

**实 验 报** **告**

**课程名称：** **统计学习方法**



班级：智研211



姓名：王利猛



学号：2021540044



教师：虞欣



人工智能研

目录

[一、 实验 1](#_Toc92980255)

[1.编程环境设置 1](#_Toc92980256)

[2.程序设计要点 1](#_Toc92980257)

[3.结果合理性分析 6](#_Toc92980258)

[二、 解题思路与分析 7](#_Toc92980259)

[2.1摘要 7](#_Toc92980260)

[2.2模拟卷积网络进行图像识别 7](#_Toc92980261)

[2.2.1算法描述： 7](#_Toc92980262)

[2.2.2实验结果 9](#_Toc92980263)

[2.3二值图像识别算法 10](#_Toc92980264)

[2.4建立全连接神经网络 10](#_Toc92980265)

[2.4.1算法介绍： 10](#_Toc92980266)

[2.4.2实验结果 12](#_Toc92980267)

# 一、 实验

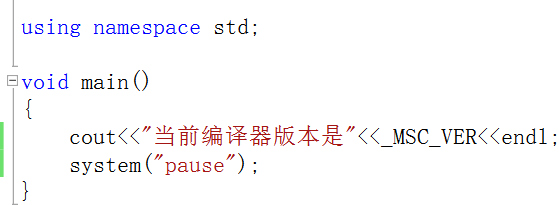
## 1.编程环境设置

代码编辑器：source insight 4

IDE工具：windows10 VC++2010

编译器版本（\_MSC\_VER）：1600

可用以下代码输出信息：





## 2.程序设计要点

程序代码在[最小距离法/mindistance.cpp](1、最小距离法/mindistance.cpp)

以下为本程序算法介绍：

一、计算训练样本（每个文件奇数行）均值与方差

S1、假设文本class1有四行：

1   0   109.230769230769    11338.8972043217

1   0   97.1496913580247    5461.40013405914

1   0   114.992723492724    142869.640907809

1   0   109.593883357041    108996.822108484

S2、只加载奇数行：

1   0   109.230769230769    11338.8972043217

1   0   114.992723492724    142869.640907809

S3、转换为向量cls有2个元素,：

(1  0   109.230769230769    11338.8972043217),

(1  0   114.992723492724    142869.6409078091)

S4、转换为向量clm有8个元素：

(1),

    (0),

    (109.230769230769),

    (11338.8972043217),

    (1),

    (0),

    (114.992723492724),

    (142869.640907809)

S5、对clm做步长为4的挑选，即选出每一列，形成一个block，共形成4个：

block1：

(1),

(1)

求得mean1，stdev1 push进result1中

block2：

(0),

(0)

求得mean2，stdev2 push进result1中

block3：

(109.230769230769),

(114.992723492724)

求得mean3，stdev3 push进result1中

block4：

(11338.8972043217);

(142869.6409078091)

求得mean4，stdev4 push进result1中

S6、输出result1，共4个元素

result1:

(mean1，stdev1),

(mean2，stdev2),

(mean3，stdev3),

(mean4，stdev4)

S7、重复以上S1-S6，循环处理其余三个文件，将class2-4奇数行的均值方差输出到result2-4

二、最小距离的计算

S1、只加载class1偶数行（测试数据）

1   0   97.1496913580247    5461.40013405914

1   0   109.593883357041    108996.822108484

S2、转换为向量cls有2个元素,：

(1    0   97.1496913580247    5461.40013405914),

(1    0   109.593883357041    108996.822108484)

S3、转换为向量clm有8个元素：

(1),

    (0),

    (97.1496913580247),

    (5461.40013405914),

    (1),

    (0),

    (109.593883357041),

    (108996.822108484)

S4、挑选，外层循环步长为4，内层循环步长为1，即选出每一行，内层从第二个元素开始（不要每一行第一列第二列，即期望值1与真实值0）,按照距离公式计算与result1-4的距离

S5、距离计算

以欧式距离为例：,其中为测试样本（每行）的每个元素，为该元素所对应列的均值。由于有四个样本中心，也就是有四组均值（result1-4[i].mean）,所以每个测试样本对应样本中心的距离应该有四组，分别是dst1-4。

第一行（对应class第二行）距离四个样本中心的欧氏距离：

dst1=

dst2=

dst3=

dst4=

第二行（对应class第四行）距离四个样本中心的欧氏距离：

dst1=

dst2=

dst3=

dst4=

此外，在程序中预设四种不同的距离函数计算规则，通过手动传参(输入1-4)的方式令程序计算对应距离dst1-4，

1：欧氏距离：

2：曼哈顿距离：

3：标准欧氏距离：其中为该元素所对应列的标准差，即==sqrt(result[i].stdev))

4:切比雪夫距离：

S6、对于每个测试样本，通过简单的搜索极小值算法找出dst1-4的最小距离。

假设对于

(1    0   97.1496913580247    5461.40013405914)

这行而言，dst2最小，则真实值real==2，由于此时在处理class1，期望值（该行第一列元素）expect==1，令混淆矩阵matrix[2-1]

[1-1]+1(matrix[1][0]位置+1)。非对角线元素+1，表示分类错误；

假设对于

(1    0   109.593883357041    108996.822108484)

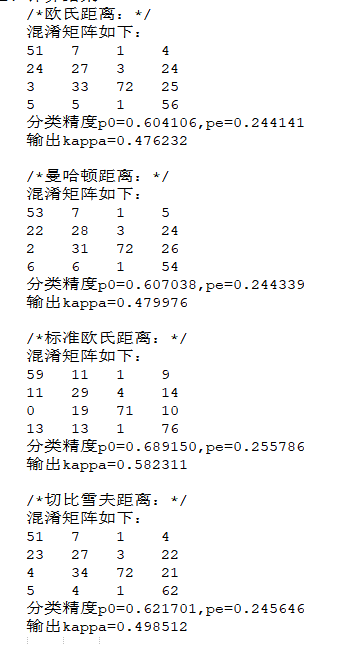
这行而言，dst1最小，则真实值为1，由于此时在处理class1，期望值为1，混淆矩阵matrix[1-1][1-1]+1(matrix[0][0]位置+1)，对角线元素+1，表示分类正确。

S7、根据matrix计算分类精度p0，pe，Kappa系数

S8、循环S1-S7，求出四种距离分类规则下的混淆矩阵matrix与Kappa

## 3.结果合理性分析

以下四种距离下输出的混淆矩阵及Kappa系数如下图（完整版输出结果见附件：最小距离法）：完整版结果在：[最小距离法/输出结果](1、最小距离法)



Kappa系数通常落在 0-1 间，有以下五种一致性级别可供参考：

0.0-0.20极低的一致性(slight)

0.21-0.40一般的一致性(fair)

0.41-0.60 中等的一致性(moderate)

0.61-0.80 高度的一致性(substantial)

0.81-1几乎完全一致(almost perfect)

本问题中，混淆矩阵的产生基于距离函数的选择，Kappa系数基于混淆矩阵。尝试使用多种距离规则对测试样本进行分类，得到的分类结果由混淆矩阵进行表示自然也不尽相同。由上图可见，用标准欧氏距离进行分类，分类精度最高，Kappa系数最高，分别达到了0.689150和0.582311。说明这四种距离函数中标准欧氏距离分类效果最好。

此外，除了以上四种距离外，这些距离（<https://www.cnblogs.com/AlvinSui/p/8931074.html>）也可供参考。本程序的不足之处在于，目前只支持四种距离。老师上课所讲到的闵可夫斯基距离和马氏距离，在运用C++编程过程中遇到了瓶颈。前者不会实现p次开方，后者不会实现矩阵求逆运算（需要引进Eigen库）(其实由数值分析上的方法，可以用LU分解系数矩阵，求n组关于自然向量的非齐次线性方程组的解，解的集合是系数矩阵的逆矩阵n是协方差矩阵维数)。我会在日后尝试用python编码进行补充实现。

# 二、 解题思路与分析

## 2.1摘要

这道题我尝试了三种解决方案。但是都没有取得很理想的结果。

第一种方法是模仿卷积网络处理像素点阵的方式。用一个8\*8的蒙版对photo矩阵进行自左而右，自上而下的点积运算。筛选出8\*56的特征矩阵，再对它进行特征提取，对特征解码出对应的图像矩阵。该方法的瓶颈在于最后一步，没有设计出解码算法。

第二种方法是受到二值图像的启发，可以把这道题看做二值图像识别问题。但是由于噪音的影响，直接识别很难。所以我查了一些关于聚类去噪算法，图像分割方面的论文。并且在Stack Overflow发帖询问本问题相关解法。但是答复好像并不合适。

第三种方法很简单。就是将2000个被噪声破坏的photo看做(2000,16,64)的三维张量作为神经网络的data，原二值化图像对应的随机数看做label。训练全连接神经网络，也就是把这道题看做是回归预测问题。这种方法好像比较合理，但是我做完以后predict的结果很差。模型还需要继续优化。

## 2.2模拟卷积网络进行图像识别

### 2.2.1算法描述：

为方便说明，先提供以下几项定义：

Photo：原始不带有噪声的二值化图像，为16\*64矩阵

Noisy\_photo:带有噪音的二值化图像，为16\*64矩阵

Fliter[i](i=0,1,2,…,9):二值化图形i，代表数字i，为8\*8方阵

Feature[i] (i=0,1,2,…,9):用fliter[i]对Noisy\_photo进行滑动点积运算后的特征矩阵

卷积神经网络处理图像的大致思想就是：针对图像的每一层，估计一个蒙版的全部参数（一般是3\*3矩阵），用该蒙版对原图像进行滑动点积运算后得到新的特征矩阵。不同的蒙版所筛选出的特征矩阵也不相同。不同的特征矩阵代表了图像的不同特征。

受到启发，可以把代表0-9这10个数字的二值化图片Fliter[i]作为蒙版，对Noisy\_photo进行滑动点积筛选得到Feature[i]。评估Feature[i]并进行相应解码运算，即找出四个可能性最大的数字后进行组合。

算法设计如下：

Step1：用Fliter[i]对Noisy\_photo进行滑动点积运算，得到Feature[i]。(i=0,1,2,…,9)

Step2：设定阈值，评估该数字出现的可能性。

比如，Fliter[0]有7个点，如果Photo中有这个单位，那么Feature[0]应有8\*8\*7=448这个元素。考虑到噪声的影响，这里我将阈值设定为 8\*8\*6=384，也就是说，Feature[0]中有384,448二者其中之一，认为Photo中有Fliter[0]。对每个Feature[i]都设定相应的阈值[阈值= (Fliter[i]内8的个数-1)\*8\*8]。挑选出有效的Feature[i]，如果Feature[i]中没有大于设定阈值的元素，Feature[i]视为无效。

Step3：获取四个有效Feature[i]，并获取他们大于阈值的元素所在坐标，根据坐标位置确定Fliter[i]空间位置，也即为i在四位数中的位数所在。

比如，获取四个有效的Feature分别为：Feature[1]，Feature[2]，Feature[3]，Feature[4]。它们的阈值分别为(8\*8\*6=384)(8\*8\*10=640)

(8\*8\*13=834)(8\*8\*6=384)，四个超出阈值的元素分别为：448,640，896,384，在Feature[i]中坐标分别为(1,15)(7,0)(2,54)(6,32)，只需要把坐标第二个数字，即列数从小到大排列，就能判断出该图像代表的数字为2143

### 2.2.2实验结果

实际操作过程远没有理想情况那么美好。由于噪声的影响，衍生了很多问题：有效的Feature[i]不止四个；Feature[i]中大于阈值的元素不唯一；如果产生的随机数中有重复的数字（如1122,3333）怎么办。我试着写了个简单的降噪算法，希望减少噪声的影响，但是效果还是不乐观。这部分程序在：[二值化图像识别/课程作业1222.cpp](2、二值化图像识别/课程作业1222.cpp)，输出结果在：[二值化图像识别/C运行结果.txt](2、二值化图像识别/C运行结果.txt)中。实际上降噪算法并没有取得很好的效果，相反，甚至还破坏了原图。这对后续的识别任务而言无疑是雪上加霜。

## 2.3二值图像识别算法

这部分我并没有写代码。查了一些二值化图像识别方面的论文。如：二值化图像聚类的文字提取算法<https://www.doc88.com/p-20096400373.html>

二值图象分割的快速聚类算法<https://max.book118.com/html/2015/1109/28972326.shtm>

都是在讲如何降噪。理论上而言，实现完美降噪的可能性很小。只有实现了，用方法一的识别算法才可能成功。而且这种聚类算法用C++很难实现。所以我试着去Stack Overflow论坛进行相关咨询。这是当时提出的链接：<https://stackoverflow.com/questions/70562893/binary-image-with-noise-recognition-problem>

有人提议用OpenCV的模型匹配方法<https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html>。其实由于噪声的存在，仅仅使用OpenCV还是难以实现。（毕竟我用眼睛都看不出来原来是什么图形）。

## 2.4建立全连接神经网络

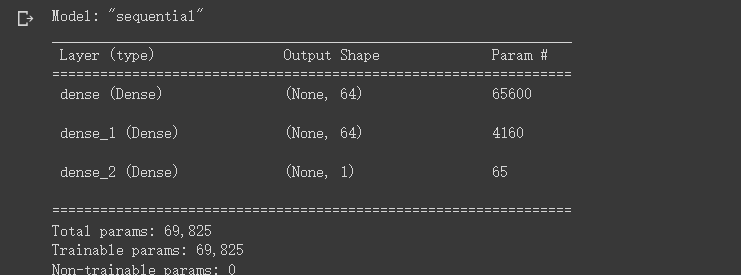
### 2.4.1算法介绍：

如果要建立数字矩阵与数字的关系，并使得输出结果为概率事件（做不到百分表准确性的预测），那么最好的方式大概就是用神经网络。可以先产生一部分数据进行网络训练，再产生2000组数据进行预测，比对准确性。所以我使用keras写了一个ipynb的脚本（位置在：[二值化图像识别/ FullyConectedNetworkTest.ipynb](2、二值化图像识别/FullyConectedNetworkTest.ipynb)）。如果电脑中有jupyter可以直接运行，没有的话用colab也可以运行。具体做法：打开[https://colab.research.google.com/notebooks/basic\_features\_overview.ipynb#scrollTo=5OCYEvK5QSHf /](https://colab.research.google.com/notebooks/basic_features_overview.ipynb#scrollTo=5OCYEvK5QSHf后点击左上角文件/打开笔记本/上传/)后点击左上角文件/打开笔记本/上传/选择文件

算法设计如下：

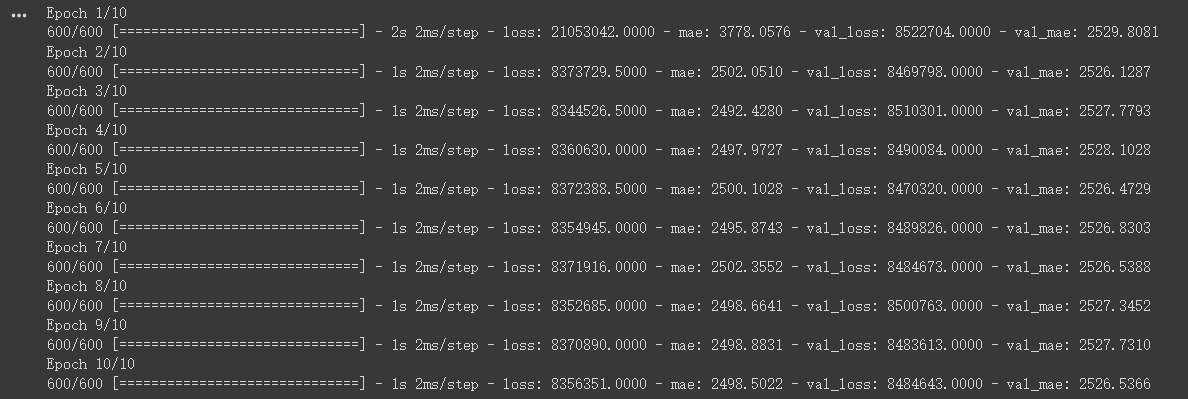
Step1:用Python重新实现C语言代码功能（产生两个Noisy\_photo见[二值化图像识别/ Python运行结果.txt](2、二值化图像识别/Python运行结果.txt)）

Step2：生成2000个随机数random\_num和对应的Noisy\_photo，视作data为(2000,16,64)的3D tensor，label是(1,2000)的1D tensor。用以下全连接网络进行训练。其中训练集：验证集：测试集的比例为6:2:2



Step3：进行400个测试集的预测评估.

### 2.4.2实验结果



实验结果很差，可以说是没有相关性。MSE与MAE大到惊人。自然预测结果也是不准确。数据高得离谱。而且预测的400个数字全是5107.9404。



很明显，程序出了bug，导致在训练的时候就出现了不正常的loss。相关问题还在排查中。或许，有个猜想，随机噪音的机制，已经将Noisy\_photo与radom\_num之间的相关性稀释得微乎其微。也就是说，神经网络并不适合解决这种问题。

总之，解决这个问题需要更好的机器学习算法，希望我能在未来找到更好的解决办法。

Written by 王利猛

Jan 13,2022