# 数据处理方式

# 非随机变换，保证每次预测结果都是一致的  
transform = transforms.Compose([  
 transforms.Resize(dest\_image\_size),  
 transforms.CenterCrop(dest\_image\_size),  
 transforms.RandomHorizontalFlip(p=1),  
 transforms.ToTensor(),  
 normalize  
])

# 模型参数

parser.add\_argument('--classes\_num', help='类别数', default=4)  
parser.add\_argument('--train\_data', help='训练文件夹的位置', default='./data/train')  
parser.add\_argument('--test\_data', help='测试文件夹的位置', default='./data/test')  
parser.add\_argument('--image\_size', help='输入图片的宽与高，B0推荐224', default=224)  
parser.add\_argument('--batch\_size', help='batchsize数', default=32)  
parser.add\_argument('--workers', help='Dataloader的worker数', default=2)  
parser.add\_argument('--epochs', help='epoch数', default=10)  
parser.add\_argument('--lr', help='学习率', default=0.001)  
parser.add\_argument('--checkpoint\_dir', help='模型保存位置', default='./checkpoints')  
parser.add\_argument('--save\_interval', help='保存间隔，每1个epoch保存一次', default=1)  
parser.add\_argument('--momentum', help='momentum动量', default=0.9)  
parser.add\_argument('--weight\_decay', help='权重衰减', default=1e-04)  
parser.add\_argument('--arch', help='使用网络结构', default='efficientnet-b0')  
parser.add\_argument('--pretrained', help='是否加载预训练模型', type=bool, default=True)

# 训练过程

C:\ProgramData\Anaconda3\python.exe C:/Users/12926/Desktop/github/EfficientNet-PyTorch/main.py

Loaded pretrained weights for efficientnet-b0

=> using pre-trained model 'efficientnet-b0'

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 0 训练过程...

batch: 1 loss:1.4044779539108276

batch: 2 loss:1.3933587074279785

batch: 3 loss:1.332332968711853

batch: 4 loss:1.327340841293335

batch: 5 loss:1.3762131929397583

batch: 6 loss:1.3522419929504395

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 1 训练过程...

batch: 1 loss:1.3339757919311523

batch: 2 loss:1.2378005981445312

batch: 3 loss:1.215526819229126

batch: 4 loss:1.2465420961380005

batch: 5 loss:1.171500563621521

batch: 6 loss:1.0980949401855469

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 2 训练过程...

batch: 1 loss:1.036608099937439

batch: 2 loss:1.2096060514450073

batch: 3 loss:1.0827313661575317

batch: 4 loss:0.9817278981208801

batch: 5 loss:1.0290940999984741

batch: 6 loss:0.9861537218093872

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 3 训练过程...

batch: 1 loss:0.9388275742530823

batch: 2 loss:1.1115992069244385

batch: 3 loss:0.8555353879928589

batch: 4 loss:0.903998851776123

batch: 5 loss:0.8245177865028381

batch: 6 loss:0.7549881339073181

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 4 训练过程...

batch: 1 loss:0.8032539486885071

batch: 2 loss:0.9774380326271057

batch: 3 loss:0.9180282354354858

batch: 4 loss:0.652799129486084

batch: 5 loss:0.7139759063720703

batch: 6 loss:0.6831961274147034

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 5 训练过程...

batch: 1 loss:0.6900100708007812

batch: 2 loss:0.7764784097671509

batch: 3 loss:0.653782308101654

batch: 4 loss:0.6021845936775208

batch: 5 loss:0.6477310657501221

batch: 6 loss:0.6676013469696045

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 6 训练过程...

batch: 1 loss:0.6381440162658691

batch: 2 loss:0.6873742938041687

batch: 3 loss:0.9147605895996094

batch: 4 loss:0.5850105285644531

batch: 5 loss:0.48319000005722046

batch: 6 loss:0.5159196853637695

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 7 训练过程...

batch: 1 loss:0.48363345861434937

batch: 2 loss:0.49987953901290894

batch: 3 loss:0.47831881046295166

batch: 4 loss:0.6673957705497742

batch: 5 loss:0.46792489290237427

batch: 6 loss:0.5429158806800842

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 8 训练过程...

batch: 1 loss:0.4422817826271057

batch: 2 loss:0.43194037675857544

batch: 3 loss:0.46592476963996887

batch: 4 loss:0.4368884265422821

batch: 5 loss:0.4106200933456421

batch: 6 loss:0.4475553631782532

\*\*\*\*\*\*\*\*\*\*\*epoch分割线\*\*\*\*\*\*\*\*\*\*\*\*\*

Epoch 9 训练过程...

batch: 1 loss:0.40364351868629456

batch: 2 loss:0.37927573919296265

batch: 3 loss:0.36346685886383057

batch: 4 loss:0.3636566400527954

batch: 5 loss:0.4105243682861328

batch: 6 loss:0.4743803143501282

# 评估结果

所有61张图片的预测类别：tensor([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1.,

1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,

1., 1., 1., 1., 1., 1., 1., 2., 2., 1., 1., 1., 2., 1., 2., 1., 1., 3.,

1., 3., 3., 3., 3., 1., 3.])

所有61张图片的实际类别：tensor([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1.,

1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,

1., 1., 1., 1., 1., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 2., 3., 3.,

3., 3., 3., 3., 3., 3., 3.])

混淆矩阵：

[[15 0 0 0]

[ 0 26 0 0]

[ 0 7 4 0]

[ 0 3 0 6]]

准确率: 0.8360655737704918

第 0 类 precision:1.0 recall:1.0

第 1 类 precision:0.7222222222222222 recall:1.0

第 2 类 precision:1.0 recall:0.36363636363636365

第 3 类 precision:1.0 recall:0.6666666666666666

