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.

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.

output

epoch 1, loss : 0.8848
epoch 2, loss : 0.6029
epoch 3, loss : 0.5937
epoch 4, loss : 0.5797
epoch 5, loss : 0.5699

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Commands + Code + Text ▷ Run all

RAM Disk

Share

[1]

```
# Lab 14: Implement a Pre-trained CNN model as a Feature Extractor
# ================================================================

import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import models, datasets, transforms
from torch.utils.data import DataLoader
import matplotlib.pyplot as plt
import os

# ---------------------------
# Step 1: Define Transformations
# ---------------------------
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
])
```

Variables Terminal

3:15 PM Python 3

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Share

Commands + Code + Text ▷ Run all

RAM Disk

[1]

```
for epoch in range(num_epochs):
    running_loss = 0.0
    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        running_loss += loss.item()

    avg_loss = running_loss / len(train_loader)
    train_loss_list.append(avg_loss)
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {avg_loss:.4f}")

print("\n✅ Training completed successfully!")

# --------------------------
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

3:15 PM Python 3

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Share

Commands + Code + Text ▷ Run all

RAM Disk

[1]

```
num_features = model.fc.in_features
model.fc = nn.Linear(num_features, 10)  # CIFAR-10 has 10 classes

# Move model to device
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = model.to(device)

# ---------------------------
# Step 4: Define Loss and Optimizer
# ---------------------------
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.fc.parameters(), lr=0.001)

# ---------------------------
# Step 5: Train Only the Classifier
# ---------------------------
num_epochs = 2  # keep small for quick run
train_loss_list = []
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

3:15 PM Python 3

ENG IN 16:03 02-11-2025

https://colab.research.google.com/drive/1zUDZXYMWADVM-GXpJDmbO2CQnWDSVIXm#scrollTo=7dEqFQWWk6ls

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Commands + Code + Text ▷ Run all

RAM Disk

Share

[1]

```
# Download a small example dataset (CIFAR-10)
train_data = datasets.CIFAR10(root='./data', train=True, download=True, transform=transform)
test_data = datasets.CIFAR10(root='./data', train=False, download=True, transform=transform)

train_loader = DataLoader(train_data, batch_size=32, shuffle=True)
test_loader = DataLoader(test_data, batch_size=32, shuffle=False)

# --------------------------
# Step 2: Load Pre-trained Model
# --------------------------
model = models.resnet18(pretrained=True)

# Freeze all convolutional layers
for param in model.parameters():
    param.requires_grad = False

# --------------------------
# Step 3: Replace the Classifier
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

3:15 PM Python 3

ENG IN 16:03 02-11-2025

https://colab.research.google.com/drive/1zUDZXYMWADVM-GXpJDmbO2CQnWDSVIXm#scrollTo=7dEqFQWWk6ls

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

Untitled1.ipynb Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

RAM Disk

Share

Variables Terminal

3:15 PM Python 3

ENG IN 16:05 02-11-2025

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Commands + Code + Text ▷ Run all

RAM Disk

Share

[1]

```
        _, predicted = torch.max(outputs, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

accuracy = 100 * correct / total
print(f"\n Test Accuracy: {accuracy:.2f}%")
```

```
100%|████████| 170M/170M [00:10<00:00, 16.0MB/s]
/usr/local/lib/python3.12/dist-packages/torchvision/models/_utils.py:208: UserWarning: The parameter 'pretrained' is deprecat
  warnings.warn(
/usr/local/lib/python3.12/dist-packages/torchvision/models/_utils.py:223: UserWarning: Arguments other than a weight enum or
  warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f3707
100%|████████| 44.7M/44.7M [00:00<00:00, 156MB/s]
Epoch [1/2], Loss: 0.8107
Epoch [2/2], Loss: 0.6423

✅ Training completed successfully!
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

3:15 PM Python 3

Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Commands + Code + Text ▷ Run all

RAM Disk

Share

[1]

```
plt.plot(train_loss_list, marker='o')
plt.title("Training Loss per Epoch")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.grid(True)
plt.show()

# --------------------------
# Step 7: Test Accuracy
# --------------------------
correct = 0
total = 0
model.eval()
with torch.no_grad():
    for images, labels in test_loader:
        images, labels = images.to(device), labels.to(device)
        outputs = model(images)
        _, predicted = torch.max(outputs, 1)
        total += labels.size(0)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

3:15 PM Python 3

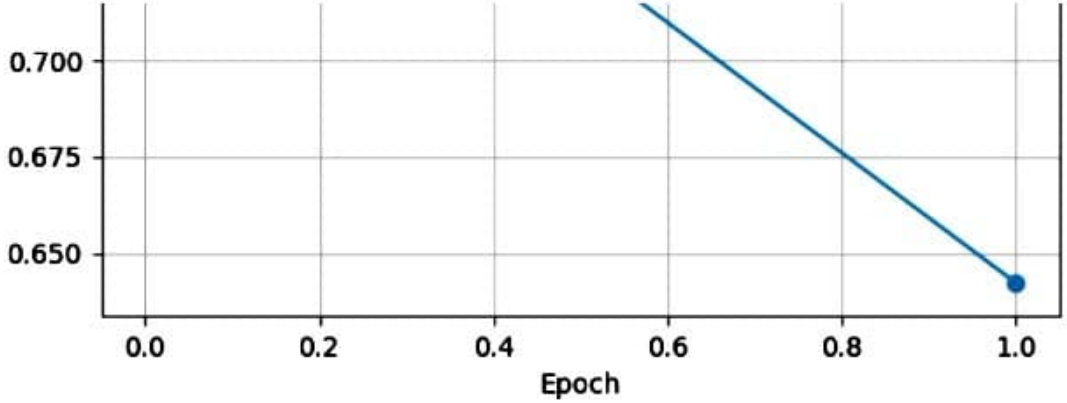
Untitled1.ipynb ☆ Saving failed since 3:54 PM

File Edit View Insert Runtime Tools Help

Share

Commands + Code + Text ▶ Run all

RAM Disk




```
Test Accuracy: 79.21%
```

[ ] Start coding or generate with AI.

Variables Terminal 3:15 PM Python 3