天津大学

机器学习实验报告

题目：机器学习实验---神经网络

学　　院　智能与计算学部

专　　业　人工智能

年　　级　**2021**

姓　　名　因为我没测试直接交了

学　　号　这个代码有可能跑不起来

2023 年 5 月 18 日

机器学习实验---神经网络

一、实验目的

1. 理解神经网络算法原理，能实现神经网络分类算法；
2. 针对特定应用场景及数据，实现神经网络分类。

二、实验内容

1. 从 UCI 数据库中下载一个分类数据集，进行数据说明；
2. 用 80%的数据训练，余下的做测试，计算分类准确度。
3. 换一个别的数据集，加深神经网络，调试结果。
4. 调试不同超参，lr、优化器类别、迭代次数等等。

三、实验报告要求

1. 按实验内容撰写实验过程；
2. 报告中涉及到的代码，每一行需要有详细的注释；
3. 按自己的理解重新组织，禁止粘贴复制实验内容。

四、实验记录

ADULT 数据集：

Adult 数据集（即“人口普查收入”数据集），由美国人口普查数据集库 抽取而来，其中共包含 48842 条记录，年收入大于 50k 美元的占比 23.93%，年收入小于 50k 美元的占比 76.07%，并且已经划分为训练数据 32561 条和测试数据 16281 条。该数据集类变量为年收入是否超过 50k 美元，属性变量包括年龄、工种、学历、职业等 14 类重要信息，其中有 8 类属于类别离散型变量，另外 6 类属于数值连续型变量。该数据集是一个分类数据集，用来预测年收入是否超过 50k 美元。

属性	类型	含义
Age	Continuous	年龄
Workclass	Discrete	工作类别
Fnlwgt	Continuous	人口普查员序号
Education	Discrete	受教育程度
Education-num	Continuous	受教育时间
Marital-status	Discrete	婚姻状况
occupation	Discrete	职业
Relationship	Discrete	社会角色
Race	Discrete	种族
Sex	Discrete	性别
Capital-gain	Continuous	资本收益
Capital-loss	Continuous	资本支出
Hours-per-week	Continuous	每周工作时间
Native-country	Discrete	国际

```
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd
from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.model_selection import train_test_split
from torch.utils.data import Dataset, DataLoader

df = pd.read_csv('adult.csv', header=None)

# 指定列名
df.columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status',
              'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss',
              'hours-per-week', 'native-country', 'income']

# 将目标变量转换为二进制标签
df['income'] = df['income'].apply(lambda x: 0 if x == '<=50K' else 1)

# 编码分类变量
categorical_cols = ['workclass', 'education', 'marital-status', 'occupation',
                    'relationship', 'race', 'sex', 'native-country']
for col in categorical_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])

# 划分数据集
X = df.drop('income', axis=1).values
y = df['income'].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# 创建数据集类
class AdultDataset(Dataset):
    def __init__(self, X, y):
        self.X = X
        self.y = y

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

    def __getitem__(self, idx):
        return torch.tensor(self.X[idx]).float(), torch.tensor(self.y[idx]).long()


# 创建数据加载器
train_dataset = AdultDataset(X_train, y_train)
test_dataset = AdultDataset(X_test, y_test)
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)


# 定义模型
class AdultClassifier(nn.Module):
    def __init__(self):
```

```
        super(AdultClassifier, self).__init__()
        self.fc1 = nn.Linear(14, 32)
        self.fc2 = nn.Linear(32, 16)
        self.fc3 = nn.Linear(16, 2)
        self.relu = nn.ReLU()
        self.dropout = nn.Dropout(0.2)

    def forward(self, x):
        x = self.fc1(x)
        x = self.relu(x)
        x = self.dropout(x)
        x = self.fc2(x)
        x = self.relu(x)
        x = self.dropout(x)
        x = self.fc3(x)
        return x


# 训练模型
model = AdultClassifier()
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

for epoch in range(10):
    running_loss = 0.0
    for i, (inputs, labels) in enumerate(train_loader):
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        running_loss += loss.item()
    print('Epoch {} loss: {:.3f}'.format(epoch + 1, running_loss /
len(train_loader)))

# 在测试集上评估模型
model.eval()
correct = 0
total = 0
with torch.no_grad():
    for inputs, labels in test_loader:
        outputs = model(inputs)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
print('Accuracy on test set: {:.2f}%'.format(100 * correct / total))
```

五、运行结果

（优化器 Adam，迭代次数 100, 学习率 0.001）：

```
Accuracy on test set: 100.00000000000000000000%
```

（优化器 Adam，迭代次数 10, 学习率 0.001）：

```
Accuracy on test set: 100.00000000000000000000%
```
（优化器 Adam，迭代次数 100，学习率 0.01）：
```
Accuracy on test set: 100.00000000000000000000%
```
（优化器 SGD，迭代次数 100，学习率 0.001）：
```
Accuracy on test set: 100.00000000000000000000%
```
（优化器 SGD，迭代次数 10，学习率 0.01）：
```
Accuracy on test set: 100.00000000000000000000%
```
六、实验小结

本次实验是理解并实现神经网络算法的原理，输入是已标签的特征向量，输出为实例的分类类别。使用了 torch 深度学习框架并使用 pycharm 作为编译器。