AlexNet Project

Tuesday, 12 November 2024 4:03 PM

AlexNet :

Complexity of the model :

- → 60 million parameters to train
- → 650,000 neurons
- → Five Convolutional layers
- → Followed by max-pooling layers.
- → Three fully connected layers
- → Final way 1000-softmax ⟹ final classes that we need predict

Training Information :

→ The paper used non-saturating neurons, which is not that famous during that time.

→ To avoid overfitting they have used "dropout" techniques.

→ Used two GPU to train the dataset.

Introduction :

ILSVRC - ImageNet Large Scale Visual Recognition Challenge

↓

(It's a subset of ImageNet)

Some statistics related to the dataset :

- → 1000 images in each 1000 categories
- → Roughly having 1.2 million training images
- → 50,000 validation images
- → 150,000 testing images.

Data Preprocessing :

→ Down-sampling the Images (256 X 256)

→ How the down sampling is done, they rescaled the image such that the shorter side was of length 256

380 ← They first targeted the side to rescale it to "256"

512

→ Substracting the mean activity over the training set for each pixel

$[1, 2, 3, 4, 5]$ ↳ The mean value of this pixel is "3"

Now, we are substracting this value across the pixels

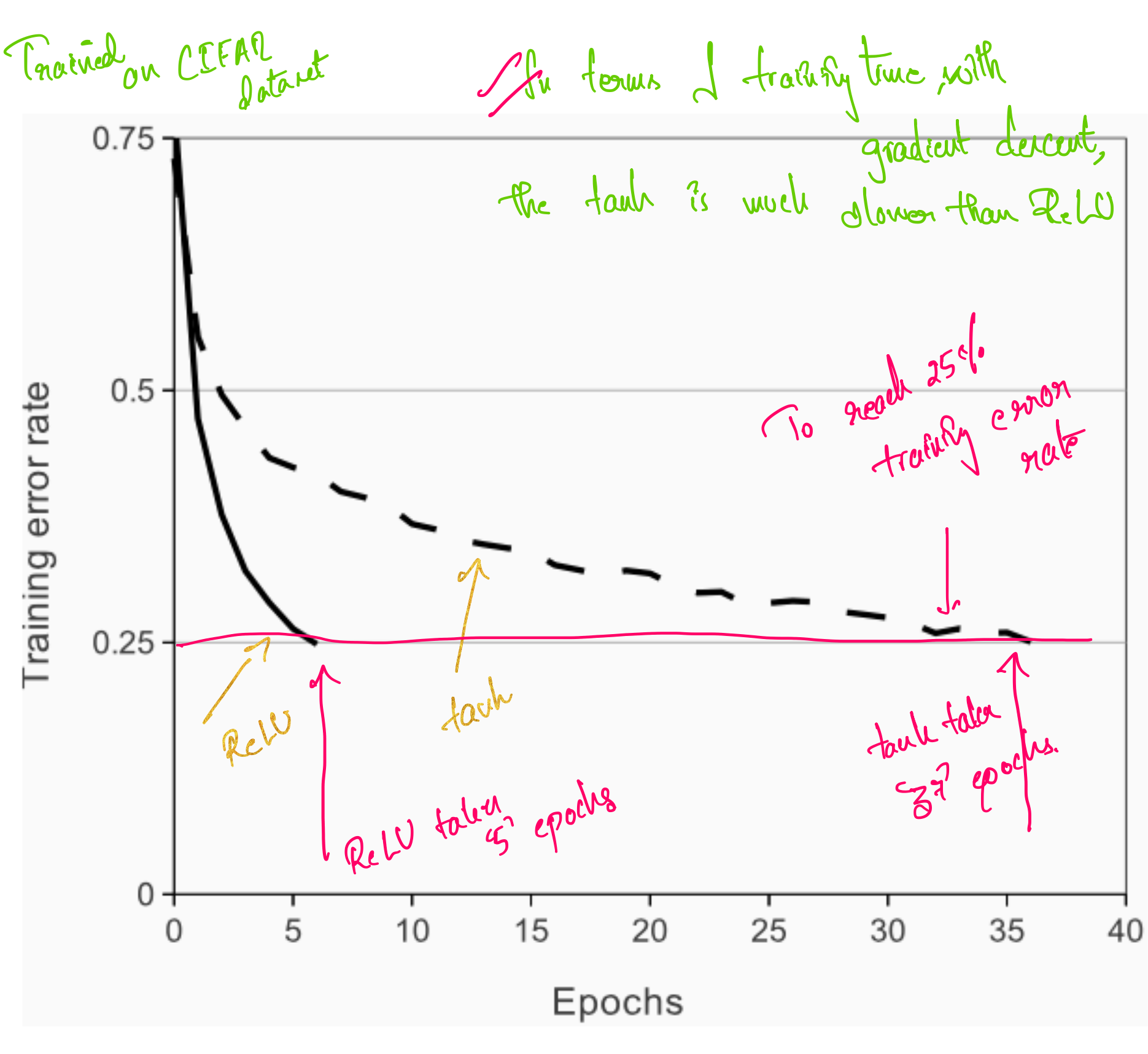
Resultant Pixels look like this ← $[-2, -1, 0, 1, 2]$

The Architecture :

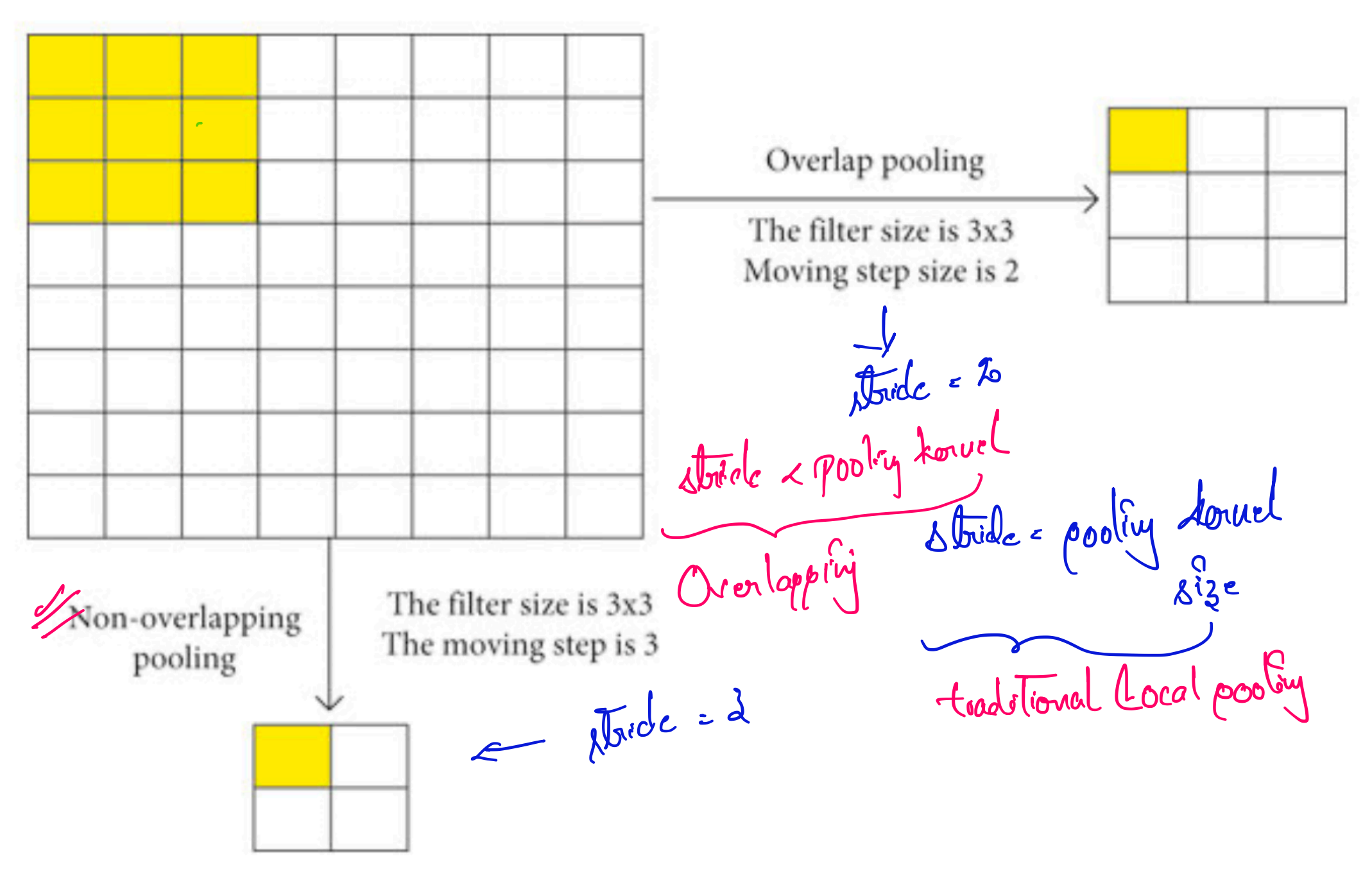
ReLU Non-Linearity : (Why ReLU we are using ? Reason)

Trained on CIFAR dataset

In terms of training time with gradient descent, the tanh is much slower than ReLU



Polling Layer :



Overlap pooling
The filter size is 3x3
Moving step size is 2
stride = 2
stride < pooling kernel
Overlapping
stride = pooling kernel size
traditional Local pooling
Non-overlapping pooling
The filter size is 3x3
The moving step is 3
stride = 3

Observation :

During training models, they observed that models with overlapping pooling find it more difficult to overfit.

Architecture :

→ Maximizes the multinomial logistic regression objective, which is equivalent to maximizing the average across training cases of the log-probability of the correct label under the prediction distribution.

→ ReLU non-linearity is applied to the output of every convolutional and fully connected layer

Layer-1

→ Images are reduced to (227, 227, 3)

→ First Convolutional layer has ⟹ 96 kernels

size: 11 X 11 X 3

⟹ With a stride of 4 pixels.

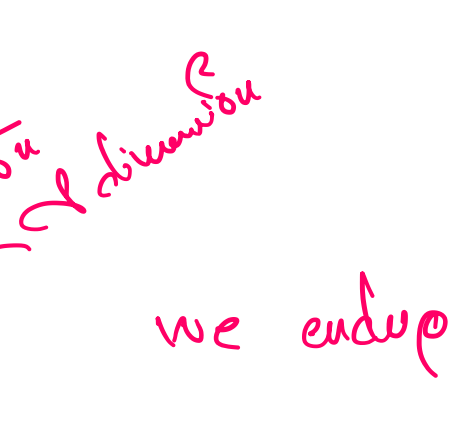
Reduction of dimension

We endup with $(227-11)/(4+1) = 55$

And, then we applied a Overlapping maximum pooling of kernel size "3X3" with a stride of "2"

Reduction of dimension

we endup with $(55-3)/(2+1) = 27$



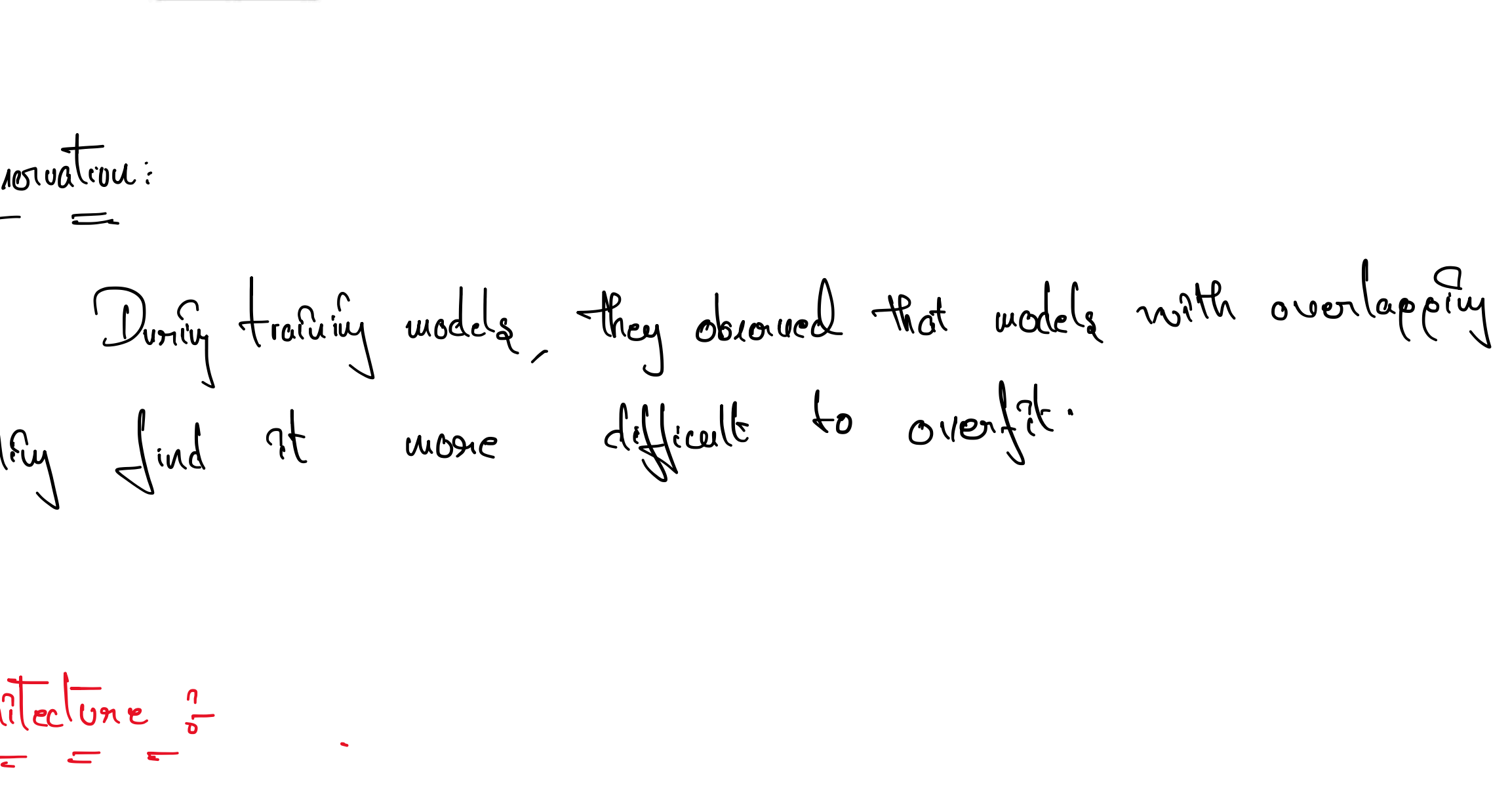
→ Second Convolution layer takes (local response normalized and pooled) ⟹ 256 kernels

↙ 5 X5 X 48

Further layer information is represented in the below screenshots

Formula for calculating output of convolutional layers

$$\text{output} = \frac{\text{Input} - \text{kernel\_size} + 2 * \text{padding}}{\text{stride}} + 1$$



227
3
227
11
11
CONV 11x11, stride=4, 96 kernels
(227-11)/4 +1 = 55
96
55
55
Overlapping Max POOL 3x3, stride=2
(55-3)/2 +1 = 27
96
27
27
CONV 5x5,pad=2 256 kernels
(27+2*2-5)/1 +1 = 27
256
27
27
Overlapping Max POOL 3x3, stride=2
(27-3)/2 +1 = 13
256
13
13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
384
13
13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
384
13
13
CONV 3x3,pad=1 256 kernels
(13+2*1-3)/1 +1 = 13
256
13
13
Overlapping Max POOL 3x3, stride=2
(13-3)/2 +1 = 6
256
6
6
9216
FC
FC
4096
4096
1000 Softmax

Reduce Overfitting :

Data Augmentation :

→ Image Translation and Horizontal reflections

→ Altering the Intensities of the RGB channels in training images (Specifically they performed PCA on the set of RGB pixel values throughout the ImageNet Training set)

Dropout :-

The dropout is used in the first two fully connected layers

without dropout, their network exhibits substantial overfitting.

Details of Learning :

→ Stochastic Gradient Descent with a batch size 128 examples, momentum = 0.9, weight decay = 0.0005

All the weights across the layers are → Initialized Weights - [Zero - Mean Gaussian Distribution with standard deviation 0.01]

Biases — second, fourth, fifth convolutional layers, and fully connected hidden layers with the constant "1",

And for remaining layers, we initialized biases as zero

learning rate

→ Initial learning rate was initialized as "0.01", whenever validation error stopped improving we divided the learning rate by 10