Introduction

This is the task project for GSoC 2025 application for the ArtExtract Project.

In this project, wikiart data was uesd. To classify the artist, genre and style of the picture, two different kinds of convolutional neural network were used which are **AlextNet model** and **ResNet model**. To save the time, I have used Res50 as the basic model and re-train the last 20 layers.

After training the two models, I use soft voting to stack the two models. In soft voting, each base model gives a probability or confidence estimate for each category, which is then averaged or weighted. The final prediction is the category with the highest average probability (or weighted average probability).

Import libaray

```
import pandas as pd
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision.transforms as transforms
from PIL import Image
from torch.utils.data import Dataset, DataLoader
from sklearn.preprocessing import LabelEncoder
import torchvision.models as models
import numpy as np
```

Loading data

Both AlexNet and ResNet take 227*227 size of inputs. Thus, before building the model, I used the DataLoader tool in torch library to contribute the dataset.

```
artist = pd.read_csv('wikiart_csv/artist_train.csv')
artist.columns = ['path', 'artist']
gener = pd.read_csv('wikiart_csv/genre_train.csv')
gener.columns = ['path', 'genre']
style = pd.read_csv('wikiart_csv/style_train.csv')
style.columns =  ['path', 'style']

label_encoder = LabelEncoder()
artist['label'] = label_encoder.fit_transform(artist['artist'])
gener['label'] = label_encoder.fit_transform(gener['genre'])
style['label'] = label_encoder.fit_transform(style['style'])

transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(15),
    transforms.RandomResizedCrop(224, scale=(0.8, 1.0)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229,
0.224, 0.225])
])

class ArtDataset(Dataset):
    def __init__(self, dataframe, img_dir, transform=None):
        self.dataframe = dataframe
        self.img_dir = img_dir
        self.transform = transform

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

    def __getitem__(self, idx):
        img_path = f"{self.img_dir}/{self.dataframe.iloc[idx]
['path']}"
        try:
            image = Image.open(img_path).convert("RGB")  # 读取图片
        except (OSError, IOError):  # 捕获损坏文件错误
            print(f"Warning: Skipping corrupted image {img_path}")
            return self.__getitem__((idx + 1) % len(self.dataframe))
# 返回下一个样本

        label = self.dataframe.iloc[idx]['label']
        if self.transform:
            image = self.transform(image)

        return image, label

img_dir = "wikiart"
artist_dataset = ArtDataset(artist, img_dir, transform)
genre_dataset = ArtDataset(gener, img_dir, transform)
style_dataset = ArtDataset(style, img_dir, transform)

train_size_artist = int(0.8 * len(artist_dataset))
train_size_genre = int(0.8 * len(genre_dataset))
train_size_style = int(0.8 * len(style_dataset))

val_size_artist = len(artist_dataset) - train_size_artist
val_size_genre = len(genre_dataset) - train_size_genre
val_size_style = len(style_dataset) - train_size_style

train_dataset_artist, val_dataset_artist =
torch.utils.data.random_split(artist_dataset, [train_size_artist,
val_size_artist])
train_dataset_genre, val_dataset_genre =
torch.utils.data.random_split(genre_dataset, [train_size_genre,
val_size_genre])
```

```
train_dataset_style, val_dataset_style =
torch.utils.data.random_split(style_dataset, [train_size_style,
val_size_style])

train_loader_artist = DataLoader(train_dataset_artist, batch_size=32,
shuffle=True, num_workers=0)
val_loader_artist = DataLoader(val_dataset_artist, batch_size=32,
shuffle=False, num_workers=0)

train_loader_genre = DataLoader(train_dataset_genre, batch_size=32,
shuffle=True, num_workers=0)
val_loader_genre = DataLoader(val_dataset_genre, batch_size=32,
shuffle=False, num_workers=0)

train_loader_style = DataLoader(train_dataset_style, batch_size=32,
shuffle=True, num_workers=0)
val_loader_style = DataLoader(val_dataset_style, batch_size=32,
shuffle=False, num_workers=0)
```

AlexNet

At this cell, I built up the frame of alexnet model.

```
class AlexNet(nn.Module):
    def __init__(self, num_classes=23):
        super(AlexNet, self).__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 64, kernel_size=11, stride=4, padding=2),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),

            nn.Conv2d(64, 192, kernel_size=5, padding=2),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),

            nn.Conv2d(192, 384, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),

            nn.Conv2d(384, 256, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),

            nn.Conv2d(256, 256, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),
        )

        self.classifier = nn.Sequential(
            nn.Dropout(),
            nn.Linear(256 * 6 * 6, 4096),
            nn.ReLU(inplace=True),
            nn.Dropout(),
            nn.Linear(4096, 4096),
            nn.ReLU(inplace=True),
            nn.Linear(4096, num_classes),
        )

    def forward(self, x):
        x = self.features(x)
        x = torch.flatten(x, 1)
        x = self.classifier(x)
        return x

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
num_epochs = 50
```

then, let's train the artist part of the data

```
print('artist part tarning...')
model = AlexNet(num_classes=23).to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)

for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_artist:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss:
{running_loss/len(train_loader_artist):.4f}, Train Accuracy:
{train_acc:.2f}%")

torch.save(model.state_dict(), "Alexnet_model_artist.pth")
```

```
print("artist part finished")
```

```
artist part tarning...
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [1/50], Loss: 2.8638, Train Accuracy: 16.15%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/50], Loss: 2.5093, Train Accuracy: 26.09%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/50], Loss: 2.2832, Train Accuracy: 31.85%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/50], Loss: 2.1623, Train Accuracy: 35.29%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/50], Loss: 2.0659, Train Accuracy: 37.88%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/50], Loss: 1.9905, Train Accuracy: 40.28%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/50], Loss: 1.9114, Train Accuracy: 42.82%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/50], Loss: 1.8289, Train Accuracy: 44.89%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/50], Loss: 1.7743, Train Accuracy: 46.07%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/50], Loss: 1.6879, Train Accuracy: 47.99%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/50], Loss: 1.6476, Train Accuracy: 49.69%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/50], Loss: 1.5899, Train Accuracy: 51.16%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/50], Loss: 1.5283, Train Accuracy: 52.85%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/50], Loss: 1.4599, Train Accuracy: 55.07%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [15/50], Loss: 1.3980, Train Accuracy: 56.65%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
```

```
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [16/50], Loss: 1.3339, Train Accuracy: 58.47%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [17/50], Loss: 1.2844, Train Accuracy: 60.10%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [18/50], Loss: 1.2172, Train Accuracy: 62.11%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [19/50], Loss: 1.1664, Train Accuracy: 62.80%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [20/50], Loss: 1.1102, Train Accuracy: 65.23%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [21/50], Loss: 1.0385, Train Accuracy: 66.78%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [22/50], Loss: 0.9763, Train Accuracy: 68.43%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [23/50], Loss: 0.9322, Train Accuracy: 69.84%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [24/50], Loss: 0.8809, Train Accuracy: 71.49%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [25/50], Loss: 0.8288, Train Accuracy: 73.55%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [26/50], Loss: 0.7643, Train Accuracy: 75.28%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [27/50], Loss: 0.7340, Train Accuracy: 76.41%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [28/50], Loss: 0.6829, Train Accuracy: 77.95%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [29/50], Loss: 0.6452, Train Accuracy: 79.04%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [30/50], Loss: 0.6026, Train Accuracy: 80.30%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [31/50], Loss: 0.5590, Train Accuracy: 81.31%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
```

```
Epoch [32/50], Loss: 0.5388, Train Accuracy: 82.77%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [33/50], Loss: 0.5312, Train Accuracy: 83.01%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [34/50], Loss: 0.4838, Train Accuracy: 84.12%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [35/50], Loss: 0.4454, Train Accuracy: 84.99%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [36/50], Loss: 0.4377, Train Accuracy: 85.58%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [37/50], Loss: 0.4281, Train Accuracy: 86.33%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [38/50], Loss: 0.3944, Train Accuracy: 86.89%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [39/50], Loss: 0.3867, Train Accuracy: 87.29%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [40/50], Loss: 0.3631, Train Accuracy: 88.36%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [41/50], Loss: 0.3469, Train Accuracy: 88.72%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [42/50], Loss: 0.3417, Train Accuracy: 89.28%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [43/50], Loss: 0.3233, Train Accuracy: 89.62%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [44/50], Loss: 0.3112, Train Accuracy: 89.77%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [45/50], Loss: 0.2950, Train Accuracy: 90.45%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [46/50], Loss: 0.2974, Train Accuracy: 90.18%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [47/50], Loss: 0.2853, Train Accuracy: 90.69%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [48/50], Loss: 0.2856, Train Accuracy: 90.79%
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [49/50], Loss: 0.2662, Train Accuracy: 91.45%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [50/50], Loss: 0.2578, Train Accuracy: 91.48%
artist part finished
```

then the genre part

```
print('genre part tarning...')

model = AlexNet(num_classes=10).to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)

for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_genre:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss:
{running_loss/len(train_loader_genre):.4f}, Train Accuracy:
{train_acc:.2f}%")

torch.save(model.state_dict(), "Alexnet_model_genre.pth")

print("genre part finished")
```

```
genre part tarning...
```

```
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/
site-packages/PIL/Image.py:3406: DecompressionBombWarning: Image size
(99962094 pixels) exceeds limit of 89478485 pixels, could be
```

```
decompression bomb DOS attack.
  warnings.warn(

Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [1/50], Loss: 1.7048, Train Accuracy: 40.55%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/50], Loss: 1.4794, Train Accuracy: 48.44%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/50], Loss: 1.3852, Train Accuracy: 51.79%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/50], Loss: 1.3203, Train Accuracy: 54.10%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/50], Loss: 1.2628, Train Accuracy: 55.96%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/50], Loss: 1.2185, Train Accuracy: 57.72%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/50], Loss: 1.1759, Train Accuracy: 59.24%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/50], Loss: 1.1372, Train Accuracy: 60.33%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/50], Loss: 1.1014, Train Accuracy: 61.50%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/50], Loss: 1.0668, Train Accuracy: 62.73%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/50], Loss: 1.0437, Train Accuracy: 63.51%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/50], Loss: 1.0155, Train Accuracy: 64.11%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/50], Loss: 0.9825, Train Accuracy: 65.39%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/50], Loss: 0.9527, Train Accuracy: 66.40%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [15/50], Loss: 0.9328, Train Accuracy: 67.20%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
```

```
Epoch [16/50], Loss: 0.9068, Train Accuracy: 67.97%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [17/50], Loss: 0.8775, Train Accuracy: 69.16%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [18/50], Loss: 0.8582, Train Accuracy: 69.70%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [19/50], Loss: 0.8317, Train Accuracy: 70.44%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [20/50], Loss: 0.8101, Train Accuracy: 71.11%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [21/50], Loss: 0.7825, Train Accuracy: 72.29%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [22/50], Loss: 0.7677, Train Accuracy: 72.72%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [23/50], Loss: 0.7450, Train Accuracy: 73.46%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [24/50], Loss: 0.7216, Train Accuracy: 74.43%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [25/50], Loss: 0.6988, Train Accuracy: 74.63%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [26/50], Loss: 0.6778, Train Accuracy: 75.63%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [27/50], Loss: 0.6694, Train Accuracy: 76.32%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [28/50], Loss: 0.6491, Train Accuracy: 76.71%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [29/50], Loss: 0.6274, Train Accuracy: 77.35%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [30/50], Loss: 0.6105, Train Accuracy: 78.00%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [31/50], Loss: 0.5951, Train Accuracy: 78.70%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [32/50], Loss: 0.5851, Train Accuracy: 78.96%
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [33/50], Loss: 0.5580, Train Accuracy: 79.86%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [34/50], Loss: 0.5541, Train Accuracy: 80.08%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [35/50], Loss: 0.5366, Train Accuracy: 80.68%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [36/50], Loss: 0.5231, Train Accuracy: 81.12%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [37/50], Loss: 0.5141, Train Accuracy: 81.46%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [38/50], Loss: 0.4935, Train Accuracy: 82.35%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [39/50], Loss: 0.4880, Train Accuracy: 82.31%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [40/50], Loss: 0.4765, Train Accuracy: 82.93%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [41/50], Loss: 0.4569, Train Accuracy: 83.62%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [42/50], Loss: 0.4569, Train Accuracy: 83.47%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [43/50], Loss: 0.4430, Train Accuracy: 84.21%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [44/50], Loss: 0.4374, Train Accuracy: 84.29%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [45/50], Loss: 0.4335, Train Accuracy: 84.51%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [46/50], Loss: 0.4137, Train Accuracy: 85.17%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [47/50], Loss: 0.4123, Train Accuracy: 85.26%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [48/50], Loss: 0.4033, Train Accuracy: 85.67%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
```

```
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [49/50], Loss: 0.3942, Train Accuracy: 85.94%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [50/50], Loss: 0.3864, Train Accuracy: 86.19%
genre part finished
```

then the style part

```
print('style part tarning...')

model = AlexNet(num_classes=27).to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)

for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_style:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss:
{running_loss/len(train_loader_style):.4f}, Train Accuracy:
{train_acc:.2f}%")

torch.save(model.state_dict(), "Alexnet_model_style.pth")

print("style part finished")
```

```
style part tarning...
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg

/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/
site-packages/PIL/Image.py:3406: DecompressionBombWarning: Image size
```

```
(107327830 pixels) exceeds limit of 89478485 pixels, could be
decompression bomb DOS attack.
  warnings.warn(

Epoch [1/50], Loss: 2.6056, Train Accuracy: 20.12%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/50], Loss: 2.3263, Train Accuracy: 27.07%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/50], Loss: 2.2158, Train Accuracy: 30.21%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/50], Loss: 2.1433, Train Accuracy: 32.00%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/50], Loss: 2.0847, Train Accuracy: 33.48%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/50], Loss: 2.0351, Train Accuracy: 34.91%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/50], Loss: 1.9892, Train Accuracy: 36.18%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/50], Loss: 1.9390, Train Accuracy: 37.62%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/50], Loss: 1.8981, Train Accuracy: 38.87%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/50], Loss: 1.8594, Train Accuracy: 39.98%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/50], Loss: 1.8161, Train Accuracy: 41.18%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/50], Loss: 1.7739, Train Accuracy: 42.02%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/50], Loss: 1.7413, Train Accuracy: 43.17%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/50], Loss: 1.6941, Train Accuracy: 44.48%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [15/50], Loss: 1.6592, Train Accuracy: 45.43%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [16/50], Loss: 1.6183, Train Accuracy: 46.39%
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [17/50], Loss: 1.5752, Train Accuracy: 47.82%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [18/50], Loss: 1.5385, Train Accuracy: 48.92%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [19/50], Loss: 1.4993, Train Accuracy: 50.28%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [20/50], Loss: 1.4567, Train Accuracy: 51.15%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [21/50], Loss: 1.4227, Train Accuracy: 52.45%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [22/50], Loss: 1.3858, Train Accuracy: 53.65%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [23/50], Loss: 1.3465, Train Accuracy: 54.63%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [24/50], Loss: 1.3165, Train Accuracy: 55.56%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [25/50], Loss: 1.2758, Train Accuracy: 56.91%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [26/50], Loss: 1.2449, Train Accuracy: 57.60%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [27/50], Loss: 1.2122, Train Accuracy: 59.01%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [28/50], Loss: 1.1710, Train Accuracy: 60.17%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [29/50], Loss: 1.1441, Train Accuracy: 60.99%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [30/50], Loss: 1.1152, Train Accuracy: 61.97%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [31/50], Loss: 1.0784, Train Accuracy: 63.11%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [32/50], Loss: 1.0491, Train Accuracy: 64.07%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
```

```
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [33/50], Loss: 1.0309, Train Accuracy: 64.60%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [34/50], Loss: 0.9995, Train Accuracy: 65.65%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [35/50], Loss: 0.9708, Train Accuracy: 66.70%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [36/50], Loss: 0.9394, Train Accuracy: 67.81%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [37/50], Loss: 0.9189, Train Accuracy: 68.24%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [38/50], Loss: 0.9024, Train Accuracy: 68.69%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [39/50], Loss: 0.8833, Train Accuracy: 69.56%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [40/50], Loss: 0.8541, Train Accuracy: 70.75%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [41/50], Loss: 0.8341, Train Accuracy: 71.18%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [42/50], Loss: 0.8182, Train Accuracy: 71.86%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [43/50], Loss: 0.7960, Train Accuracy: 72.67%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [44/50], Loss: 0.7817, Train Accuracy: 72.93%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [45/50], Loss: 0.7618, Train Accuracy: 73.64%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [46/50], Loss: 0.7397, Train Accuracy: 74.47%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [47/50], Loss: 0.7351, Train Accuracy: 74.81%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [48/50], Loss: 0.7134, Train Accuracy: 75.38%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
```

```
Epoch [49/50], Loss: 0.7021, Train Accuracy: 75.75%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [50/50], Loss: 0.6811, Train Accuracy: 76.54%
style part finished
```

Resnet model

```python
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
num_epochs = 15
```

First train the artist data

```python
model = models.resnet50()
model.load_state_dict(torch.load("/Users/adamlee/.cache/torch/hub/checkpoints/resnet50-0676ba61.pth"))

for param in list(model.parameters())[:-20]:
    param.requires_grad = False

num_ftrs = model.fc.in_features
model.fc = nn.Linear(num_ftrs, 23)

model = model.to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)


for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_artist:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss:
{running_loss/len(train_loader_artist):.4f}, Train Acc:
{train_acc:.2f}%")

torch.save(model.state_dict(), "Resnet_model_artist.pth")

print("Artist part done.")
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [1/15], Loss: 1.4498, Train Acc: 57.77%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/15], Loss: 0.8179, Train Acc: 75.66%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/15], Loss: 0.6093, Train Acc: 81.69%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/15], Loss: 0.4932, Train Acc: 84.89%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/15], Loss: 0.3938, Train Acc: 87.66%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/15], Loss: 0.3251, Train Acc: 89.93%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/15], Loss: 0.2755, Train Acc: 91.29%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/15], Loss: 0.2232, Train Acc: 93.11%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/15], Loss: 0.2222, Train Acc: 93.15%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/15], Loss: 0.1807, Train Acc: 94.20%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/15], Loss: 0.1673, Train Acc: 94.28%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/15], Loss: 0.1481, Train Acc: 95.37%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/15], Loss: 0.1391, Train Acc: 95.63%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/15], Loss: 0.1322, Train Acc: 95.73%
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [15/15], Loss: 0.1387, Train Acc: 95.60%
Artist part done.
```

Then the genre part

```python
model = models.resnet50()
model.load_state_dict(torch.load("/Users/adamlee/.cache/torch/hub/checkpoints/resnet50-0676ba61.pth"))

for param in list(model.parameters())[:-20]:
    param.requires_grad = False

num_ftrs = model.fc.in_features
model.fc = nn.Linear(num_ftrs, 10)

model = model.to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)


for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_genre:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {running_loss/len(train_loader_genre):.4f}, Train Acc: {train_acc:.2f}%")

torch.save(model.state_dict(), "Resnet_model_genre.pth")

print("Genre part done.")
```

```
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/
site-packages/PIL/Image.py:3406: DecompressionBombWarning: Image size
(99962094 pixels) exceeds limit of 89478485 pixels, could be
decompression bomb DOS attack.
  warnings.warn(

Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [1/15], Loss: 0.9048, Train Acc: 68.65%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/15], Loss: 0.7347, Train Acc: 74.12%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/15], Loss: 0.6623, Train Acc: 76.64%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/15], Loss: 0.6102, Train Acc: 78.26%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/15], Loss: 0.5569, Train Acc: 79.93%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/15], Loss: 0.5140, Train Acc: 81.55%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/15], Loss: 0.4716, Train Acc: 83.18%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/15], Loss: 0.4268, Train Acc: 84.66%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/15], Loss: 0.3901, Train Acc: 85.77%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/15], Loss: 0.3636, Train Acc: 86.69%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/15], Loss: 0.3274, Train Acc: 87.97%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/15], Loss: 0.3036, Train Acc: 88.95%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/15], Loss: 0.2828, Train Acc: 89.66%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/15], Loss: 0.2612, Train Acc: 90.40%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-
van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
```

```
Epoch [15/15], Loss: 0.2433, Train Acc: 91.06%
Genre part done.
```

finaly the style part

```
model = models.resnet50()
model.load_state_dict(torch.load("/Users/adamlee/.cache/torch/hub/
checkpoints/resnet50-0676ba61.pth"))

for param in list(model.parameters())[:-20]:
    param.requires_grad = False

num_ftrs = model.fc.in_features
model.fc = nn.Linear(num_ftrs, 27)

model = model.to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)


for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0

    for images, labels in train_loader_style:
        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 = outputs.max(1)
        correct += predicted.eq(labels).sum().item()
        total += labels.size(0)

    train_acc = 100 * correct / total
    print(f"Epoch [{epoch+1}/{num_epochs}], Loss:
{running_loss/len(train_loader_style):.4f}, Train Acc: {train_acc:.2f}
%")

torch.save(model.state_dict(), "Resnet_model_style.pth")

print("Style part done.")
```

```
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [1/15], Loss: 1.7185, Train Acc: 43.77%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [2/15], Loss: 1.4030, Train Acc: 53.09%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [3/15], Loss: 1.2667, Train Acc: 57.05%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [4/15], Loss: 1.1663, Train Acc: 60.12%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [5/15], Loss: 1.0771, Train Acc: 62.90%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [6/15], Loss: 0.9988, Train Acc: 65.64%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [7/15], Loss: 0.9275, Train Acc: 68.10%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [8/15], Loss: 0.8618, Train Acc: 69.72%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [9/15], Loss: 0.7924, Train Acc: 72.33%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [10/15], Loss: 0.7384, Train Acc: 74.09%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [11/15], Loss: 0.6924, Train Acc: 75.84%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [12/15], Loss: 0.6416, Train Acc: 77.40%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [13/15], Loss: 0.6029, Train Acc: 78.91%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [14/15], Loss: 0.5713, Train Acc: 79.73%
Warning: Skipping corrupted image wikiart/Post_Impressionism/vincent-van-gogh_l-arlesienne-portrait-of-madame-ginoux-1890.jpg
Epoch [15/15], Loss: 0.5280, Train Acc: 81.54%
Style part done.
```

Stacking

```
import torchvision.models as models

model_Alex_artist = AlexNet(num_classes=23).to(device)
model_Alex_artist.load_state_dict(torch.load("Alexnet_model_artist.pth
"))
model_Alex_artist.eval()

model_Alex_genre = AlexNet(num_classes=10).to(device)
model_Alex_genre.load_state_dict(torch.load("Alexnet_model_genre.pth")
)
model_Alex_genre.eval()

model_Alex_style = AlexNet(num_classes=27).to(device)
model_Alex_style.load_state_dict(torch.load("Alexnet_model_style.pth")
)
model_Alex_style.eval()

model_Res_artist = models.resnet50().to(device)
num_ftrs = model_Res_artist.fc.in_features
model_Res_artist.fc = torch.nn.Linear(num_ftrs, 23)
model_Res_artist.load_state_dict(torch.load('Resnet_model_artist.pth')
)
model_Res_artist.eval()

model_Res_genre = models.resnet50().to(device)
num_ftrs = model_Res_genre.fc.in_features
model_Res_genre.fc = torch.nn.Linear(num_ftrs, 10)
model_Res_genre.load_state_dict(torch.load('Resnet_model_genre.pth'))
model_Res_genre.eval()

model_Res_style = models.resnet50().to(device)
num_ftrs = model_Res_style.fc.in_features
model_Res_style.fc = torch.nn.Linear(num_ftrs, 27)
model_Res_style.load_state_dict(torch.load('Resnet_model_style.pth'))
model_Res_style.eval()
```

```
ResNet(
  (conv1): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2),
padding=(3, 3), bias=False)
  (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (relu): ReLU(inplace=True)
  (maxpool): MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1,
ceil_mode=False)
  (layer1): Sequential(
    (0): Bottleneck(
      (conv1): Conv2d(64, 64, kernel_size=(1, 1), stride=(1, 1),
bias=False)
```

```
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(64, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (downsample): Sequential(
        (0): Conv2d(64, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
        (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): Bottleneck(
      (conv1): Conv2d(256, 64, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(64, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (2): Bottleneck(
      (conv1): Conv2d(256, 64, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(64, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
  )
  (layer2): Sequential(
    (0): Bottleneck(
      (conv1): Conv2d(256, 128, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(2, 2),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(128, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (downsample): Sequential(
        (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(2, 2),
bias=False)
        (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): Bottleneck(
      (conv1): Conv2d(512, 128, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(128, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (2): Bottleneck(
      (conv1): Conv2d(512, 128, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(128, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (3): Bottleneck(
      (conv1): Conv2d(512, 128, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(128, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
  )
  (layer3): Sequential(
    (0): Bottleneck(
      (conv1): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(2, 2),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (downsample): Sequential(
        (0): Conv2d(512, 1024, kernel_size=(1, 1), stride=(2, 2),
bias=False)
        (1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): Bottleneck(
      (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (2): Bottleneck(
      (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (3): Bottleneck(
      (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (4): Bottleneck(
      (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (5): Bottleneck(
      (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
  )
  (layer4): Sequential(
    (0): Bottleneck(
      (conv1): Conv2d(1024, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(2, 2),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(512, 2048, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (downsample): Sequential(
        (0): Conv2d(1024, 2048, kernel_size=(1, 1), stride=(2, 2),
bias=False)
        (1): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): Bottleneck(
      (conv1): Conv2d(2048, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(512, 2048, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
    (2): Bottleneck(
      (conv1): Conv2d(2048, 512, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (conv3): Conv2d(512, 2048, kernel_size=(1, 1), stride=(1, 1),
bias=False)
      (bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
    )
  )
  (avgpool): AdaptiveAvgPool2d(output_size=(1, 1))
  (fc): Linear(in_features=2048, out_features=27, bias=True)
)

import torch
import torch.nn.functional as F

def combined_prediction(alex_model, res_model, dataloader,
device,al,re):
    alex_model.eval()
    res_model.eval()

    correct = 0
    total = 0

    with torch.no_grad():
        for images, labels in dataloader:
            images, labels = images.to(device), labels.to(device)

            alex_outputs = alex_model(images)
            res_outputs = res_model(images)

            alex_probs = F.softmax(alex_outputs, dim=1)
            res_probs = F.softmax(res_outputs, dim=1)

            combined_probs = al * alex_probs + re * res_probs
            _, predicted = torch.max(combined_probs, 1)

            correct += (predicted == labels).sum().item()
            total += labels.size(0)

    accuracy = 100 * correct / total
    print(f"Combined Model Accuracy: {accuracy:.2f}%")
    return accuracy
```

test the accuracy on the test dataset

```
# 评估模型
model_Alex_artist.eval()
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_artist:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Alex_artist(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Alex Validation Accuracy in artist: {val_acc:.2f}%')
```

```
Alex Validation Accuracy in artist: 89.28%
```

```
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_artist:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Res_artist(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Res Validation Accuracy in artist: {val_acc:.2f}%')
```

```
Res Validation Accuracy in artist: 94.94%
```

```
combined_accuracy = combined_prediction(alex_model=model_Alex_artist, 
res_model=model_Res_artist, dataloader=val_loader_artist, 
device=device,al=0.4,re=0.6)
```

```
Combined Model Accuracy: 97.30%
```

test the accuracy on the genre test data

```
# 评估模型
model_Alex_genre.eval()
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_genre:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Alex_genre(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Alex Validation Accuracy in genre: {val_acc:.2f}%')
```

```
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/
site-packages/PIL/Image.py:3406: DecompressionBombWarning: Image size
(107327830 pixels) exceeds limit of 89478485 pixels, could be
decompression bomb DOS attack.
  warnings.warn(
```

```
Alex Validation Accuracy in genre: 86.03%
```

```
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_genre:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Res_genre(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Res Validation Accuracy in genre: {val_acc:.2f}%')
```

```
Res Validation Accuracy in genre: 75.31%
```

```
combined_accuracy = combined_prediction(alex_model=model_Alex_genre,
res_model=model_Res_genre, dataloader=val_loader_genre,
device=device,al = 0.6,re = 0.4)
```

```
Combined Model Accuracy: 87.18%
```

test the accuracy of the style test data

```
# 评估模型
model_Alex_style.eval()
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_style:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Alex_style(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Alex Validation Accuracy in style: {val_acc:.2f}%')
```

```
Alex Validation Accuracy in style: 79.83%
```

```
correct = 0
total = 0

with torch.no_grad():
    for images, labels in val_loader_style:
        images, labels = images.to(device), labels.to(device)
        outputs = model_Res_style(images)
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

val_acc = 100 * correct / total
print(f'Res Validation Accuracy in style: {val_acc:.2f}%')
```

```
Res Validation Accuracy in style: 56.56%
```

```
combined_accuracy = combined_prediction(alex_model=model_Alex_style,
res_model=model_Res_style, dataloader=val_loader_style,
device=device,al=0.7,re=0.3)
```

```
Combined Model Accuracy: 82.03%
```

Result

In this part, I have uesed the val files to build some data loaders. With this data, we can clearly tell the perfomance of the model.

```
artist_val = pd.read_csv('wikiart_csv/artist_val.csv')
artist_val.columns = ['path', 'artist']
genre_val = pd.read_csv('wikiart_csv/genre_val.csv')
genre_val.columns = ['path', 'genre']
style_val = pd.read_csv('wikiart_csv/style_val.csv')
```

```
style_val.columns =  ['path', 'style']

artist_val['label'] =
label_encoder.fit_transform(artist_val['artist'])
genre_val['label'] = label_encoder.fit_transform(genre_val['genre'])
style_val['label'] = label_encoder.fit_transform(style_val['style'])


img_dir = "wikiart"
artist_val_dataset = ArtDataset(artist_val, img_dir, transform)
genre_val_dataset = ArtDataset(genre_val, img_dir, transform)
style_val_dataset = ArtDataset(style_val, img_dir, transform)

val_loader_artist_f = DataLoader(artist_val_dataset, batch_size=32,
shuffle=False, num_workers=0)
val_loader_genre_f = DataLoader(genre_val_dataset, batch_size=32,
shuffle=False, num_workers=0)
val_loader_style_f = DataLoader(style_val_dataset, batch_size=32,
shuffle=False, num_workers=0)

print('----Artist data final acc----')
combined_accuracy = combined_prediction(alex_model=model_Alex_artist,
res_model=model_Res_artist, dataloader=val_loader_artist_f,
device=device,al=0.4,re=0.6)
print('----Genre data final acc----')
combined_accuracy = combined_prediction(alex_model=model_Alex_genre,
res_model=model_Res_genre, dataloader=val_loader_genre_f,
device=device,al = 0.6,re=0.4)
print('----Style data final acc----')
combined_accuracy = combined_prediction(alex_model=model_Alex_style,
res_model=model_Res_style, dataloader=val_loader_style_f,
device=device,al=0.7,re=0.3)
```

```
----Artist data final acc----
Combined Model Accuracy: 80.79%
----Genre data final acc----
Combined Model Accuracy: 71.41%
----Style data final acc----
Combined Model Accuracy: 49.50%
```

We can build another fuction to calculate the top 3 probable label of the input

```
def top3_combined_prediction(alex_model, res_model, dataloader,
device, al, re):
    alex_model.eval()
    res_model.eval()

    correct = 0
    total = 0
    with torch.no_grad():
        for images, labels in dataloader:
            images, labels = images.to(device), labels.to(device)

            alex_outputs = alex_model(images)
            res_outputs = res_model(images)

            alex_probs = F.softmax(alex_outputs, dim=1)
            res_probs = F.softmax(res_outputs, dim=1)

            combined_probs = al * alex_probs + re * res_probs  # 权重加
权组合

            top3_probs, top3_indices = torch.topk(combined_probs, 3,
dim=1)  # 取前三个最大概率的类别

            correct += (top3_indices == labels.view(-1,
1)).sum().item()  # 计算是否真实标签在前3个预测值中
            total += labels.size(0)

    accuracy = 100 * correct / total
    print(f"Top-3 Combined Model Accuracy: {accuracy:.2f}%")
    return accuracy

print('----artist data----')
top3_combined_prediction(model_Alex_artist,model_Res_artist,
val_loader_artist_f, device,0.4,0.6)
print('----genre data----')
top3_combined_prediction(model_Alex_genre,model_Res_genre,
val_loader_genre_f, device,0.6,0.4)
print('----style data----')
top3_combined_prediction(model_Alex_style,model_Res_style,
val_loader_style_f, device,0.7,0.3)
```

```
----artist data----
Top-3 Combined Model Accuracy: 91.97%
----genre data----
Top-3 Combined Model Accuracy: 93.51%
----style data----
Top-3 Combined Model Accuracy: 80.42%

80.42178542178542
```

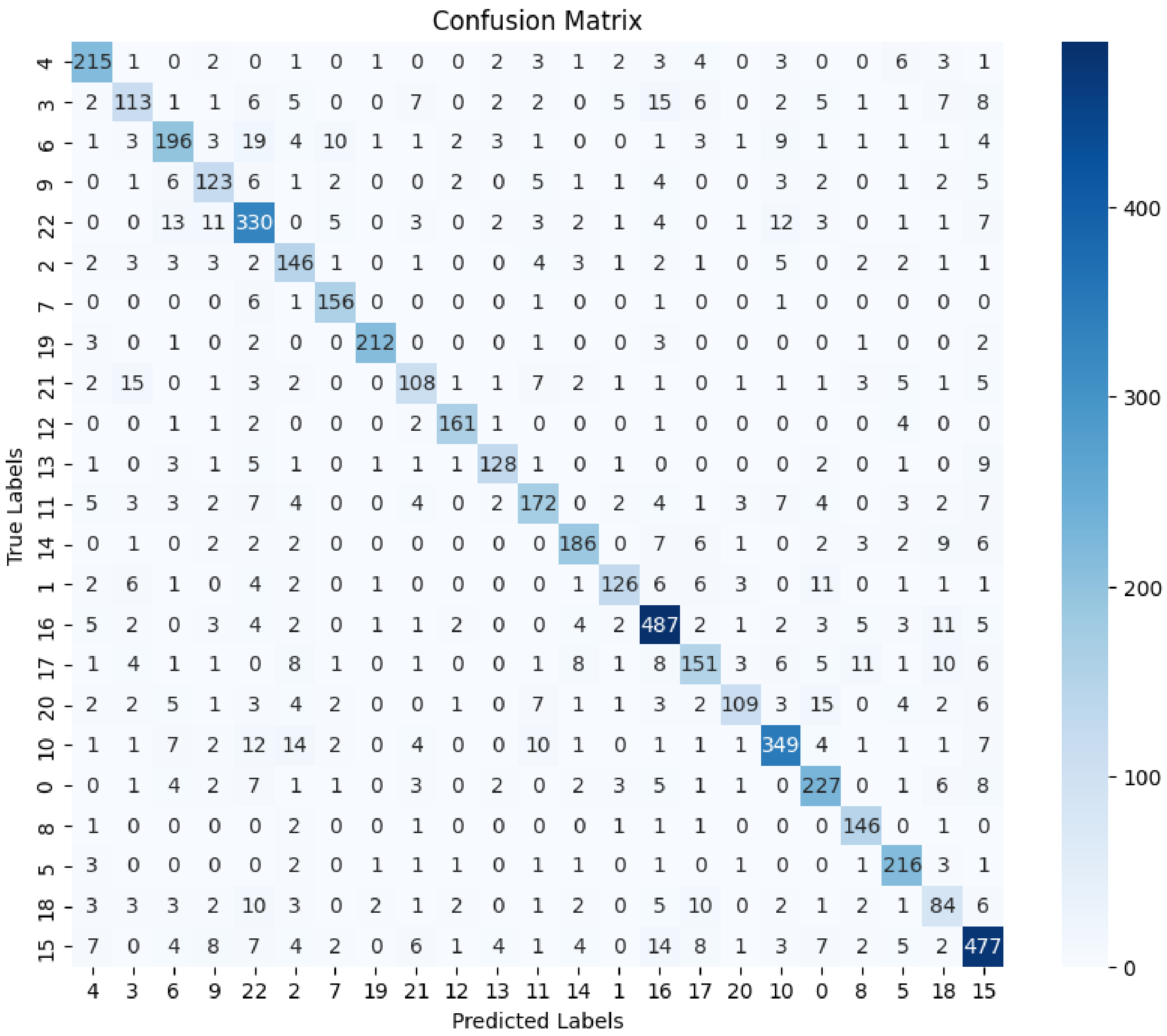
Additionally, we can calculate the per-class accuracy of the model, with the aim of find the outliars of the data.

```
import torch
import torch.nn.functional as F
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix
```

```
import seaborn as sns
import matplotlib.pyplot as plt

def per_class_accuracy(alex_model,res_model, dataloader, device,
class_names,al,re):

    alex_model.eval()
    res_model.eval()

    y_true = []
    y_pred = []

    with torch.no_grad():
        for images, labels in dataloader:
            images, labels = images.to(device), labels.to(device)

            alex_outputs = alex_model(images)
            res_outputs = res_model(images)

            alex_probs = F.softmax(alex_outputs, dim=1)
            res_probs = F.softmax(res_outputs, dim=1)

            combined_probs = al * alex_probs + re * res_probs

            _, predicted = torch.max(combined_probs, 1)
            y_true.extend(labels.cpu().numpy())
            y_pred.extend(predicted.cpu().numpy())

    report = classification_report(y_true, y_pred,
target_names=class_names, digits=4)
    print("Classification Report:\n", report)

    cm = confusion_matrix(y_true, y_pred)

    # Plot Confusion Matrix
    plt.figure(figsize=(10, 8))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
xticklabels=class_names, yticklabels=class_names)
    plt.xlabel("Predicted Labels")
    plt.ylabel("True Labels")
    plt.title("Confusion Matrix")
    plt.show()

    return report, cm

print('----artist part per-class acc----')
class_names = [str(name) for name in
artist_val["label"].unique().tolist()]
report, cm = per_class_accuracy(model_Alex_artist, model_Res_artist,
val_loader_artist_f, device, class_names, 0.4, 0.6)
print(report)
```

```python
print(cm)

print('----genre part per-class acc----')
class_names = [str(name) for name in
genre_val["label"].unique().tolist()]
report, cm = per_class_accuracy(model_Alex_genre, model_Res_genre,
val_loader_genre_f, device, class_names, 0.6, 0.4)
print(report)
print(cm)

print('----style part per-class acc----')
class_names = [str(name) for name in
style_val["label"].unique().tolist()]
report, cm = per_class_accuracy(model_Alex_style, model_Res_style,
val_loader_style_f, device, class_names, 0.7, 0.3)
print(report)
print(cm)
```

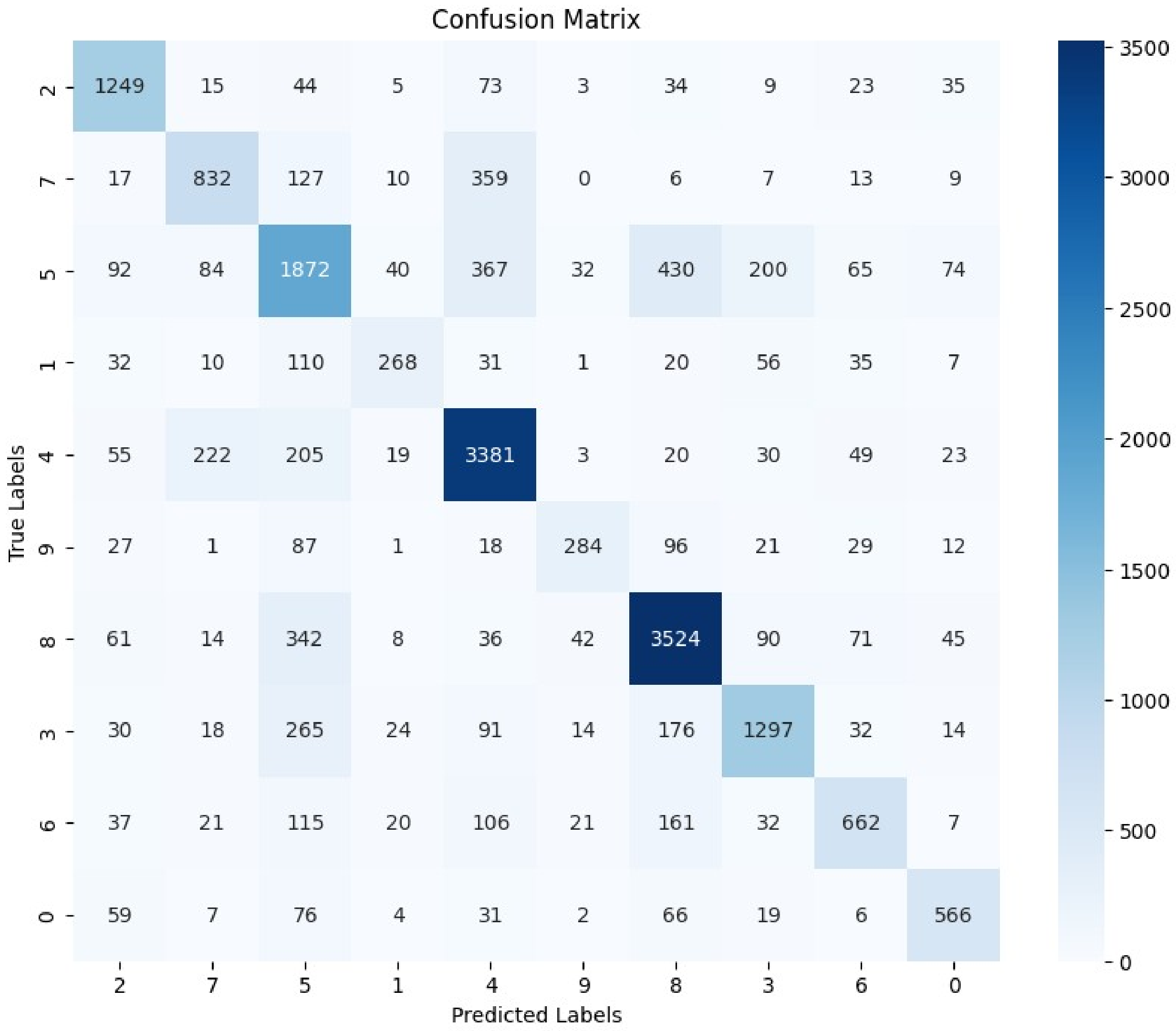
```
----artist part per-class acc----
Classification Report:
               precision    recall  f1-score   support

           4     0.8398    0.8669    0.8532       248
           3     0.7107    0.5979    0.6494       189
           6     0.7778    0.7368    0.7568       266
           9     0.7278    0.7455    0.7365       165
          22     0.7551    0.8271    0.7895       399
           2     0.6986    0.7978    0.7449       183
           7     0.8571    0.9398    0.8966       166
          19     0.9636    0.9422    0.9528       225
          21     0.7448    0.6708    0.7059       161
          12     0.9253    0.9306    0.9280       173
          13     0.8707    0.8205    0.8449       156
          11     0.7783    0.7319    0.7544       235
          14     0.8493    0.8122    0.8304       229
           1     0.8514    0.7326    0.7875       172
          16     0.8440    0.8936    0.8681       545
          17     0.7438    0.6623    0.7007       228
          20     0.8583    0.6301    0.7267       173
          10     0.8554    0.8310    0.8430       420
           0     0.7747    0.8255    0.7993       275
           8     0.8156    0.9481    0.8769       154
           5     0.8308    0.9270    0.8763       233
          18     0.5676    0.5874    0.5773       143
          15     0.8339    0.8413    0.8376       567

    accuracy                         0.8095      5705
   macro avg     0.8032    0.7956    0.7972      5705
weighted avg     0.8100    0.8095    0.8080      5705
```



	4	3	6	9	22	2	7	19	21	12	13	11	14	1	16	17	20	10	0	8	5	18	15
4	215	1	0	2	0	1	0	1	0	0	2	3	1	2	3	4	0	3	0	0	6	3	1
3	2	113	1	1	6	5	0	0	7	0	2	2	0	5	15	6	0	2	5	1	1	7	8
6	1	3	196	3	19	4	10	1	1	2	3	1	0	0	1	3	1	9	1	1	1	1	4
9	0	1	6	123	6	1	2	0	0	2	0	5	1	1	4	0	0	3	2	0	1	2	5
22	0	0	13	11	330	0	5	0	3	0	2	3	2	1	4	0	1	12	3	0	1	1	7
2	2	3	3	3	2	146	1	0	1	0	0	4	3	1	2	1	0	5	0	2	2	1	1
7	0	0	0	0	6	1	156	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0
19	3	0	1	0	2	0	0	212	0	0	0	1	0	0	3	0	0	0	0	1	0	0	2
21	2	15	0	1	3	2	0	0	108	1	1	7	2	1	1	0	1	1	1	3	5	1	5
12	0	0	1	1	2	0	0	0	2	161	1	0	0	0	1	0	0	0	0	0	4	0	0
13	1	0	3	1	5	1	0	1	1	1	128	1	0	1	0	0	0	0	2	0	1	0	9
11	5	3	3	2	7	4	0	0	4	0	2	172	0	2	4	1	3	7	4	0	3	2	7
14	0	1	0	2	2	2	0	0	0	0	0	0	186	0	7	6	1	0	2	3	2	9	6
1	2	6	1	0	4	2	0	1	0	0	0	0	1	126	6	6	3	0	11	0	1	1	1
16	5	2	0	3	4	2	0	1	1	2	0	0	4	2	487	2	1	2	3	5	3	11	5
17	1	4	1	1	0	8	1	0	1	0	0	1	8	1	8	151	3	6	5	11	1	10	6
20	2	2	5	1	3	4	2	0	0	1	0	7	1	1	3	2	109	3	15	0	4	2	6
10	1	1	7	2	12	14	2	0	4	0	0	10	1	0	1	1	1	349	4	1	1	1	7
0	0	1	4	2	7	1	1	0	3	0	2	0	2	3	5	1	1	0	227	0	1	6	8
8	1	0	0	0	0	2	0	0	1	0	0	0	0	1	1	1	0	0	0	146	0	1	0
5	3	0	0	0	0	2	0	1	1	1	0	1	1	0	1	0	1	0	0	1	216	3	1
18	3	3	3	2	10	3	0	2	1	2	0	1	2	0	5	10	0	2	1	2	1	84	6
15	7	0	4	8	7	4	2	0	6	1	4	1	4	0	14	8	1	3	7	2	5	2	477

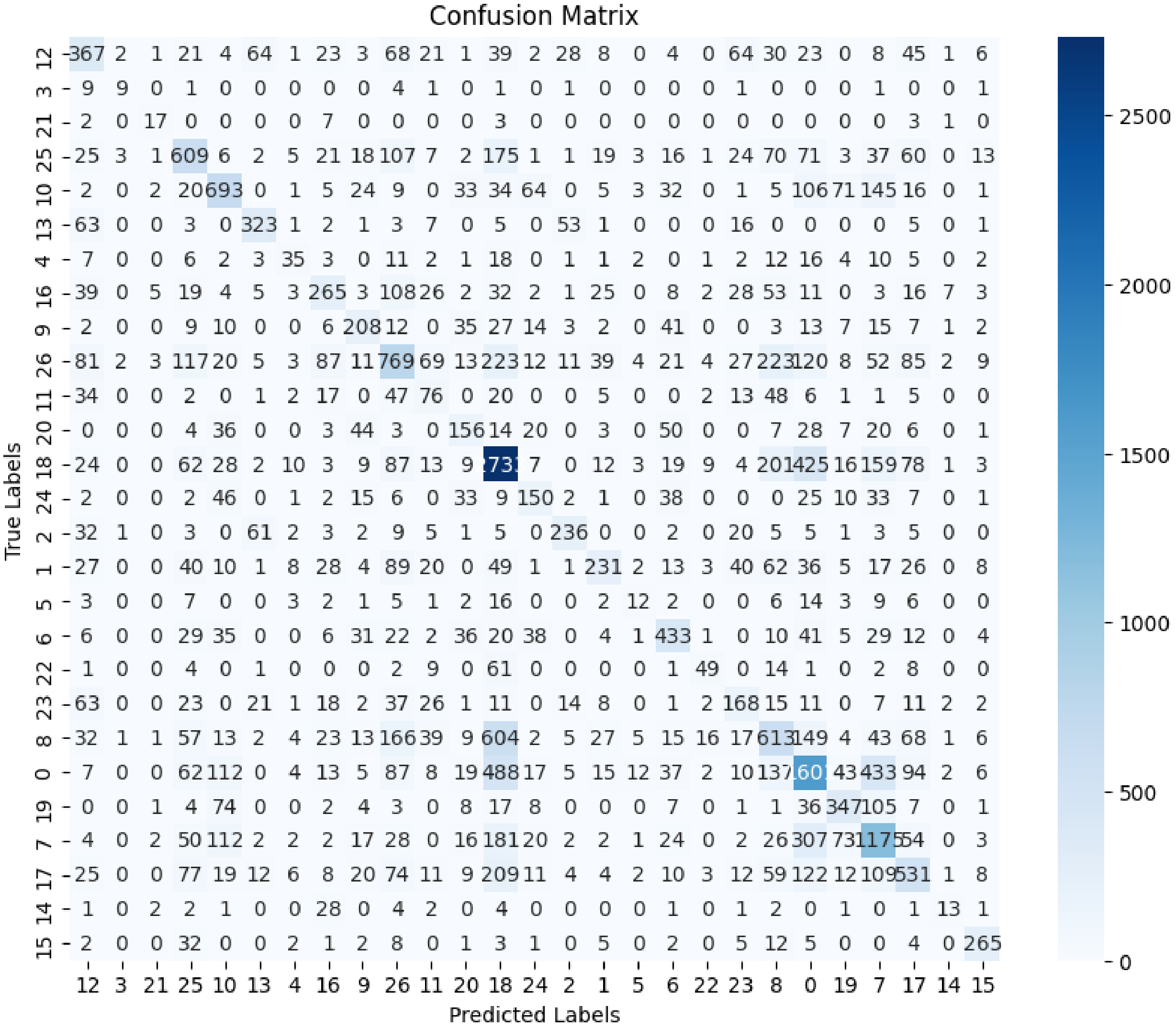
```
              precision    recall  f1-score   support

           4     0.8398    0.8669    0.8532       248
           3     0.7107    0.5979    0.6494       189
           6     0.7778    0.7368    0.7568       266
           9     0.7278    0.7455    0.7365       165
          22     0.7551    0.8271    0.7895       399
           2     0.6986    0.7978    0.7449       183
           7     0.8571    0.9398    0.8966       166
          19     0.9636    0.9422    0.9528       225
          21     0.7448    0.6708    0.7059       161
          12     0.9253    0.9306    0.9280       173
          13     0.8707    0.8205    0.8449       156
          11     0.7783    0.7319    0.7544       235
          14     0.8493    0.8122    0.8304       229
           1     0.8514    0.7326    0.7875       172
          16     0.8440    0.8936    0.8681       545
          17     0.7438    0.6623    0.7007       228
          20     0.8583    0.6301    0.7267       173
          10     0.8554    0.8310    0.8430       420
           0     0.7747    0.8255    0.7993       275
           8     0.8156    0.9481    0.8769       154
           5     0.8308    0.9270    0.8763       233
          18     0.5676    0.5874    0.5773       143
          15     0.8339    0.8413    0.8376       567

    accuracy                         0.8095      5705
   macro avg     0.8032    0.7956    0.7972      5705
weighted avg     0.8100    0.8095    0.8080      5705

[[215   1   0   2   0   1   0   1   0   0   2   3   1   2   3   4   0
3
    0   0   6   3   1]
 [  2 113   1   1   6   5   0   0   7   0   2   2   0   5  15   6   0
2
    5   1   1   7   8]
 [  1   3 196   3  19   4  10   1   1   2   3   1   0   0   1   3   1
9
    1   1   1   1   4]
 [  0   1   6 123   6   1   2   0   0   2   0   5   1   1   4   0   0
3
    2   0   1   2   5]
 [  0   0  13  11 330   0   5   0   3   0   2   3   2   1   4   0   1
12
    3   0   1   1   7]
 [  2   3   3   3   2 146   1   0   1   0   0   4   3   1   2   1   0
5
    0   2   2   1   1]
 [  0   0   0   0   6   1 156   0   0   0   0   1   0   0   1   0   0
1
    0   0   0   0   0]
 [  3   0   1   0   2   0   0 212   0   0   0   1   0   0   3   0   0
0
    0   1   0   0   2]
 [  2  15   0   1   3   2   0   0 108   1   1   7   2   1   1   0   1
1
    1   3   5   1   5]
 [  0   0   1   1   2   0   0   0   2 161   1   0   0   0   1   0   0
0
    0   0   4   0   0]
 [  1   0   3   1   5   1   0   1   1   1 128   1   0   1   0   0   0
0
    2   0   1   0   9]
 [  5   3   3   2   7   4   0   0   4   0   2 172   0   2   4   1   3
7
```

```
    4   0   3   2   7]
 [  0   1   0   2   2   2   0   0   0   0   0   0 186   0   7   6   1
0
    2   3   2   9   6]
 [  2   6   1   0   4   2   0   1   0   0   0   0   1 126   6   6   3
0
   11   0   1   1   1]
 [  5   2   0   3   4   2   0   1   1   2   0   0   4   2 487   2   1
2
    3   5   3  11   5]
 [  1   4   1   1   0   8   1   0   1   0   0   1   8   1   8 151   3
6
    5  11   1  10   6]
 [  2   2   5   1   3   4   2   0   0   1   0   7   1   1   3   2 109
3
   15   0   4   2   6]
 [  1   1   7   2  12  14   2   0   4   0   0  10   1   0   1   1   1
349
    4   1   1   1   7]
 [  0   1   4   2   7   1   1   0   3   0   2   0   2   3   5   1   1
0
  227   0   1   6   8]
 [  1   0   0   0   0   2   0   0   1   0   0   0   0   1   1   1   0
0
    0 146   0   1   0]
 [  3   0   0   0   0   2   0   1   1   1   0   1   1   0   1   0   1
0
    0   1 216   3   1]
 [  3   3   3   2  10   3   0   2   1   2   0   1   2   0   5  10   0
2
    1   2   1  84   6]
 [  7   0   4   8   7   4   2   0   6   1   4   1   4   0  14   8   1
3
    7   2   5   2 477]]
----genre part per-class acc----
Classification Report:
               precision    recall  f1-score   support

           2     0.7529    0.8383    0.7933      1490
           7     0.6797    0.6029    0.6390      1380
           5     0.5772    0.5749    0.5761      3256
           1     0.6717    0.4702    0.5531       570
           4     0.7525    0.8438    0.7955      4007
           9     0.7065    0.4931    0.5808       576
           8     0.7774    0.8325    0.8040      4233
           3     0.7365    0.6614    0.6969      1961
           6     0.6721    0.5601    0.6110      1182
           0     0.7146    0.6770    0.6953       836
    accuracy                         0.7149     19491
   macro avg     0.7041    0.6554    0.6745     19491
weighted avg     0.7117    0.7149    0.7106     19491
```



```
              precision    recall  f1-score   support

           2     0.7529    0.8383    0.7933      1490
           7     0.6797    0.6029    0.6390      1380
           5     0.5772    0.5749    0.5761      3256
           1     0.6717    0.4702    0.5531       570
           4     0.7525    0.8438    0.7955      4007
           9     0.7065    0.4931    0.5808       576
           8     0.7774    0.8325    0.8040      4233
           3     0.7365    0.6614    0.6969      1961
           6     0.6721    0.5601    0.6110      1182
           0     0.7146    0.6770    0.6953       836

    accuracy                         0.7149     19491
   macro avg     0.7041    0.6554    0.6745     19491
weighted avg     0.7117    0.7149    0.7106     19491

[[1249   15   44    5   73    3   34    9   23   35]
 [  17  832  127   10  359    0    6    7   13    9]
 [  92   84 1872   40  367   32  430  200   65   74]
 [  32   10  110  268   31    1   20   56   35    7]
 [  55  222  205   19 3381    3   20   30   49   23]
 [  27    1   87    1   18  284   96   21   29   12]
 [  61   14  342    8   36   42 3524   90   71   45]
 [  30   18  265   24   91   14  176 1297   32   14]
 [  37   21  115   20  106   21  161   32  662    7]
 [  59    7   76    4   31    2   66   19    6  566]]
----style part per-class acc----
Classification Report:
               precision    recall  f1-score   support

          12     0.4267    0.4400    0.4333       834
           3     0.5000    0.3103    0.3830        29
          21     0.4857    0.5152    0.5000        33
          25     0.4814    0.4685    0.4749      1300
          10     0.5657    0.5448    0.5551      1272
          13     0.6396    0.6674    0.6532       484
           4     0.3723    0.2431    0.2941       144
          16     0.4585    0.3955    0.4247       670
           9     0.4760    0.4988    0.4871       417
          26     0.4350    0.3807    0.4060      2020
          11     0.2203    0.2714    0.2432       280
          20     0.4031    0.3881    0.3954       402
          18     0.5465    0.6977    0.6129      3917
          24     0.4054    0.3916    0.3984       383
           2     0.6413    0.5885    0.6138       401
           1     0.5513    0.3204    0.4053       721
           5     0.2400    0.1277    0.1667        94
           6     0.5573    0.5660    0.5616       765
          22     0.5158    0.3203    0.3952       153
          23     0.3684    0.3784    0.3733       444
           8     0.3798    0.3168    0.3454      1935
           0     0.5047    0.4974    0.5010      3219
          19     0.5588    0.5543    0.5565       626
           7     0.4863    0.5582    0.5198      2105
          17     0.4558    0.3910    0.4209      1358
          14     0.4062    0.2031    0.2708        64
          15     0.7637    0.7571    0.7604       350

    accuracy                         0.4948     24420
   macro avg     0.4758    0.4368    0.4501     24420
```

```
weighted avg     0.4910    0.4948    0.4893     24420
```



	12	3	21	25	10	13	4	16	9	26	11	20	18	24	2	1	5	6	22	23	8	0	19	7	17	14	15
12	367	2	1	21	4	64	1	23	3	68	21	1	39	2	28	8	0	4	0	64	30	23	0	8	45	1	6
3	9	9	0	1	0	0	0	0	0	4	1	0	1	0	1	0	0	0	0	1	0	0	0	1	0	0	1
21	2	0	17	0	0	0	0	7	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	3	1	0
25	25	3	1	609	6	2	5	21	18	107	7	2	175	1	1	19	3	16	1	24	70	71	3	37	60	0	13
10	2	0	2	20	693	0	1	5	24	9	0	33	34	64	0	5	3	32	0	1	5	106	71	145	16	0	1
13	63	0	0	3	0	323	1	2	1	3	7	0	5	0	53	1	0	0	0	16	0	0	0	0	5	0	1
4	7	0	0	6	2	3	35	3	0	11	2	1	18	0	1	1	2	0	1	2	12	16	4	10	5	0	2
16	39	0	5	19	4	5	3	265	3	108	26	2	32	2	1	25	0	8	2	28	53	11	0	3	16	7	3
9	2	0	0	9	10	0	0	6	208	12	0	35	27	14	3	2	0	41	0	0	3	13	7	15	7	1	2
26	81	2	3	117	20	5	3	87	11	769	69	13	223	12	11	39	4	21	4	27	223	120	8	52	85	2	9
11	34	0	0	2	0	1	2	17	0	47	76	0	20	0	0	5	0	0	2	13	48	6	1	1	5	0	0
20	0	0	0	4	36	0	0	3	44	3	0	156	14	20	0	3	0	50	0	0	7	28	7	20	6	0	1
18	24	0	0	62	28	2	10	3	9	87	13	9	2731	7	0	12	3	19	9	4	201	425	16	159	78	1	3
24	2	0	0	2	46	0	1	2	15	6	0	33	9	150	2	1	0	38	0	0	0	25	10	33	7	0	1
2	32	1	0	3	0	61	2	3	2	9	5	1	5	0	236	0	0	2	0	20	5	5	1	3	5	0	0
1	27	0	0	40	10	1	8	28	4	89	20	0	49	1	1	231	2	13	3	40	62	36	5	17	26	0	8
5	3	0	0	7	0	0	3	2	1	5	1	2	16	0	0	2	12	2	0	0	6	14	3	9	6	0	0
6	6	0	0	29	35	0	0	6	31	22	2	36	20	38	0	4	1	433	1	0	10	41	5	29	12	0	4
22	1	0	0	4	0	1	0	0	0	2	9	0	61	0	0	0	0	1	49	0	14	1	0	2	8	0	0
23	63	0	0	23	0	21	1	18	2	37	26	1	11	0	14	8	0	1	2	168	15	11	0	7	11	2	2
8	32	1	1	57	13	2	4	23	13	166	39	9	604	2	5	27	5	15	16	17	613	149	4	43	68	1	6
0	7	0	0	62	112	0	4	13	5	87	8	19	488	17	5	15	12	37	2	10	137	1601	43	433	94	2	6
19	0	0	1	4	74	0	0	2	4	3	0	8	17	8	0	0	0	7	0	1	1	36	347	105	7	0	1
7	4	0	2	50	112	2	2	2	17	28	0	16	181	20	2	2	1	24	0	2	26	307	73	1175	54	0	3
17	25	0	0	77	19	12	6	8	20	74	11	9	209	11	4	4	2	10	3	12	59	122	12	109	531	1	8
14	1	0	2	2	1	0	0	28	0	4	2	0	4	0	0	0	0	1	0	1	2	0	1	0	1	13	1
15	2	0	0	32	0	0	2	1	2	8	0	1	3	1	0	5	0	2	0	5	12	5	0	0	4	0	265

```
              precision    recall  f1-score   support

          12     0.4267    0.4400    0.4333       834
           3     0.5000    0.3103    0.3830        29
          21     0.4857    0.5152    0.5000        33
          25     0.4814    0.4685    0.4749      1300
          10     0.5657    0.5448    0.5551      1272
          13     0.6396    0.6674    0.6532       484
           4     0.3723    0.2431    0.2941       144
          16     0.4585    0.3955    0.4247       670
           9     0.4760    0.4988    0.4871       417
          26     0.4350    0.3807    0.4060      2020
          11     0.2203    0.2714    0.2432       280
          20     0.4031    0.3881    0.3954       402
          18     0.5465    0.6977    0.6129      3917
          24     0.4054    0.3916    0.3984       383
           2     0.6413    0.5885    0.6138       401
           1     0.5513    0.3204    0.4053       721
           5     0.2400    0.1277    0.1667        94
           6     0.5573    0.5660    0.5616       765
          22     0.5158    0.3203    0.3952       153
          23     0.3684    0.3784    0.3733       444
           8     0.3798    0.3168    0.3454      1935
           0     0.5047    0.4974    0.5010      3219
          19     0.5588    0.5543    0.5565       626
           7     0.4863    0.5582    0.5198      2105
          17     0.4558    0.3910    0.4209      1358
          14     0.4062    0.2031    0.2708        64
          15     0.7637    0.7571    0.7604       350

    accuracy                         0.4948     24420
   macro avg     0.4758    0.4368    0.4501     24420
weighted avg     0.4910    0.4948    0.4893     24420

[[ 367    2    1   21    4   64    1   23    3   68   21    1   39
2
    28    8    0    4    0   64   30   23    0    8   45    1    6]
 [   9    9    0    1    0    0    0    0    0    4    1    0    1
0
     1    0    0    0    0    1    0    0    0    1    0    0    1]
 [   2    0   17    0    0    0    0    7    0    0    0    0    3
0
     0    0    0    0    0    0    0    0    0    0    3    1    0]
 [  25    3    1  609    6    2    5   21   18  107    7    2  175
1
     1   19    3   16    1   24   70   71    3   37   60    0   13]
 [   2    0    2   20  693    0    1    5   24    9    0   33   34
64
     0    5    3   32    0    1    5  106   71  145   16    0    1]
 [  63    0    0    3    0  323    1    2    1    3    7    0    5
0
    53    1    0    0    0   16    0    0    0    0    5    0    1]
 [   7    0    0    6    2    3   35    3    0   11    2    1   18
0
     1    1    2    0    1    2   12   16    4   10    5    0    2]
 [  39    0    5   19    4    5    3  265    3  108   26    2   32
2
     1   25    0    8    2   28   53   11    0    3   16    7    3]
 [   2    0    0    9   10    0    0    6  208   12    0   35   27
14
     3    2    0   41    0    0    3   13    7   15    7    1    2]
 [  81    2    3  117   20    5    3   87   11  769   69   13  223
```

```
12
    11   39    4   21    4   27  223  120    8   52   85    2    9]
 [  34    0    0    2    0    1    2   17    0   47   76    0   20
0
     0    5    0    0    2   13   48    6    1    1    5    0    0]
 [   0    0    0    4   36    0    0    3   44    3    0  156   14
20
     0    3    0   50    0    0    7   28    7   20    6    0    1]
 [  24    0    0   62   28    2   10    3    9   87   13    9 2733
7
     0   12    3   19    9    4  201  425   16  159   78    1    3]
 [   2    0    0    2   46    0    1    2   15    6    0   33    9
150
     2    1    0   38    0    0    0   25   10   33    7    0    1]
 [  32    1    0    3    0   61    2    3    2    9    5    1    5
0
   236    0    0    2    0   20    5    5    1    3    5    0    0]
 [  27    0    0   40   10    1    8   28    4   89   20    0   49
1
     1  231    2   13    3   40   62   36    5   17   26    0    8]
 [   3    0    0    7    0    0    3    2    1    5    1    2   16
0
     0    2   12    2    0    0    6   14    3    9    6    0    0]
 [   6    0    0   29   35    0    0    6   31   22    2   36   20
38
     0    4    1  433    1    0   10   41    5   29   12    0    4]
 [   1    0    0    4    0    1    0    0    0    2    9    0   61
0
     0    0    0    1   49    0   14    1    0    2    8    0    0]
 [  63    0    0   23    0   21    1   18    2   37   26    1   11
0
    14    8    0    1    2  168   15   11    0    7   11    2    2]
 [  32    1    1   57   13    2    4   23   13  166   39    9  604
2
     5   27    5   15   16   17  613  149    4   43   68    1    6]
 [   7    0    0   62  112    0    4   13    5   87    8   19  488
17
     5   15   12   37    2   10  137 1601   43  433   94    2    6]
 [   0    0    1    4   74    0    0    2    4    3    0    8   17
8
     0    0    0    7    0    1    1   36  347  105    7    0    1]
 [   4    0    2   50  112    2    2    2   17   28    0   16  181
20
     2    2    1   24    0    2   26  307   73 1175   54    0    3]
 [  25    0    0   77   19   12    6    8   20   74   11    9  209
11
     4    4    2   10    3   12   59  122   12  109  531    1    8]
 [   1    0    2    2    1    0    0   28    0    4    2    0    4
0
```

```
     0    0    0    1    0    1    2    0    1    0    1   13    1]
 [   2    0    0   32    0    0    2    1    2    8    0    1    3
1
     0    5    0    2    0    5   12    5    0    0    4    0  265]]
```