深度學習專題:水果分類

成員:林家平

引用資料集


功能表
kaggle
搜索
創造
探索
比賽
數據
法典
討論
學習
更多
您的工作
最近查看
CNN的水果和蔬菜分類
查看活動事件
2582
新筆記本
下載（1GB）
水果360
包含 131 種水果和蔬菜的 90380 張圖像的數據集
數據卡
代碼（471）
討論（24）
關於數據集
水果 360 數據集：包含水果和蔬菜的圖像數據集
版本： 2020.05.18.0
可用性
8.75
許可證
CC BY-SA 4.0
預期更新頻率

算法:CNN

```
class FruitCnnModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.network = nn.Sequential(

            nn.Conv2d(3, 16, kernel_size=2, padding=1), #99
            nn.BatchNorm2d(16),
            nn.ReLU(),
            nn.MaxPool2d(2, 2), #99/2=50


            nn.Conv2d(16, 32, kernel_size=2, stride=1, padding=1), #49
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(2, 2), #49/2=25


            nn.Conv2d(32, 64, kernel_size=2, stride=1, padding=1),#24
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(5, 5), #24/5=5


            nn.Flatten(),
            nn.Dropout(0.5),
            nn.ReLU(),
            nn.Linear(64*5*5, 131))

    def forward(self, xb):
      return self.network(xb)
```

訓練

```
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
n_epochs = 4
valid_loss_min = np.Inf
```

```
running epoch: 1
        Training Loss: 1.306246         Validation Loss: 0.146474
Validation loss decreased (inf --> 0.146474).  Saving model ...
running epoch: 2
        Training Loss: 0.137370         Validation Loss: 0.041651
Validation loss decreased (0.146474 --> 0.041651).  Saving model ...
running epoch: 3
        Training Loss: 0.060066         Validation Loss: 0.013181
Validation loss decreased (0.041651 --> 0.013181).  Saving model ...
running epoch: 4
        Training Loss: 0.035998         Validation Loss: 0.007642
Validation loss decreased (0.013181 --> 0.007642).  Saving model ...
```

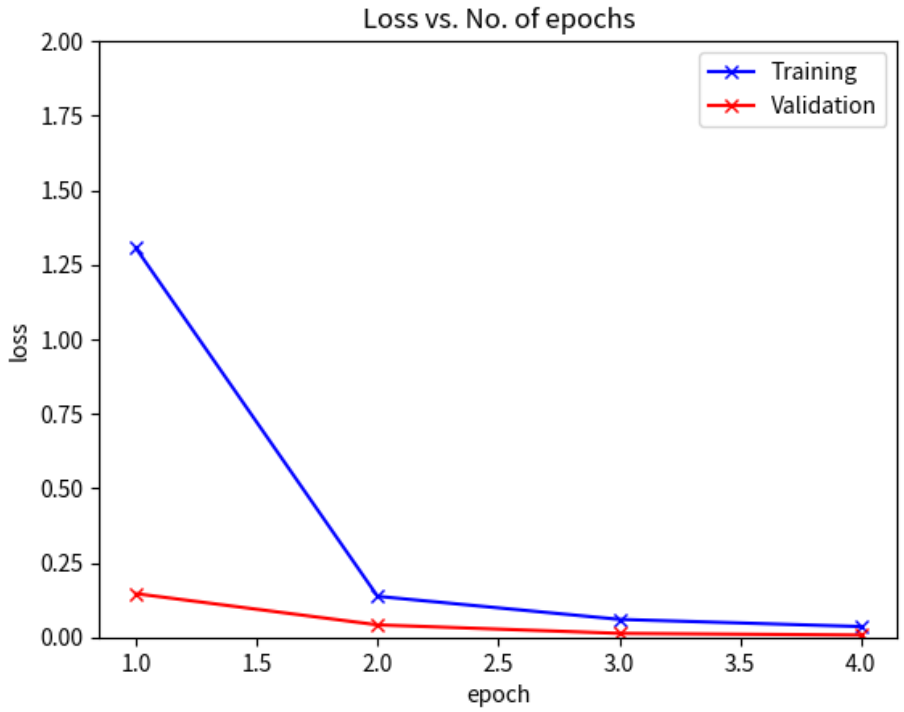
測試結果

```
def test(loaders, model, criterion, use_cuda):


    test_loss = 0.
    correct = 0.
    total = 0.
    model.eval()
    for batch_idx, (data, target) in enumerate(loaders):

        if use_cuda:
            data, target = data.cuda(), target.cuda()

        output = model(data)

        loss = criterion(output, target)

        test_loss = test_loss + ((1 / (batch_idx + 1)) * (loss.data - test_loss))

        pred = output.data.max(1, keepdim=True)[1]

        correct += np.sum(pred.eq(target.data.view_as(pred)).cpu().numpy())
        total += data.size(0)

    print('Test Loss: {:.6f}'.format(test_loss))
    print('Test Accuracy: %2d%% (%2d/%2d)' % (100. * correct / total, correct, total))


use_cuda = torch.cuda.is_available()
test(test_loader, model, criterion, use_cuda)
```

```
Test Loss: 0.077323
Test Accuracy: 98% (22305/22688)
```

損失值曲線圖

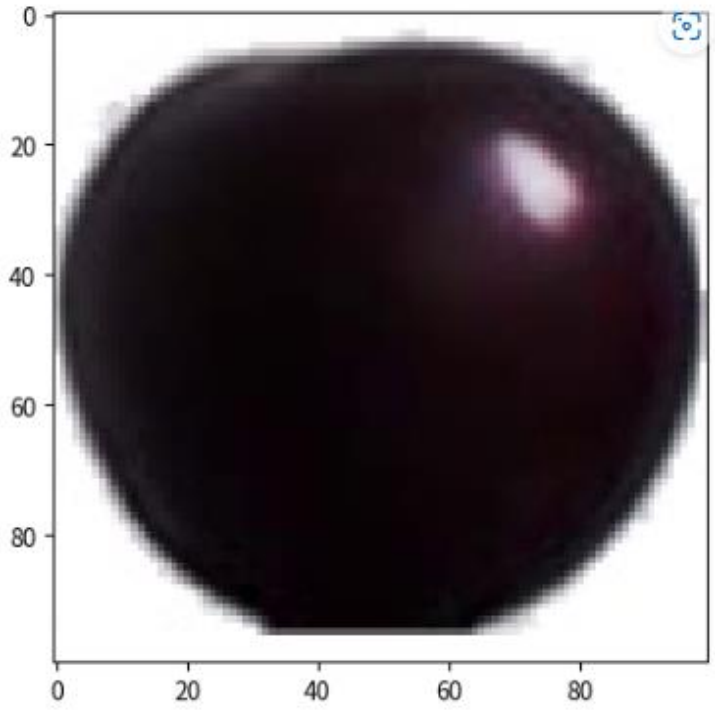
```
train_losses = [x for x in train_losses]
epochs=[1,2,3,4]
val_losses = [x for x in valid_losses]
plt.plot(epochs,train_losses, '-bx')
plt.plot(epochs,val_losses, '-rx')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.ylim(0,2)
plt.legend(['Training', 'Validation'])
plt.title('Loss vs. No. of epochs');

plt.show()
```



模型測試

```
In [11]: img, label = test_data[5000]
         plt.imshow(img.permute(1, 2, 0))
         print('Label:', train_data.classes[label], 'Predicted:', predict_image(img, model))
```

```
Label: Cherry Wax Black Predicted: Cherry Wax Black
```



蝴蝶和飛蛾圖像分類


GERRY · UPDATED 3 MONTHS AGO
86
New Notebook
Download (476 MB)
Butterfly & Moths Image Classification 100 species
12639 train, 500 test, 500 validation images 224 X 224 X 3 jpg format
Data Card
Code (29)
Discussion (1)

算法:CNN

```
class Butterfly_vs_Classifier_CnnModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.network = nn.Sequential(

            nn.Conv2d(3, 16, kernel_size=5, padding=0),
            nn.BatchNorm2d(16),
            nn.ReLU(),
            nn.MaxPool2d(2, 2),


            nn.Conv2d(16, 32, kernel_size=5, stride=1, padding=0),
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(2, 2),


            nn.Conv2d(32, 16, kernel_size=3, stride=1, padding=0),
            nn.BatchNorm2d(16),
            nn.ReLU(),
            nn.MaxPool2d(2, 2),

            nn.Conv2d(16, 8, kernel_size=3, stride=1, padding=0),
            nn.BatchNorm2d(8),
            nn.ReLU(),
            nn.MaxPool2d(2, 2),


            nn.Flatten(),
            nn.Dropout(0.5),
            nn.ReLU(),
            nn.Linear(8*11*11, 100))

    def forward(self, xb):
      return self.network(xb)


model = FruitCnnModel()
model
```

訓練

```python
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)


n_epochs = 50

valid_loss_min = np.Inf
```

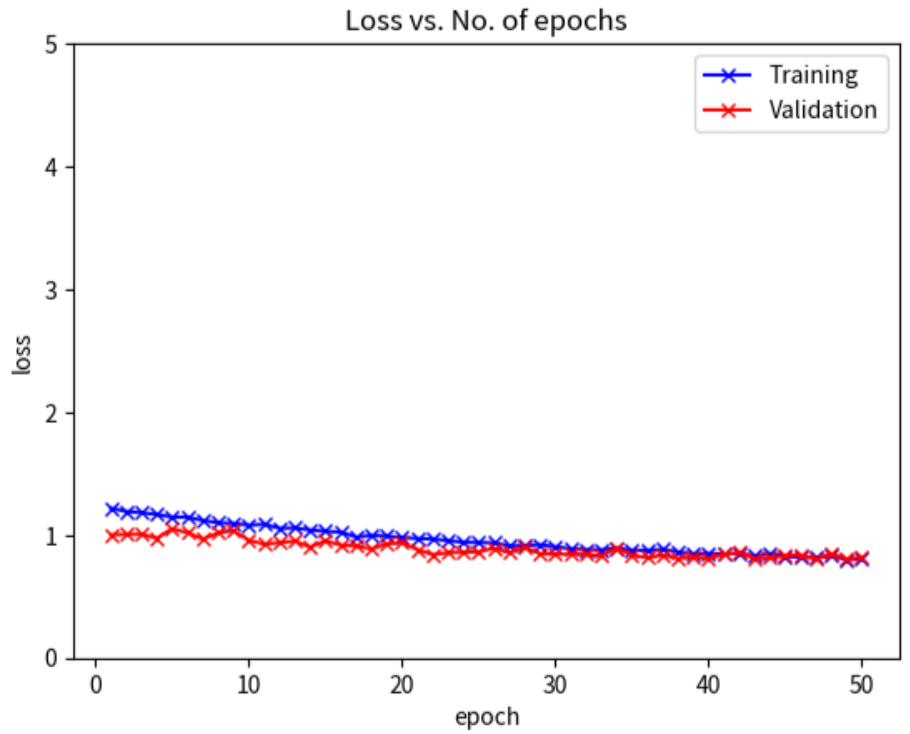
```
running epoch: 1
        Training Loss: 1.219365         Validation Loss: 1.002321
Validation loss decreased (inf --> 1.002321).  Saving model ...
running epoch: 2
        Training Loss: 1.191668         Validation Loss: 1.005435
running epoch: 3
        Training Loss: 1.182129         Validation Loss: 1.006855
running epoch: 4
        Training Loss: 1.167862         Validation Loss: 0.974304
Validation loss decreased (1.002321 --> 0.974304).  Saving model ...
running epoch: 5
        Training Loss: 1.146752         Validation Loss: 1.051923
running epoch: 6
        Training Loss: 1.145451         Validation Loss: 1.018808
running epoch: 7
        Training Loss: 1.119549         Validation Loss: 0.967221
Validation loss decreased (0.974304 --> 0.967221).  Saving model ...
running epoch: 8
        Training Loss: 1.100453         Validation Loss: 1.018015
running epoch: 9
        Training Loss: 1.095090         Validation Loss: 1.034428
running epoch: 10
        Training Loss: 1.080947         Validation Loss: 0.958184
Validation loss decreased (0.967221 --> 0.958184).  Saving model ...
running epoch: 11
        Training Loss: 1.087827         Validation Loss: 0.925947
Validation loss decreased (0.958184 --> 0.925947).  Saving model ...
running epoch: 12
        Training Loss: 1.055455         Validation Loss: 0.938659
running epoch: 13
        Training Loss: 1.058292         Validation Loss: 0.950806
running epoch: 14
        Training Loss: 1.041591         Validation Loss: 0.902710
Validation loss decreased (0.925947 --> 0.902710).  Saving model ...
running epoch: 15
        Training Loss: 1.031274         Validation Loss: 0.952339
running epoch: 16
        Training Loss: 1.024050         Validation Loss: 0.918353
running epoch: 17
        Training Loss: 0.984280         Validation Loss: 0.911466
running epoch: 18
        Training Loss: 0.997803         Validation Loss: 0.887306
Validation loss decreased (0.902710 --> 0.887306).  Saving model ...
running epoch: 19
        Training Loss: 0.991826         Validation Loss: 0.928942
running epoch: 20
        Training Loss: 0.979805         Validation Loss: 0.937860
running epoch: 21
        Training Loss: 0.972732         Validation Loss: 0.881339
Validation loss decreased (0.887306 --> 0.881339).  Saving model ...
running epoch: 22
        Training Loss: 0.967737         Validation Loss: 0.840881
Validation loss decreased (0.881339 --> 0.840881).  Saving model ...
running epoch: 23
        Training Loss: 0.955617         Validation Loss: 0.857133
running epoch: 24
        Training Loss: 0.941583         Validation Loss: 0.860032
running epoch: 25
        Training Loss: 0.939470         Validation Loss: 0.863752
running epoch: 26
        Training Loss: 0.936656         Validation Loss: 0.892765
running epoch: 27
        Training Loss: 0.912217         Validation Loss: 0.859003
running epoch: 28
        Training Loss: 0.913842         Validation Loss: 0.899580
running epoch: 29
        Training Loss: 0.922358         Validation Loss: 0.848848
running epoch: 30
        Training Loss: 0.905812         Validation Loss: 0.843210
running epoch: 31
        Training Loss: 0.889700         Validation Loss: 0.845844
running epoch: 32
        Training Loss: 0.881824         Validation Loss: 0.845093
running epoch: 33
        Training Loss: 0.879473         Validation Loss: 0.829454
Validation loss decreased (0.840881 --> 0.829454).  Saving model ...
running epoch: 34
        Training Loss: 0.894274         Validation Loss: 0.895647
running epoch: 35
        Training Loss: 0.877130         Validation Loss: 0.834756
running epoch: 36
        Training Loss: 0.875628         Validation Loss: 0.817081
Validation loss decreased (0.829454 --> 0.817081).  Saving model ...
running epoch: 37
        Training Loss: 0.883569         Validation Loss: 0.836547
running epoch: 38
        Training Loss: 0.863131         Validation Loss: 0.810686
Validation loss decreased (0.817081 --> 0.810686).  Saving model ...
running epoch: 39
        Training Loss: 0.848675         Validation Loss: 0.826044
running epoch: 40
        Training Loss: 0.851950         Validation Loss: 0.809089
Validation loss decreased (0.810686 --> 0.809089).  Saving model ...
running epoch: 41
        Training Loss: 0.844693         Validation Loss: 0.843245
running epoch: 42
        Training Loss: 0.852901         Validation Loss: 0.859733
running epoch: 43
        Training Loss: 0.829086         Validation Loss: 0.808046
Validation loss decreased (0.809089 --> 0.808046).  Saving model ...
running epoch: 44
        Training Loss: 0.850447         Validation Loss: 0.815440
running epoch: 45
        Training Loss: 0.824561         Validation Loss: 0.831608
running epoch: 46
        Training Loss: 0.820571         Validation Loss: 0.837813
running epoch: 47
        Training Loss: 0.814792         Validation Loss: 0.805094
Validation loss decreased (0.808046 --> 0.805094).  Saving model ...
running epoch: 48
        Training Loss: 0.828432         Validation Loss: 0.848318
running epoch: 49
        Training Loss: 0.801110         Validation Loss: 0.806240
running epoch: 50
        Training Loss: 0.804431         Validation Loss: 0.819413
Test Loss: 0.752096
Test Accuracy: 80% (403/500)
```

測試結果

```
Test Loss: 0.752096
Test Accuracy: 80% (403/500)
```

•損失值曲線圖

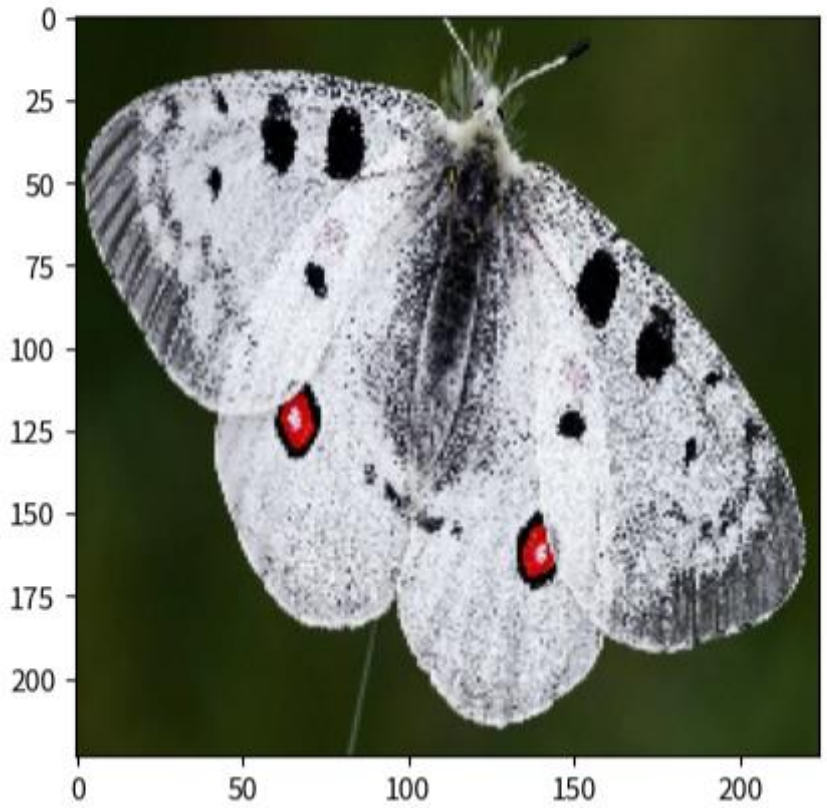
```
train_losses = [x for x in train_losses]
epochs=[i for i in range(1,51)]
val_losses = [x for x in valid_losses]
plt.plot(epochs,train_losses, '-bx')
plt.plot(epochs,val_losses, '-rx')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.ylim(0,5)
plt.legend(['Training', 'Validation'])
plt.title('Loss vs. No. of epochs');

plt.show()
```



模型測試

```
img, label = test_data[20]
plt.imshow(img.permute(1, 2, 0))
print('Label:', train_data.classes[label], 'Predicted:', predict_image(img, model))
```

```
Label: APPOLLO Predicted: APPOLLO
```



謝謝觀看