Dogvscat_classifier

April 16, 2025

Install & Import

[44]:
```
!pip install torch torchvision matplotlib
```

```
Requirement already satisfied: torch in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (2.6.0)
Requirement already satisfied: torchvision in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages
(0.21.0)
Requirement already satisfied: matplotlib in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages
(3.10.1)
Requirement already satisfied: filelock in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (4.13.1)
Requirement already satisfied: networkx in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (3.4.2)
Requirement already satisfied: jinja2 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (3.1.6)
Requirement already satisfied: fsspec in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (2025.3.2)
Requirement already satisfied: setuptools in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (78.1.0)
Requirement already satisfied: sympy==1.13.1 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torch) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
sympy==1.13.1->torch) (1.3.0)
Requirement already satisfied: numpy in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torchvision) (2.2.4)
```

```
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
torchvision) (11.1.0)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (4.57.0)
Requirement already satisfied: kiwisolver>=1.3.1 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
python-dateutil>=2.7->matplotlib) (1.17.0)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\wahid\appdata\local\programs\python\python313\lib\site-packages (from
jinja2->torch) (3.0.2)
```

[46]:
```python
import os
import torch
import torchvision
from torch.utils.data import DataLoader, Dataset
from torchvision import datasets, transforms
from torchvision.models import resnet18, ResNet18_Weights
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import torch.nn as nn
import torch.optim as optim
```

Define Path

[48]:
```python
data_dir = r"D:\AFC\Cat vs Dog\datasets\datasets"
train_dir = os.path.join(data_dir, "train")
```

```
val_dir = os.path.join(data_dir, "val")
test_dir = os.path.join(data_dir, "test")

BATCH_SIZE = 32
IMG_SIZE = 128
```

Transforms and Dataset Prep

[49]:
```
train_transform = transforms.Compose([
    transforms.Resize((IMG_SIZE, IMG_SIZE)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.ToTensor(),
    transforms.Normalize([0.5], [0.5])
])

val_transform = transforms.Compose([
    transforms.Resize((IMG_SIZE, IMG_SIZE)),
    transforms.ToTensor(),
    transforms.Normalize([0.5], [0.5])
])

test_transform = transforms.Compose([
    transforms.Resize((IMG_SIZE, IMG_SIZE)),
    transforms.ToTensor(),
    transforms.Normalize([0.5], [0.5])
])
```

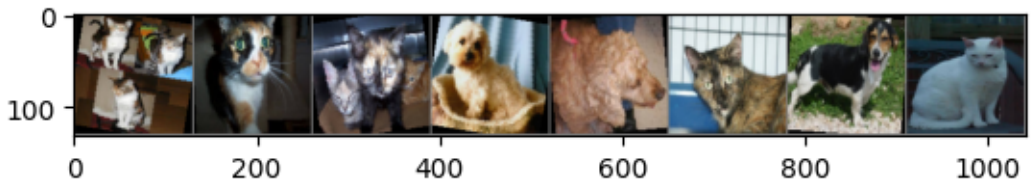
Load Dataset

[52]:
```
train_dataset = datasets.ImageFolder(train_dir, transform=train_transform)
val_dataset = datasets.ImageFolder(val_dir, transform=val_transform)

class TestDataset(Dataset):
    def __init__(self, test_dir, transform=None):
        self.image_paths = sorted([os.path.join(test_dir, img) for img in os.
 ↪listdir(test_dir) if img.endswith('.jpg')])
        self.transform = transform

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

    def __getitem__(self, idx):
        img_path = self.image_paths[idx]
        img = Image.open(img_path).convert("RGB")
        if self.transform:
            img = self.transform(img)
        img_id = os.path.basename(img_path)
```

```
        return img, img_id

test_dataset = TestDataset(test_dir, transform=test_transform)

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

Visualize Sample

[53]:
```
def imshow(img):
    img = img / 2 + 0.5
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.show()

dataiter = iter(train_loader)
images, labels = next(dataiter)
imshow(torchvision.utils.make_grid(images[:8]))
print("Labels:", labels[:8].tolist())
```




```
Labels: [0, 0, 0, 1, 1, 0, 1, 0]
```

ResNet18

[55]:
```
weights = ResNet18_Weights.DEFAULT
model = resnet18(weights=weights)


for param in model.parameters():
    param.requires_grad = False

model.fc = nn.Linear(model.fc.in_features, 2)

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

print("ResNet18 with updated weights loaded successfully.")
```

```
ResNet18 with updated weights loaded successfully.
```

Model Training

[56]:
```
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.fc.parameters(), lr=0.001)

EPOCHS = 5

for epoch in range(EPOCHS):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 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()
        _, predicted = torch.max(outputs, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{EPOCHS}] - Train Loss: {running_loss:.3f} - Train␣
 ↪Acc: {train_acc:.2f}%")

    # Validation
    model.eval()
    val_loss = 0.0
    val_correct = 0
    val_total = 0
    with torch.no_grad():
        for val_images, val_labels in val_loader:
            val_images, val_labels = val_images.to(device), val_labels.
 ↪to(device)
            val_outputs = model(val_images)
            loss = criterion(val_outputs, val_labels)
            val_loss += loss.item()
            _, val_predicted = torch.max(val_outputs, 1)
            val_total += val_labels.size(0)
            val_correct += (val_predicted == val_labels).sum().item()

    val_acc = 100 * val_correct / val_total
```

```
    print(f"           -> Val Loss: {val_loss:.3f} - Val Acc: {val_acc:.
↪2f}%\\n")
```

```
Epoch [1/5] - Train Loss: 174.564 - Train Acc: 87.61%
           -> Val Loss: 29.125 - Val Acc: 92.16%\n
Epoch [2/5] - Train Loss: 151.677 - Train Acc: 89.17%
           -> Val Loss: 27.689 - Val Acc: 92.68%\n
Epoch [3/5] - Train Loss: 154.582 - Train Acc: 89.22%
           -> Val Loss: 28.118 - Val Acc: 92.20%\n
Epoch [4/5] - Train Loss: 148.440 - Train Acc: 89.53%
           -> Val Loss: 27.445 - Val Acc: 92.54%\n
Epoch [5/5] - Train Loss: 147.823 - Train Acc: 89.75%
           -> Val Loss: 28.173 - Val Acc: 92.16%\n
```

Predictions

[57]:
```
model.eval()
predictions = []

with torch.no_grad():
    for images, image_ids in test_loader:
        images = images.to(device)
        outputs = model(images)
        _, predicted = torch.max(outputs, 1)

        for img_id, label in zip(image_ids, predicted.cpu().numpy()):
            predictions.append({'id': img_id, 'label': int(label)})

submission_df = pd.DataFrame(predictions)
submission_df = submission_df.sort_values(by="id")
submission_df.to_csv("submission.csv", index=False)

print("submission.csv file created!")
submission_df.head()
```

```
submission.csv file created!
```

[57]:

	id	label
0	1.jpg	0
1	10.jpg	1
2	100.jpg	0
3	101.jpg	0
4	102.jpg	1

Visualizing Predictions

[58]:
```
model.eval()

for images, image_ids in test_loader:
```

```
    outputs = model(images.to(device))
    _, preds = torch.max(outputs, 1)

    for i in range(3):  # Show 3 predictions
        img = images[i] / 2 + 0.5
        plt.imshow(np.transpose(img.numpy(), (1, 2, 0)))
        plt.title(f"Predicted: {'Dog' if preds[i]==1 else 'Cat'}")
        plt.axis('off')
        plt.show()
    break
```

Predicted: Dog

Predicted: Cat