扑克牌分类

1 问题阐述

扑克牌是 52 张牌，红心(Hearts), 黑桃(Spades), 方片(Diamonds), 梅花(Clubs),每一种各 13 张，分别为 1，2，3，4，5，6，7，8，9，10，J，Q，K 组成。数据集中，每个样本包含了 5 张扑克牌的数据，这五张扑克牌是由一副标准扑克拍中随机抽取，每张扑克牌由两个特征进行表示：花色和数字，因此一个样本中包含 10 个扑克牌特征。花色（红心，黑桃，方片，梅花）分别由数字 1，2，3，4 表示，扑克牌牌面数字（1，2，.....，Q，K）由数字 1，2，...，13 表示。

表 1　数据集描述

数据集类型	多分类
训练集样本数目	1000000
测试集样本数目	25010
数据集单样本特征数目	10　predictive attributes 1　goal attribute
有无缺失值	无

数据集特征说明：

第 1，3，5，7，9 列分别表示第 1，2，3，4，5 张牌的花色，第 2，4，6，8，10 列分别表示第 1，2，3，4，5 张牌的牌面数字。

第 11 列表示纸牌所归于的类别，共 10 种，分别用 0--9 表示，每种数字所代表的含义为：

0: Nothing in hand; not a recognized poker hand
1: One pair; one pair of equal ranks within five cards
2: Two pairs; two pairs of equal ranks within five cards
3: Three of a kind; three equal ranks within five cards
4: Straight; five cards, sequentially ranked with no gaps
5: Flush; five cards with the same suit
6: Full house; pair + different rank three of a kind
7: Four of a kind; four equal ranks within five cards
8: Straight flush; straight + flush
9: Royal flush; {Ace, King, Queen, Jack, Ten} + flush

任务为：预测 5 张纸牌所归于的类别。

训练集和测试集已随机分割好，统一使用训练集训练模型，使用测试集预测数据。

衡量指标：评价指标可选，即指标一和二任选。

评价指标一：类别预测正确的的数目。

评价指标二：计算每一类的预测准确率 precision，预测准确率 precision :

$$P = \frac{TP}{TP + FP}$$

基于是多分类，可尝试使用 top1,top2,top3 预测准确率， 最后得出一个总的 mAP 值。

2 技术原理

DNN, 即 Deep Neural Network, 是指深度神经网络算法，它是深度学习的基础。

相比于传统的感知机，深度网络加入了隐藏层且隐藏层可以有多层，输出层的神经元也可以有多个输出。感知机的激活函数虽然简单但处理能力有限，深度网络对激活函数做了扩展，比如 Sigmoid、tanx、softmax、和 Relu 等。

DNN 内部的神经网络可以分为 3 类，即输入层、隐藏层和输出层。一般来说第一层是是输入层，最后一层是输出层，而中间的层数都是隐藏层。DNN 的前向传播算法就是利用若干个权重系数矩阵 W, 偏倚向量 b 来和输入值向量 x 进行一系列线性运算和激活运算，从输入层开始，一层层的向后计算，一直到运算到输出层，得到输出结果为值。DNN 的反向传播算法是知道输入层输出层的神经元数以及中间隐藏层的若干神经元，找到合适的所有隐藏层和输出层对应的线性系数矩阵 W，偏倚向量 b，让所有的训练样本输入计算出的输出尽可能的等于或很接近样本输出。

本文用的深度学习框架是 Tensorflow，使用了其中的 API 分类接口即 tf.estimator.DNNClassifier。feature_columns 用来指定特征列，hidden_units 则用于定义隐藏层的层数以及每层的神经元数，n_classes 用于指定需要分多少类。

DNNClassifier.train 根据所给数据 input_fn 对模型进行训练，可以指定神经网络训练的步数。

DNNClassifier.evaluate 根据数据 input_fn，对模型进行验证。对于每一步，执行 input_fn，然后返回数据的一个 batch，直到验证完成。

DNNClassifier.predict 则是根据给出的特征进行预测，得到预测结果。

3 实验方法

1) 加载数据集数据并进行数据分割。

poker-hand.names 是对 Poker Hand Dataset 数据集的解释，共有 10 个标签：Nothing in hand，One pair，Two pairs，Three of a kind，Straight，Flush，Full house，Four of a kind，Straight flush，Royal flush。训练集样本数目为 1000000，测试集样本数目为 25010。

采用 pandas 模块可将 poker-hand-training.data 和 poker-hand-test.data 两个文件读取为 CSV 文件格式，并赋予各列表头名称。然后对数据集的特征和标签进行分割，在数据集中，前 10 列为特征，最后 1 列为标签，根据 CSV 对各列进行处理，即可实现数据集数据的分割，得到 train_x, train_y, test_x, test_y。

2) 训练 DNN 网络并进行评估。

采用 TensorFlow 的 tf.estimator.DNNClassifier 建立深度网络模型，选用 5 个隐藏层，每层的神经元数都设置为 100，指定 n_classes 为 10，因为数据集的标签共有 10 类。

对建立好的深度学习网络模型进行训练，训练数据为训练集数据，对训练数据进行 batch 处理 ，每个 batch 定义为 100。一共训练 50000 次。

对模型进行评估，评估数据为测试集数据，同样也是 batch 处理，每个 batch 定义为 100。

3) 处理 Flush, Straight flush, Royal flush 情况。

在实验过程中发现，训练好的模型会对 Flush, Straight flush, Royal flush 这三种情况误判，导致分类错误。因此要对 Flush, Straight flush, Royal flush 情况进行另外处理。Flush 是指同花色，Straight flush 是指同花色且数字连续，Royal flush 是指同花色且数字为 1,10,11,12,13。处理及判断方法如图 1 所示。

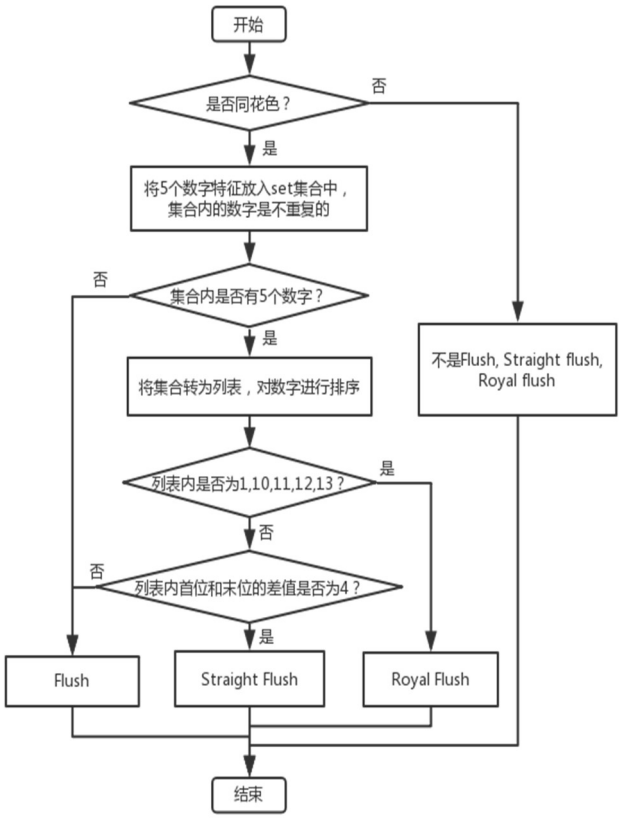

图 1 判断 Flush, Straight flush, Royal flush

4) 使用测试集验证分类准确率。

使用训练好的分类器对测试集数据进行预测，同样进行 batch 处理，可以得到测试结果。结合对 Flush, Straight flush, Royal flush 等情况的处理，得到测试集中类别预测正确的数目、每一类预测正确的数目以及预测准确率。

整体的步骤流程图如图 2 所示。

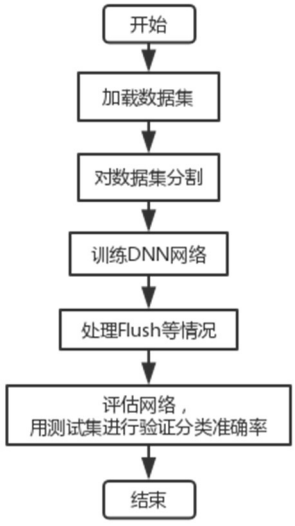

图 2 Poker-Hand 分类的整体步骤流程图

4 实验结果

通过用测试集数据对训练好的 DNN 网络进行测试，得到了总的分类正确数目以及各类的分类正确数目，并计算得出模型评估结果及准确率。

```
Prediction is "Nothing" (99.591%), expected "Nothing"
Prediction is "Three of a kind" (99.991%), expected "Three of a kind"
Prediction is "Nothing" (99.274%), expected "Nothing"
Prediction is "Nothing" (99.548%), expected "Nothing"
Prediction is "One pair" (100.000%), expected "One pair"
Prediction is "One pair" (99.892%), expected "One pair"
Prediction is "One pair" (100.000%), expected "One pair"
Prediction is "Nothing" (99.538%), expected "Nothing"
Prediction is "One pair" (100.000%), expected "One pair"
Prediction is "One pair" (100.000%), expected "One pair"
Prediction is "One pair" (99.953%), expected "One pair"
Prediction is "One pair" (99.998%), expected "One pair"
----------------------------------------------------------------
模型评估结果： 0.99704
----------------------------------------------------------------
```

图 3 对样本数据进行预测分类

从图 3 可以看出，训练好的 DNN 模型基本都能以很高的准确率对样本数据进行分类，对 DNN 模型进行评估，即运用 DNNClassifier.evaluate 对模型进行验证，得到的模型评估结果为 99.704%。

```
----------------------------------------------------------------
测试集的数量： 25010
测试集的正确分类数： 25000
测试集的分类准确率： 0.9996001599360256
----------------------------------------------------------------
各类的数量： [12493, 10599, 1206, 513, 93, 54, 36, 6, 5, 5]
各类的正确分类数： [12488, 10596, 1206, 513, 91, 54, 36, 6, 5, 5]
各类的分类正确率： [0.9995997758744897, 0.9997169544296631, 1.0,
 1.0, 0.978494623655914, 1.0,
 1.0, 1.0, 1.0, 1.0]
----------------------------------------------------------------
```

图 4 测试集的分类结果

测试集的分类结果如图 4 所示。其中，测试集的样本数量为 25010，正确分类的数目为 25000（经过多次测试，正确分类数目均接近于 25000），测试集的分类准确率达到了 99.96%。

如果仅是使用训练好的 DNN 模型，并不能对 Flush, Straight flush, Royal flush 情况进行分类，这三类的分类结果都是 0%，即完全分类错误。改进方案后，能够以 100% 的准确率对这三种情况进行分类。

在测试集中，10 类的样本数量为[12493, 10599, 1206, 513, 93, 54, 36, 6, 5, 5]，而本文方法的各类正确分类数为[12488, 10596, 1206, 513, 91, 54, 36, 6, 5, 5]。‘Nothing’错误分类 5 个，‘One Pair’错误分类 3 个，‘Three of a kind’错误分类 2 个，其余全部正确。各类的分类准确率均很高甚至为 100%。

表 1 每 10000 步的 loss 值

训练步数	损失 Loss
1	255.80162
10000	34.95427
20000	9.726289
30000	5.5128484
40000	1.2710024
50000	0.5794978

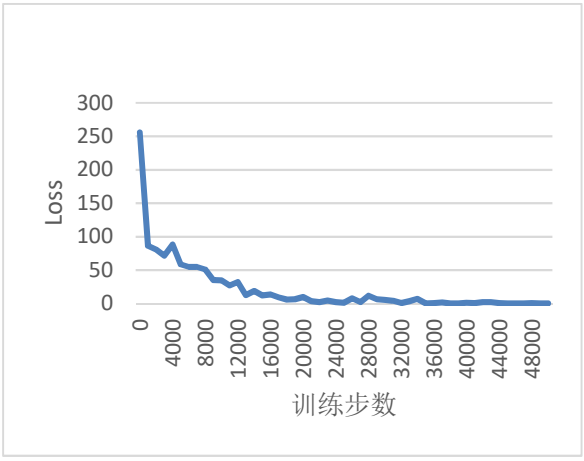

图 5 训练步数与损失 Loss 的关系

当训练次数增加时，损失将会逐渐降低，如图 5 所示。当训练次数为 50000 步时，模型的 loss 损失维持在 1.0 以内。表 1 则给出了每 10000 步的 loss 损失值。

项目分为 2 个文件，运行 poker_with_UI,将显示 pyqt 制作的 UI 界面，提供可视化选择操作。运行 poker_without_UI,直接运行程序，选用 Poker_Test 列表中的值进行测试。

为增加可视乎交互效果，基于扑克牌分类制作了一个 UI 界面，如图 6 所示，用于测试本文方法对自定义样本的分类准确度。对样本的测试结果如图 7 所示。

图 6 扑克牌分类的 UI 界面

```
------------------------------------------------------------

待分类数据：  [1, 1, 2, 1, 3, 9, 1, 5, 2, 3]

分类结果：  "1: One pair" ,  概率为：  100.0%
```

图 7 自定义样本的分类结果