scenerecognition

December 16, 2023

0.1 Importing Libraries

[1]:
```
############################
## Import Torch Libaries
import torch
import torch.nn as nn
from torch.utils.data import Dataset
from torchvision import transforms
import torch.nn.functional as F
import torch.optim as optim
from torch.utils.data import DataLoader
from torch.utils.data import random_split
from torchvision import transforms
from torchvision.datasets import ImageFolder
############################
## Import standard libaries
import pandas as pd
import numpy as np
import os
import random
#########################
## Import PIL
from PIL import Image

#####################
## import SK learn libaries
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score, recall_score, accuracy_score,␣
 ↪f1_score
###################
## import matplot
import matplotlib.pyplot as plt
```

[2]:
```
#####################
##Enable CUDA
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

0.2 Figuring out the dimension of the images

[3]:
```
folder_path = './Images'

# Define a transform without resizing (to maintain original dimensions)
transform = transforms.Compose([transforms.ToTensor()])

# Create an ImageFolder dataset
dataset = ImageFolder(root=folder_path, transform=transform)

# Choose an index for the image you want to check
index_to_check = 0

# Retrieve the image and its dimensions
image, label = dataset[index_to_check]
original_dimensions = image.shape

# Print the original dimensions
print(f"Original dimensions of the image: {original_dimensions}")
```

```
Original dimensions of the image: torch.Size([3, 256, 256])
```

0.3 View Class

[4]:
```
class View(nn.Module):

    def __init__(self, shape):

        super().__init__()
        self.shape = shape,

    def forward(self, x):
        return x.view(*self.shape)
```

0.4 Mapping labels with images

[5]:
```
class SceneDataset(Dataset):
    def __init__(self, data_path, transform=None):
        self.data_path = data_path
        self.transform = transform
        self.classes = os.listdir(data_path)
        self.images = []
        self.labels = []

        for i, scene_class in enumerate(self.classes):
            class_path = os.path.join(data_path, scene_class)
            for image_name in os.listdir(class_path):
```

```
                image_path = os.path.join(class_path, image_name)
                self.images.append(image_path)
                self.labels.append(i)

    def __len__(self):
        return len(self.images)

    def __getitem__(self, index):
        image_path = self.images[index]
        label = self.labels[index]

        # Open the image
        image = Image.open(image_path).convert("RGB")

        if self.transform:
            image = self.transform(image)

        # Convert label to a single index
        label = torch.tensor(label, dtype=torch.long)

        return image, label
```

[6]:
```
data_path = './Images'
transform = transforms.Compose([
    transforms.Resize((48, 48)),
    transforms.ToTensor(),
])

scene_dataset = SceneDataset(data_path=data_path, transform=transform)
```

[7]:
```
index = 0
image, label = scene_dataset[index]
```

[8]:
```
image.shape
```

[8]:
```
torch.Size([3, 48, 48])
```

[9]:
```
# label are based on the directory name which means 0 is the first folder name,␣
 ↪1 is the second folder name, etc.
label
```

[9]:
```
tensor(0)
```

0.5 Model

[10]:
```
class Classifier_CNN_3_Layers(nn.Module):
    def __init__(self):
        super(Classifier_CNN_3_Layers, self).__init__()
        self.conv1 = nn.Conv2d(3, 64, kernel_size=3, stride=1, padding=1)
        self.pool1= nn.MaxPool2d(kernel_size=2, stride=2)
        self.conv2 = nn.Conv2d(64, 128, kernel_size=3, stride=1, padding=1)
        self.pool2 = nn.MaxPool2d(kernel_size=2, stride=2)
        self.conv3 = nn.Conv2d(128, 256, kernel_size=3, padding=1)
        self.pool3 = nn.MaxPool2d(kernel_size=2, stride=2)
        self.relu = nn.ReLU()
        self.fc1 = nn.Linear(256 * 6 * 6, 512)
        self.fc2 = nn.Linear(512, 4)

    def forward(self, x):
        x = self.conv1(x)
        x = self.relu(x)
        x = self.pool1(x)

        x = self.conv2(x)
        x = self.relu(x)
        x = self.pool2(x)

        x = self.conv3(x)
        x = self.relu(x)
        x = self.pool3(x)

        x = x.view(-1, 256 * 6 * 6)
        x = self.fc1(x)
        x = self.relu(x)
        x = self.fc2(x)

        return F.softmax(x, dim=1)
```

0.6 Splitting the dataset into training and testing

[11]:
```
train_size = int(0.8 * len(scene_dataset))
test_size = len(scene_dataset) - train_size
train_dataset, test_dataset = random_split(scene_dataset, [train_size,␣
↪test_size])
```

0.7 Creating dataloader

[12]:
```
batch_size = 32

train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False)
```

[13]:
```
%%time
cnn_classifier = Classifier_CNN_3_Layers()

optimizer = optim.SGD(cnn_classifier.parameters(), lr=0.1)
criterion = nn.CrossEntropyLoss()

epochs = 10

for epoch in range(epochs):
    # Training
    cnn_classifier.train()
    for i, (images, labels) in enumerate(train_loader):
        images, labels = images.to(device), labels.to(device)
        optimizer.zero_grad()
        outputs = cnn_classifier(images)

        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        print(f"Epoch {epoch + 1}/{epochs}, Iteration {i + 1}/
 ↪{len(train_loader)}, Loss: {loss.item()}")
```

```
Epoch 1/10, Iteration 1/51, Loss: 1.3863255977630615
Epoch 1/10, Iteration 2/51, Loss: 1.3855677843093872
Epoch 1/10, Iteration 3/51, Loss: 1.3867197036743164
Epoch 1/10, Iteration 4/51, Loss: 1.3860418796539307
Epoch 1/10, Iteration 5/51, Loss: 1.388322353363037
Epoch 1/10, Iteration 6/51, Loss: 1.3853520154953003
Epoch 1/10, Iteration 7/51, Loss: 1.386749029159546
Epoch 1/10, Iteration 8/51, Loss: 1.3875092267990112
Epoch 1/10, Iteration 9/51, Loss: 1.3873761892318726
Epoch 1/10, Iteration 10/51, Loss: 1.3854031562805176
Epoch 1/10, Iteration 11/51, Loss: 1.3869595527648926
Epoch 1/10, Iteration 12/51, Loss: 1.386850118637085
Epoch 1/10, Iteration 13/51, Loss: 1.3857885599136353
Epoch 1/10, Iteration 14/51, Loss: 1.38649582862854
Epoch 1/10, Iteration 15/51, Loss: 1.3859394788742065
Epoch 1/10, Iteration 16/51, Loss: 1.3856518268585205
Epoch 1/10, Iteration 17/51, Loss: 1.3859920501708984
Epoch 1/10, Iteration 18/51, Loss: 1.385383129119873
Epoch 1/10, Iteration 19/51, Loss: 1.385954737663269
```

```
Epoch 1/10, Iteration 20/51, Loss: 1.385561227798462
Epoch 1/10, Iteration 21/51, Loss: 1.3856768608093262
Epoch 1/10, Iteration 22/51, Loss: 1.3852664232254028
Epoch 1/10, Iteration 23/51, Loss: 1.3865736722946167
Epoch 1/10, Iteration 24/51, Loss: 1.3864105939865112
Epoch 1/10, Iteration 25/51, Loss: 1.385994553565979
Epoch 1/10, Iteration 26/51, Loss: 1.3862547874450684
Epoch 1/10, Iteration 27/51, Loss: 1.38524329662323
Epoch 1/10, Iteration 28/51, Loss: 1.3855466842651367
Epoch 1/10, Iteration 29/51, Loss: 1.3842264413833618
Epoch 1/10, Iteration 30/51, Loss: 1.3847676515579224
Epoch 1/10, Iteration 31/51, Loss: 1.3842161893844604
Epoch 1/10, Iteration 32/51, Loss: 1.385377287864685
Epoch 1/10, Iteration 33/51, Loss: 1.3867875337600708
Epoch 1/10, Iteration 34/51, Loss: 1.3848837614059448
Epoch 1/10, Iteration 35/51, Loss: 1.3853428363800049
Epoch 1/10, Iteration 36/51, Loss: 1.3851298093795776
Epoch 1/10, Iteration 37/51, Loss: 1.3876628875732422
Epoch 1/10, Iteration 38/51, Loss: 1.3834290504455566
Epoch 1/10, Iteration 39/51, Loss: 1.3894826173782349
Epoch 1/10, Iteration 40/51, Loss: 1.385858178138733
Epoch 1/10, Iteration 41/51, Loss: 1.386263132095337
Epoch 1/10, Iteration 42/51, Loss: 1.3859691619873047
Epoch 1/10, Iteration 43/51, Loss: 1.3854432106018066
Epoch 1/10, Iteration 44/51, Loss: 1.3853274583816528
Epoch 1/10, Iteration 45/51, Loss: 1.3854448795318604
Epoch 1/10, Iteration 46/51, Loss: 1.3843648433685303
Epoch 1/10, Iteration 47/51, Loss: 1.385180115699768
Epoch 1/10, Iteration 48/51, Loss: 1.3852659463882446
Epoch 1/10, Iteration 49/51, Loss: 1.3856571912765503
Epoch 1/10, Iteration 50/51, Loss: 1.3871879577636719
Epoch 1/10, Iteration 51/51, Loss: 1.3834770917892456
Epoch 2/10, Iteration 1/51, Loss: 1.386706829071045
Epoch 2/10, Iteration 2/51, Loss: 1.3826663494110107
Epoch 2/10, Iteration 3/51, Loss: 1.3862252235412598
Epoch 2/10, Iteration 4/51, Loss: 1.3838086128234863
Epoch 2/10, Iteration 5/51, Loss: 1.3845325708389282
Epoch 2/10, Iteration 6/51, Loss: 1.3841279745101929
Epoch 2/10, Iteration 7/51, Loss: 1.3854058980941772
Epoch 2/10, Iteration 8/51, Loss: 1.3852262496948242
Epoch 2/10, Iteration 9/51, Loss: 1.384321928024292
Epoch 2/10, Iteration 10/51, Loss: 1.3834251165390015
Epoch 2/10, Iteration 11/51, Loss: 1.3860976696014404
Epoch 2/10, Iteration 12/51, Loss: 1.385496973991394
Epoch 2/10, Iteration 13/51, Loss: 1.380041241645813
Epoch 2/10, Iteration 14/51, Loss: 1.3856678009033203
Epoch 2/10, Iteration 15/51, Loss: 1.3837108612060547
Epoch 2/10, Iteration 16/51, Loss: 1.3897407054901123
```

```
Epoch 2/10, Iteration 17/51, Loss: 1.38555109500885
Epoch 2/10, Iteration 18/51, Loss: 1.3834830522537231
Epoch 2/10, Iteration 19/51, Loss: 1.3836373090744019
Epoch 2/10, Iteration 20/51, Loss: 1.3901963233947754
Epoch 2/10, Iteration 21/51, Loss: 1.3814423084259033
Epoch 2/10, Iteration 22/51, Loss: 1.3796865940093994
Epoch 2/10, Iteration 23/51, Loss: 1.3868986368179321
Epoch 2/10, Iteration 24/51, Loss: 1.386818528175354
Epoch 2/10, Iteration 25/51, Loss: 1.3880821466445923
Epoch 2/10, Iteration 26/51, Loss: 1.3858022689819336
Epoch 2/10, Iteration 27/51, Loss: 1.3832712173461914
Epoch 2/10, Iteration 28/51, Loss: 1.380305290222168
Epoch 2/10, Iteration 29/51, Loss: 1.3830244541168213
Epoch 2/10, Iteration 30/51, Loss: 1.386609673500061
Epoch 2/10, Iteration 31/51, Loss: 1.3843801021575928
Epoch 2/10, Iteration 32/51, Loss: 1.3818551301956177
Epoch 2/10, Iteration 33/51, Loss: 1.387414813041687
Epoch 2/10, Iteration 34/51, Loss: 1.3840992450714111
Epoch 2/10, Iteration 35/51, Loss: 1.3844940662384033
Epoch 2/10, Iteration 36/51, Loss: 1.3838597536087036
Epoch 2/10, Iteration 37/51, Loss: 1.3801897764205933
Epoch 2/10, Iteration 38/51, Loss: 1.3836179971694946
Epoch 2/10, Iteration 39/51, Loss: 1.3805738687515259
Epoch 2/10, Iteration 40/51, Loss: 1.3865573406219482
Epoch 2/10, Iteration 41/51, Loss: 1.3847932815551758
Epoch 2/10, Iteration 42/51, Loss: 1.3850892782211304
Epoch 2/10, Iteration 43/51, Loss: 1.3848234415054321
Epoch 2/10, Iteration 44/51, Loss: 1.3829818964004517
Epoch 2/10, Iteration 45/51, Loss: 1.380954384803772
Epoch 2/10, Iteration 46/51, Loss: 1.3831086158752441
Epoch 2/10, Iteration 47/51, Loss: 1.3849384784698486
Epoch 2/10, Iteration 48/51, Loss: 1.3841074705123901
Epoch 2/10, Iteration 49/51, Loss: 1.3813765048980713
Epoch 2/10, Iteration 50/51, Loss: 1.3799351453781128
Epoch 2/10, Iteration 51/51, Loss: 1.392775535583496
Epoch 3/10, Iteration 1/51, Loss: 1.374671459197998
Epoch 3/10, Iteration 2/51, Loss: 1.3859714269638062
Epoch 3/10, Iteration 3/51, Loss: 1.3847641944885254
Epoch 3/10, Iteration 4/51, Loss: 1.3763930797576904
Epoch 3/10, Iteration 5/51, Loss: 1.3811434507369995
Epoch 3/10, Iteration 6/51, Loss: 1.3838255405426025
Epoch 3/10, Iteration 7/51, Loss: 1.383687138557434
Epoch 3/10, Iteration 8/51, Loss: 1.3823559284210205
Epoch 3/10, Iteration 9/51, Loss: 1.3780770301818848
Epoch 3/10, Iteration 10/51, Loss: 1.3874449729919434
Epoch 3/10, Iteration 11/51, Loss: 1.384675145149231
Epoch 3/10, Iteration 12/51, Loss: 1.381494164466858
Epoch 3/10, Iteration 13/51, Loss: 1.377591848373413
```

```
Epoch 3/10, Iteration 14/51, Loss: 1.3847477436065674
Epoch 3/10, Iteration 15/51, Loss: 1.3838427066802979
Epoch 3/10, Iteration 16/51, Loss: 1.380424976348877
Epoch 3/10, Iteration 17/51, Loss: 1.3824347257614136
Epoch 3/10, Iteration 18/51, Loss: 1.3731622695922852
Epoch 3/10, Iteration 19/51, Loss: 1.3878134489059448
Epoch 3/10, Iteration 20/51, Loss: 1.3768266439437866
Epoch 3/10, Iteration 21/51, Loss: 1.3836740255355835
Epoch 3/10, Iteration 22/51, Loss: 1.3677666187286377
Epoch 3/10, Iteration 23/51, Loss: 1.382016897201538
Epoch 3/10, Iteration 24/51, Loss: 1.374485969543457
Epoch 3/10, Iteration 25/51, Loss: 1.3854750394821167
Epoch 3/10, Iteration 26/51, Loss: 1.3755673170089722
Epoch 3/10, Iteration 27/51, Loss: 1.3841779232025146
Epoch 3/10, Iteration 28/51, Loss: 1.3918694257736206
Epoch 3/10, Iteration 29/51, Loss: 1.3764839172363281
Epoch 3/10, Iteration 30/51, Loss: 1.3903992176055908
Epoch 3/10, Iteration 31/51, Loss: 1.3788412809371948
Epoch 3/10, Iteration 32/51, Loss: 1.3703073263168335
Epoch 3/10, Iteration 33/51, Loss: 1.3996269702911377
Epoch 3/10, Iteration 34/51, Loss: 1.3705449104309082
Epoch 3/10, Iteration 35/51, Loss: 1.395817756652832
Epoch 3/10, Iteration 36/51, Loss: 1.3797526359558105
Epoch 3/10, Iteration 37/51, Loss: 1.3840075731277466
Epoch 3/10, Iteration 38/51, Loss: 1.383246898651123
Epoch 3/10, Iteration 39/51, Loss: 1.378709316253662
Epoch 3/10, Iteration 40/51, Loss: 1.3863223791122437
Epoch 3/10, Iteration 41/51, Loss: 1.3843035697937012
Epoch 3/10, Iteration 42/51, Loss: 1.371706485748291
Epoch 3/10, Iteration 43/51, Loss: 1.392154574394226
Epoch 3/10, Iteration 44/51, Loss: 1.3747234344482422
Epoch 3/10, Iteration 45/51, Loss: 1.3880559206008911
Epoch 3/10, Iteration 46/51, Loss: 1.3801298141479492
Epoch 3/10, Iteration 47/51, Loss: 1.3856104612350464
Epoch 3/10, Iteration 48/51, Loss: 1.3767240047454834
Epoch 3/10, Iteration 49/51, Loss: 1.3833792209625244
Epoch 3/10, Iteration 50/51, Loss: 1.3769564628601074
Epoch 3/10, Iteration 51/51, Loss: 1.3740026950836182
Epoch 4/10, Iteration 1/51, Loss: 1.38695228099823
Epoch 4/10, Iteration 2/51, Loss: 1.3781960010528564
Epoch 4/10, Iteration 3/51, Loss: 1.379722237586975
Epoch 4/10, Iteration 4/51, Loss: 1.3825886249542236
Epoch 4/10, Iteration 5/51, Loss: 1.3844343423843384
Epoch 4/10, Iteration 6/51, Loss: 1.3740800619125366
Epoch 4/10, Iteration 7/51, Loss: 1.3754298686981201
Epoch 4/10, Iteration 8/51, Loss: 1.3853108882904053
Epoch 4/10, Iteration 9/51, Loss: 1.384234070777893
Epoch 4/10, Iteration 10/51, Loss: 1.3923641443252563
```

```
Epoch 4/10, Iteration 11/51, Loss: 1.3784681558609009
Epoch 4/10, Iteration 12/51, Loss: 1.38225257396698
Epoch 4/10, Iteration 13/51, Loss: 1.3804545402526855
Epoch 4/10, Iteration 14/51, Loss: 1.3709526062011719
Epoch 4/10, Iteration 15/51, Loss: 1.3718726634979248
Epoch 4/10, Iteration 16/51, Loss: 1.374783992767334
Epoch 4/10, Iteration 17/51, Loss: 1.3828010559082031
Epoch 4/10, Iteration 18/51, Loss: 1.3762835264205933
Epoch 4/10, Iteration 19/51, Loss: 1.3833940029144287
Epoch 4/10, Iteration 20/51, Loss: 1.3647631406784058
Epoch 4/10, Iteration 21/51, Loss: 1.3743025064468384
Epoch 4/10, Iteration 22/51, Loss: 1.3922361135482788
Epoch 4/10, Iteration 23/51, Loss: 1.3598263263702393
Epoch 4/10, Iteration 24/51, Loss: 1.3881330490112305
Epoch 4/10, Iteration 25/51, Loss: 1.3705641031265259
Epoch 4/10, Iteration 26/51, Loss: 1.3760534524917603
Epoch 4/10, Iteration 27/51, Loss: 1.3700560331344604
Epoch 4/10, Iteration 28/51, Loss: 1.3696249723434448
Epoch 4/10, Iteration 29/51, Loss: 1.3703316450119019
Epoch 4/10, Iteration 30/51, Loss: 1.362291932106018
Epoch 4/10, Iteration 31/51, Loss: 1.3756262063980103
Epoch 4/10, Iteration 32/51, Loss: 1.380618691444397
Epoch 4/10, Iteration 33/51, Loss: 1.3810791969299316
Epoch 4/10, Iteration 34/51, Loss: 1.3694038391113281
Epoch 4/10, Iteration 35/51, Loss: 1.3469326496124268
Epoch 4/10, Iteration 36/51, Loss: 1.391782522201538
Epoch 4/10, Iteration 37/51, Loss: 1.370482087135315
Epoch 4/10, Iteration 38/51, Loss: 1.3581230640411377
Epoch 4/10, Iteration 39/51, Loss: 1.3525081872940063
Epoch 4/10, Iteration 40/51, Loss: 1.3674906492233276
Epoch 4/10, Iteration 41/51, Loss: 1.3564096689224243
Epoch 4/10, Iteration 42/51, Loss: 1.3760509490966797
Epoch 4/10, Iteration 43/51, Loss: 1.3564112186431885
Epoch 4/10, Iteration 44/51, Loss: 1.3781852722167969
Epoch 4/10, Iteration 45/51, Loss: 1.355769395828247
Epoch 4/10, Iteration 46/51, Loss: 1.3583849668502808
Epoch 4/10, Iteration 47/51, Loss: 1.3873332738876343
Epoch 4/10, Iteration 48/51, Loss: 1.3553041219711304
Epoch 4/10, Iteration 49/51, Loss: 1.3719290494918823
Epoch 4/10, Iteration 50/51, Loss: 1.3684450387954712
Epoch 4/10, Iteration 51/51, Loss: 1.3892581462860107
Epoch 5/10, Iteration 1/51, Loss: 1.3799145221710205
Epoch 5/10, Iteration 2/51, Loss: 1.3677353858947754
Epoch 5/10, Iteration 3/51, Loss: 1.3918260335922241
Epoch 5/10, Iteration 4/51, Loss: 1.3629629611968994
Epoch 5/10, Iteration 5/51, Loss: 1.3507184982299805
Epoch 5/10, Iteration 6/51, Loss: 1.3626960515975952
Epoch 5/10, Iteration 7/51, Loss: 1.4026674032211304
```

```
Epoch 5/10, Iteration 8/51, Loss: 1.3581939935684204
Epoch 5/10, Iteration 9/51, Loss: 1.3918421268463135
Epoch 5/10, Iteration 10/51, Loss: 1.362274408340454
Epoch 5/10, Iteration 11/51, Loss: 1.36281418800354
Epoch 5/10, Iteration 12/51, Loss: 1.4139680862426758
Epoch 5/10, Iteration 13/51, Loss: 1.387580394744873
Epoch 5/10, Iteration 14/51, Loss: 1.3458096981048584
Epoch 5/10, Iteration 15/51, Loss: 1.3349955081939697
Epoch 5/10, Iteration 16/51, Loss: 1.3977874517440796
Epoch 5/10, Iteration 17/51, Loss: 1.3721944093704224
Epoch 5/10, Iteration 18/51, Loss: 1.3555183410644531
Epoch 5/10, Iteration 19/51, Loss: 1.3697714805603027
Epoch 5/10, Iteration 20/51, Loss: 1.3797862529754639
Epoch 5/10, Iteration 21/51, Loss: 1.3577760457992554
Epoch 5/10, Iteration 22/51, Loss: 1.3776450157165527
Epoch 5/10, Iteration 23/51, Loss: 1.3634672164916992
Epoch 5/10, Iteration 24/51, Loss: 1.3700693845748901
Epoch 5/10, Iteration 25/51, Loss: 1.3741295337677002
Epoch 5/10, Iteration 26/51, Loss: 1.3552683591842651
Epoch 5/10, Iteration 27/51, Loss: 1.4185105562210083
Epoch 5/10, Iteration 28/51, Loss: 1.3377041816711426
Epoch 5/10, Iteration 29/51, Loss: 1.3273518085479736
Epoch 5/10, Iteration 30/51, Loss: 1.360275149345398
Epoch 5/10, Iteration 31/51, Loss: 1.370519995689392
Epoch 5/10, Iteration 32/51, Loss: 1.3578639030456543
Epoch 5/10, Iteration 33/51, Loss: 1.3434356451034546
Epoch 5/10, Iteration 34/51, Loss: 1.3811633586883545
Epoch 5/10, Iteration 35/51, Loss: 1.3548251390457153
Epoch 5/10, Iteration 36/51, Loss: 1.392517328262329
Epoch 5/10, Iteration 37/51, Loss: 1.3487628698349
Epoch 5/10, Iteration 38/51, Loss: 1.345414638519287
Epoch 5/10, Iteration 39/51, Loss: 1.3796813488006592
Epoch 5/10, Iteration 40/51, Loss: 1.335068941116333
Epoch 5/10, Iteration 41/51, Loss: 1.3797965049743652
Epoch 5/10, Iteration 42/51, Loss: 1.352555274963379
Epoch 5/10, Iteration 43/51, Loss: 1.3428033590316772
Epoch 5/10, Iteration 44/51, Loss: 1.3220868110656738
Epoch 5/10, Iteration 45/51, Loss: 1.3627904653549194
Epoch 5/10, Iteration 46/51, Loss: 1.3147170543670654
Epoch 5/10, Iteration 47/51, Loss: 1.3411952257156372
Epoch 5/10, Iteration 48/51, Loss: 1.3689690828323364
Epoch 5/10, Iteration 49/51, Loss: 1.3488759994506836
Epoch 5/10, Iteration 50/51, Loss: 1.3226161003112793
Epoch 5/10, Iteration 51/51, Loss: 1.3463720083236694
Epoch 6/10, Iteration 1/51, Loss: 1.357224702835083
Epoch 6/10, Iteration 2/51, Loss: 1.3998193740844727
Epoch 6/10, Iteration 3/51, Loss: 1.3584829568862915
Epoch 6/10, Iteration 4/51, Loss: 1.340855360031128
```

```
Epoch 6/10, Iteration 5/51, Loss: 1.313616394996643
Epoch 6/10, Iteration 6/51, Loss: 1.379714846611023
Epoch 6/10, Iteration 7/51, Loss: 1.3791396617889404
Epoch 6/10, Iteration 8/51, Loss: 1.3247158527374268
Epoch 6/10, Iteration 9/51, Loss: 1.3442461490631104
Epoch 6/10, Iteration 10/51, Loss: 1.368298888206482
Epoch 6/10, Iteration 11/51, Loss: 1.3233336210250854
Epoch 6/10, Iteration 12/51, Loss: 1.338156819343567
Epoch 6/10, Iteration 13/51, Loss: 1.374997615814209
Epoch 6/10, Iteration 14/51, Loss: 1.336864948272705
Epoch 6/10, Iteration 15/51, Loss: 1.3556426763534546
Epoch 6/10, Iteration 16/51, Loss: 1.3166756629943848
Epoch 6/10, Iteration 17/51, Loss: 1.3457545042037964
Epoch 6/10, Iteration 18/51, Loss: 1.3922901153564453
Epoch 6/10, Iteration 19/51, Loss: 1.3908545970916748
Epoch 6/10, Iteration 20/51, Loss: 1.3513180017471313
Epoch 6/10, Iteration 21/51, Loss: 1.3240519762039185
Epoch 6/10, Iteration 22/51, Loss: 1.3897936344146729
Epoch 6/10, Iteration 23/51, Loss: 1.3386698961257935
Epoch 6/10, Iteration 24/51, Loss: 1.3199931383132935
Epoch 6/10, Iteration 25/51, Loss: 1.334703803062439
Epoch 6/10, Iteration 26/51, Loss: 1.3608640432357788
Epoch 6/10, Iteration 27/51, Loss: 1.3763443231582642
Epoch 6/10, Iteration 28/51, Loss: 1.4014699459075928
Epoch 6/10, Iteration 29/51, Loss: 1.369899868965149
Epoch 6/10, Iteration 30/51, Loss: 1.337683081626892
Epoch 6/10, Iteration 31/51, Loss: 1.304563045501709
Epoch 6/10, Iteration 32/51, Loss: 1.36648690700531
Epoch 6/10, Iteration 33/51, Loss: 1.3582866191864014
Epoch 6/10, Iteration 34/51, Loss: 1.343612551689148
Epoch 6/10, Iteration 35/51, Loss: 1.3855332136154175
Epoch 6/10, Iteration 36/51, Loss: 1.290418267250061
Epoch 6/10, Iteration 37/51, Loss: 1.3217931985855103
Epoch 6/10, Iteration 38/51, Loss: 1.39431631565094
Epoch 6/10, Iteration 39/51, Loss: 1.3460050821304321
Epoch 6/10, Iteration 40/51, Loss: 1.3594471216201782
Epoch 6/10, Iteration 41/51, Loss: 1.3094661235809326
Epoch 6/10, Iteration 42/51, Loss: 1.2947744131088257
Epoch 6/10, Iteration 43/51, Loss: 1.4540646076202393
Epoch 6/10, Iteration 44/51, Loss: 1.360634446144104
Epoch 6/10, Iteration 45/51, Loss: 1.3699809312820435
Epoch 6/10, Iteration 46/51, Loss: 1.3548210859298706
Epoch 6/10, Iteration 47/51, Loss: 1.3326880931854248
Epoch 6/10, Iteration 48/51, Loss: 1.339020848274231
Epoch 6/10, Iteration 49/51, Loss: 1.3182870149612427
Epoch 6/10, Iteration 50/51, Loss: 1.3451390266418457
Epoch 6/10, Iteration 51/51, Loss: 1.3218929767608643
Epoch 7/10, Iteration 1/51, Loss: 1.2971755266189575
```

```
Epoch 7/10, Iteration 2/51, Loss: 1.30586576461792
Epoch 7/10, Iteration 3/51, Loss: 1.3142163753509521
Epoch 7/10, Iteration 4/51, Loss: 1.3930367231369019
Epoch 7/10, Iteration 5/51, Loss: 1.3370604515075684
Epoch 7/10, Iteration 6/51, Loss: 1.3087828159332275
Epoch 7/10, Iteration 7/51, Loss: 1.3093476295471191
Epoch 7/10, Iteration 8/51, Loss: 1.4130712747573853
Epoch 7/10, Iteration 9/51, Loss: 1.3318830728530884
Epoch 7/10, Iteration 10/51, Loss: 1.2949652671813965
Epoch 7/10, Iteration 11/51, Loss: 1.2842668294906616
Epoch 7/10, Iteration 12/51, Loss: 1.3206874132156372
Epoch 7/10, Iteration 13/51, Loss: 1.2979750633239746
Epoch 7/10, Iteration 14/51, Loss: 1.3760366439819336
Epoch 7/10, Iteration 15/51, Loss: 1.377354621887207
Epoch 7/10, Iteration 16/51, Loss: 1.3734169006347656
Epoch 7/10, Iteration 17/51, Loss: 1.325095295906067
Epoch 7/10, Iteration 18/51, Loss: 1.3422292470932007
Epoch 7/10, Iteration 19/51, Loss: 1.3051929473876953
Epoch 7/10, Iteration 20/51, Loss: 1.2023259401321411
Epoch 7/10, Iteration 21/51, Loss: 1.2931722402572632
Epoch 7/10, Iteration 22/51, Loss: 1.357719898223877
Epoch 7/10, Iteration 23/51, Loss: 1.3676592111587524
Epoch 7/10, Iteration 24/51, Loss: 1.3557571172714233
Epoch 7/10, Iteration 25/51, Loss: 1.436278223991394
Epoch 7/10, Iteration 26/51, Loss: 1.3319933414459229
Epoch 7/10, Iteration 27/51, Loss: 1.336578130722046
Epoch 7/10, Iteration 28/51, Loss: 1.300376057624817
Epoch 7/10, Iteration 29/51, Loss: 1.2470815181732178
Epoch 7/10, Iteration 30/51, Loss: 1.2832744121551514
Epoch 7/10, Iteration 31/51, Loss: 1.2822461128234863
Epoch 7/10, Iteration 32/51, Loss: 1.35573410987854
Epoch 7/10, Iteration 33/51, Loss: 1.3006446361541748
Epoch 7/10, Iteration 34/51, Loss: 1.4065918922424316
Epoch 7/10, Iteration 35/51, Loss: 1.3486359119415283
Epoch 7/10, Iteration 36/51, Loss: 1.3573676347732544
Epoch 7/10, Iteration 37/51, Loss: 1.282822847366333
Epoch 7/10, Iteration 38/51, Loss: 1.3338501453399658
Epoch 7/10, Iteration 39/51, Loss: 1.380858302116394
Epoch 7/10, Iteration 40/51, Loss: 1.324094533920288
Epoch 7/10, Iteration 41/51, Loss: 1.3466840982437134
Epoch 7/10, Iteration 42/51, Loss: 1.315816044807434
Epoch 7/10, Iteration 43/51, Loss: 1.4126423597335815
Epoch 7/10, Iteration 44/51, Loss: 1.2622703313827515
Epoch 7/10, Iteration 45/51, Loss: 1.258102297782898
Epoch 7/10, Iteration 46/51, Loss: 1.2634211778640747
Epoch 7/10, Iteration 47/51, Loss: 1.2739360332489014
Epoch 7/10, Iteration 48/51, Loss: 1.268617868423462
Epoch 7/10, Iteration 49/51, Loss: 1.260399580001831
```

```
Epoch 7/10, Iteration 50/51, Loss: 1.3975214958190918
Epoch 7/10, Iteration 51/51, Loss: 1.3398865461349487
Epoch 8/10, Iteration 1/51, Loss: 1.4199315309524536
Epoch 8/10, Iteration 2/51, Loss: 1.3328018188476562
Epoch 8/10, Iteration 3/51, Loss: 1.288203477859497
Epoch 8/10, Iteration 4/51, Loss: 1.3473209142684937
Epoch 8/10, Iteration 5/51, Loss: 1.3763225078582764
Epoch 8/10, Iteration 6/51, Loss: 1.3295656442642212
Epoch 8/10, Iteration 7/51, Loss: 1.3439006805419922
Epoch 8/10, Iteration 8/51, Loss: 1.3306763172149658
Epoch 8/10, Iteration 9/51, Loss: 1.2604190111160278
Epoch 8/10, Iteration 10/51, Loss: 1.3028994798660278
Epoch 8/10, Iteration 11/51, Loss: 1.335435152053833
Epoch 8/10, Iteration 12/51, Loss: 1.3502962589263916
Epoch 8/10, Iteration 13/51, Loss: 1.3996046781539917
Epoch 8/10, Iteration 14/51, Loss: 1.2909140586853027
Epoch 8/10, Iteration 15/51, Loss: 1.3518503904342651
Epoch 8/10, Iteration 16/51, Loss: 1.2913964986801147
Epoch 8/10, Iteration 17/51, Loss: 1.2828510999679565
Epoch 8/10, Iteration 18/51, Loss: 1.328622817993164
Epoch 8/10, Iteration 19/51, Loss: 1.3238328695297241
Epoch 8/10, Iteration 20/51, Loss: 1.3366901874542236
Epoch 8/10, Iteration 21/51, Loss: 1.3665302991867065
Epoch 8/10, Iteration 22/51, Loss: 1.259798288345337
Epoch 8/10, Iteration 23/51, Loss: 1.3317463397979736
Epoch 8/10, Iteration 24/51, Loss: 1.3337218761444092
Epoch 8/10, Iteration 25/51, Loss: 1.2791352272033691
Epoch 8/10, Iteration 26/51, Loss: 1.2432348728179932
Epoch 8/10, Iteration 27/51, Loss: 1.349863052368164
Epoch 8/10, Iteration 28/51, Loss: 1.3093595504760742
Epoch 8/10, Iteration 29/51, Loss: 1.3214609622955322
Epoch 8/10, Iteration 30/51, Loss: 1.2931331396102905
Epoch 8/10, Iteration 31/51, Loss: 1.324876308441162
Epoch 8/10, Iteration 32/51, Loss: 1.2938873767852783
Epoch 8/10, Iteration 33/51, Loss: 1.3703593015670776
Epoch 8/10, Iteration 34/51, Loss: 1.3775700330734253
Epoch 8/10, Iteration 35/51, Loss: 1.2799962759017944
Epoch 8/10, Iteration 36/51, Loss: 1.2782807350158691
Epoch 8/10, Iteration 37/51, Loss: 1.2019504308700562
Epoch 8/10, Iteration 38/51, Loss: 1.2993453741073608
Epoch 8/10, Iteration 39/51, Loss: 1.2979419231414795
Epoch 8/10, Iteration 40/51, Loss: 1.2729140520095825
Epoch 8/10, Iteration 41/51, Loss: 1.346835970878601
Epoch 8/10, Iteration 42/51, Loss: 1.2909259796142578
Epoch 8/10, Iteration 43/51, Loss: 1.3652023077011108
Epoch 8/10, Iteration 44/51, Loss: 1.3648829460144043
Epoch 8/10, Iteration 45/51, Loss: 1.322844386100769
Epoch 8/10, Iteration 46/51, Loss: 1.2150834798812866
```

```
Epoch 8/10, Iteration 47/51, Loss: 1.4321626424789429
Epoch 8/10, Iteration 48/51, Loss: 1.2980741262435913
Epoch 8/10, Iteration 49/51, Loss: 1.313760757446289
Epoch 8/10, Iteration 50/51, Loss: 1.3823378086090088
Epoch 8/10, Iteration 51/51, Loss: 1.3121141195297241
Epoch 9/10, Iteration 1/51, Loss: 1.387795090675354
Epoch 9/10, Iteration 2/51, Loss: 1.2375462055206299
Epoch 9/10, Iteration 3/51, Loss: 1.2976828813552856
Epoch 9/10, Iteration 4/51, Loss: 1.22669517993927
Epoch 9/10, Iteration 5/51, Loss: 1.3178385496139526
Epoch 9/10, Iteration 6/51, Loss: 1.3286579847335815
Epoch 9/10, Iteration 7/51, Loss: 1.3183794021606445
Epoch 9/10, Iteration 8/51, Loss: 1.2461178302764893
Epoch 9/10, Iteration 9/51, Loss: 1.452799916267395
Epoch 9/10, Iteration 10/51, Loss: 1.4087084531784058
Epoch 9/10, Iteration 11/51, Loss: 1.2285206317901611
Epoch 9/10, Iteration 12/51, Loss: 1.4379804134368896
Epoch 9/10, Iteration 13/51, Loss: 1.3713487386703491
Epoch 9/10, Iteration 14/51, Loss: 1.3417251110076904
Epoch 9/10, Iteration 15/51, Loss: 1.2927192449569702
Epoch 9/10, Iteration 16/51, Loss: 1.259028434753418
Epoch 9/10, Iteration 17/51, Loss: 1.327049970626831
Epoch 9/10, Iteration 18/51, Loss: 1.269821286201477
Epoch 9/10, Iteration 19/51, Loss: 1.3977148532867432
Epoch 9/10, Iteration 20/51, Loss: 1.352725863456726
Epoch 9/10, Iteration 21/51, Loss: 1.3217060565948486
Epoch 9/10, Iteration 22/51, Loss: 1.3724021911621094
Epoch 9/10, Iteration 23/51, Loss: 1.3328899145126343
Epoch 9/10, Iteration 24/51, Loss: 1.3209797143936157
Epoch 9/10, Iteration 25/51, Loss: 1.2599215507507324
Epoch 9/10, Iteration 26/51, Loss: 1.3307760953903198
Epoch 9/10, Iteration 27/51, Loss: 1.2573063373565674
Epoch 9/10, Iteration 28/51, Loss: 1.2945159673690796
Epoch 9/10, Iteration 29/51, Loss: 1.2915977239608765
Epoch 9/10, Iteration 30/51, Loss: 1.2915652990341187
Epoch 9/10, Iteration 31/51, Loss: 1.1881637573242188
Epoch 9/10, Iteration 32/51, Loss: 1.273995280265808
Epoch 9/10, Iteration 33/51, Loss: 1.3114374876022339
Epoch 9/10, Iteration 34/51, Loss: 1.235366702079773
Epoch 9/10, Iteration 35/51, Loss: 1.2101165056228638
Epoch 9/10, Iteration 36/51, Loss: 1.2567013502120972
Epoch 9/10, Iteration 37/51, Loss: 1.3101158142089844
Epoch 9/10, Iteration 38/51, Loss: 1.357598066329956
Epoch 9/10, Iteration 39/51, Loss: 1.3237708806991577
Epoch 9/10, Iteration 40/51, Loss: 1.2507705688476562
Epoch 9/10, Iteration 41/51, Loss: 1.2280998229980469
Epoch 9/10, Iteration 42/51, Loss: 1.281008243560791
Epoch 9/10, Iteration 43/51, Loss: 1.2397758960723877
```

```
Epoch 9/10, Iteration 44/51, Loss: 1.2847076654434204
Epoch 9/10, Iteration 45/51, Loss: 1.1983968019485474
Epoch 9/10, Iteration 46/51, Loss: 1.300742745399475
Epoch 9/10, Iteration 47/51, Loss: 1.3076956272125244
Epoch 9/10, Iteration 48/51, Loss: 1.2693543434143066
Epoch 9/10, Iteration 49/51, Loss: 1.4299739599227905
Epoch 9/10, Iteration 50/51, Loss: 1.3823060989379883
Epoch 9/10, Iteration 51/51, Loss: 1.256052017211914
Epoch 10/10, Iteration 1/51, Loss: 1.2878408432006836
Epoch 10/10, Iteration 2/51, Loss: 1.2748873233795166
Epoch 10/10, Iteration 3/51, Loss: 1.3392090797424316
Epoch 10/10, Iteration 4/51, Loss: 1.2988275289535522
Epoch 10/10, Iteration 5/51, Loss: 1.3276333808898926
Epoch 10/10, Iteration 6/51, Loss: 1.2603216171264648
Epoch 10/10, Iteration 7/51, Loss: 1.3047809600830078
Epoch 10/10, Iteration 8/51, Loss: 1.388969898223877
Epoch 10/10, Iteration 9/51, Loss: 1.2372839450836182
Epoch 10/10, Iteration 10/51, Loss: 1.299720048904419
Epoch 10/10, Iteration 11/51, Loss: 1.2506247758865356
Epoch 10/10, Iteration 12/51, Loss: 1.2964940071105957
Epoch 10/10, Iteration 13/51, Loss: 1.2201929092407227
Epoch 10/10, Iteration 14/51, Loss: 1.1737185716629028
Epoch 10/10, Iteration 15/51, Loss: 1.312487244606018
Epoch 10/10, Iteration 16/51, Loss: 1.2516611814498901
Epoch 10/10, Iteration 17/51, Loss: 1.293478012084961
Epoch 10/10, Iteration 18/51, Loss: 1.392001748085022
Epoch 10/10, Iteration 19/51, Loss: 1.3567914962768555
Epoch 10/10, Iteration 20/51, Loss: 1.3276644945144653
Epoch 10/10, Iteration 21/51, Loss: 1.2452112436294556
Epoch 10/10, Iteration 22/51, Loss: 1.2162225246429443
Epoch 10/10, Iteration 23/51, Loss: 1.2767384052276611
Epoch 10/10, Iteration 24/51, Loss: 1.2455507516860962
Epoch 10/10, Iteration 25/51, Loss: 1.2848973274230957
Epoch 10/10, Iteration 26/51, Loss: 1.2135199308395386
Epoch 10/10, Iteration 27/51, Loss: 1.3187758922576904
Epoch 10/10, Iteration 28/51, Loss: 1.2222694158554077
Epoch 10/10, Iteration 29/51, Loss: 1.3637676239013672
Epoch 10/10, Iteration 30/51, Loss: 1.2624255418777466
Epoch 10/10, Iteration 31/51, Loss: 1.273742437362671
Epoch 10/10, Iteration 32/51, Loss: 1.2443742752075195
Epoch 10/10, Iteration 33/51, Loss: 1.2838006019592285
Epoch 10/10, Iteration 34/51, Loss: 1.2319786548614502
Epoch 10/10, Iteration 35/51, Loss: 1.291395664215088
Epoch 10/10, Iteration 36/51, Loss: 1.2434614896774292
Epoch 10/10, Iteration 37/51, Loss: 1.3956407308578491
Epoch 10/10, Iteration 38/51, Loss: 1.376810908317566
Epoch 10/10, Iteration 39/51, Loss: 1.2605206966400146
Epoch 10/10, Iteration 40/51, Loss: 1.192931890487671
```

```
Epoch 10/10, Iteration 41/51, Loss: 1.2804604768753052
Epoch 10/10, Iteration 42/51, Loss: 1.3234710693359375
Epoch 10/10, Iteration 43/51, Loss: 1.3088494539260864
Epoch 10/10, Iteration 44/51, Loss: 1.446096658706665
Epoch 10/10, Iteration 45/51, Loss: 1.388561725616455
Epoch 10/10, Iteration 46/51, Loss: 1.2555091381072998
Epoch 10/10, Iteration 47/51, Loss: 1.2418718338012695
Epoch 10/10, Iteration 48/51, Loss: 1.3174375295639038
Epoch 10/10, Iteration 49/51, Loss: 1.279547095298767
Epoch 10/10, Iteration 50/51, Loss: 1.3169975280761719
Epoch 10/10, Iteration 51/51, Loss: 1.3594677448272705
CPU times: total: 10min 48s
Wall time: 2min 44s
```

0.8 evaluate train

[14]:
```python
############################
## evaluate train
def evaluate_model(model, train_loader, device):
    ## Set the model to evaluation mode
    cnn_classifier.eval()
    all_predictions = []
    all_targets = []

    with torch.no_grad():
        for inputs, targets in train_loader:
            inputs, targets = inputs.to(device), targets.to(device)
            outputs = model(inputs)
            _, predictions = torch.max(outputs, 1)

            all_predictions.extend(predictions.cpu().numpy())
            all_targets.extend(targets.cpu().numpy())

    accuracy  = accuracy_score(all_targets, all_predictions)
    confmat   = confusion_matrix(y_true=all_targets, y_pred=all_predictions)
    precision = precision_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted', zero_division=1)
    recall    = recall_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted', zero_division=1)
    f1 = f1_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted')

    print(f'Confusion Matrix:\n{confmat}')
    print(f'Accuracy: {accuracy:.2f}')
    print(f'Precision: {precision:.2f}')
    print(f'Recall: {recall:.2f}')
    print(f'F1-score: {f1:.2f}')
```

[15]:
```
evaluate_model(cnn_classifier, train_loader, device)
```

```
Confusion Matrix:
[[ 46   9 335   4]
 [ 36  12 358   8]
 [  1   0 408   0]
 [ 11   5 365  22]]
Accuracy: 0.30
Precision: 0.47
Recall: 0.30
F1-score: 0.19
```

0.9 Evaluate Test

[16]:
```
#########################
## evaluate model based on test data loader
def evaluate_model(model, test_loader, device):
    ## Set the model to evaluation mode
    cnn_classifier.eval()
    all_predictions = []
    all_targets = []

    with torch.no_grad():
        for inputs, targets in test_loader:
            inputs, targets = inputs.to(device), targets.to(device)
            outputs = model(inputs)
            _, predictions = torch.max(outputs, 1)

            all_predictions.extend(predictions.cpu().numpy())
            all_targets.extend(targets.cpu().numpy())

    accuracy  = accuracy_score(all_targets, all_predictions)
    confmat   = confusion_matrix(y_true=all_targets, y_pred=all_predictions)
    precision = precision_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted', zero_division=1)
    recall    = recall_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted', zero_division=1)
    f1 = f1_score(y_true=all_targets, y_pred=all_predictions,␣
 ↪average='weighted')

    print(f'Confusion Matrix:\n{confmat}')
    print(f'Accuracy: {accuracy:.2f}')
    print(f'Precision: {precision:.2f}')
    print(f'Recall: {recall:.2f}')
    print(f'F1-score: {f1:.2f}')
```

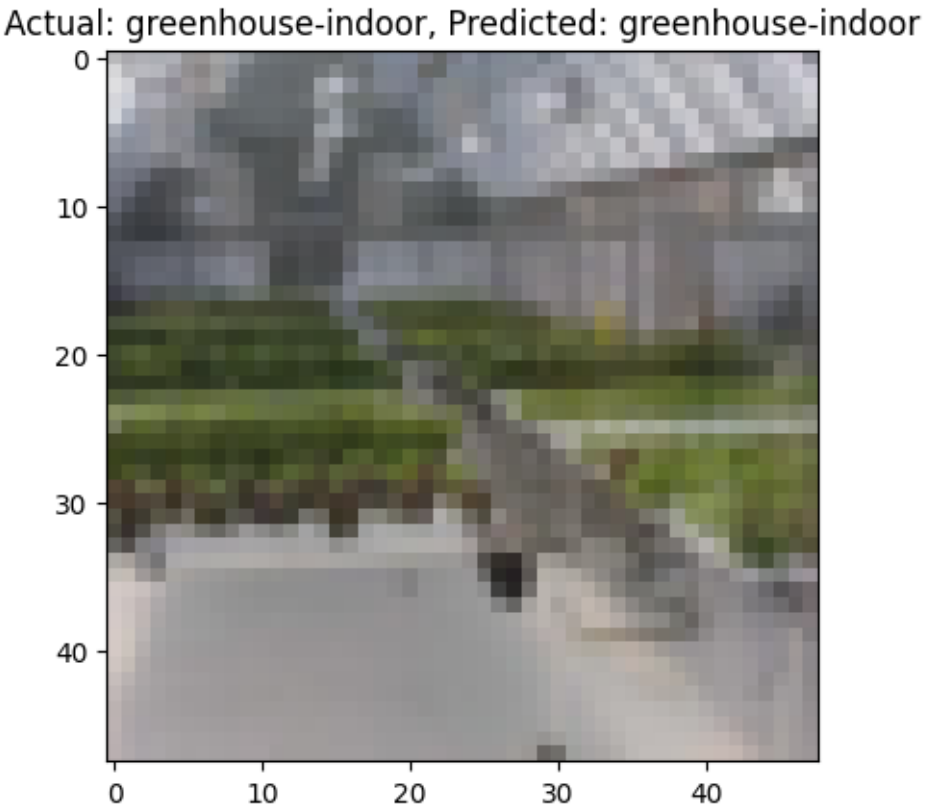
[17]:
```
evaluate_model(cnn_classifier, test_loader, device)
```

```
Confusion Matrix:
[[12  2 96  2]
 [ 6  3 84  0]
 [ 0  0 97  0]
 [ 4  1 96  2]]
Accuracy: 0.28
Precision: 0.46
Recall: 0.28
F1-score: 0.17
```

0.10 Test based off random image from train dataset

[21]:
```
############################
## Test on train data
cnn_classifier.eval()
# Define class names
class_names = ["amusement_park", "bridge", "greenhouse-indoor", "mountain"]
# Get a random index from the test set
random_index = random.randint(0, len(train_dataset) - 1)
# Get the image and label at the random index
image, label = train_dataset[random_index]
image = image.to(device)
label = label.to(device)

output = cnn_classifier(image.unsqueeze(0))  # Unsqueeze to add batch dimension

# Apply threshold for multiclass classification
_, predicted_label = torch.max(output, 1)

# Convert labels to class names
actual_class = class_names[int(label)]
predicted_class = class_names[int(predicted_label)]

# Check if the prediction is correct
is_correct = predicted_label == int(label)

# Print results
print(f"Actual Class: {actual_class}, Predicted Class: {predicted_class},␣
 ↪Correct: {is_correct}")

# Display the image
image = image.cpu().permute(1, 2, 0).numpy()  # Convert to numpy and rearrange␣
 ↪dimensions
plt.imshow(image)
plt.title(f"Actual: {actual_class}, Predicted: {predicted_class}")
plt.show()
```

```
Actual Class: greenhouse-indoor, Predicted Class: greenhouse-indoor, Correct:
tensor([True])
```



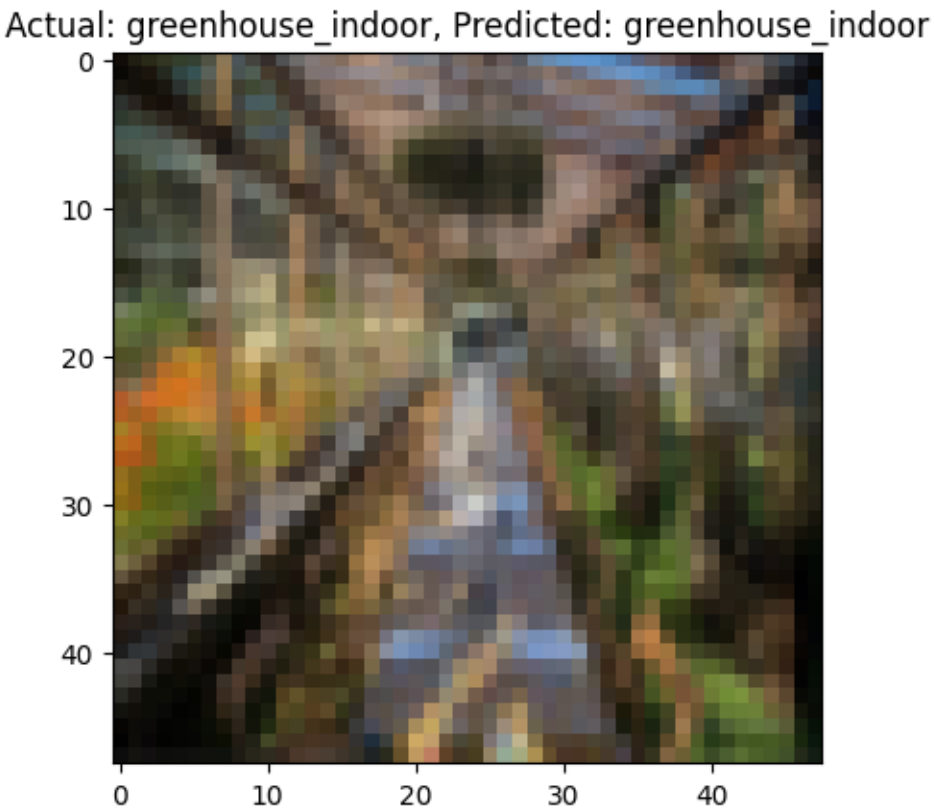
0.11 Test based off random image from test dataset

[22]:
```
cnn_classifier.eval()

# Define class names
class_names = ["amusement_park", "bridge" ,"greenhouse_indoor", "mountain"]

# Get a random index from the test set
random_index = random.randint(0, len(test_dataset) - 1)

# Get the image and label at the random index
image, label = test_dataset[random_index]
image = image.to(device)
label = label.to(device)
```

[23]:
```
output = cnn_classifier(image.unsqueeze(0))  # Unsqueeze to add batch dimension

# Apply threshold for multiclass classification
```

```
_, predicted_label = torch.max(output, 1)

# Convert labels to class names
actual_class = class_names[int(label)]
predicted_class = class_names[int(predicted_label)]

# Check if the prediction is correct
is_correct = predicted_label == int(label)

# Print results
print(f"Actual Class: {actual_class}, Predicted Class: {predicted_class},␣
 ↪Correct: {is_correct}")

# Display the image
image = image.cpu().permute(1, 2, 0).numpy()  # Convert to numpy and rearrange␣
 ↪dimensions
plt.imshow(image)
plt.title(f"Actual: {actual_class}, Predicted: {predicted_class}")
plt.show()
```

```
Actual Class: greenhouse_indoor, Predicted Class: greenhouse_indoor, Correct:
tensor([True])
```



[ ]: