_{12}}$$

$$\frac{\partial L}{\partial w_{21}} = \frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial w_{21}} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial w_{21}} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial w_{21}} + \frac{\partial L}{\partial z_{22}} \times \frac{\partial z_{22}}{\partial w_{21}}$$

$$\frac{\partial L}{\partial w_{21}} = \frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial w_{21}} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial w_{21}} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial w_{21}} + \frac{\partial L}{\partial z_{22}} \times \frac{\partial z_{22}}{\partial w_{21}}$$

$$\frac{\partial L}{\partial w_{22}} = \frac{\partial L}{\partial z_{11}} \times \frac{\partial z_{11}}{\partial w_{22}} + \frac{\partial L}{\partial z_{12}} \times \frac{\partial z_{12}}{\partial w_{24}} + \frac{\partial L}{\partial z_{21}} \times \frac{\partial z_{21}}{\partial w_{22}} + \frac{\partial L}{\partial z_{32}} \times \frac{\partial z_{22}}{\partial w_{22}}$$

$$\frac{\partial L}{\partial w_{11}} = \frac{\partial L}{\partial z_{11}} x_{11} + \frac{\partial L}{\partial z_{12}} x_{12} + \frac{\partial L}{\partial z_{21}} x_{21} + \frac{\partial L}{\partial z_{22}} x_{22}$$

$$\frac{\partial L}{\partial w_{12}} = \frac{\partial L}{\partial z_{11}} x_{12} + \frac{\partial L}{\partial z_{12}} x_{13} + \frac{\partial L}{\partial z_{21}} x_{22} + \frac{\partial L}{\partial z_{22}} x_{23}$$

$$\frac{\partial L}{\partial w_{21}} = \frac{\partial L}{\partial z_{11}} x_{21} + \frac{\partial L}{\partial z_{12}} x_{22} + \frac{\partial L}{\partial z_{21}} x_{31} + \frac{\partial L}{\partial z_{22}} x_{32}$$

$$\frac{\partial L}{\partial w_{22}} = \frac{\partial L}{\partial z_{11}} x_{22} + \frac{\partial L}{\partial z_{12}} x_{23} + \frac{\partial L}{\partial z_{21}} x_{32} + \frac{\partial L}{\partial z_{22}} x_{33}$$

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \qquad \frac{\partial L}{\partial Z_1} = \begin{bmatrix} \frac{\partial L}{\partial z_{11}} & \frac{\partial L}{\partial z_{12}} \\ \frac{\partial L}{\partial z_{21}} & \frac{\partial L}{\partial z_{22}} \end{bmatrix}$$

$$\frac{\partial L}{\partial W_1} = \text{conv}\left(X, \frac{\partial L}{\partial Z_1}\right)$$

$$\frac{\partial L}{\partial W_1} = \text{conv}\left(X, \frac{\partial L}{\partial Z_1}\right)$$

$$\frac{\partial L}{\partial b_1} = \text{sum}\left(\frac{\partial L}{\partial Z_1}\right)$$

Why use Pretrained models?

03 October 2022 12:52

1) Data hungry → Image data → labelled data → cat dog → google

pinterest

labelling

10000 photos

manually

2) Time → model → training

Pretrained model

ImageNET Dataset

03 October 2022 12:36

Visual Database of images (Why What and How)

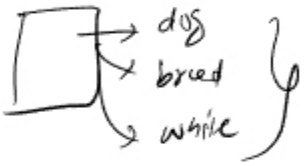
Why

2006 → Fei fei Li → model and algorithms

1.4 million images → 1.4 crore image → 20,000 categories → labelled

1 million images → to label → bounding box

↳ object local



How → crowd sourcing  → Amazon mechanical Turk

↳ Dataset → Deep learning

ImageNET challenge

ILSVRC

03 October 2022 12:38

Imagenet challenge → 2010 → Dataset → subset → 1 million → 10 → 20,000 classes → 1000 classes

1 crore

100 image

Machine learning → feature extraction → SIFT / HOG

2010 → 28%.

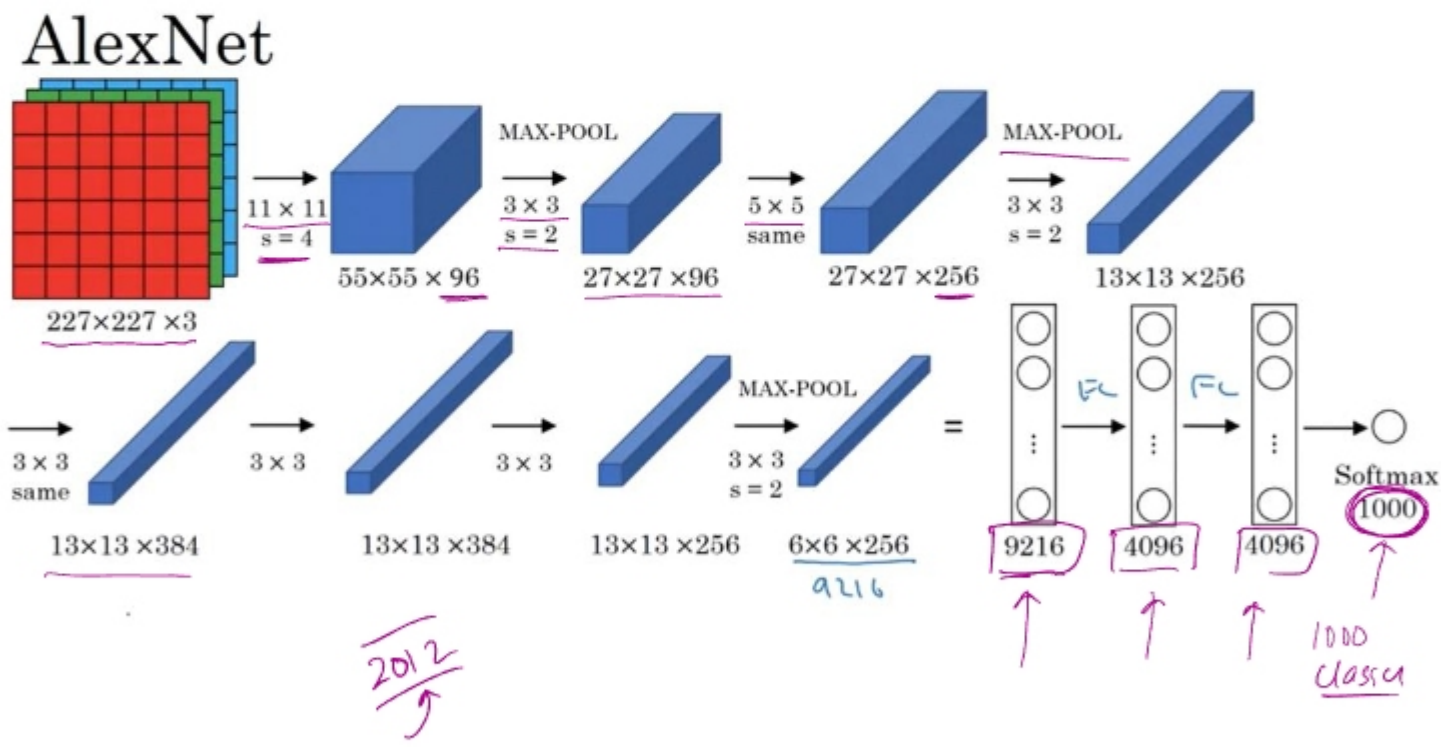
2011 → 25%.

2012 → 16%. → CNN

2013 →

2012 → DL → AlexNET → CNN → relu

Deep GPU

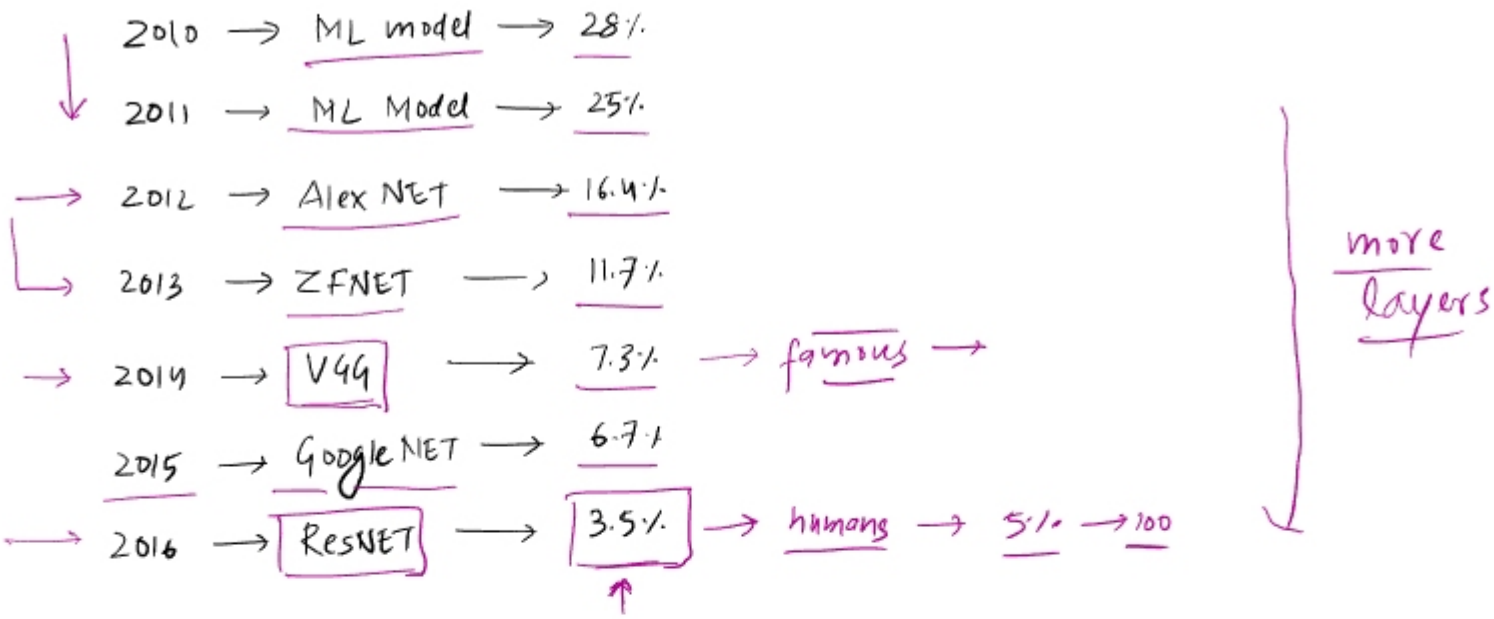


[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

Andrew Ng

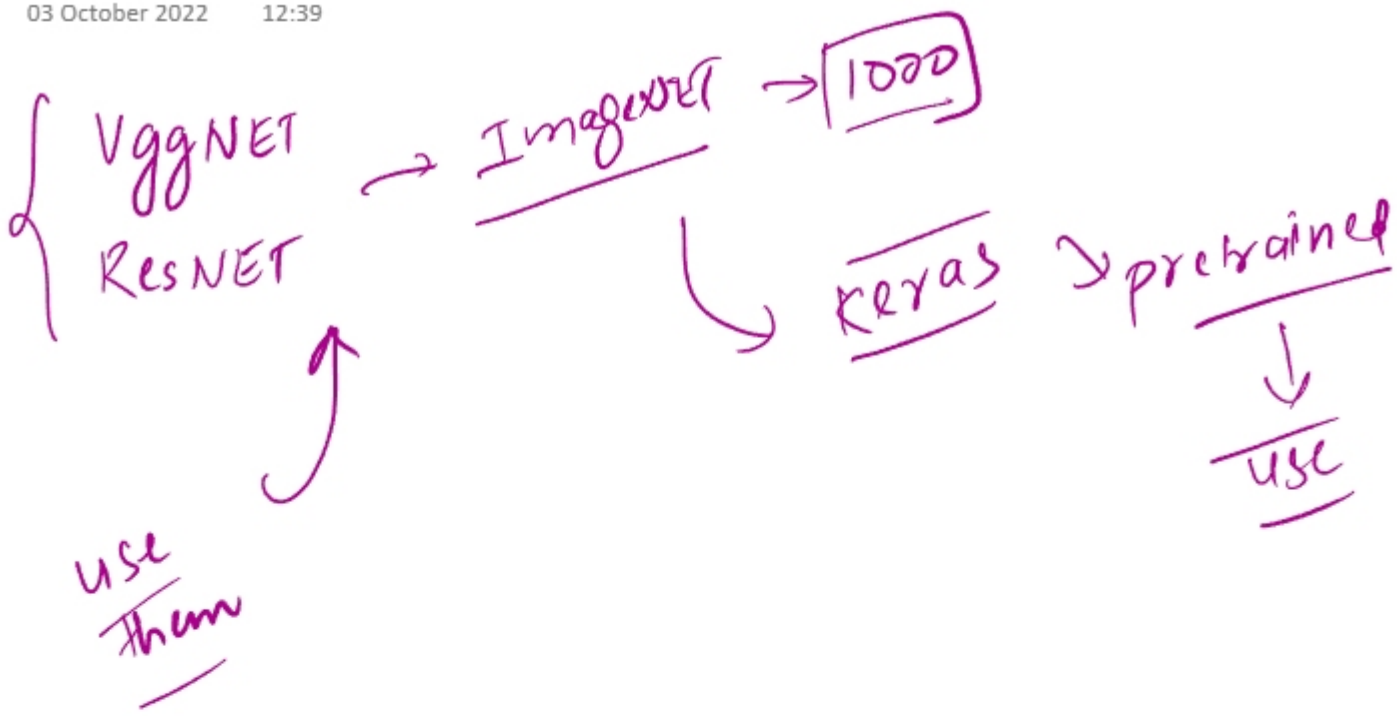
Famous Architectures

03 October 2022 12:39



Idea of Pretrained Models

03 October 2022 12:39



Keras Demo

03 October 2022 12:40

Problem with training your own model

10 October 2022 10:48

1) Data hungry → labelled → 10,000 → google

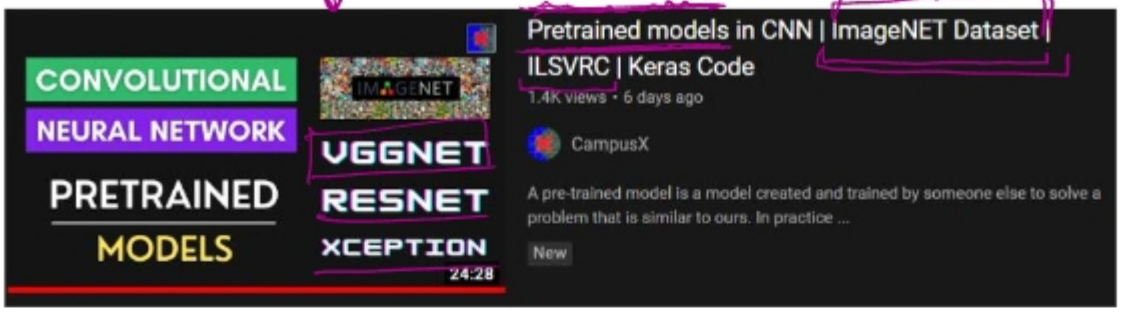
↓

cat/dog → manual labour

2) lot of time → days/ weeks/

Using Pretrained Models

10 October 2022 10:48



Transfer Learning

10 October 2022 10:49

Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.