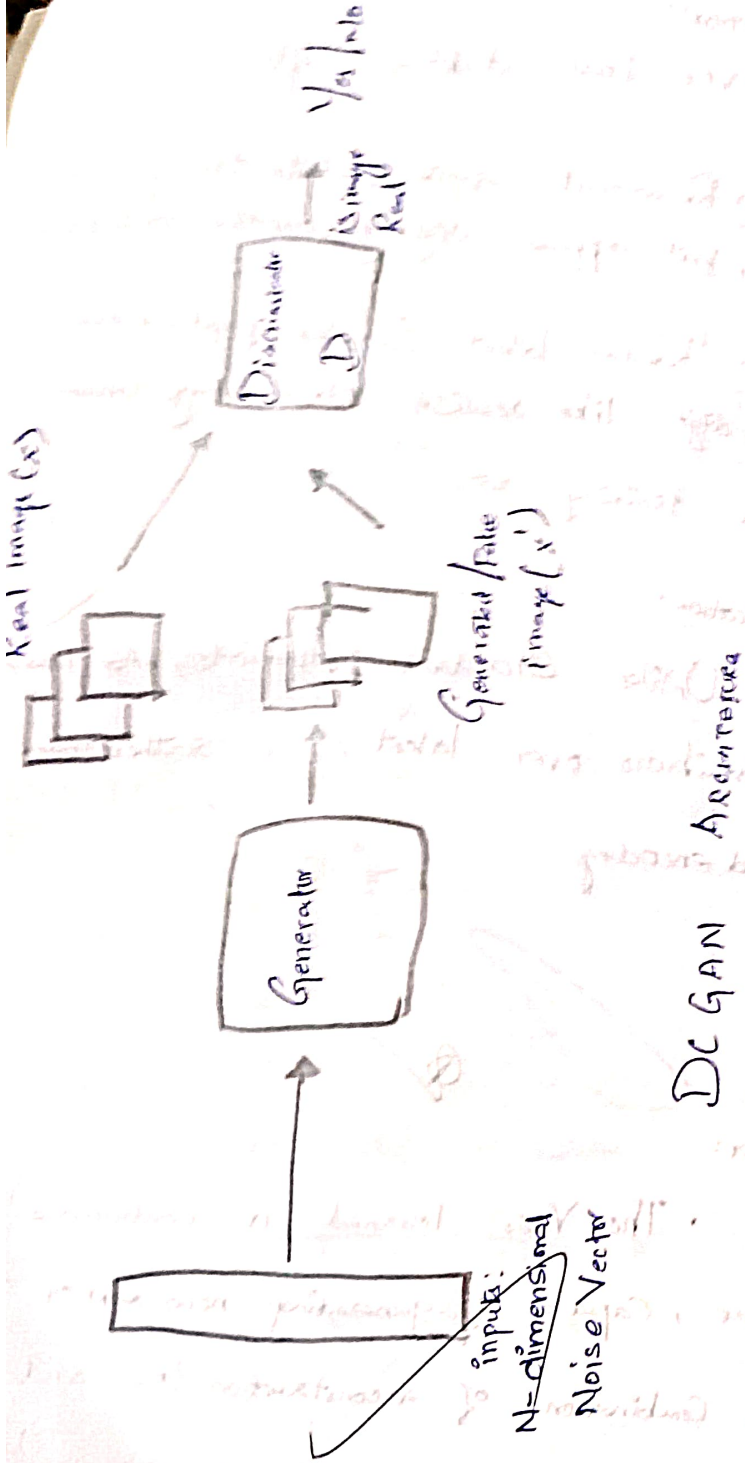
RA2311047010038
Deep learning Techniques

INDEX

NAME : Tamilselvan M STD : _____ SEC : _____ ROLL NO.: _____ SUB _____

S.No.	Date	Title	Page No.	Teacher's Sign / Remarks
10	8/10/25	Perform Compression on MNIST dataset using auto encoder		
11	8/10/25	Experiments using Variational Autoencoder		
12	17/10/25	Implementation a Pre-trained GAN Deep Convolution Generative Adversial Network (DCGAN) to generate Complex Color Images		
13	17/10/25	Understand the Architecture of Pre-trained Models		
14	27/10/25	Implementation a Pre-trained CNN model as feature extraction		
15	27/10/25	IMPLEMENT A Yolo Model to Detect Objects		



DC GAN Architecture

Exp. no 12
17 Marks

Implementation a Deep learning Convolution Gan to generate Complex Color Images

Aim:

To implement a deep Convolution ~~Gan~~ Generative Adversarial Network (DCGAN) using Pytorch to generate Complex Color images from the Noise.

OBJECTIVES:

→ To implement GAN and adversial loss.

→ To design a DCGAN architecture of generating RGB images.

→ To evaluate the learning Progress through generator and discriminator losses.

Pseudocode:

1. Import required libraries
2. load a Color Image.
3. Defino
   - Generator.
   - Discriminator.
4. Initialize both networks with Xavier Initialization.
5. Train with Adversial loss:

1	1.372	0.873
2	1.102	0.2
3	0.984	1.4
4	0.84	1.6
5	0.75	1.9
6	0.70	21
7	0.66	2.3
8	0.64	2.4
9	0.61	2.5
10	0.59	2.7

- Generator: learn to fool the discriminator.

6. After training, generating new fake images using random noise.

7. Visualize generated color images.

OBSERVATIONS:

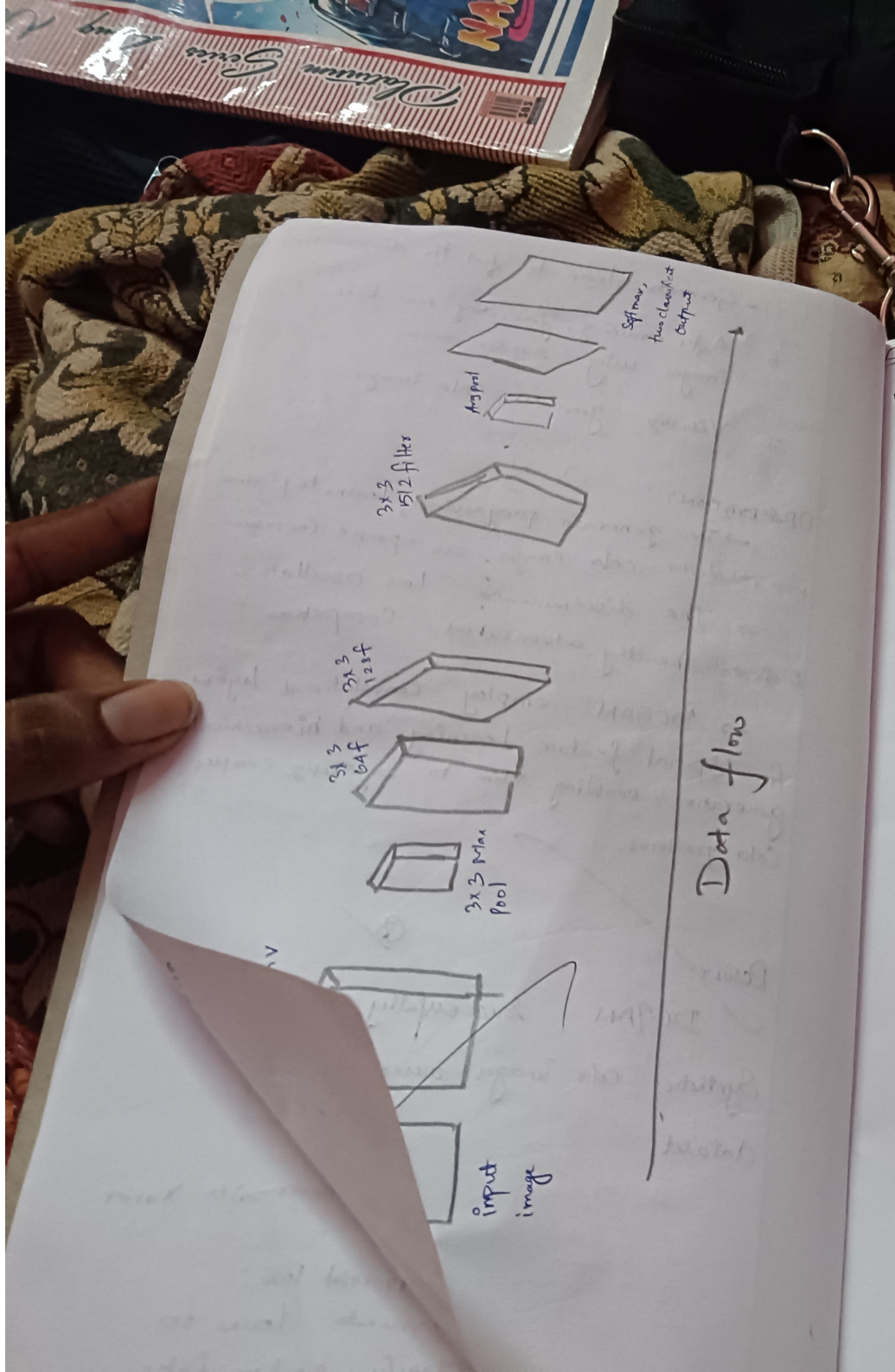
→ The generator progressively learns to produce more relalistic color images as epochs increase.

→ The discriminator loss oscillates indicates healthy adversarial Competion.

→ DCGAN employ convolutional layers for spatial feature learning and hierachical generation, enabling them to synthesize complex color patterns.

RESULT:

DCGAN succesfully Generated Synthetic color images resembling the CIFAR-1 dataset.



Exp No: 13
17/10/25

Understand the Architecture of Pre-Trained Models.

AIM:

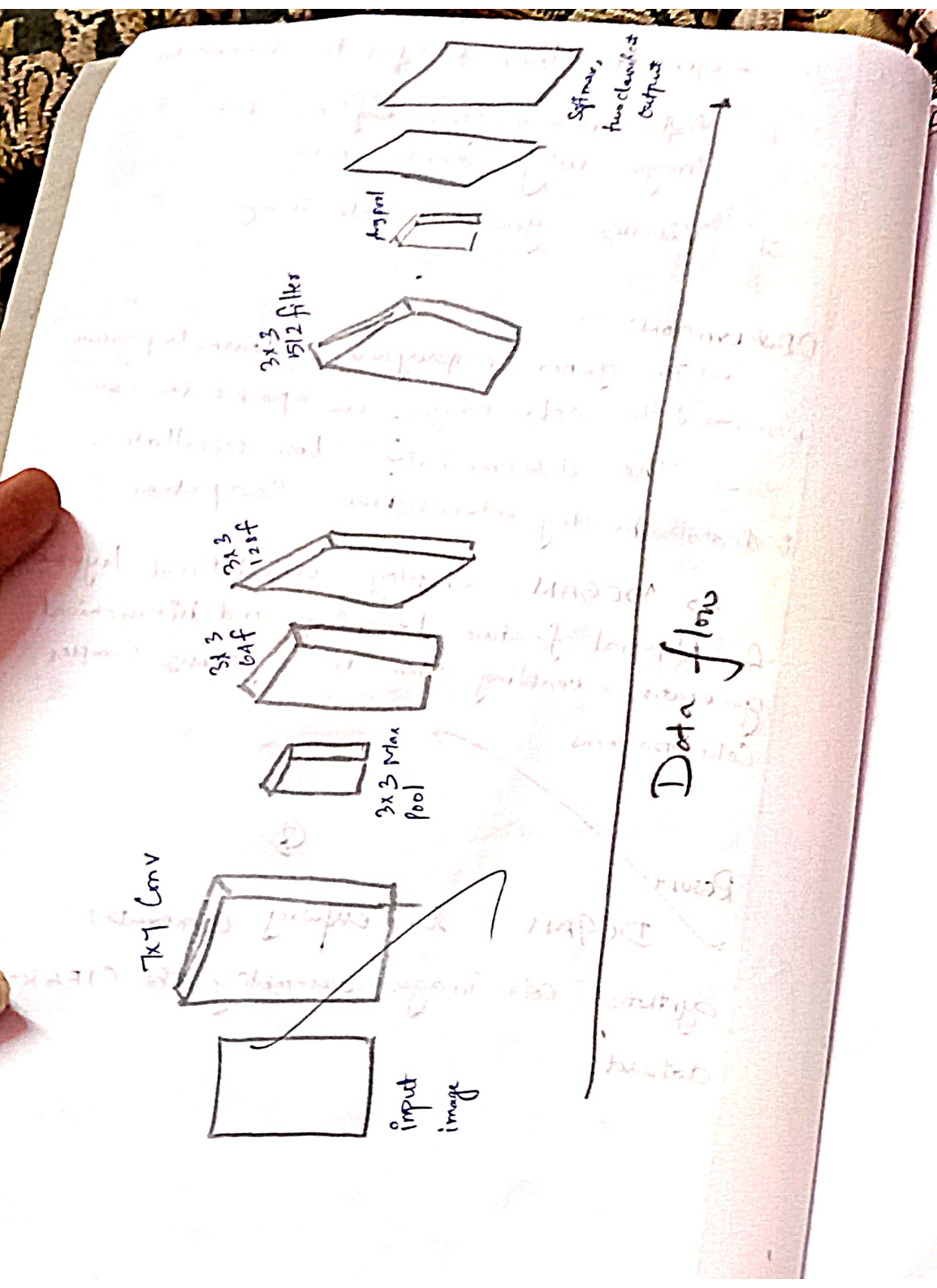
To analyse and understand the architecture and feature extraction of a Pre-trained deep learning model in PyTorch.

OBJECTIVES:

- To load and explore a pre-trained CNN model from Pytorch's model zoo
- To understand the structure of Convolution, Pooling, fully Connected layers
- To visualized intermediate feature maps and comprehend learned representation

PSEUDOCODE:

1. Import library
2. Print model architecture layer-wise
3. Pass a sample image through a model.
4. Visualize the feature map
5. Observe how filters detect low to high level features.



Exp No: 13 Understand the Architecture of
17/10/25 Pre-Trained Models.

AIM:

To analyse and understand the architecture and feature extraction of a Pre-trained deep learning model in PyTorch.

OBJECTIVES:

- To load and explore a pre-trained CNN model from Pytorch's model zoo.
- To understand the structure of Convolution / Pooling, fully Connected layers.
- To visualized intermediate feature maps and comprehend learned representation.

PSEUDOCODE:

1. Import library
2. Print model architecture layer-wise
3. Pass a sample image through a model.
4. Visualize the feature map
5. Observe how filters detect low to high level features.

Epoch

Discriminator

Layer	Feature type	Description
Conv1	Edges, Colors	Basic Visual fea
layer1	Textures	Detect Stripes
layer 2	Shapes	Recognize Pattern
layer3	object	Detect Object
layer 4	Semantic	High level Context

OBSERVATION:

→ The initial Convolution layers detect edges and color gradient

→ Deeper layers capture Complex features like shapes and textures.

→ The hierarchical structure helps the network learn from low-level to semantic representation.

→ Pre trained models like ResNet, VGG are rich hierarchical feature extraction

RESULT:

A Pre trained ResNet 18 model was succesfully analyzed and used to Visualize interral features map.