Softmax回归的从零开始实现

Eng Teong Cheah

目录

Softmax回归的从零开始实现

Softmax回归的从零开始实现

Softmax回归的从零开始实现

```
import d2l
from mxnet import autograd, nd, gluon
from IPython import display
```

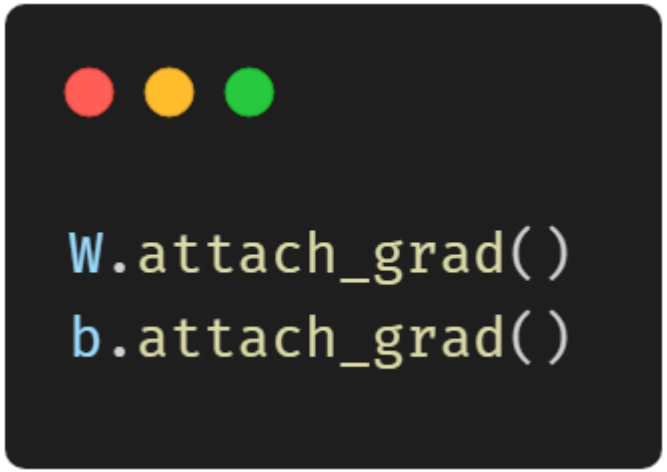
Softmax回归的从零开始实现

```
batch_size = 256
train_iter, test_iter = d2l.load_data_fashion_mnist(batch_size)
```

初始化模型参数

```
num_inputs = 784
num_outputs = 10

W = nd.random.normal(scale=0.01, shape=(num_inputs,
b = nd.zeros(num_outputs)
```

初始化模型参数



实现softmax运算

```
X = nd.array([[1, 2, 3], [4, 5, 6]])
X.sum(axis=0, keepdims=True), X.sum(axis=1,
keepdims=True)
```

实现softmax运算

```
def softmax(X):
    X_exp = X.exp()
    partition = X_exp.sum(axis=1, keepdims=True)
    return X_exp / partition  # The broadcast mechanism is applied here
```

实现softmax运算

```
X = nd.random.normal(shape=(2,
X_prob = softmax(X)
X_prob, X_prob.sum(axis=1)
```

定义模型

```
def net(X):
    return softmax(nd.dot(X.reshape((-1, num_inputs)), W) +
b)
```

定义损失函数

```
y_hat = nd.array([[0.1, 0.3, 0.6], [0.3, 0.2,
0.5]])
y = nd.array([0, 2], dtype='int32')
nd.pick(y_hat, y)
```

定义损失函数

```
def cross_entropy(y_hat, y):
    return - nd.pick(y_hat, y).log()
```

计算分类准确率

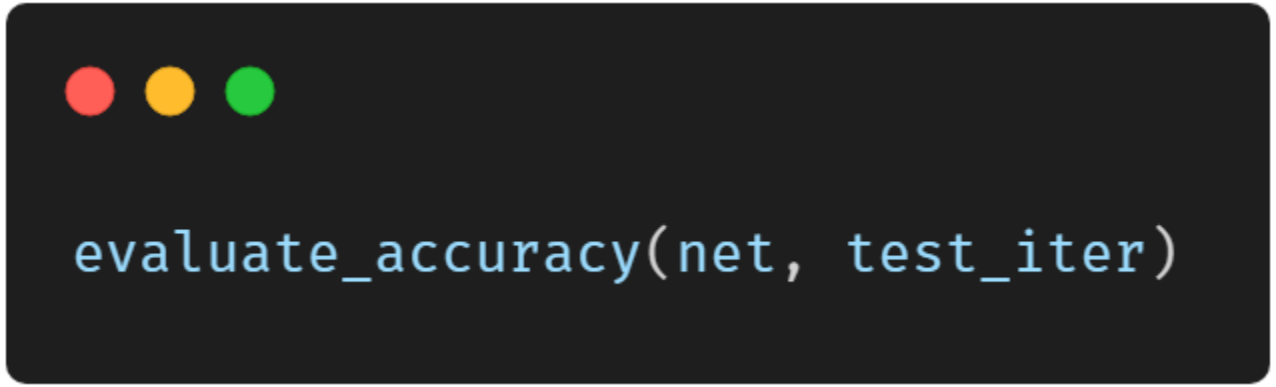
```
# Save to the d2l package.
def accuracy(y_hat, y):
    return (y_hat.argmax(axis=1) ==
y.astype('float32')).sum().asscalar()
```

```
accuracy(y_hat, y) / len(y)
```

计算分类准确率

```python
# Save to the d2l package.
def evaluate_accuracy(net, data_iter):
    metric = Accumulator(2) # num_corrected_examples, num_examples
    for X, y in data_iter:
        y = y.astype('float32')
        metric.add(accuracy(net(X), y), y.size)
    return metric[0] / metric[1]
```

计算分类准确率

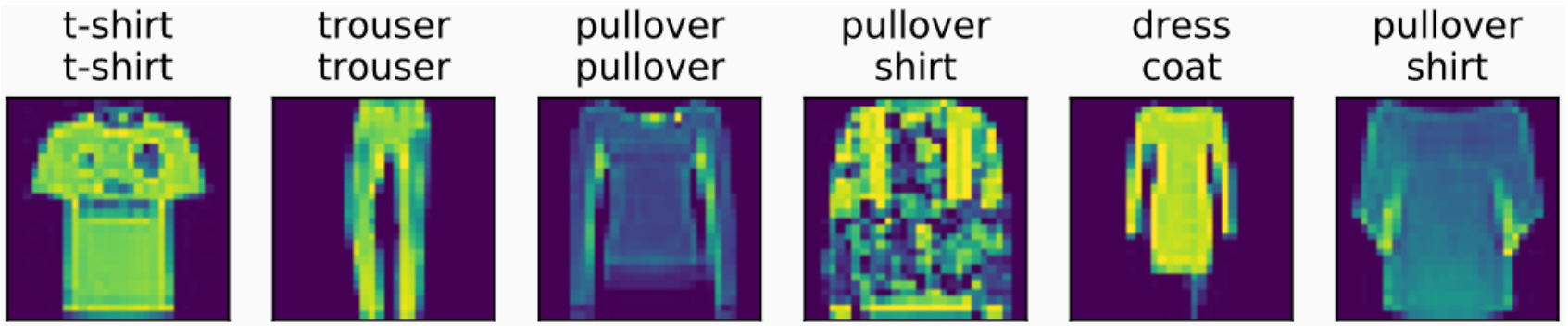


训练模型

```
num_epochs, lr = 5, 0.1

# 本函数已保存在d2lzh包中方便以后使用
def train_ch3(net, train_iter, test_iter, loss, num_epochs, batch_size,
              params=None, lr=None, trainer=None):
    for epoch in range(num_epochs):
        train_l_sum, train_acc_sum, n = 0.0, 0.0, 0
        for X, y in train_iter:
            with autograd.record():
                y_hat = net(X)
                l = loss(y_hat, y).sum()
            l.backward()
            if trainer is None:
                d2l.sgd(params, lr, batch_size)
            else:
                trainer.step(batch_size)  # “softmax回归的简洁实现”一节将用到
            y = y.astype('float32')
            train_l_sum += l.asscalar()
            train_acc_sum += (y_hat.argmax(axis=1) == y).sum().asscalar()
            n += y.size
        test_acc = evaluate_accuracy(test_iter, net)
        print('epoch %d, loss %.4f, train acc %.3f, test acc %.3f'
              % (epoch + 1, train_l_sum / n, train_acc_sum / n, test_acc))

train_ch3(net, train_iter, test_iter, cross_entropy, num_epochs, batch_size,
          [W, b], lr)
```

预测

```
# Save to the d2l package.
def predict_ch3(net, test_iter, n=6):
    for X, y in test_iter:
        break
    trues = d2l.get_fashion_mnist_labels(y.asnumpy())
    preds =
d2l.get_fashion_mnist_labels(net(X).argmax(axis=1).asnumpy())
    titles = [true +'\n' + pred for true, pred in zip(trues, preds)]
    d2l.show_images(X[0:n].reshape((n,28,28)), 1, n, titles=titles[0:n])

predict_ch3(net, test_iter)
```



谢谢!

Does anyone have any questions?

Twitter: @walkercet
Blog: https://ceteongvanness.wordpress.com

资源

Dive into Deep Learning