EX.NO:14 IMPLEMENT A PRE-TRAINED CNN MODEL AS A FEATURE EXTRACTOR USING TRANSFER LEARNING MODEL

27.10.25

Aim

To implement a pre-trained CNN model as a feature extractor using transfer learning model.

Objective

To understand how a CNN pre-trained on ImageNet can extract visual features.

To freeze convolutional layers and use their output features for new tasks.

To train only the classifier head for faster and more accurate learning.

psoudocode

Import libraries

Load a pre-trained CNN

freeze all convolution layers to prevent retrained

Replace final classifier layer for new number of classes.

Load small dataset (CIFAR-10)

Extract features

Train only new classifier head

Evaluate accuracy

objectives observation

The pre-trained ResNet18 model was successfully loaded.

frozen layers acted as fixed feature extractor, capturing low-level edges, corners, textures.

output

epoch 1, loss : 0.8348
epoch 2, loss : 0.6829
epoch 3, loss : 0.5937
epoch 4, loss : 0.5797
epoch 5, loss : 0.5699

Training was significantly faster because only the final classifier parameter wire updated leading stable convergence.

This experiment demonstrated how transport learning can drastically reduce training and computational cost.

Result

Successfully Implemented pre-trained CNN Model.

EXP NO: 15

27.10.25

IMPLEMENT A YOLO MODEL TO DETECT OBJECTS

AIM

To implement a YOLO model to detect objects

Objective

To study how YOLO detects multiple objects in a single image.

To understand the architecture and working a pre-trained YOLO model.

To use transfer learning for custom object decisions.

pseudocode

Install and import (YOLO package)

load a pre-trained YOLO model

load a test image or use a camera frame.

Run the model predict() method to detect objects.

Display bonding boxes and class label.

Save the annotated output image

Observation

The pre-trained YOLO VSS model successfully detected multiple objects such as cars, buses person in single frame.

image 1 | 1 | content | IMG-20250523-WA00

3 persons . 7.8 ms 640 X 480

Speed : 8 preprocess : 7.3 ms inference

2.5 ms

postprocess per image A shape

(1, 3, 640, 480)

output

Detected objects

Person

Person

Person

Each object was enclosed in a bounding box with a class label and confidence score.

The inference time per image was very low

The model demonstrated strong generalization without additional training.

The visualization clearly showed YOLO's ability to detect overlapping objects complex scenes.

Result

Successfully Implement pre-trained YOLO.

RA2311047010011

10.1.38.19

Index

Superior® Note Books

Name : L DHARSHINI

Subject : DEEP LEARNING TECHNIQUES

Sec. : A Roll No. 011

School :

No.	Date	Title	Marks	Signature
01	30.07.25	EXPLORING THE DEEP LEARNING PLATFORMS		
02	07.08.25	IMPLEMENT A CLASSIFIER USING OPEN SOURCE DATASET		
03	07.08.25	STUDY OF THE CLASSIFIER WITH RESPECT TO STATISTICAL PARAMETERS		
04	14.08.25	BUILD A SIMPLE FEED FORWARD NEURAL NETWORK TO RECOGNIZE CHARACTER		
05	22.08.25	STUDY OF ACTIVATION FUCNTION AND ITS ROLE		
06	09.09.25	IMPLEMENT GRADIENT DESCENT & BACKPROPAGATION IN DNN		
07	16.09.25	BUILD A CNN MODEL TO CLASSIFY CAT & DOG IMAGE		
09	09.10.25	BUILD A RECURRENT NEURAL NETWORK		
08	09.10.25	EXPERIMENT USING LSTM ON IMDB DATASET		
10	09.10.25	PERFORM COMPRESSION ON MNIST DATASET USING AUTO ENCODER		
11	27.10.25	EXPERIMENT USING VAE		
12	27.10.25	IMPLEMENT A DEEP CONVOLUTIONAL GAN TO GENERATE COMPLEX IMAGES		
13	27.10.25	UNDERSTADING THE ARCHITECTURE OF PRE TRAINED MODEL		
14	27.10.25	IMPLEMENT A PRE TRAINED CNN MODEL AS A FEATURE USINGTLM		
15	27.10.25	IMPLEMENT A YOLO MODEL TO DETECT OBJECTS		