Marine Bird Classification

Niranga Udumulla

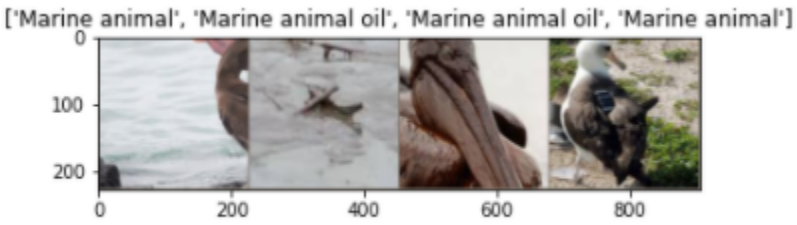
December 11, 2020

Abstract

In this report represents how to separate marine birds from a set of birds covered with oils.

1 Introduction

Firstly, we downloaded 100 images for two categories that represents the birds with oils and without oils in marine areas.We used 80/20 split and aim is to identify the oily birds correctly.Following figure shows the example images for 2 categories.



2 Procedure

We made a simple classifier to make classification using pytorch.Model is based on cross-entropy loss and stochastic gradient decent algorithm used to find out the points which have minimum loss.After that resnet18 weights used as the weight of our classifier.After running several epochs,accuracy values for training and validation have recorded to check the performance of the model.

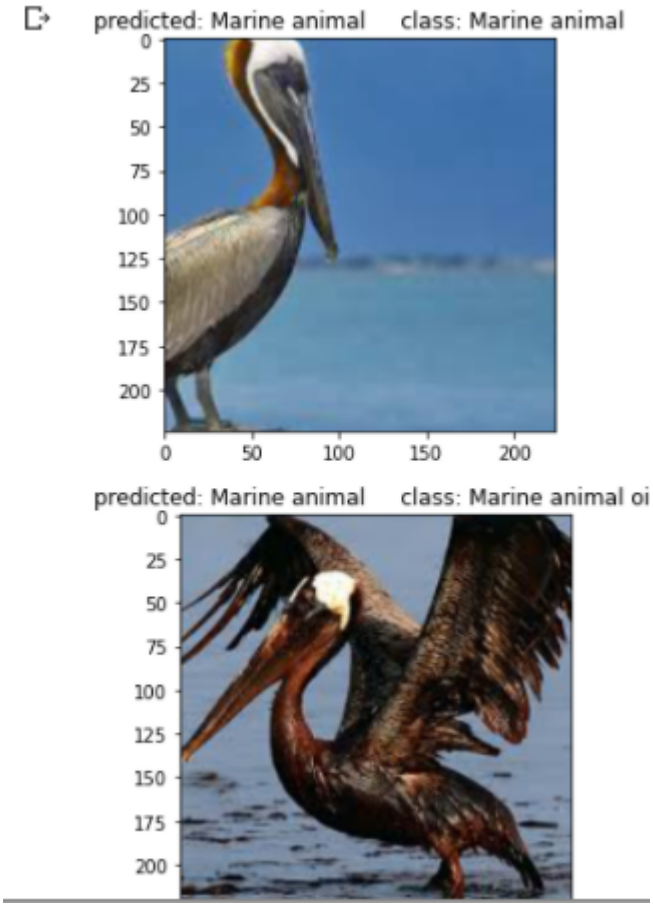
```
def train_model(model, num_epochs=25):

    model = model.to(device)
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(model.parameters(), lr=0.001, momentum=0.9)
    scheduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)
```

```
    for epoch in range(num_epochs):

        print('Epoch: ',epoch+1,'/',num_epochs)

        ###Train
        model.train()
        running_corrects = 0
        for inputs, labels in Bar(dataloaders['train']):
            inputs = inputs.to(device)
            labels = labels.to(device)

            optimizer.zero_grad()

            outputs = model(inputs)

            preds = torch.max(outputs, 1)[1]
            running_corrects += torch.sum(preds == labels.data)

            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

print("Train ", 'Acc: {:.2f}'.format(running_corrects.
double()/dataset_sizes['train']))

        scheduler.step()

        ###Val
        model.eval()
        running_corrects = 0
        for inputs, labels in Bar(dataloaders['valid']):
            inputs = inputs.to(device)
            labels = labels.to(device)

            outputs = model(inputs)
            preds = torch.max(outputs, 1)[1]
            running_corrects += torch.sum(preds == labels.data)

print("Valid ", 'Acc: {:.2f}'.format(running_corrects.double()/
dataset_sizes['valid']))
print("##########################")
    return model
```

3 Prediction using Resnet18

Finally we check the testing image with their actual and predicted class.The code is as follows.

```
model = train_model(model, num_epochs=3)#train the model

def visualize_model(model, num_images=16):
    model.eval()
    index = 0
    for i, (inputs, labels) in enumerate(dataloaders['valid']):
        inputs = inputs.to(device)
        labels = labels.to(device)

        outputs = model(inputs)

        preds = torch.max(outputs, 1)[1]

for j in range(inputs.size()[0]):
index += 1
title1 = 'predicted: ' +dataset_labels[int(class_names[preds[j]])-1] + '
class: ' + dataset_labels[int(class_names[labels[j]])-1]
imshow(inputs.cpu().data[j], title1)

if index == num_images:
return
```



3.1 Prediction for new image

```
image = io.imread('https://images-na.ssl-images-
amazon.com/images/I/51dZp-%2B4W9L._AC_.jpg')
plt.imshow(image);
img = apply_transforms(image).
clone().detach().requires_grad_(True).to(device)
outputs = model(img)
preds = torch.max(outputs, 1)[1]
print('predicted: ' + dataset_labels[int(class_names[preds])-1])
```


```
[ ]  1
```

```
[21]  1 img = apply_transforms(image).clone().detach().requires_grad_(True).to(device)
```

```
[22]  1 outputs = model(img)
      2 preds = torch.max(outputs, 1)[1]
```

```
[23]  1 print('predicted: ' + dataset_labels[preds])
```

```
predicted: Marine animal
```

Finally we calculate the precision,recall and F1 score for each category and calculated the classification accuracy using our confusion matrix.

3.2 Classification Report

- True positive (TP) The number of correctly identified samples.
- True negative (TN) The number of correctly identified negative samples
- False positive (FP) The number of wrongly identified samples, i.e., a commonly called a "false alarm".
- False negative (FN) The number of wrongly identified negative samples.

.

3.2.1 Precision

Precision (PREC) This metric is also frequently called the positive predictive value, and shows the ratio of samples that are correctly identified as positive.

- Precision for Without oil bird = 0.8181818181818182
- Precision for With oil bird = 0.8888888888888888

3.2.2 Recall

Recall (REC) This metric is also frequently called sensitivity, probability of detection and true positive rate, and it is the ratio of samples that are correctly identified as positive among all existing positive samples.

- Recall for Without oil bird = 0.9
- Recall for With oil bird = 0.8

3.2.3 F1 Score

F1 score (F1) A measure of a test's accuracy by calculating the harmonic mean of the precision and recall.

- F1 score Without oil bird = 0.8590909090909091
- F1 scorer With oil bird = 0.8444444444444444

3.2.4 Accuracy

Accuracy (ACC) The percentage of correctly identified true and false samples.

- Classification-Accuracy = 0.851