04-Feb-2025

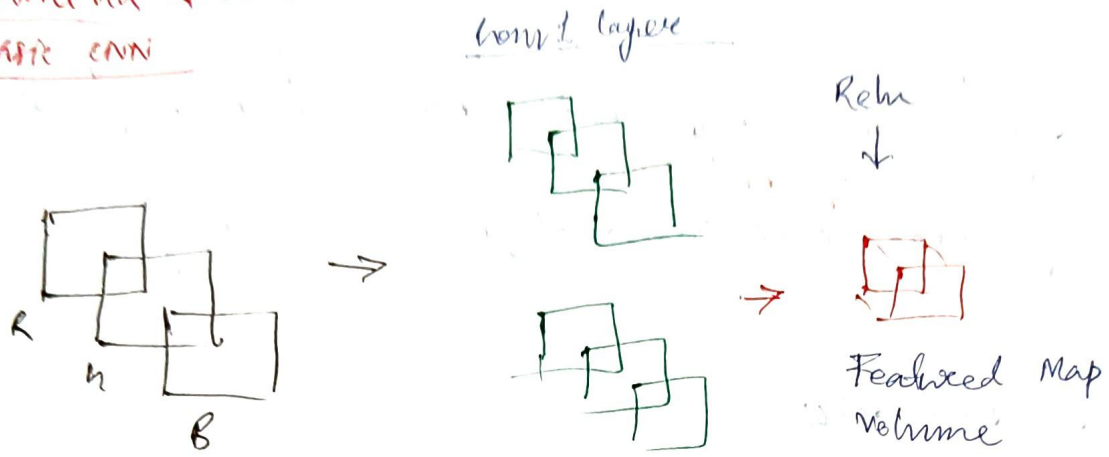
> Vanilla architecture - there is no convolution

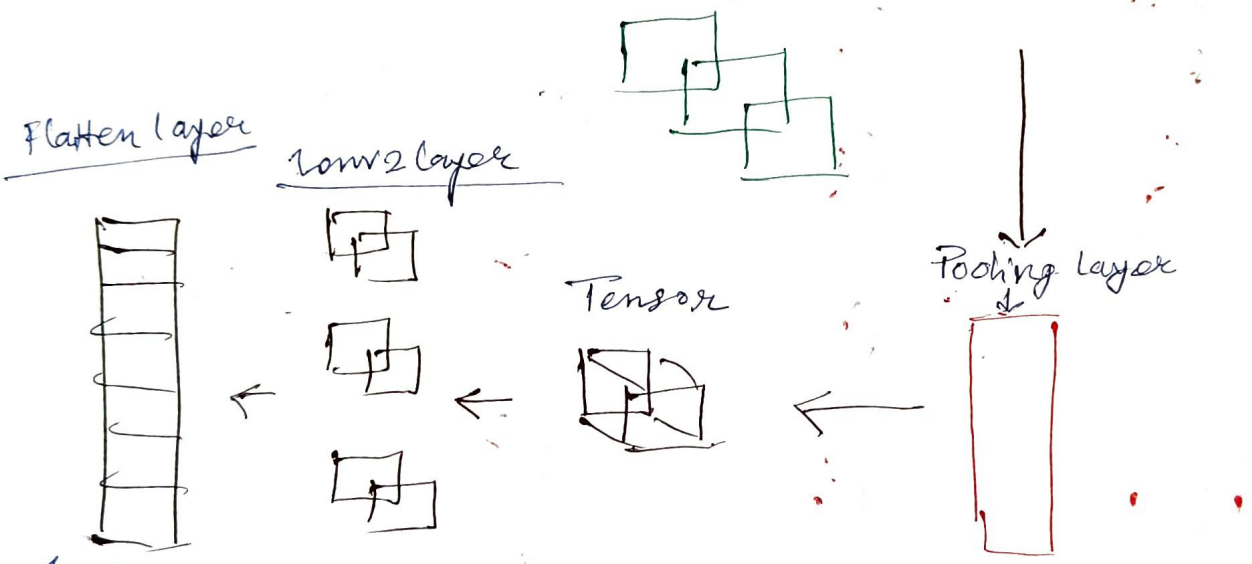
(Gray scale) mnist → 28 X 28 X 1 → channel

(RGB) Cifar → 32, X 32 X 3 → channel

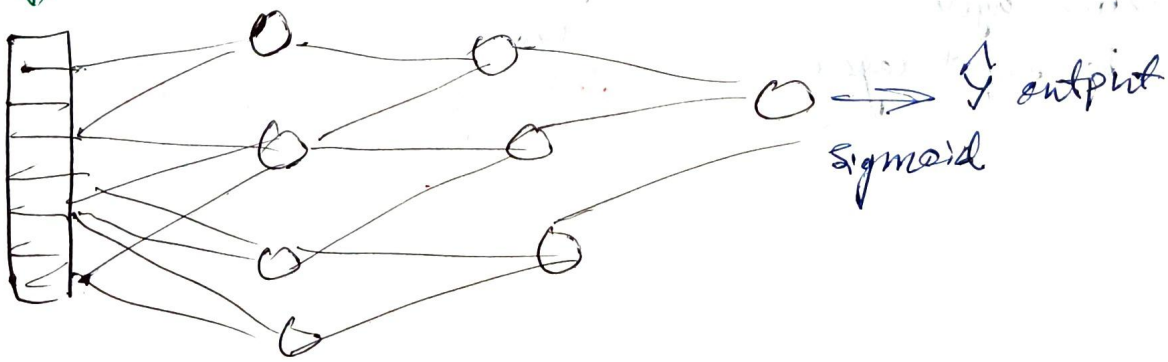
Vanilla Architecture

Basic CNN





ANN



ImageNet Competition

→ Researchers has used this dataset to build different CNN architecture.

> People changes the architecture by
1. convolution layer
2. filter size
3. stride
4. Padding
5. Fully connected layers (ANN)
6. activation function
7. Dropout
8. Batch Normalization

History of CNN

1. Lenet -5
2. Alexnet
3. VGG (VGG16 or VGG19)
4. Inception (GoogleNet)
5. ResNet (Microsoft)

Note: there are many CNN architecture available.

1. Lenet -5 (1st application of computer vision)

> It's a classification architecture from a paper called "Gradient based learning for Document classification."

> Handwritten digit classification / character > VS Many

> Postal Serve - developed this architecture to recognize pincode.

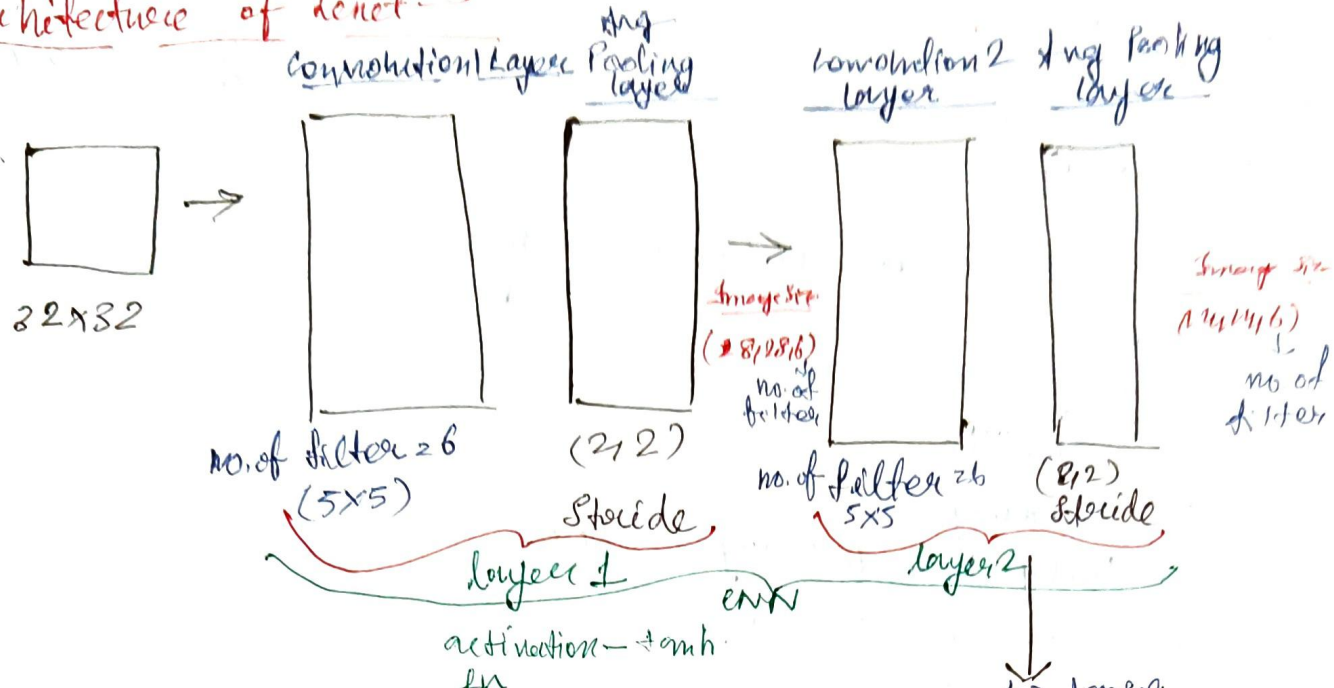
[1245] - Pincode

> Published in proceeding of the IEEE (1998)

Input Size - 32 × 32 × 1 (It will work for this input only)

> This papers was published by Yann Lecun, Leon Bottou and, Yoshua Bangio and Patrick Haffner.

Architecture of Lenet-5



$$\text{Image dimension} = \frac{(n+2p-f)+1}{s}$$

1st conv layer

$$= \frac{(32+2\times0-5)+1}{1}$$

$$= 28$$

Padding, $p=0$

(Don't have stride in 1st layer) $s=1$

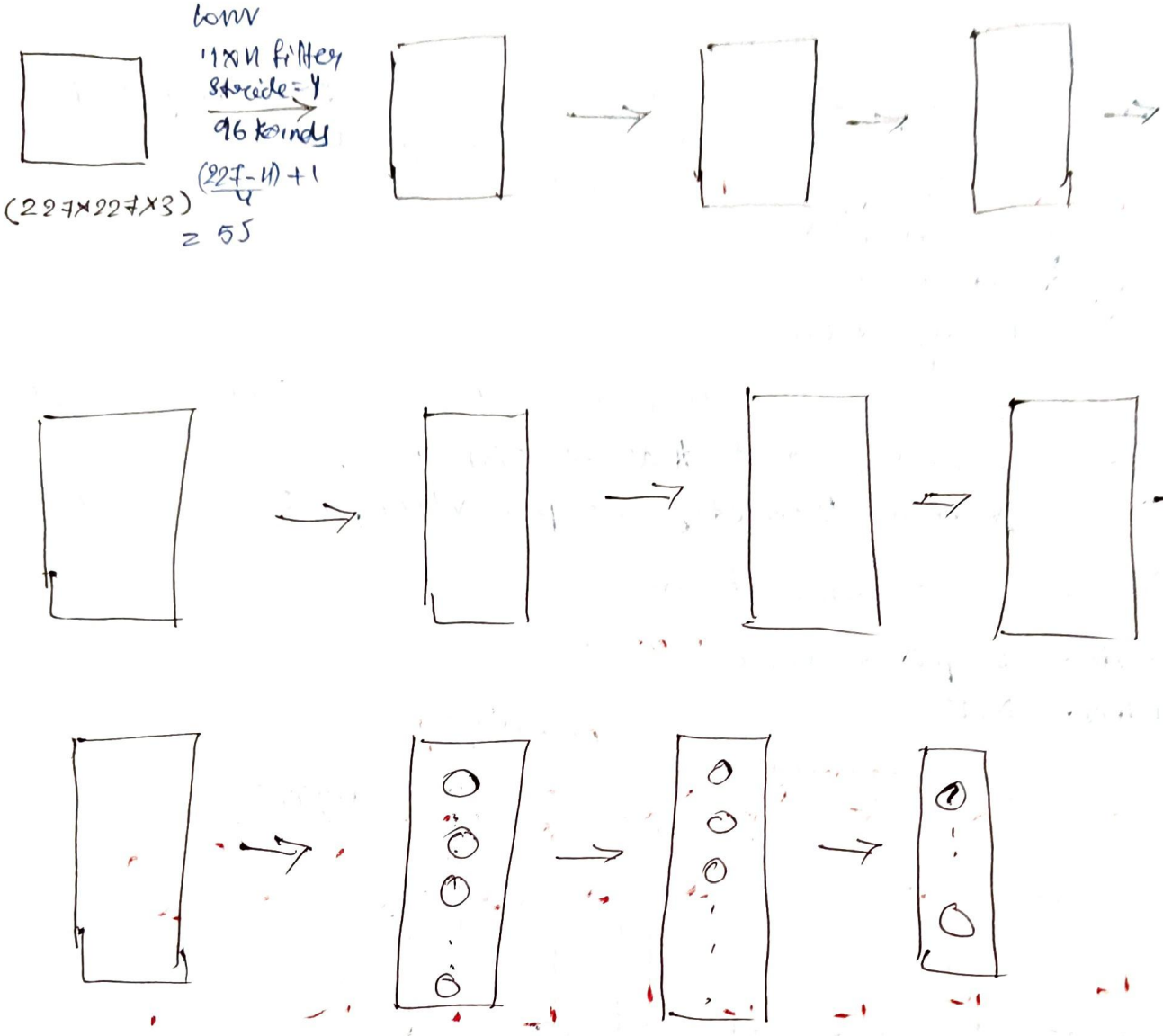
Image dimension $= (28\times28\times6)$

filter

> RNN Performance on RNN image model be very poor because no. of layer is very less.

2(5-5)+1 = 54

② AlexNet



05-Feb-2022

⑧ VGG-16 (16 layers)

> It's a classification architecture/model, from very deep convolutional networks for large-scale image recognition.

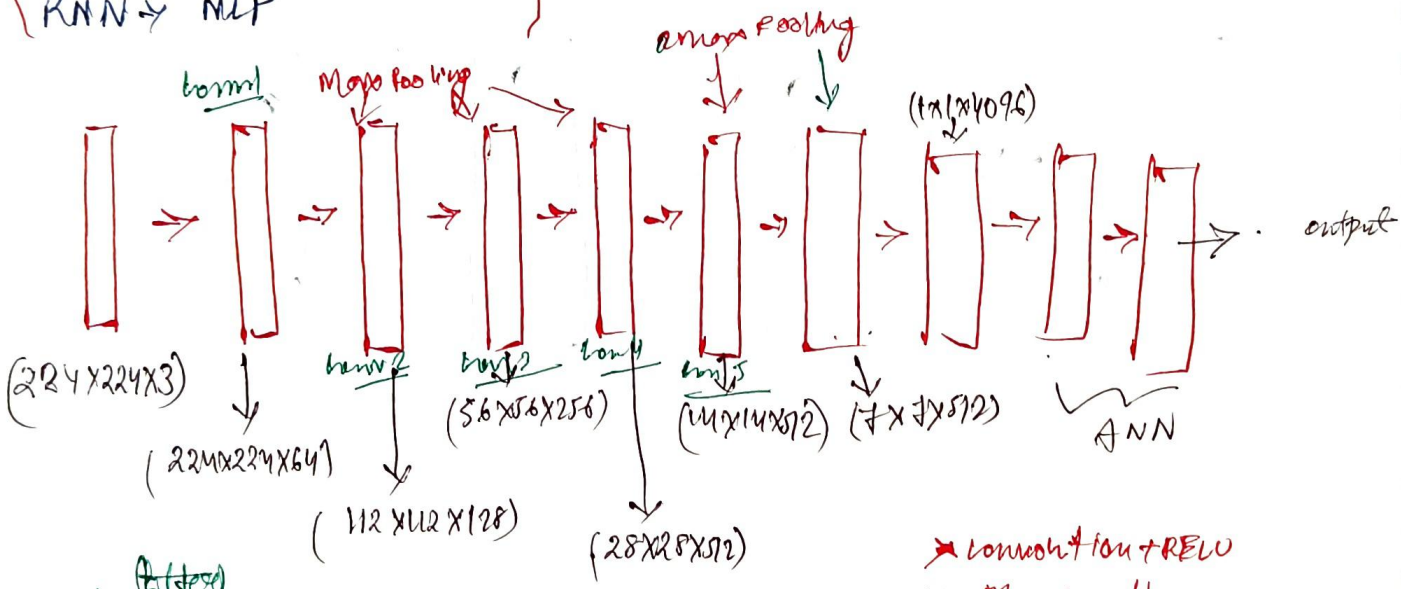
Application of object computer vision

① Image classification
② Object detection
③ Face Recognition

> It is published as a conference paper at ICLR in 2015, by Karen simonyan and Andrew Zisserman.
Visual Geometry Group – VGG → from OXFORD

> Video is a sequence of ~~ex~~ images.

{ CNN → computer vision } use cases
{ RNN → NLP }



> No. of ~~filter~~ Kernel is getting increased.
no. of filter

* Convolution + RELU
* Max pooling
* Fully connected + RELU

> All convolution layers has filter size = 3x3. But in Alexnet filter size is different for different layer. In VGG, all stride is 1 and padding is is same.

> In max pooling layers, pooling size is 2x2 and stride is 2x2.

(4) VGG-19

- It has 19 layers.

⇒ So practical for all the architecture.

& Why we use pretrained model?

> Biggest challenge in DL is availability of data. DL are techniques are data hungry.

> Always labeled the data

> Model training time.

→ To overcome the above time, we use pretrained model.

Imagenet Dataset

> It's a visual representation of image.

Why?

In 2006: main focus Model and algo building to get better accuracy.

> Garbage in, garbage out.

Bad data → good model → Bad Result

good data → Bad model → good Result

> After this one researcher thought to build big good database. He collected around 1.4 M images and labelled the images and their features like colour, tail etc

> He used crowd ~~sourcing & search~~ sourcing to collect the data. Using captcha technique

> collected data is known as ImageNet. It is the largest dataset of the image classification in the world as of now.

ILSVRC / ImageNet challenge

> ImageNet Large Scale Recognition challenge (ILSVRC)

1.4 M data - 20,000 class

1 million, 1000 class } ImageNet competition

Machine learning
Human ← 5% error

2010 → 28% error rate } ML model
2011 → 25% error rate

Finally 2012 → Reduce the error to 16% } DL

Geoffry Hinton, He propose deep learning based architecture called AlexNet and trained on GPU

In 2013 → Error rate was 11.7%

In 2014 → VGG → 7.3% Error rate

In 2015 → Google Net → 6.7% Error rate

In 2016 → ResNet → 3.5% error rate.

} Deep learning

Exception

Image classification : Why to build from scratch?

> we will use pre trained model.

Eg to build a car, we use same technique to build wheel. Because invention has been done already.

Transfer Learning

> use CNN part and drop ANN part.

↳ feature extraction

> If we train convolutional layers, then it is called fine tuned.

↓ use when dataset is completely new.

11-Feb-2022

~~Computer Vision~~

Filter | Kernel | Feature Extractions
— Use is to extract features

Eg a B C d e D f F How to decide whether it is name or not? using filters?

Pixels
> Smallest component
> After grouping of pixel, we can infer something.
> Only based on pixel, we cannot decide anything.

Pixels → Edges/Gradients → Patterns → Textures → Parts of object → (eye, leg etc) → Image object → Scene ← Image/Video (Series of frames)

> 5 layers to get scene

Image RGB ≥ h×w×3

Purpose of filter

Small to Big
{Pixel to Scene/object}

Big to Small

Image → object → Parts of object → Textures → Patterns → Edges → Pixels

convolutional → This can be anything

200 × 200 × 3 (RGB)

200 × 200 × 12
↓
12 different channel / feature map
↓
Group of channels

Image
colour

Feature maps
No colour

> Decision will be made on shape not on colour.

12 filters → input
200 × 200 × 12 → output

Why 12 filters?
> If decide Kernel then no. of kernel
Which is best?
> Why 3×3 best?
→ It has less parameters.

1×1
3×3
5×5
7×7
11×11

Eg
① 36 × 36 image ← (input)

3×3 ×12
(filters) 12 times

> $(n-2p-f)+1 = (36-0-3)+1 = 34$ (p=0, filter)

output = 34×34×12

> If I use 3×3, 2 pixel will be losed.

> 3×3 = 9 (2)
Params = 9+9 = 18
less information loss
with 2 3×3 filters compare to single 5×5 filter.

② input ⇒ 36 × 36 × ?
filter ← 5×5×12 ⇒ 12 channel
output = 32×32×12

$(n-2p-f)+1 = (36-0-5+1) = 32$

12 → no. of filters, It is a hyperparameter.

5×5 = 25 (single)
Params = 25.
more information loss with single filter

Eg:-

360 × 360 × (?) → 3 × 3 × (?) = 9 → 358 × 358 × (?)

Single convolution operation

360 × 360 × 3

5 × 5 × (?) (= 25) Params

356 × 356 × (1)

358 × 358 × (?) → 3 × 3 × (?) = 9 → 356 × 356 × (?)

Second convolution operation

3×3

Params 9 + 9 = 18

> So, 3×3 is better than 5×5 based on information loss and parameters.

Information loss

> When we are using 2 filters, information can be stored in feature map. But in 5×5 we directly loss information.

Striding

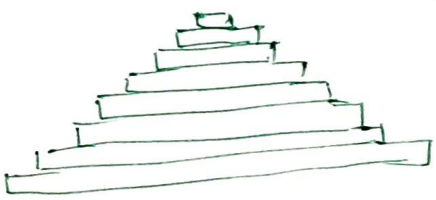
> Jump

Padding

> Looked up names
> Black pixels
> Adding padding so that we can do more convolutional.

Eg Input h × w × 3

output 1 × 1 × 256 (Before flatten) (We need in this shape)

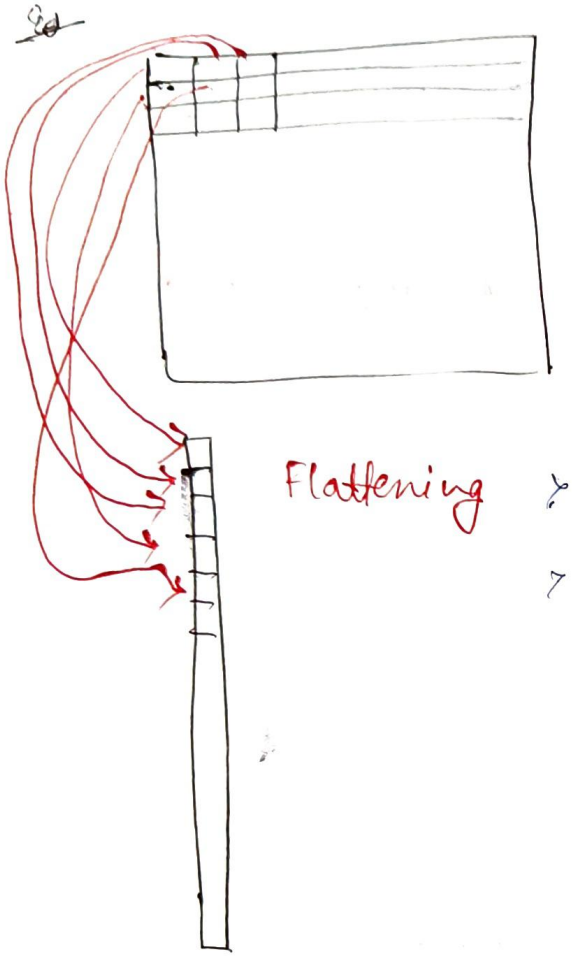


> Convolutional means moving of the Kernel on top of the image to extract features.

ANN

→ It learn by learning more complex features.

-3 · 5 → ~~Distinguish~~ identification can't be done for complex image.



→ decay all the pixels medically

Flattening

> If we directly flatten then it will not hold shape, it is of no use.
> 1st we have to perform convolutional operation to hold the shape.

Q can we use flatten in b/w?

~~Ans~~ Dense Dense Flatten Dense.

→ Yes it can be.

> Using too much dense is bad,

Dense
Dense
Dropout
Dense
Flatten
Dense (10) o/p layer.
Softmax

CNN Architecture

1. Lenet
2. Alexnet
3. VGG

} All has fully connected layer.

Q. Can we do without fully connected layer?

Convolutional

Input
Conv
Conv
Max Pooling
Conv
Conv
Conv
Conv
Max Pooling
Flatten
FC - Fully connected
Softmax

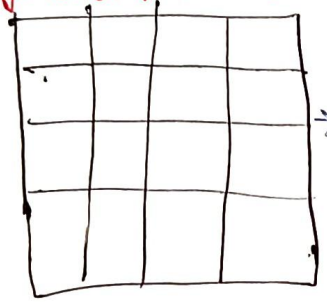
} Shrinking 256×256×3 (image size)

} Closing 1×1×(?) (closing size)

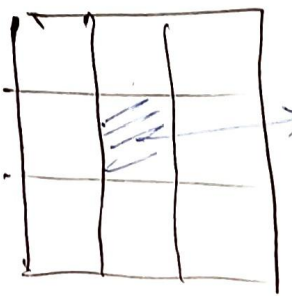
→ We can have only one flatten.
→ Flatten disrupts the initial information.

Q. Why odd filter is better?

Eg:



→ There is no middle



→ Middle Pixel (2,2)

Total Pixel = 9

Total Pixel = 16

→ Symmetry issue

↓

One to symmetry due we use odd size filter.

Eg 36×36×3 (image size)
8×8 (filter size)

29×29 (output)

→ Center has is getting focused more.
→ Hardware use also, 3×3 is faster and good.

- All our assumptions in CV is that object is always in the middle
- For translation invariance is better and less reduced by pooling is better.

- We use max pooling to avoid lots of layers.
- More the no. of neuron, more will be the complexity