电子科技大学 计算机 学院

标 准 实 验 报 告

（实验）课程名称 人工智能综合实验 I

学生姓名：刘洋岑　　学 号： **2020080601018**

指导教师：姬艳丽

实验地点：主楼 **A2-412**

实验时间：**2022.3.16**

电子科技大学教务处制表

电 子 科 技 大 学

实 验 报 告

一、实验项目名称：K 近邻法实现分类任务

二、实验内容：

实验一：编程实现 K 近邻分类器模型，在西瓜数据集 3.0α（表 4.5）上验证 K 近邻分类器。
（1）实现 K 不同取值时分类实验，分析实验结果。
（2）采用 Lp 距离、欧氏距离等不同距离计算验证 K 近邻分类结果，并分析讨论。
实验二：在 Iris 数据集上实现 KNN 分类。本实验的数据以 csv 文件的形式给出，已上传到指定文件夹。
（1）实现 K 不同取值时分类实验，分析实验结果。
（2）采用 Lp 距离、欧氏距离等不同距离计算验证 K 近邻分类结果，并分析讨论
实验三（补充实验）：对于工业上人脸识别的多类别、添加类别问题进行简单的实验。（“作者”： 刘洋岑（2020080601018） 谭一谈（2020080601022））
（1）使用 resnet50 作为 backbone 使用 triple net 对于人脸学习映射，并模拟添加类别（注册人脸）操作。
（2）使用 KNN 对于 triple net 提取高维特征进行分类。并分析实验结果。

三、实验算法设计：

实验一

由于西瓜数据集本身比较小，所以仅仅使用了 KNN 直接进行套用。大致的算法为：

1. 将训练数据的 X 与 y 进行保存

2. 对于每一个输入的数据 x，根据设定的 K 值以及规定的距离度量模式，与 X 中所有样本进行距离计算，并排序（从小到大），取前 K 个。

3. 在这前 K 个中找到类别最多的一个，根据先验假设使得使用出现类别最多的一类作为最终的预测分类。

实验二

方法与实验一几乎相同，然而对于数据首先使用了一个 PCA 降维，并且降维之后发现十分容易进行处理，则使用了 PCA 降维之后的二维特征进行实验。

实验三

与实验一，二不同，由于人脸数据是非常复杂的分布，仅仅使用一些简单的范数是不可能的，我们的思路是“学一种复杂高维的范数”。并且我们的任务考虑了人脸分类的加类别问题。

为什么不直接使用 CNN：CNN 异或是最新的一些深度学习方法在人脸分类问题已经取得了极好的效果。但是工业界有“添加类别问题”，即比如支付宝人脸扫描识别任务可能不停有新注册的用户，且类别数目可能达到亿的数量级，即使在公司内部耗费大量的资源来训练一个这样的模型，这样的 CNN 参数量巨大，不便于实际应用，且添加类别后，必须对于全体参数进行重新学习，或者最后几层参数进行 fine tune。这个任务更加倾向于 meta learning 中的思想：“学习如何学习一个新类别”。

我与合作者（谭一谈 2020080601022）查阅了资料，发现这个问题引入了一种高维映射思想。如果能够把人脸这种复杂的分布映射到一个高维空间，使得相同类别之间二范数接近，不同类别之间二范数距离较远，那么就可以学习到这种分类映射。对于新注册的类别，我们仅仅需要计算它经过这种映射得到的高维特征，在最后一层就可以使用一些传统的机器学习方法进行计算了，而不需要更新这个映射的全体参数。

我们在最后一个高维映射空间，使用一些机器学习方法，比如 KNN,SVM 即可。虽然工业界有更加复杂的处理方式，但是我们的实验简单的使用 KNN 作为最后一层的分类器。

四、算法及创新：

1.代码：代码由于写成了 repo 形式并且比较复杂，所以给出 github 链接：

https://github.com/Randle-Github/Machine-Learning-Experiment/tree/main/experiment2

其中 KNN 部分在 model/classifier_helper.py 为本次实验 KNN 核心，实现了一个 sklearn 的 Kneighborsclassifier 以及一个自己仿照手写的。

其中 config/config.yaml 控制全局参数。

2.个人创新：

1.创作形式：使用了标准的 repository 管理模式进行写作。使用 config.yaml 进行整体参数估计。并且使用了标准的深度学习框架 pytorch。

2.任务方法：引入 meta learning 的一些方法，类别拓展。在机器学习课程上，老师让我

们讨论了人脸识别技术。而我意识到：工业上的人脸识别技术不可能简单的使用 CNN 模型，由于数据集有以下特征：1)极大类别数，可能以亿为数量级。2)可能有新注册的类别，如果简单的使用卷积神经网络，无法进行所有权值的更新。即使仅仅只使用最后几层的 fine-tune，也无法进行如此进行大规模的参数更新。3)使用了 triple net 对于参数进行更新。4)最终的 model 仍然具有 end-to-end 形式。

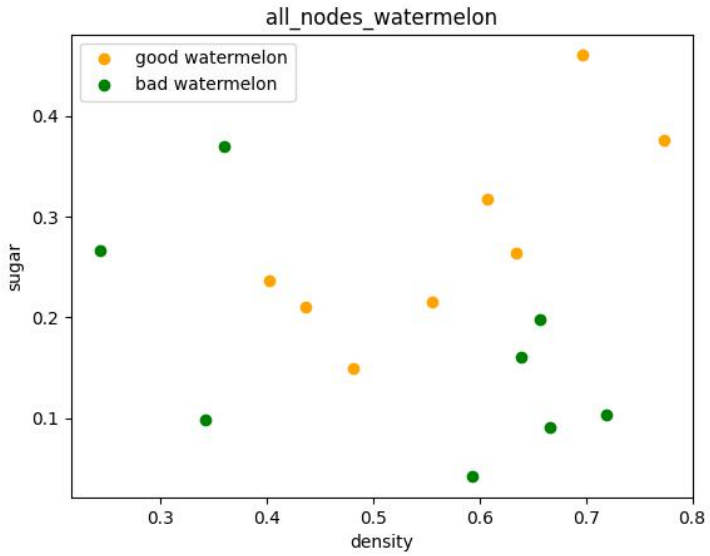
五、实验数据及结果分析：

由于代码量较大，所有代码存在 github 仓库中，链接如下：

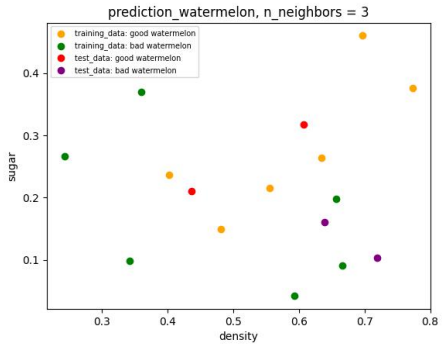
https://github.com/Randle-Github/Machine-Learning-Experiment/tree/main/experiment2

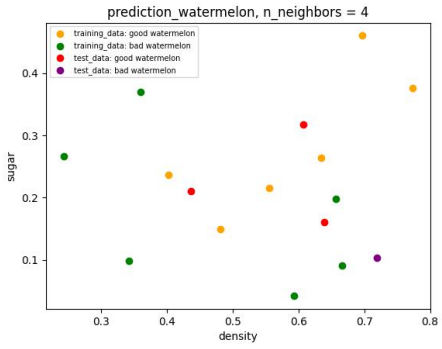
1. 西瓜数据集

使用了简单的 KNN 算法对于西瓜数据集进行分类。由于数据量太少，并且只有二维，首先进行可视化：



发现数据分布也很简单，所以直接使用了 3:1 的 train:test 比例。Metric 直接使用 L2 范数，调试 K 值为 1,2,3 均为 100%正确率。调 K 值为 4 出现一个分类错误的情况。



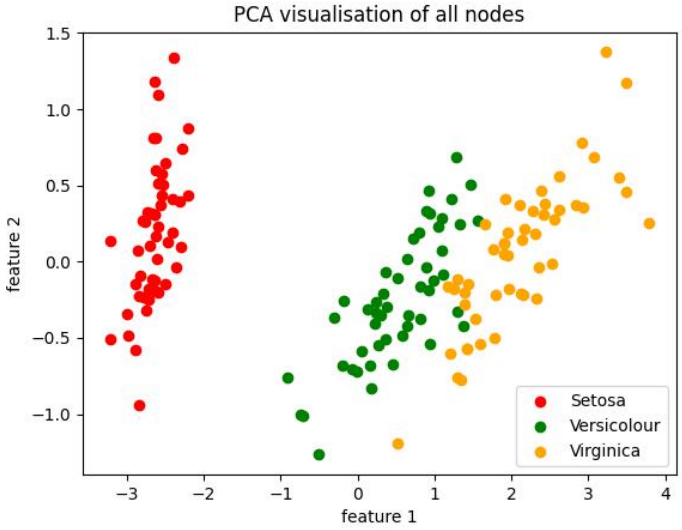


由于数据集比较简单，比较难以进行具体性质的研究。

2. Iris 数据集

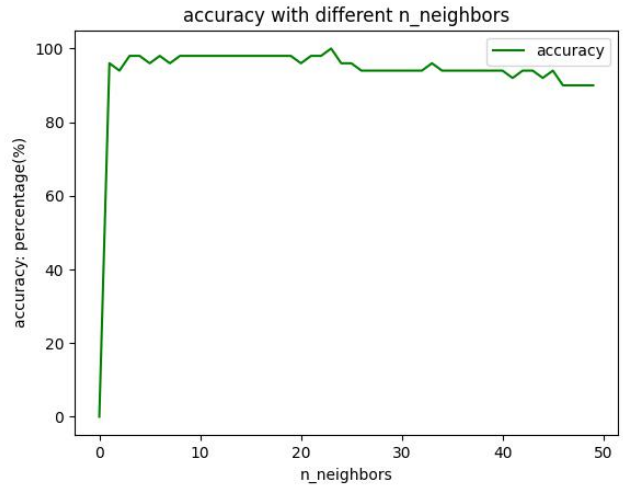
使用 sklearn 中自带的鸢尾花数据集（load_iris）进行简单的实验。其中进行了 KNeighborsClassifier 以及自己写的 KNN 对数据集分类正确率的实验。其中测试了不同 n_neigbors，metric 以及对于准确率的影响。其中包含了 150 条记录，每一行包括花萼长度、花萼宽度、花瓣长度和花瓣宽度四个特征，以及花的种类（包含 Setosa, Versicolour, Virginica 三类，数据已经处理为了 0,1,2 三个部分）。

在 config.yaml 中设置是否进行 PCA 降维可视化的选项。降维后得到的可视化效果：



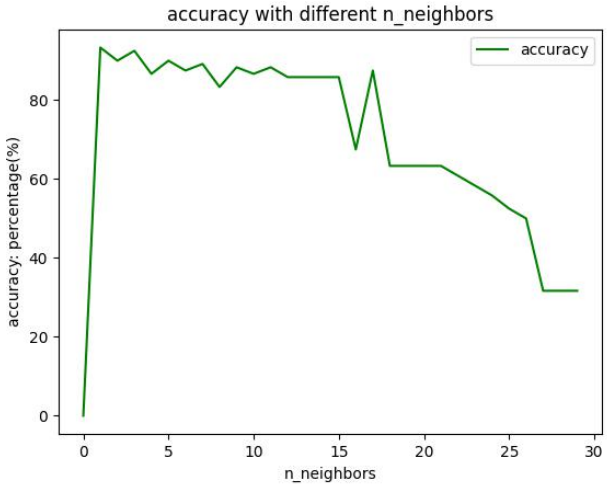
发现其实使用 2 维已经可以得到很好的实验效果了。

首先将训练集以及测试集以 13:2 的比例进行分割，并且使用 2 范数进行了实验。得到的结果为下图所示。



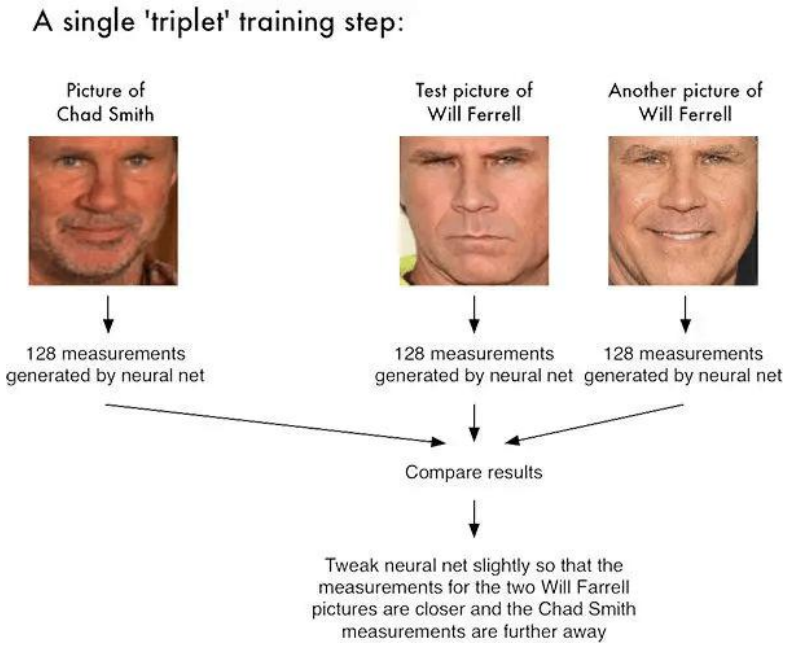
发现测试准确率太高了，将训练集与测试集 1:4 的比例进行分割，得到的结果如下图所

示：



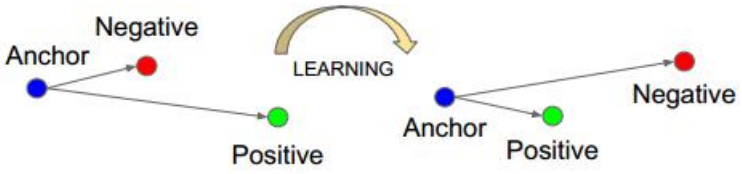
3. Face

此部分与谭一谈（2020080601022 合作完成）



使用了小型公开人脸数据集 CASIA-FaceV5（包含 500 个亚洲人，每个人 5 张脸）：

使用了 triplenet 进行损失函数的传递：



我们通过对每一张图像数据进行读取（使用 random crop 方法进行裁剪），将图像 RGB 的数据信息读取出来后，将图像的 RGB 矩阵双线性插值到 3x244x244 的三元组。我们的映射函数 metricnet 基于 resnet 的 backbone（其中 config 中设置了 resnet18,resnet32,resnet50,resnet101 的选项）。我们使用这个 backbone 提取出 256 维的高维特征。后续使用一个简单的线性映射得到 128 维特征。

对于这个 128 维特征，我们进行 triplenet 进行以下处理。设 triplenet 中的三元组分别为 Anchor(主样本)、Negative（负样本）、Positive(正样本)。其中 Positive 与 Anchor 是统一标签的样本，而 Negative 是与 Anchor 不是统一标签的样本。triplet loss 函数的基本思想如下所示（对于一个 batch 中 n 个样本）：

$$loss = \sum_{i}^{N}[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2]$$

我们的目的就是使 loss 在训练迭代中下降的越小越好，也就是要使得 Anchor 与 Positive 越接近越好，Anchor 与 Negative 越远越好。

经本方法训练的人脸识别模型，训练的是对于人脸的映射，因为在现实中处理问题的时候，在数据集样本非常大的时候，若对新输入的人脸，需要将整个模型重新训练一遍，代价过大。所以使用 triplnet 方法能够训练出好的映射模型。

关于这个实验我们的数据处理部分(dataloader,datasets)由于时间仓促还没有写完，后续会完善。仓库结构如下：

```
"https://github.com/Randle-Github/Machine-Learning-Experiment/tree/main/experiment2"
root\
    |--config\
    |   |--config.yaml 进行整个工作的设置
    |--datasets\
    |   |--iris.py  对于iris的数据处理
    |   |--watermelon.py 对于西瓜数据集的数据处理
    |   |--face.py 对于CASIA-FaceV5的数据处理（未完成）
    |--model\
    |   |--backward_helper.py 对于triplenet的参数训练
    |   |--classifier_helper.py KNN部分
    |   |--metricnet.py 人脸映射网络
    |   |--triplenet.py triplenet网络
    |   |--resnet_helper.py 包含了各种resnet模型
    |   |--visualise.py 可视化
    |--output\
    |   |--... 输出文件
    |--run.py 运行代码
```

Config.yaml 内部展示：

```yaml
DATASET:
  NAME: watermelon # watermelon iris face 三个数据集
  VISUALISE: False # 是否进行均一化

VISUALISE: False # 是否可视化
MODEL:
  n_neighbors: 3 # KNN的近邻数
  metric: Minkowski # KNN使用的距离度量
  p: 3 # 对于Minkowski的距离度量参数
  SELF_MODEL: True # 是否使用自定义KNN（或者使用sklearn）
  PCA: True # 是否PCA降维至二维，便于可视化
  NET: resnet34 # metric使用的backbone模型:resnet18,34,50,101
  MAPPING_DIM: 256 # 将人脸映射出的维度

STEP:
  epoches: [10, 10, 10] # 迭代轮数
  learning_rt: [0.1, 0.01, 0.001] # 迭代学习率
  method: SGD # 学习形式
  preserved: net_param # 数据存储位置

OUTPUT_DIR: output # 输出文件
ENV_PATH: D:\Work\Paper\machine learning\experiment2 # 环境路径
```

六、总结及心得体会：

体会最深刻的一点是：对于 KNeighborsClassifier，最重要的部分其实是 metric 方式。对于简单的数据集，仅仅使用他们的数据集本身的性质使用某种范数就可以达到几乎 100%的预测准确率，但是对于人脸这种高维特征，需要将数据分布映射到另外的一个空间进行操作。

KNN 虽然是一种较为早期的机器学习方法，并且效果受到了距离度量的影响，但是与深度学习结合，仍然具有自己的优势。其在可拓展性部分具有极大的优势，在后续 meta learning 的阶段，KNN 的思想十分简单但是不可忽视。

七、对本实验过程及方法、手段的改进建议：

改进建议暂无。

报告评分：

指导教师签字：