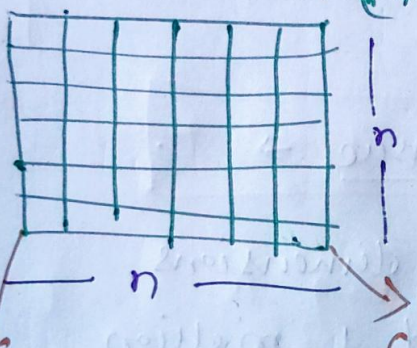
CNN (Convolution Neural Networks)

→ It is used in text / images / Visualization.

→ It is particularly well-suited for image recognition and processing tasks.

→ It is made up of multiple layers.

Image



Convolution means we use some filters to on given image (Gray scale or RGB) and traverse filters to retrieve some features from images.

Gray scale
↓
Black/White
6 x 6
[0-255]
↓
tells about intensity of image

Coloured → {6x6x6}
{ R, G, B
Red
Green
Blue }
[0-255]

* **Convolution** → retrive features from image by applying filters on image

Q Why Convolutions ?

Ans → (1) Spatial invariance or loss features →

→ The Spatial features of a 2D image are lost when it is flattened to a 1D Vector but. Before feeding an image to the hidden layers of an MLP, we must flatten the image matrix to a 1D Vector, this implies that all of the image's 2D information is discarded.

(2) Increase in Parameter Issue →

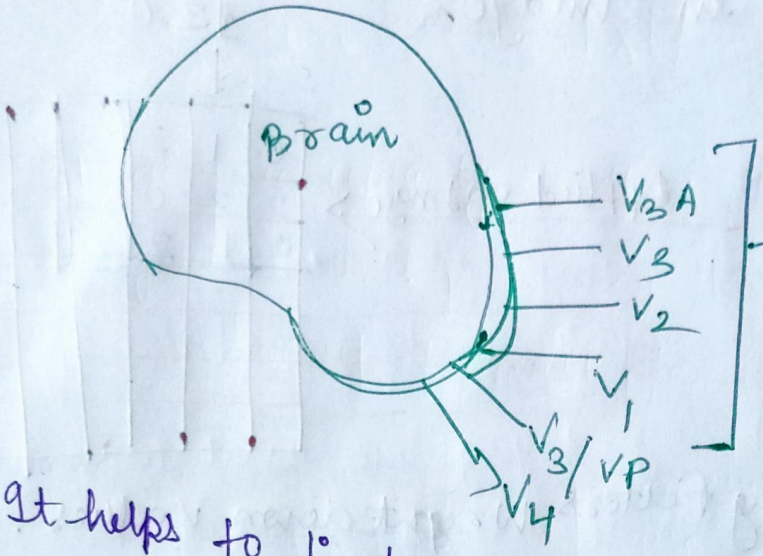
→ If we have an image with dimensions 1000 × 1000, it will yield 1 million Parameters for each node in the first hidden layer. so, flattened image is necessary.

(3) Local Connected Neural Net →

It is used in image recognition to focus on parts of images.

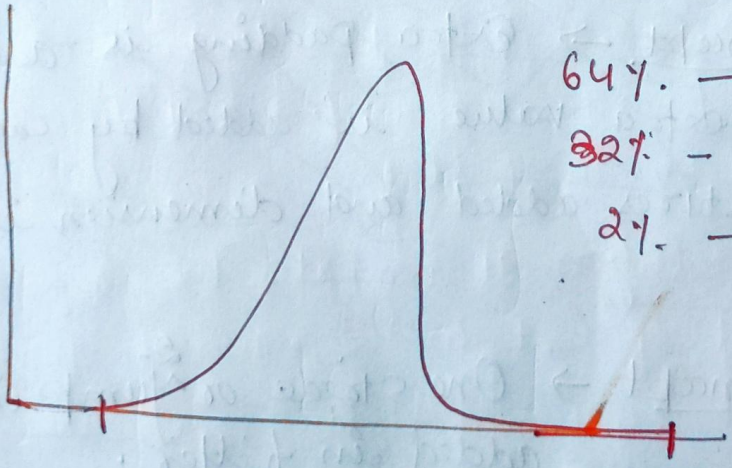
Flatten → used to flatten images to 1D Vector.

* Human Brain Visual Cortex Processing →



→ It helps to find which types of images are given.

* Human Eye Color sensitivity →



Cones are
64% — Red sensitivity
32% — Green sensitive
2% — blue sensitive

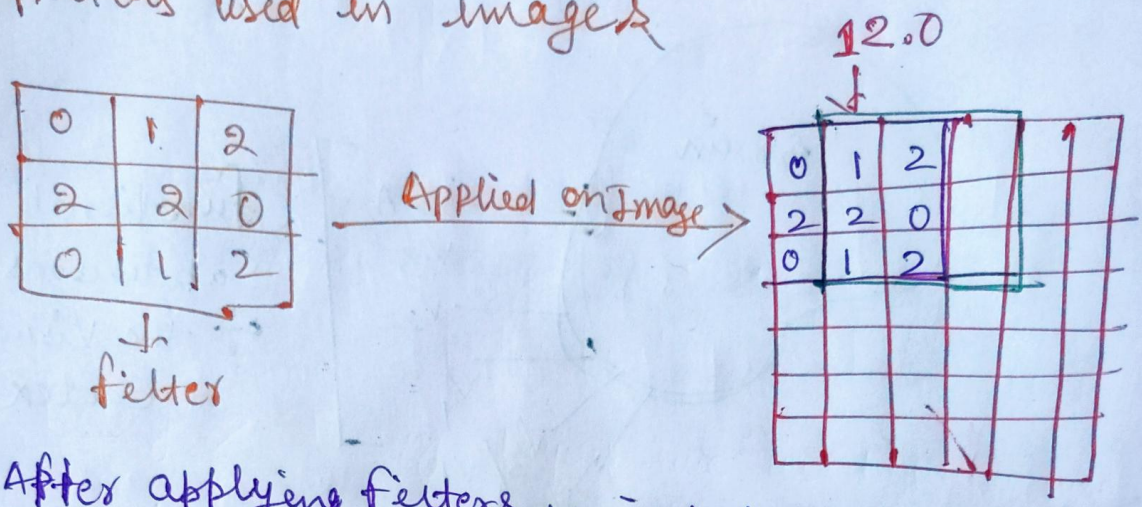
* Visualizing the Process →

(1) Simple Convolution → By using filters select features from images.

→ Gradient used in backpropagation also here.

* Matrix calculation →

Filters used in images



After applying filters write down values.

→ this is dot product value

Filter * Input value = Dot product.

* Padding concept → Extra padding is added so, one extra value is added by convolution so, extra features added and dimension is preserved.

* Stridge concept → One stride or 2-jump is extra added in filter.

so, 2×2 convolution features are preserved.

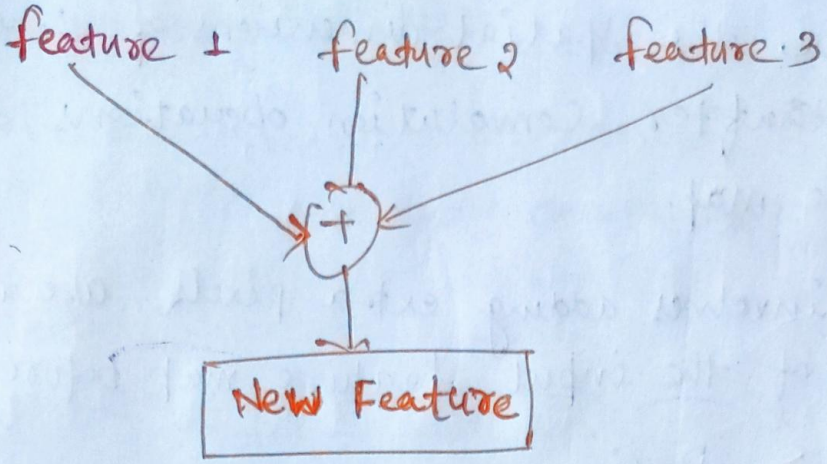
2-jump.

* Feature Accumulation →

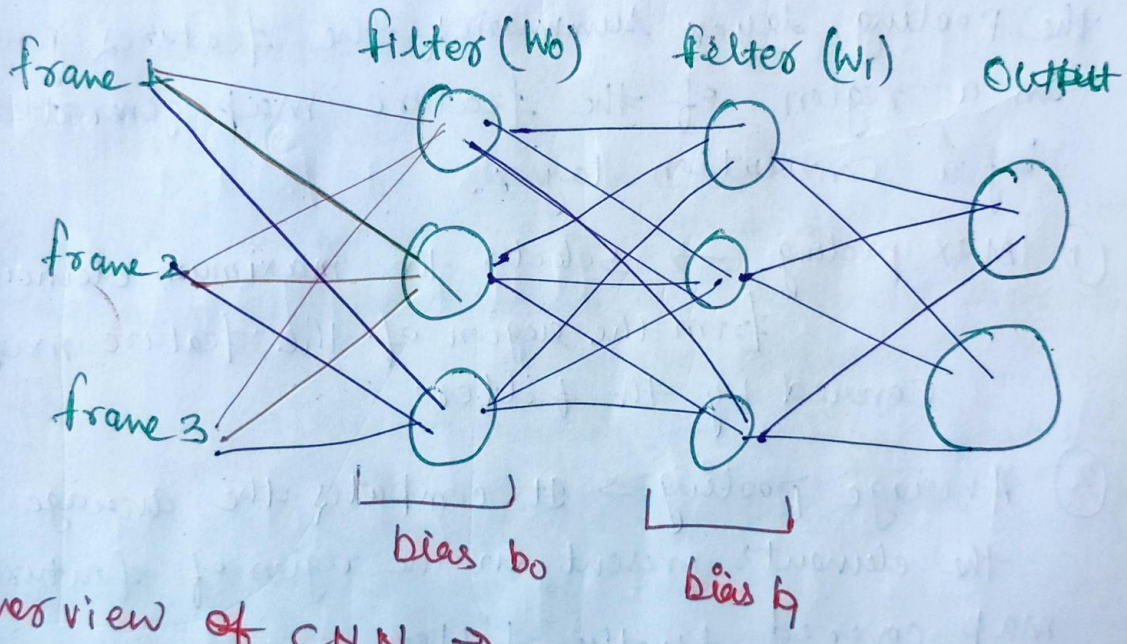
filter 1 → filter 2 → filter 3

feature 1 feature 2 feature 3

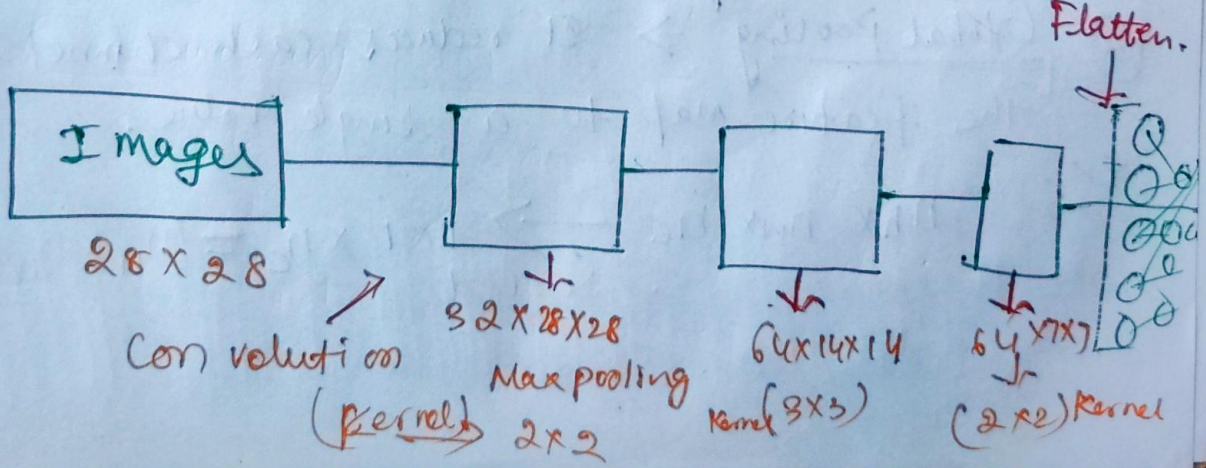
* Feature Aggregation →



Convolution operation ⇒



Overview of CNN →



* **Padding** ⇒ It is a technique used to preserve the spatial dimensions of the input image after convolution operations on a feature map.

- Padding involves adding extra pixels around the border of the input feature map before convolution.

* **Pooling** ⇒ Pooling layers are used to reduce the dimensions of the feature maps.

The Pooling layer summarises the features present in a region of the feature map generated by a convolution layer.

(1) Max pooling → Selects the maximum element from the region of the feature map covered by the filter.

(2) Average Pooling → It computes the average of the elements present in the region of feature map covered by the filter.

(3) **Global Pooling** → It reduces each channel in the feature map to a single value.

Ex →

$$n_h \times n_w \times n_c \xrightarrow[\text{Seleted}]{} 1 \times 1 \times n_c = (n_h \times n_w)$$

Advantages of pooling →

(1) Dimensionality reduction

(2) Translation invariance → position of an object in the image does not affect the classification result.

(3) Feature selection

Disadvantage of pooling ⇒

(1) Information loss

(2) Over-smoothing

(3) Hyperparameter tuning

(3) Stride in CNN →

→ It is a parameter of the neural network's filter that modifies the amount of movement over the image or video.

→ the no. of pixels turning to the input matrix is known as the strides.

no of strides = no. of filters

Important formulas

(1) Input dimensions (IMAGE) ⇒

Width (W) → Width of image

Height (H) → Height of image

channels (C) → the no. of color planes in the input image (1 for Grayscale, 3 for RGB

Convolutional Layer →

Kernel size (K) – Width & Height of the Convolutional Kernel or filter.

Padding (P) → the no. of zeros added to the border of the input image to preserve spatial dimensions.

Stride (S) :- the step size at which the kernel moves across the input image.

No. of filters (N) → no. of convolutional filters applied in the layer.

* **Output Dimensions** →

Width (W_out) : $((W-K+2P)/S)+1$

Height (H_out) : $((H-K+2P)/S)+1$

Channels (C_out) : N

Pooling layer →

Pool size (P) : Width & height of the Pooling Window

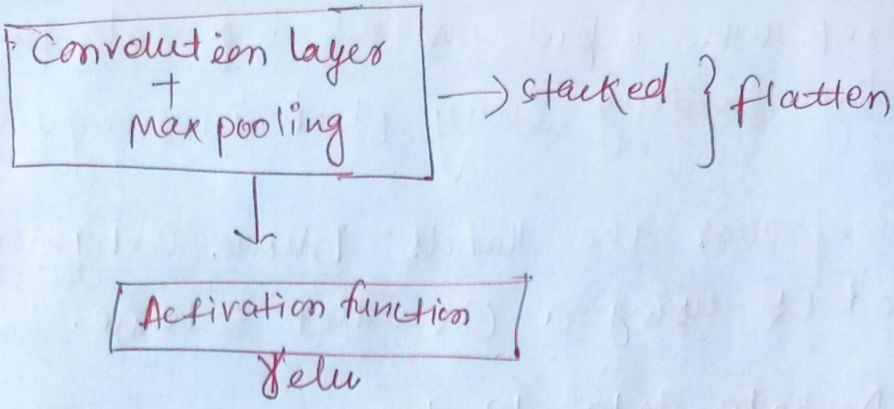
Stride (S) : – the step size at which the Pooling window moves across the input.

Output Dimension ⇒

(W_out) : $((W-P)/S)+1$

(H_out) : $(H-P)/S)+1$

(C_out) : C (remains the same)



* ~~Kernel~~ ⇒

* Image Kernel →

A image kernel is a small matrix used to apply effects like the ones you might find in Photoshop or Gimp, such as blurring, sharpening, outlining or embossing.

* Practical Intuition with code in CNN

(1.) Load MNIST Database

MNIST is one of the most famous datasets in the field of machine learning.

- It has 70,000 images of hand-written digits.
- Very straight forward to download.
- Images dimensions are 28 x 28
- Gray scale Images.

(2.) Visualize the first six Training Images.

(3.) View an image in more detail

(4) Preprocess input images : Rescale the images by dividing Every Pixel in Every Image by 255

(5) Preprocess the labels: Encode Categorical Integer Labels using a One-hot scheme.

(6) Reshape data to fit our CNN (and input shape)

(7) Define the Model Architecture =>

filters - no. of filters

Kernels - no. specifying both height & width of the (square) Convolution window.

Stride - by default set to 1.

Padding - by default - Valid

Activation - (relu)

(8) Compile Model

(9) Train the Model

(10) Load the model with the Best classification Accuracy on the validation set.

(11) Calculate the Classification Accuracy on the Test set.

CNN-4 LENET

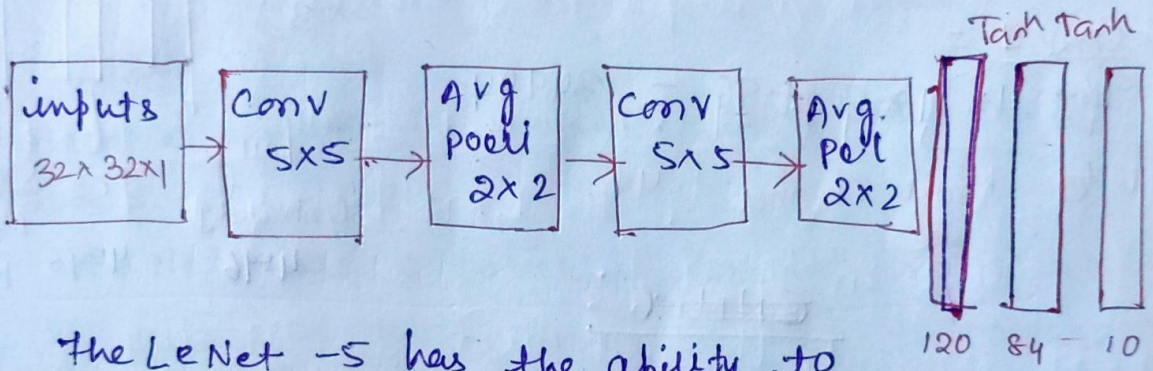
LENET → LENET-5 is one of the simplest architectures.

→ It has 2 convolutional and 3 fully-connected layers.

→ It is the most widely known CNN Architecture.

→ It was introduced in 1998 and is widely used for handwritten method digit recognition.

→ LENET-5, architecture has 60,000 parameters.



→ The LeNet-5 has the ability to process higher one-resolution images that require larger and more CNN Convolutional layers.

CNN-5

- **ALEXNET** → [2012]

→ It consists of 8 layers, including 5 conv. layers and 3 fully connected layers.

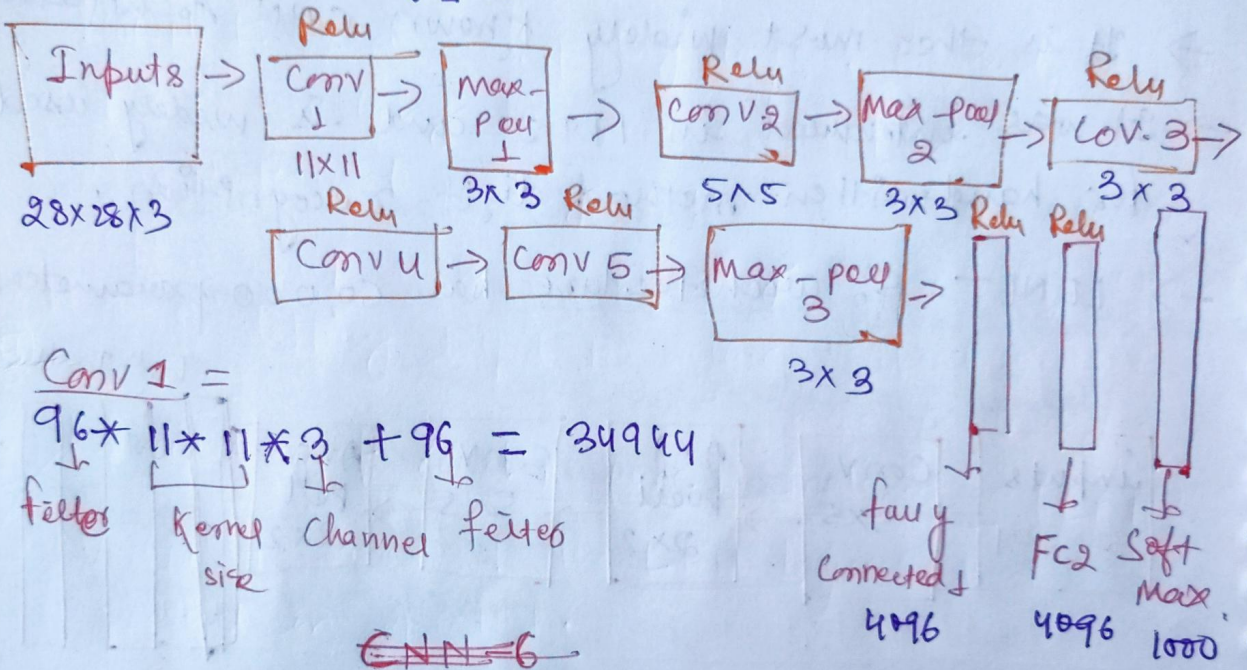
→ It uses traditional stacked Convolutional Layers with max-pooling in between.

→ It allows for the extraction of complex features from images.

Activation function → Relu used.

→ Activation function: and dropout Regularization, which enhances the model's ability to capture non-linear relationships within the data.

→ Batch size is of 128



Conv 1 =

$96 * 11 * 11 * 3 + 96 = 34944$

96 → filter, 11 * 11 → Kernel size, 3 → Channel, + 96 → filter

~~CNN-6~~

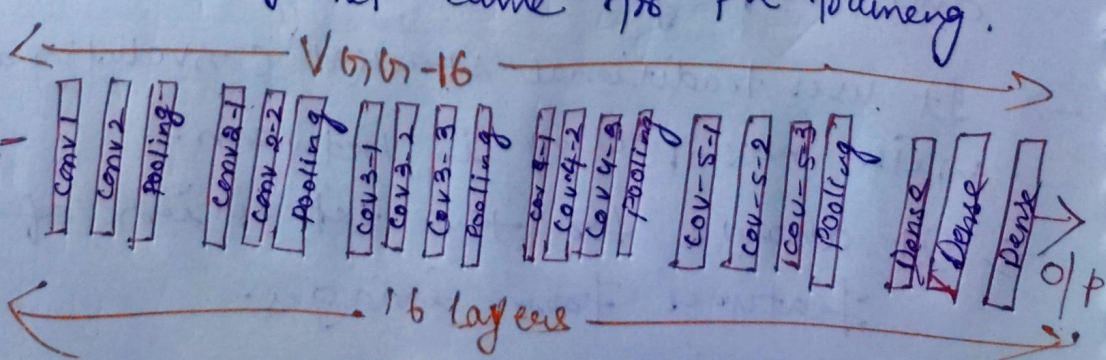
* VGGNET * [Visual Geometry Group]

- Vgg 16 [16 - no. of layers)
- Vgg 19 [19 - no. of layers)

with trainable parameters)

In 2015 → Image net came for pre training.

Inputs :-



- In VGG NET

→ All convolution layers were made to have kernel sizes equal to 3×3.

→ CNN is very deep as 16 and 19 layers for large scale image recognition.

→ LRN - removed.

→ Max pooling → 2×2 and stride is (2) so, no overlapping occured.

*

* Some key changes made in VGG Net from AlexNet

(1) Deeper Architecture ⇒

VGGNet introduced a much deeper architecture compared to AlexNet.

→ AlexNet has 8 layers, VGGNet has 16 or 19 layers. The increased depth allowed VGGNet to capture more complex features and patterns.

(2) Smaller filter Sizes ⇒

VGGNet utilized small 3×3 conv. filters throughout the network, where as AlexNet used a mixture of 3×3 and 5×5 filters.

The use of small filters for a more localized feature representation and enabled deeper networks without a significant increase in parameters.

(3) Uniform structures →

→ VGGNet maintained a uniform structure with a stack of conv. layers followed by max pooling layers.

→ In VGGNet helped to reduce spatial resolution more gradually.

→ The uniform made it easier to design and understand the network architecture.

④ NO Overlapping Pooling ⇒

→ VGGNet used max pooling layers with a stride of 2, resulting in non-overlapping pooling regions

→ AlexNet used pooling regions with a stride of 2 but with Overlapping regions of size 3×3. Non-overlapping helped to reduce spatial resolution more gradually.

⑤ Elimination of Local Respone Normalization (LRN)

→ VGGNet removed the LRN layers that were present in AlexNet.

LRN is initially introduced in AlexNet as a technique to enhance local contrast.

→ However, subsequent research found that LRN has limited impact on performance and was computationally expensive.

⑥ Smaller Fully Connected Layers ⇒

→ VGGNet used smaller fully connected layers compared to AlexNet.

→ VGGNet used 4096 neurons in the last two fully connected layers reduced the no. of neurons

to 1000 in the output layers.

→ the change helped to reduce the no. of parameters and computational complexity.

⇒ [If padding is same then in conv. NN dimension can't reduced.]

RESNET

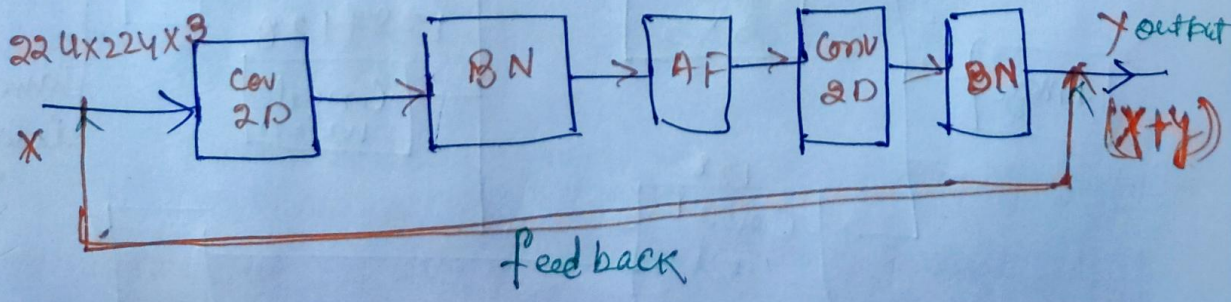
[Residual Networks]

→ In this network, we use a technique called Skip connections.

→ ~~Skipping~~

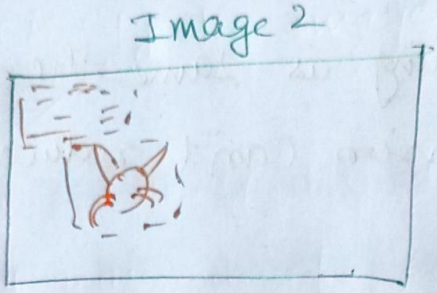
→ the skip connection connects activations of a layer to further layers by skipping some layers in between.

Residual Block (USP) → used to solve vanishing/exploding gradient problem.



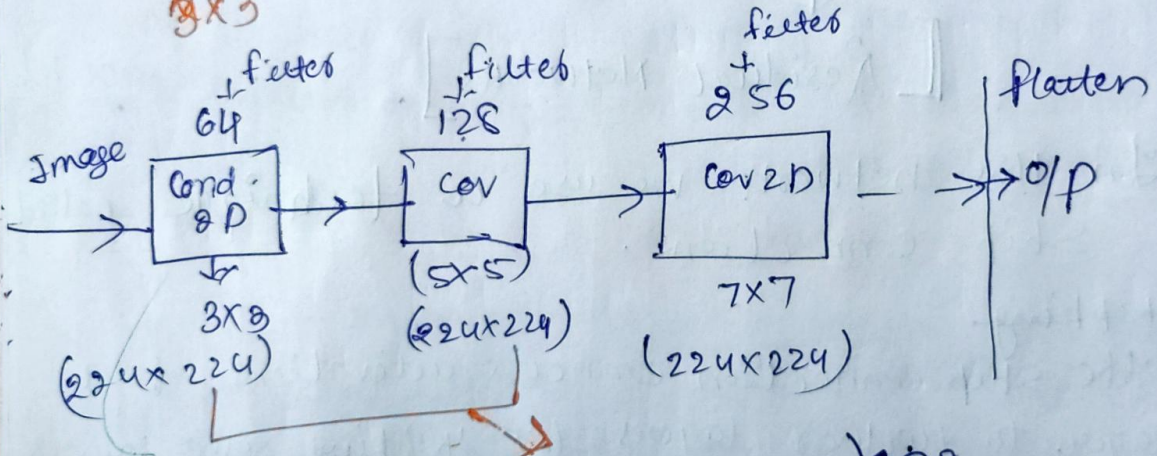
This Residual block was the main USP for the sucess of RESNET

Inception



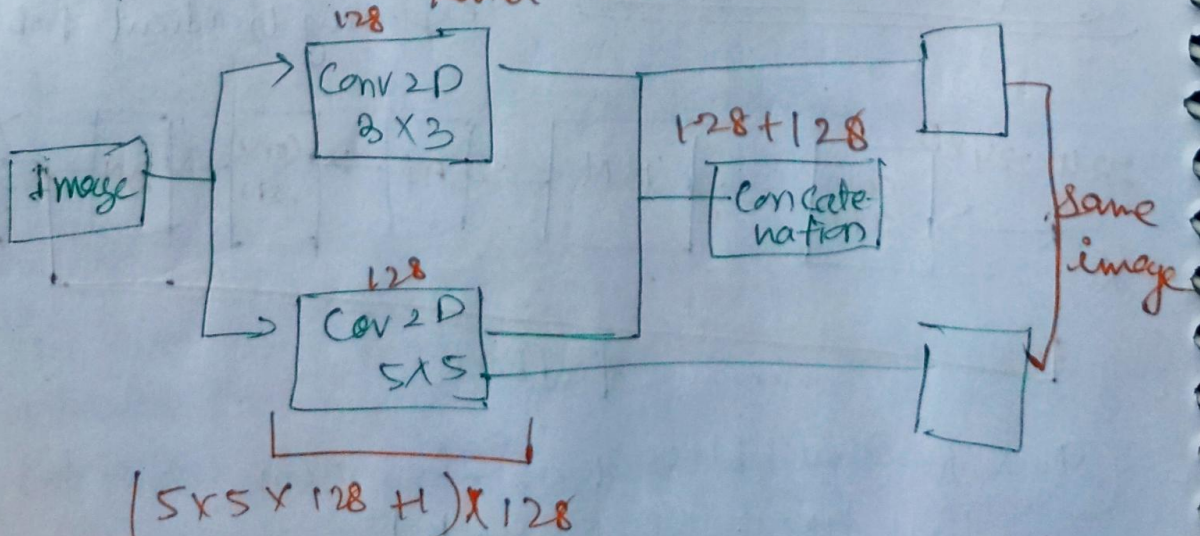
Feature map

~~9~~ X 3



Parameters :- $(3 \times 3 \times 64 + 1)128$

(3 × 3 → Kernel, 64 → bias, 128 → filter)



→ In short, this parallel approach of applying on different dimensions of filters to our filter map in images and concatenating them to get the desired output with the same dimension.

* Less no. of trainable parameters.
* Multiple filter sizes.

Features of Inception :→

(1) It adds non-linearity in the architecture.

(2) No. of Parameters remains same.

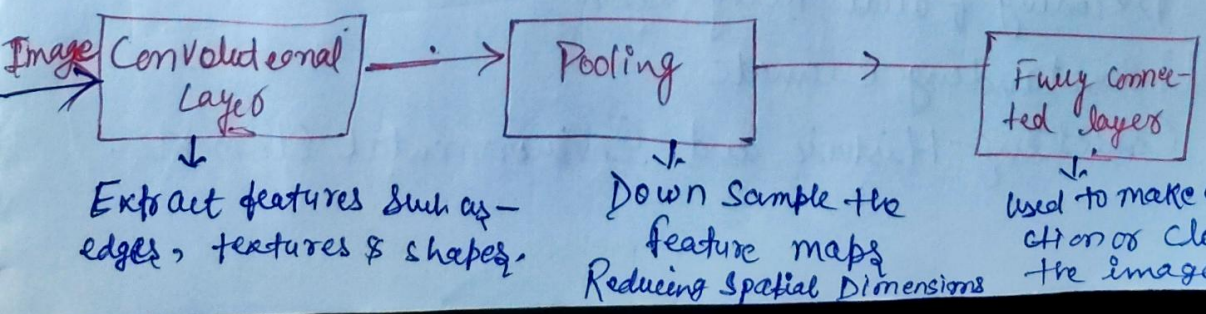
(3) Inception Architecture creates multiple sizes of images which helps in feature engineering.

CNN (Convolutional Neural Network) ⇒

→ It is well-suited for image recognition and processing tasks.

→ It consists of multiple layers :-

(i) Convolutional layer
(ii) Pooling Layer
(iii) Fully connected layers.



→ Output of convolutional layer is passed to Pooling layers and then output of pooling layers are passed to fully connected layers for prediction or classify the image.

Types of CNN Models ⇒

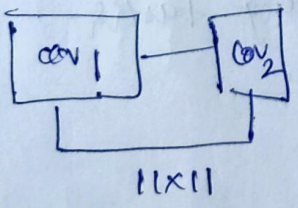
① **LeNet** → It was specifically designed for handwritten digit recognition, and was one of the first successful CNNs for image recognition.

→ It was trained on the MNIST dataset, which consists of 70,000 images of handwritten digits, and was able to achieve high recognition accuracy.

② **AlexNet** ⇒

→ It was the first CNN to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a major image recognition competition.

→ Conv2D = 5, Pooling layer = 3, fully connected layer = 3



→ first 2 conv0 layer uses 11×11 size of Kernel.

Application →

- Decoding facial Recognition
- Understanding Climate
- Collecting Historic and Environmental Elements.

R-CNN (Region-based CNN)

[Used for Object Detection]

Object Detection :->

-> RCNN (Naive Approach)
-> Fast RCNN
-> Faster RCNN
-> YOLO
-> SSD

* Exhaustive Search -> Search types of images of no. of images.
* Selective Search -> Search iteratively
  ↳ It is greedy algo.

Steps ->

Step 1 -> Cropped images

Methods to extract Cropped images :-

(i) Exhaustive Search (ii) Selective Search.

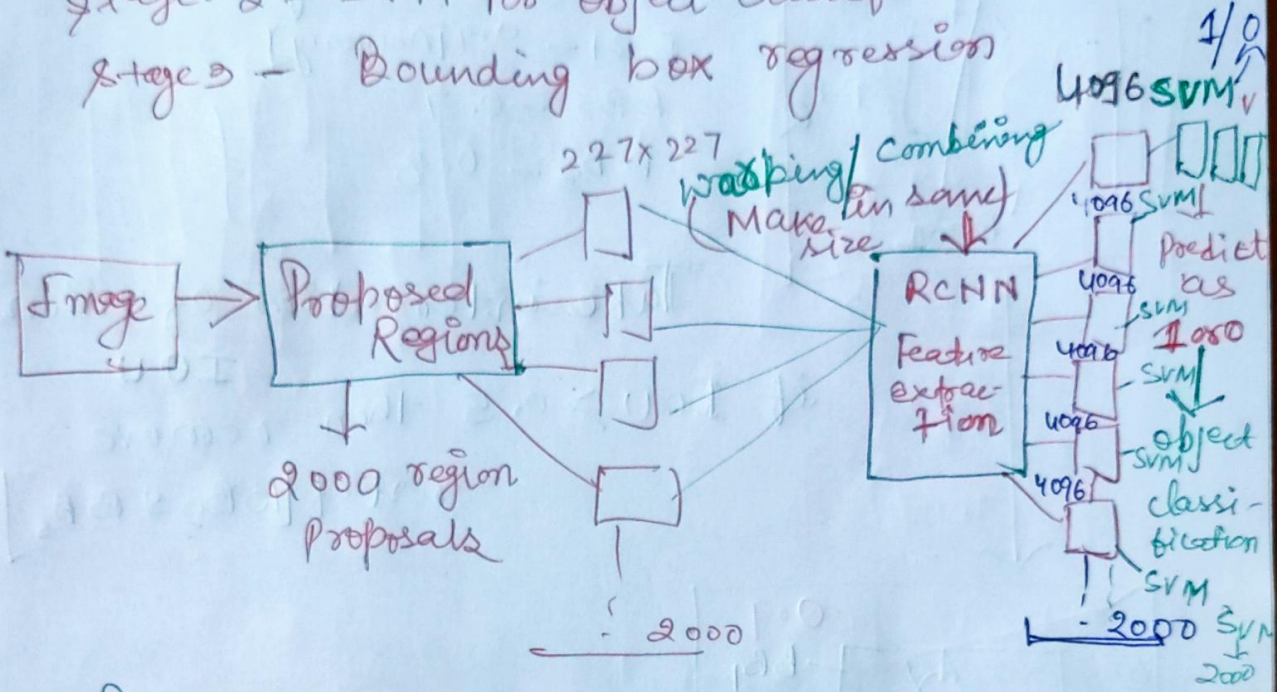
* Regions extracted from R-CNN.

(i) Selective Search -> Region Proposal Method
  ↳ used in RCNN is selective search around 2000 regions proposals.

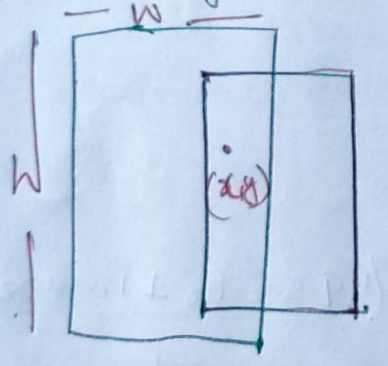
-> this algorithm works by generating sub-segmentations of the image that could belong to one object - based on color, texture, size and shape - and iteratively combining similar regions to form objects.

RCNN Pipeline ⇒

① stage 1 - Feature extraction from Region Proposals.
stage-2 - SVM for object classification
stage 3 - Bounding box regression



→ Bounding Box Regression →

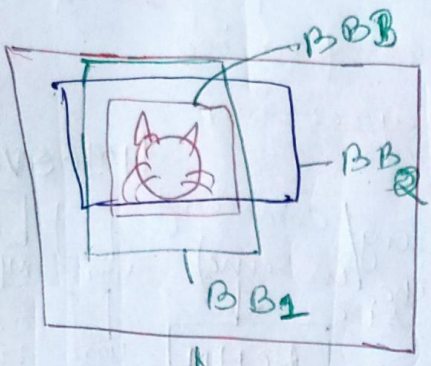


(dx, dy, dw, dh) → Regress on these points for fine tunning.

* Steps of RCNN →

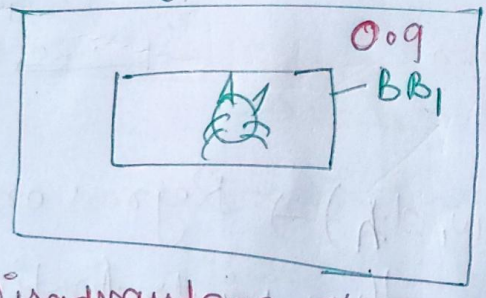
① Region Proposals [by exhaustive search]
② Warping and Resize
③ Pretrained CNN [AlexNet, VGGNet used]
↳ for fine tunning or Categorise image type.
④ Pretrained SVM for every image.
⑤ Clean up images → [NON MAX-Suppression]
⑥ Bound Box Regression

Non-Max Suppression ⇒ For clean-up or fine tunning for best accuracy of Model.



BB1 – 0.9 [Accuracy]
BB2 – 0.7
BB3 – 0.8

Take intersection of union
IOU
if IOU > 0.5 then remove rest of [BB2 & BB3]



Disadvantage of RCNN →

1. Very computationaly expensive.
2. Extremly time consuming (49 sec for 1 image)
3. We need to pretrain [CNN, SVM]
4. huge amount of time, storage & Computation power used.

Fast R-CNN

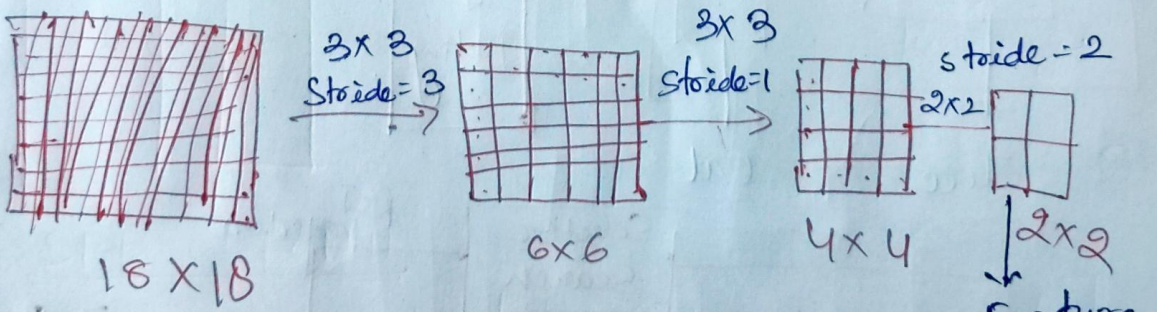
→ Overcomes Drawbacks of RCNN.

It uses Selective Search for generating Region of proposals.

* Fast CNN Features :→

1. Subsampling
2. ROI Projection
3. ROI pooling

(1) Subsampling ⇒ A method that reduces data size by selecting a subset of the original data.

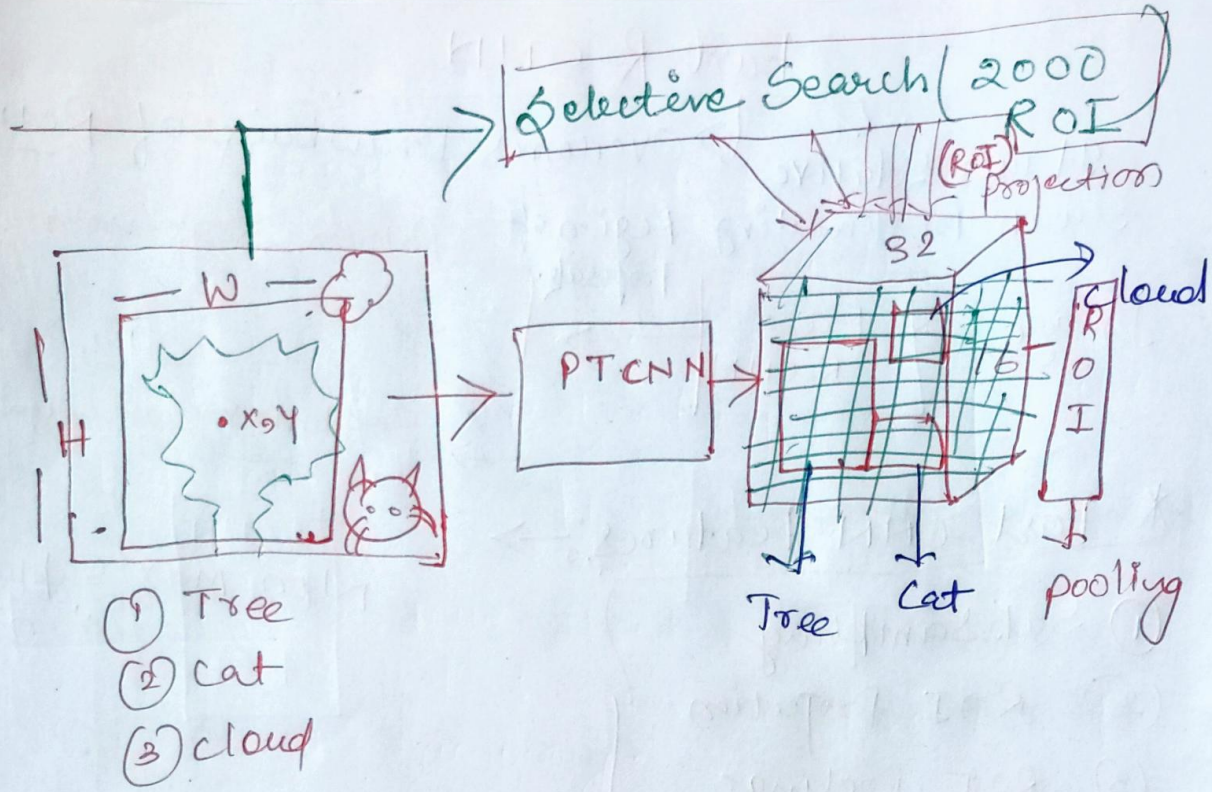


Subsampling Ratio = 1/9

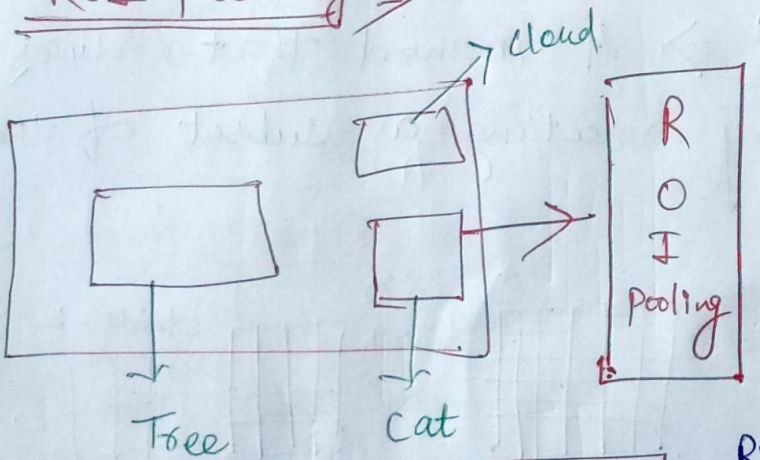
(II) ROI (Region of Interest) ⇒

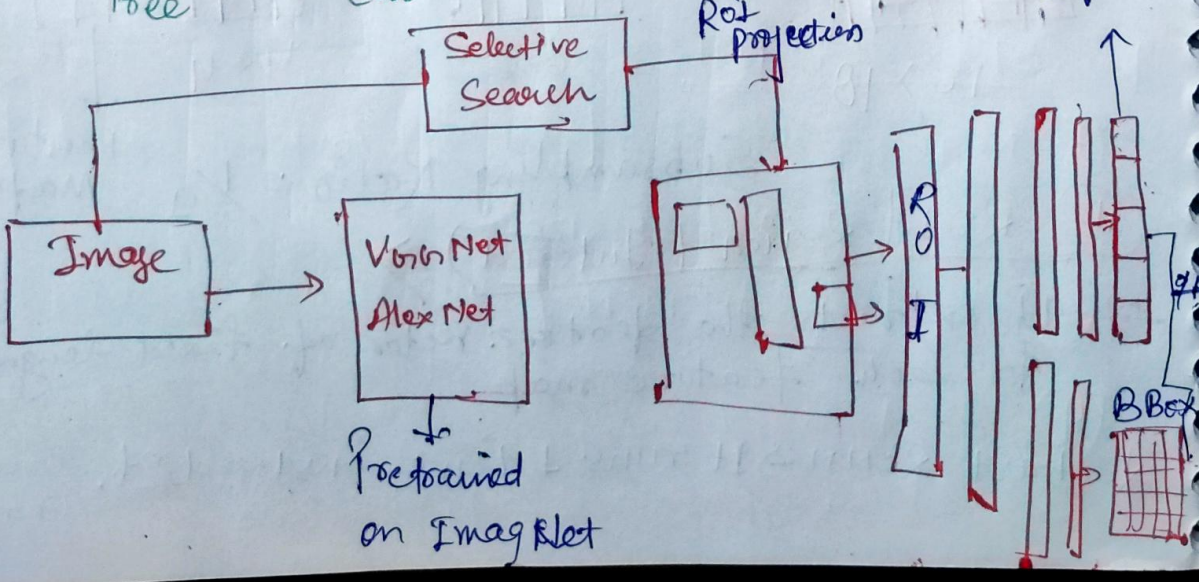
→ It extracts the feature vector of fixed length for each feature map.

* Fast RCNN → It runs 1 time instead of 2000 times.


Selective Search (2000 ROI)
(ROI projection)
W
H
x, y
PTCNN
32
6
ROI
cloud
Tree
Cat
pooling
1 Tree
2 cat
3 cloud

ROI pooling =>

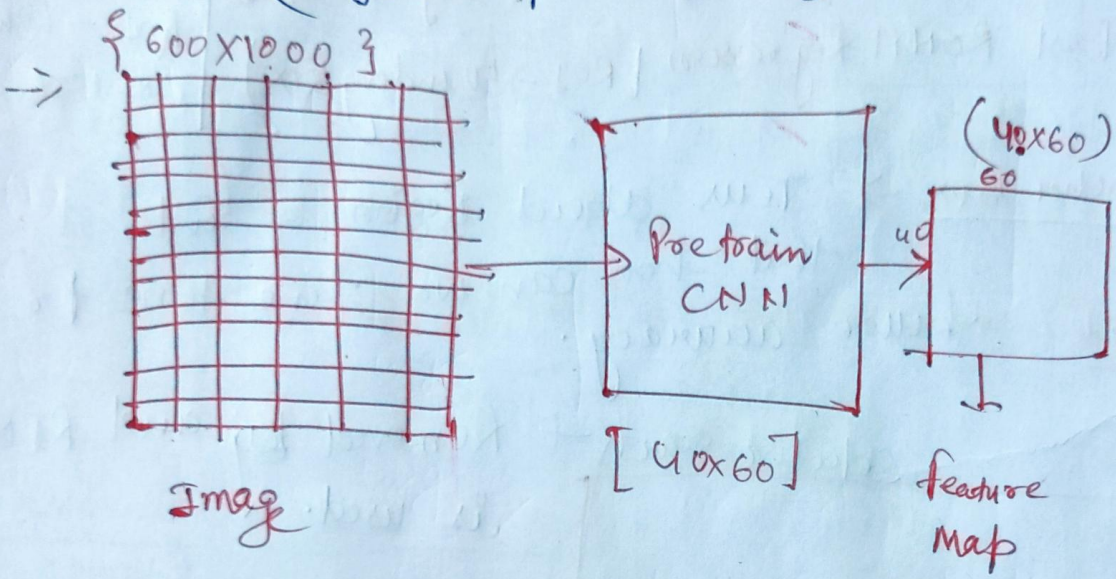

cloud
ROI Pooling
Tree
Cat


Selective Search
ROI projection
Softmax
Image
VGG Net
Alex Net
Pretrained on ImageNet
ROI
FC
BBox

Faster RCNN → RPN used to generate ROI.

[RPN + Fast RCNN]

(Region Proposal Network)



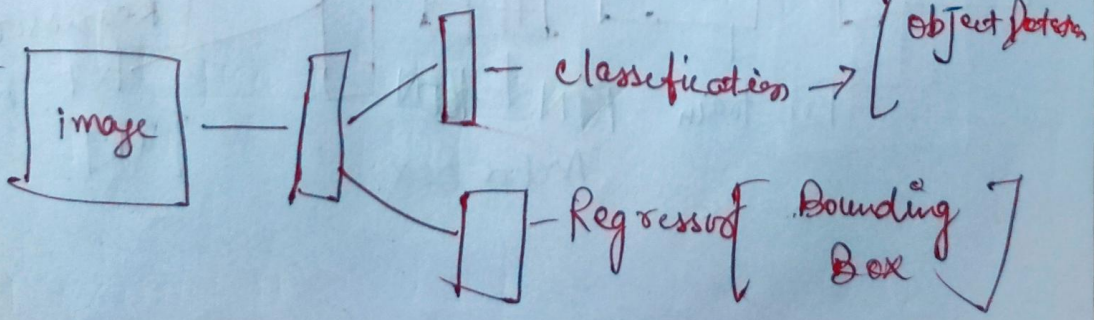
40x60x9 ≥ 20000

↓

no. of sliding windows

→ Comparing with my Anchor Box Result, with selective search it gives 10 times more ROI's.

So, this problem is solved



* The Faster R-CNN has 4 losses: —

(I) RPN Classification (object foreground / background) → Binary classifier

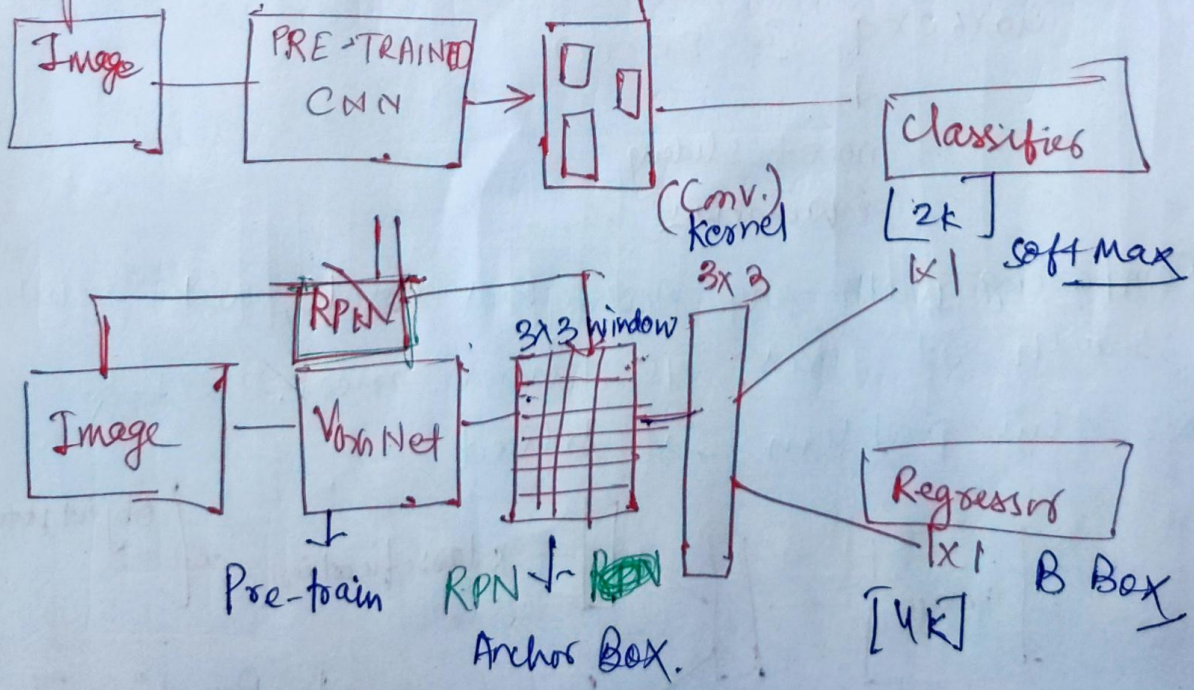
(II) RPN Regression (Anchor → ROI)

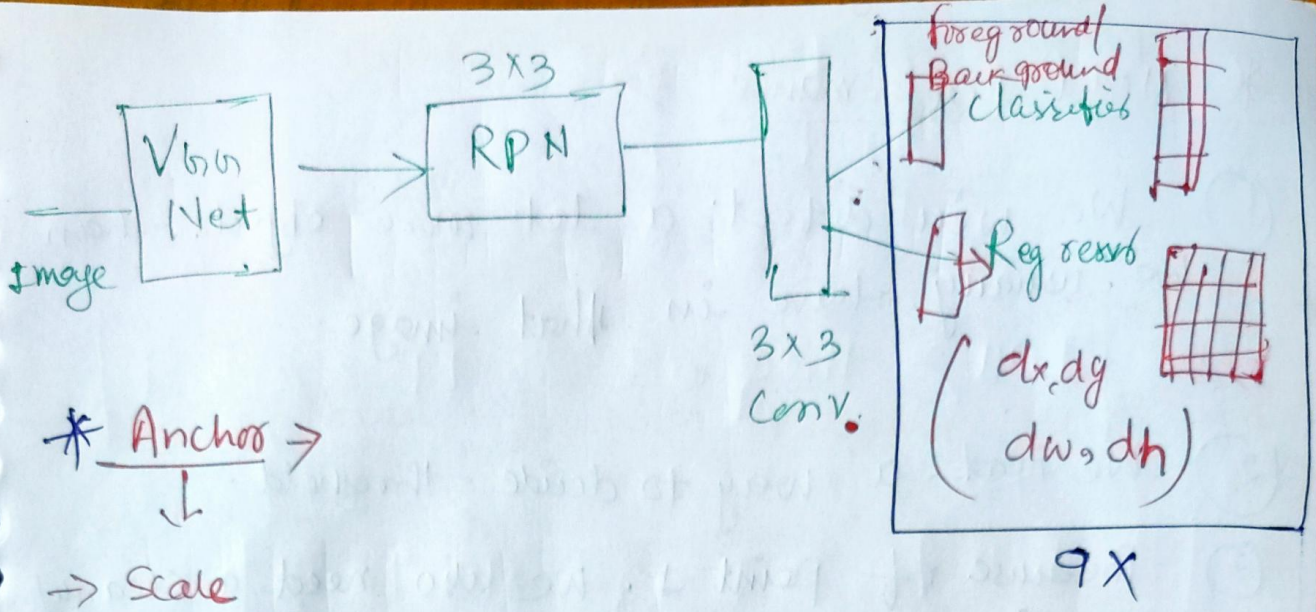
(III) Fast RCNN Classification (object classes)] object classes

(IV) Fast RCNN Regression [ROI → Bounding Box)] ROI – Boundary dx, dy, dw, dh

Anchor Box → Tells about Region of interest that how can we fine tune for better accuracy.

Selective search → Removed and RPN is used.



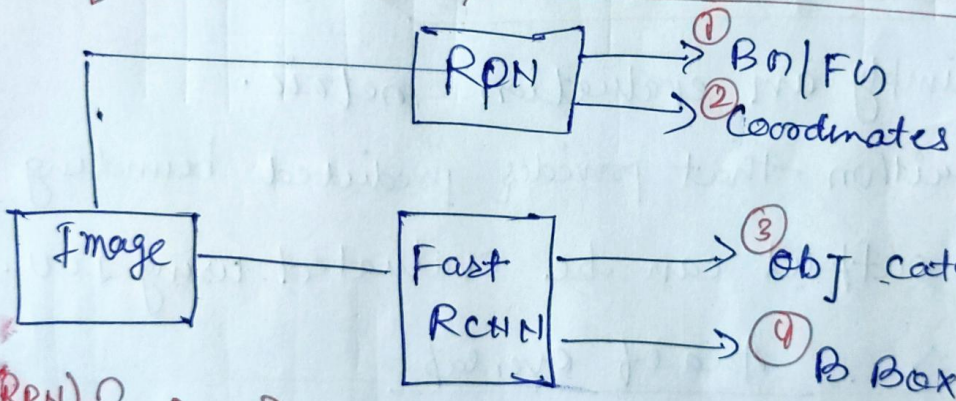


* Anchor →

→ Scale

→ Aspect Ratio

[Faster-RCNN Detailed ~~Out~~ Understanding]



(RPN) Region Proposal Network ⇒

* Model Performance ⇒

(1) We will detect a lot more objects than we usually have in that image.

(2) We need a way to decide threshold.

(3) Because of point 1, we also need a cleanup process, by this predictions will be limited.

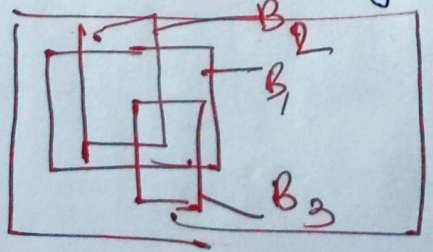
IOU (Intersection over union) ⇒

→ It is simply an evaluation metric.

→ Any algorithm that provides predicted bounding boxes as output can be evaluated using IOU.

$$IOU = \frac{\text{Area of Overlap}}{\text{Area of union}}$$

Area of overlap = Between predicted bounding box and the ground-Truth bounding box.



$$IOU = B_1 \cap B_2 \cup B_3$$

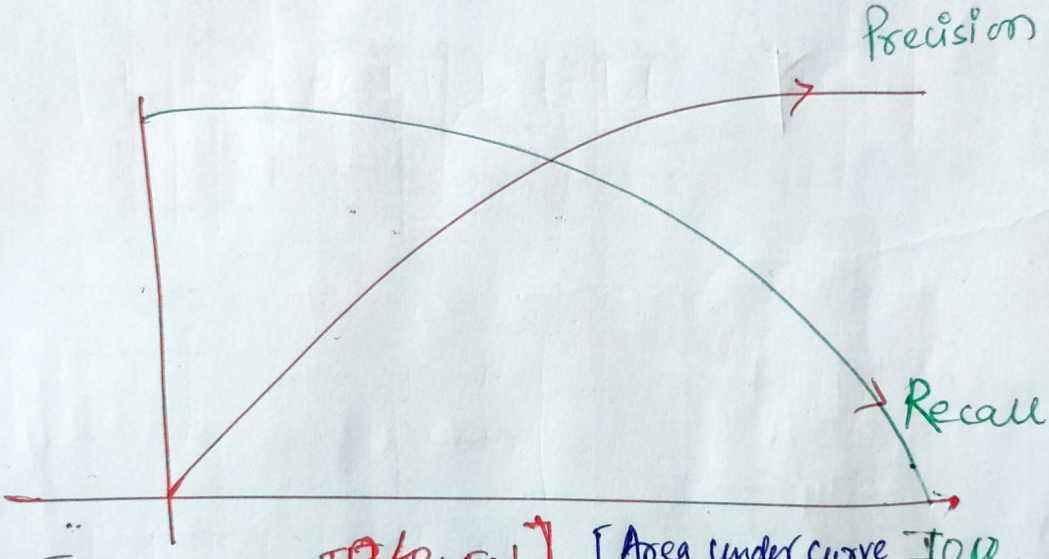
* **Mean Average Precision** → It is a metric used to evaluate object detection model.

$$\text{Precision} = \frac{TP}{TP + FP}$$

It Varies [0 - 1]

→ IOU, Recall, Precision, Confusion Matrix etc.

$$MAP = \sum_{i=1}^{n} AP_i$$



[Recall = TP/TP+FN] [Area under curve IOU = Mean Avg. Precision]

Non - Max Suppression =>

(1) Remove all B Box with confidence level < (0.75)

(2) (a) find B Box with max confidence level.

(b) All B Box have IOU > 0.5 then remove all B Box except large value B Box.

Recall = $\frac{TP}{\text{Actual Ground Truth of class}}$

→ IOU > threshold → TP considered

→ If IOU ↑es → Precision also ↑es