

学 号 2020302181189

# 武汉大学本科实验报告

## 人工智能课程设计： 一种基于 KNN 的手写数字识别模型

院(系)名称： 国家网络安全学院

专业名称： 信息安全

学生姓名： 李勃衡

指导教师： 汪润 副研究员

二〇二一年十二月

# UNDERGRADUATE EXPERIMENT REPORT OF WUHAN UNIVERSITY

A handwritten digit recognition model based on  
KNN

School (Department): SCHOOL OF CYBER SCIENCE AND ENGINEERING

Major: INFORMATION SECURITY

Candidate: LI BOHENG

Supervisor: A.P. WANG RUN



WUHAN UNIVERSITY

December, 2021

# 郑 重 声 明

本人呈交的课程设计实验报告, 是在导师的指导下, 独立进行课程  
设计工作所取得的成果, 所有数据、图片资料真实可靠. 尽我所知, 除  
文中已经注明引用的内容外, 本课程设计实验报告不包含他人享有著  
作权的内容. 对本课程设计实验报告所涉及的研究工作做出贡献的其  
他个人和集体, 均已在文中以明确的方式标明. 本课程设计实验报告的  
知识产权归属于培养单位.

本人签名: \_\_\_\_\_

日期: \_\_\_\_\_

## 摘 要

本文主要讨论了一种基于 K 最邻近 (KNN, K-NearestNeighbor) 分类算法的手写数字识别模型. 本文简要介绍了 KNN 分类算法的基本原理与在 MNIST 手写数字识别数据集上的应用, 并探讨了多种距离评估方法的应用表现和超参数的选取方法, 并通过实验选取了合适的超参数并施以验证, 得到了令人满意的结果. 最后介绍了 KNN 算法的弊端、总结了实验的不足之处并讨论了可能的改进措施.

关键词: KNN; 手写数字识别; 分类算法

# ABSTRACT

This article mainly discusses a handwritten digit recognition model based on the K-Nearest Neighbor classification algorithm. It briefly introduces the basic principles of the KNN classification algorithm and its application on the MNIST handwritten digit recognition data set. It also discusses the application performance of various optimization methods and the selection method of hyperparameters, introduces the drawbacks of the KNN algorithm, and discusses some possible improvement measures.

**Key words:** KNN; handwritten digit recognition; classification algorithms

# 目 录

摘要	III
ABSTRACT	IV
<b>1 引言</b>	<b>1</b>
1.1 手写数字识别简介 . . . . .	1
1.2 KNN 分类算法简介 . . . . .	2
1.3 实验环境与配置 . . . . .	3
<b>2 实验数据选取与预处理</b>	<b>4</b>
2.1 实验数据的获取 . . . . .	4
2.2 实验数据预处理 . . . . .	4
2.2.1 向量维度变换 . . . . .	5
2.2.2 归一化 . . . . .	5
<b>3 计算待分类样本和已分类集合样本间的距离</b>	<b>6</b>
3.1 L0 距离 . . . . .	6
3.2 L1 距离 . . . . .	7
3.3 L2 距离 . . . . .	9
<b>4 K 值的选取</b>	<b>10</b>
<b>5 决策方式</b>	<b>12</b>
5.1 多数表决法 . . . . .	12
5.2 加权平均法 . . . . .	12
5.3 选取合适的决策算法 . . . . .	13

<b>6</b>	<b>实验结果验证</b>	<b>15</b>
<b>7</b>	<b>实验弊端与改进方法分析</b>	<b>16</b>
7.1	KNN 算法优劣性分析 . . . . .	16
7.2	实验过程中的缺陷 . . . . .	18
7.2.1	在测试集上验证 . . . . .	18
7.2.2	没有交叉验证 . . . . .	18
	<b>参考文献</b>	<b>20</b>

# 1 引言

## 1.1 手写数字识别简介

手写数字识别是计算机视觉领域一个重要的问题，目前仍有大量手写数字需要录入计算机方便管理，如财务账单、手写支票与快递单等。如果能通过计算机算法自动识别，将节约很多人力物力。手写数字识别属于人工智能应用中的“分类”(classification)问题，设计者需接收输入的图片，然后经过处理后输出预测的标签。

$$f(\text{3}) = 3$$

图 1.1 手写数字识别问题的数学模型

如图1.1所示，设计者需要设计一个分类函数  $f(x)$ ，预测并输出图片的目标标签。从原理上，分类算法有基于统计特征分类、基于结构特征分类和基于神经网络分类三种。本文介绍的 KNN 算法是基于统计特征分类的人工智能算法。

手写数字数据集有许多，本实验选取的 MNIST 数据集是一个庞大的入门级手写数字数据集，其是由 Google 实验室、纽约大学柯朗研究所及微软研究院整理的不同年龄段人的手写数字构成的集合，它包含 60,000 张训练数据集图片和 10,000 张测试集图片，每张图片都是 0-9 之间的一个已知的手写数字，并都被处理为如图1.2所示的大小为 28\*28 的灰度图片。相较于 USPS、UCI 等手写数字识别数据集，MNIST 数据集具有数据规模大、笔迹多样等优点，数据规模和特征都较为符合本实验的要求。



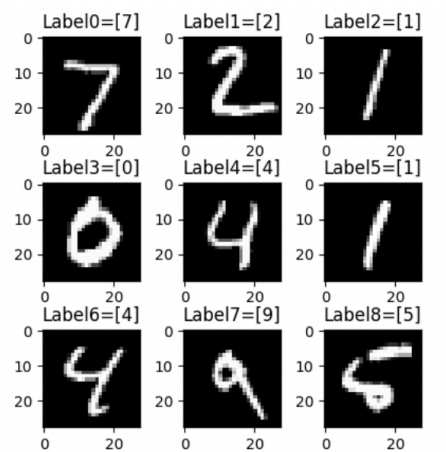


图 1.2 MNIST 数据集的一些示例图片

## 1.2 KNN 分类算法简介

KNN 分类算法，全称 K 最近邻分类算法，是一种基于统计特征分类的基本人工智能分类与回归算法。它通过选取分类集合中与待分类样本最为相似的 K 个样本值，然后通过决策预测待分类样本的类别。

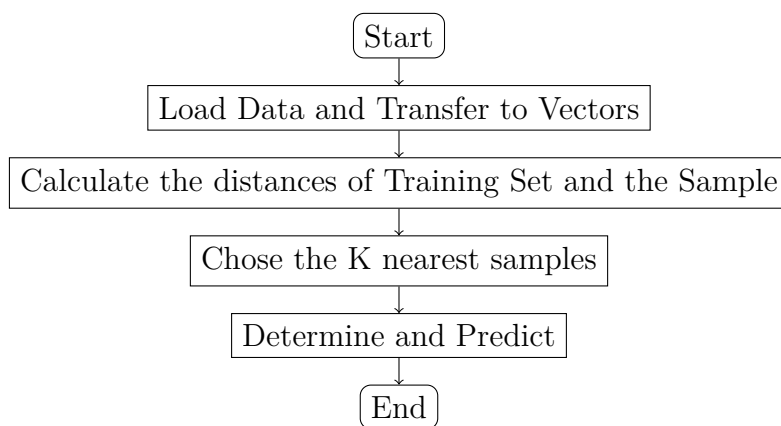


图 1.3 KNN 算法流程图

容易发现，在 KNN 算法中最为重要的三个要素是计算样本间距离的方法、K 值的选取以及决策方法。样本间距离的计算方法决定了算法寻找“最为相似”的标准，越优秀的计算方法找到的临近样本和待预测样本相似度越高，标签和待预测样本重合的可能性越大；K 值的选取需要综合考虑，太小的 K 值将造成模型

识别的精确率和容错率降低，太大的  $K$  值不仅影响算法效率，还增大了初始样本数这一无关因素对识别准确率的影响（特别地，当  $K$  等于训练集数据个数相等时，识别的结果仅与初始训练集中各个标签的数据的个数占比相关，而与数据相似度无关）；决策方法决定了从  $K$  个最近样本中选出最终的预测结果的方法，越优秀的决策方法识别准确率越高。因此，后续笔者将从这三点出发，通过实际实验数据改进 KNN 算法，并最终得到符合预期的结果。

### 1.3 实验环境与配置

本实验完全在 Google Colab 云端环境完成。Google Colaboratory 是一个免费的 Jupyter 笔记本环境，不需要进行任何设置就可以使用，并且完全在云端运行。使用 Jupyter Notebook 的好处在于可以在类 markdown 富文本环境中实时执行 python 代码，并保留代码块的运行结果。附件中有导出的 .ipynb Jupyter Notebook 文件和可以运行的 .py Python 代码，同时笔者提供一个经过测试可以运行的本地环境：

---

Operation System	Ubuntu 20.04
Python Version	Python 3.8.12 64-bit
Dependency Packages	tensorflow, numpy, pandas, matplotlib

---

值得提及的是，虽然依赖中包含 tensorflow 库和 pandas 库，但在实际应用中其仅用于下载并导入数据集、数据编码等工作，核心的 KNN 代码是完全依靠 numpy 库自己实现的。本实验报告中的部分图片是用 matplotlib 库绘制的。

## 2 实验数据选取与预处理

### 2.1 实验数据的获取

直接调用

```
(train_images, train_labels), (test_images, test_labels) =  
tf.keras.datasets.mnist.load_data()
```

函数即可加载 MNIST 数据集的训练集和测试集。

### 2.2 实验数据预处理

需要注意的是，对于 MNIST 数据集，虽然人眼看到的是  $28 \times 28$  的灰度图片，但图片二进制文件在计算机中实际是以一个  $28 \times 28$  的向量存储的。其中二维矩阵中的每个元素为一个  $[0, 255]$  的数值，表示该像素点的灰度值。图2.1展示了 MNIST 数据集中的一张灰度图片以及其向量。



图 2.1 MNIST 数据集中的一张灰度图片和它对应的向量

### 2.2.1 向量维度变换

加载到数据集中的数据是 60,000 个 28\*28 的训练集矩阵和 10,000 个 28\*28 的测试集矩阵，故当前变量中放置的是一个 60,000(10,000)\*28\*28 的三维矩阵。为了方便后续计算距离，我们将矩阵做向量维度变换，将每个二维图片矩阵变换为一维。调用 numpy 数组的 reshape 函数即可。

```
train_images = train_images.reshape(60000, 28 * 28)
test_images = test_images.reshape(10000, 28 * 28)
```

如此做以后，数据集被变换为二维矩阵，每一行为一个 28\*28(784) 大小的一维向量。

### 2.2.2 归一化

对于一张图片中的每一个像素，其数字表示均为一个 [0, 255] 的像素灰度值，而灰度值的数字大小并不能直观体现出两个图片的差距，因此我们对每个像素的灰度值进行归一化处理，即是将每个像素都按比例离散化为一个 [0, 1] 之间的数值。只需要对整个矩阵除以常数 255 即可。

### 3 计算待分类样本和已分类集合样本间的距离

计算待分类样本和分类集合样本间的距离是 KNN 算法中较为重要的一环，在这一步骤中它要求我们选取一距离评估函数  $dis(x)$ ，并合理估计两样本之间的距离（即两样本之间的差异值）。容易发现，距离计算函数越合理，越能正确评估两样本的差异大小，从而选取出最为相似近邻的已分类样本。因此，正确选取距离评估函数能大大提高 KNN 算法分类的准确率。目前常见的评估方法有 L0 距离、L1 距离、L2 距离三种。下面本文将一一介绍它们并通过实验选取最为合适的距离评估算法。

#### 3.1 L0 距离

L0 距离的评估方式非常简单：它认为，两张相似的数字图片应当在相同的像素点有相同的灰度值，而在不同的位置有不同的灰度值。因此只需要将两张图片对应的一维向量相减，然后统计结果中 0 的个数即可。容易发现，结果中 0 的个数越多，代表两张图片相同的灰度值越多，在 L0 距离的评估方式下两张图片即越为相似。因此如公式(3.1)所示，统计向量差中非 0 元素的个数作为 L0 距离的值。容易发现 L0 距离越小，两图片差异越小。

$$d_0(I_1, I_2) = \#(p | I_1^p - I_2^p \neq 0) \quad (3.1)$$

图3.1展示了一张手写数字 3 和其他 0-9 的手写数字图片的距离，图3.2展示了数据集中和该手写数字 3 最相似的前 5 张图片。

可以看出，在数据集较为庞大时，使用 L0 距离能筛选出和待分类样本较为相似的样本。但很明显 L0 距离只考虑相同像素的做法容错率极低，也极易构造对抗样本：如图3.3所示，考虑一张图片，只需要将这张图片的每个像素灰度值调整一个单位，得到的图片在视觉上仍然是相同的手写数字，但由于新生成图像的每个

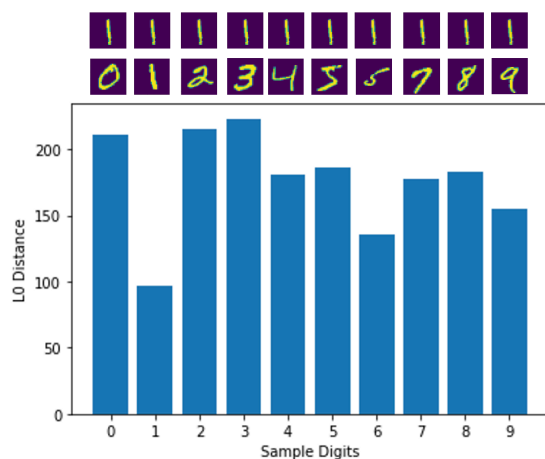


图 3.1 手写数字 1 与其他数字样本的 L0 距离

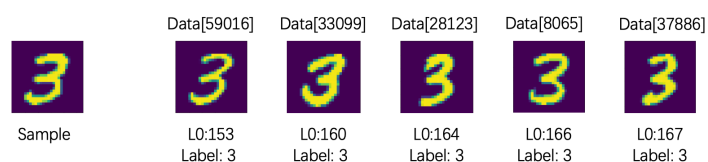


图 3.2 训练集中与样本手写数字 3 中 L0 距离最近的前 5 个样本

像素都和原图像不同，L0 距离的计算方法将得到此值的上限（即像素点总个数），并认为这两张图片的相似度为 0。

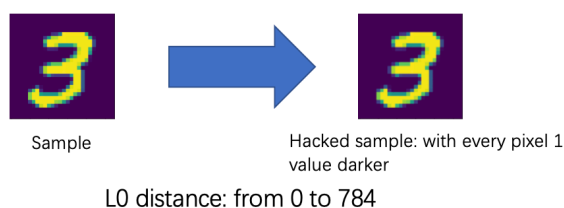


图 3.3 构造在人眼视觉上几乎没有差异的对抗样本

## 3.2 L1 距离

L1 距离又称曼哈顿距离 (Manhattan Distance)，与 L0 距离只考虑相同像素点的做法不同，L1 距离将两个像素点的相似程度也考虑在内，它求得两张图片每

个像素点灰度的相对差值，并将差值的和作为两张图片的距离，如公式3.2所示。

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p| \quad (3.2)$$

L1 距离能较好地比较两个图像相同像素点之间的灰度差异值，同时可以避免 L0 距离因为一些轻微差异就导致图片识别结果天差地别的问题。例如图3.3中的对抗样本在 L1 距离下只会造成一个像素点的距离误差，对整体判断结果几乎没有影响。

经过实验，如果使用 L1 距离最近的图片的标签作为预测的结果，在训练集前 50 张、前 500 张、前 5,000 张图片中，预测准确率分别为 96.0%, 94.8% 和 92.0%。总体优于 L0 距离的评估表现。

**表 3.1 选取距离最近图片的标签作为预测结果时的准确率**

Distance Metrics	50 images	500 images	5,000 images
Chose randomly	14.0%	7.9%	8.7%
L0 distance	88.0%	78.8%	68.5%
L1 distance	96.0%	94.8%	92.8%

但 L1 距离也并非无懈可击——它同样存在预测原理的不足之处。由于只考虑了相同像素点之间的差异，轻微的平移、旋转或者缩放就能使在视觉上差异不大的两张图片拥有差异较大的 L1 距离。同时，也可以轻易地构造对抗样本——选取一张纯白色的图片，然后上面染黑若干和待测试图片相同位置的像素点，就能使这张构造出来的图片和原图片拥有极小的 L1 距离，事实上，如果将这个样本加入训练集，它将从众多数字中脱颖而出，成为和原图 L1 距离最为接近的图片——然而事实上，它甚至没有一个合理的形状，只是一堆零散的点的组合。

图3.4展示了实验中部分预测错误的样本以及它们和最接近样本、实际标签相同的样本以及对抗样本之间的 L1 距离。可以看到，L1 距离只考虑相同位置像素点而忽略整体形状的做法存在不足之处。





Sample type	Mispredicted sample	Nearest image	Same label image	Adversarial sample
Image				
Label	2	1	2	X
L1 distance	0	54.95	114.50	14.70
Performance	-	✓	×	✓✓

图 3.4 被错误分类的样本及其与其他样本以及对抗样本的 L1 距离

### 3.3 L2 距离

L2 距离又称欧几里得距离 (Euclidean Distance)，它使用两个向量在 N 维空间的直线距离代表两个向量的距离，如公式3.3所示。

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2} \quad (3.3)$$

选取 L2 距离最近图片的标签作为预测结果，分别在 50、500、5,000 张测试集图片上做预测，得到的准确率结果如下表所示：

表 3.2 选取距离最近图片的标签作为预测结果时的准确率

Distance Metrics	50 images	500 images	5,000 images
Chose randomly	14.0%	7.9%	8.7%
L0 distance	88.0%	78.8%	68.5%
L1 distance	96.0%	94.8%	92.8%
L2 distance	100.0%	96.6%	93.6%

从实验结果可以看出，L2 距离的表现在三种距离度量方式中最好，虽然其也容易构造对抗样本（构造原理与 L1 距离类似），但总体准确率较高。L2 距离相较于 L1 距离的不足之处在于需要对每个向量都进行平方运算，速度较 L1 慢。考虑到综合表现，后续实验中将采取 L2 距离作为距离度量方法。



## 4 K 值的选取

在 KNN 算法中，K 值的选取是算法超参数中最为核心的部分。它指出算法将选取和待分类样本最相近的 K 个样本，并参与到后续的决策过程中。容易发现，合适的 K 值能保证算法的鲁棒性和准确度，避免偶然样本的出现影响算法判断。若 K 值过小（如在之前的实验中，我们选取最近邻样本标签作为预测结果，此时  $K = 1$ ），则算法容易被偶然出现的相似但标签不同的样本误导，从而作出错误的预测。但 K 值也并非越大越好——过大的 K 值一方面会影响算法的效率，另一方面也会选入更多标签不同的样本参与到决策过程中来，极端地，如果 K 值等于训练集中样本的总数，如果决策过程采取多数表决方法，那么预测的结果将永远等于集合中样本个数最多的标签，而不再和距离相关。图4.1展示了选取合适 K 值的重要性。



图 4.1 不合适的 K 值影响决策的结果

因此，需要结合理论和实验选取最为合适的 K 值。下文将结合实验选取最合适的 K 值。

下面的步骤展示了如何通过实验选取最合适的 K 值。

**Step 1** 选取 L2 距离作为评估方法，并采取多数表决的决策方式，初始化  $K=1$ ，运行算法预测测试集中前 200 个图片（为了节约时间，算法选取了训练集的前 20,000 张图片作为训练集图片）。

**Step 2** 将预测标签和实际标签进行比较，并计算预测准确率。

**Step 3** K++, 回到 Step2, 直到 K 大于 100.

**Step 4** 统计实验数据并绘制表格和折线图，选取折线图中准确率最大的 K 作为最终应用的 K 值。

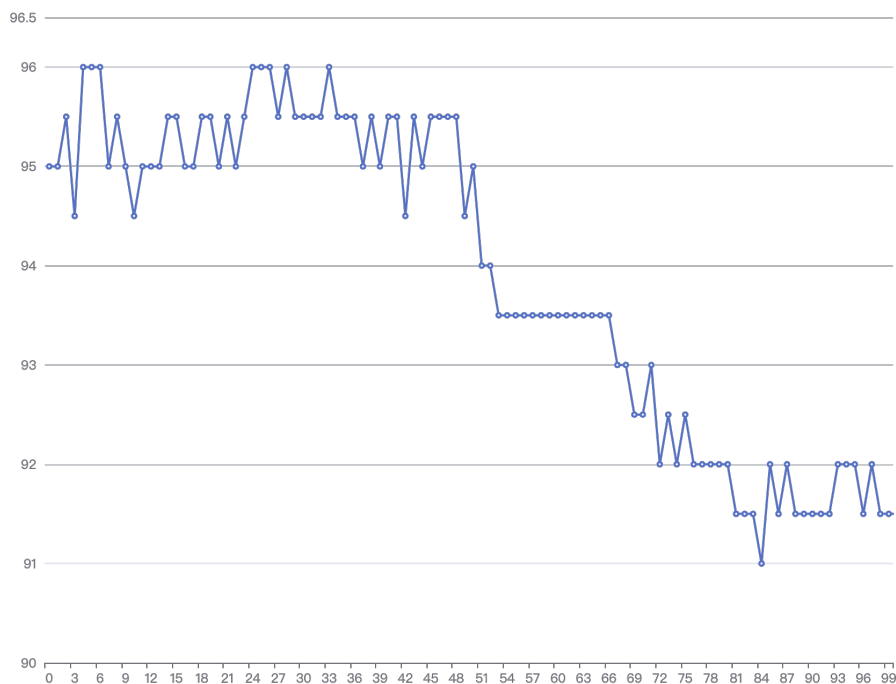


图 4.2 不同 K 值的准确率

统计实验数据绘得的折线图如图4.2所示，容易看出，由于训练集较为简单且评估方式较为合理等因素，准确率在 91% 到 96% 之间波动，均为较优秀的准确率。我们选取 5 作为最终的 K 值。

## 5 决策方式

决策方式决定了 KNN 算法如何从已经选出的 K 个最近邻样本集合中得出最终的预测结果。通常的预测方法有多数表决法和加权平均法。

### 5.1 多数表决法

多数表决法顾名思义，它认为 K 值选取合适时，集合中最多的标签就最可能是待分类样本的标签。因此，在多数表决法中，集合中的每一个样本都具有相同的地位。图5.1展示了多数表决法的决策过程。

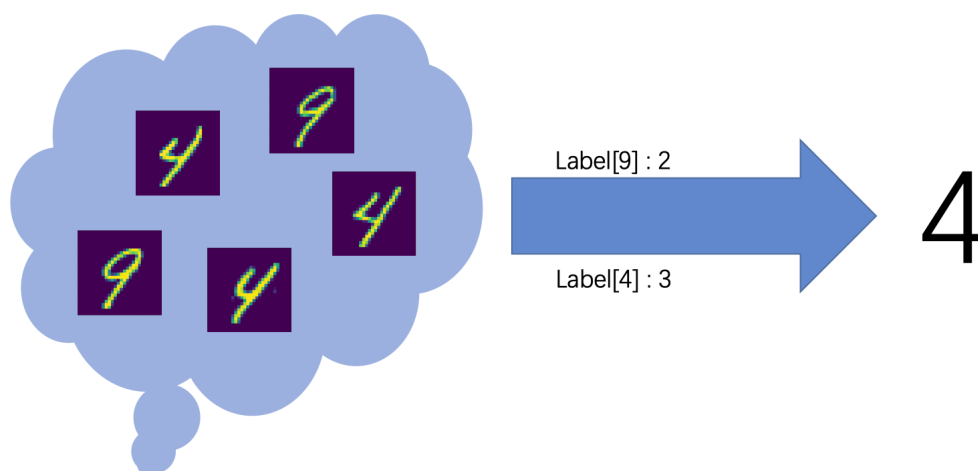


图 5.1 多数表决法的决策过程

### 5.2 加权平均法

加权平均法认为，最终待分类样本的标签不仅和集合中标签的数量有关，还和集合中图片与待分类样本图片的相似度有关，即：集合中图片和待分类样本相

似度越高，它的标签能获得的权重就越大。加权平均法中，一个样本的权重可以是它与待分类样本的距离，也可以是它在集合中的排名。

$$Result = \max(i \mid p_i * weight(p_i))$$

### 5.3 选取合适的决策算法

多数表决法和加权平均法都有其合理之处：多数表决法忽略集合中样本的特性，只按数量进行决策，避免了强相似或对抗样本对于集合的干扰；而加权平均法在数量之外还考虑了样本的相似性，能保证最终选出的样本在其评估方式下一定为最佳，在很多情况表现良好，但无法避免强干扰和强对抗样本对结果的影响。因此，决策算法的选取需要结合实际实验结果。

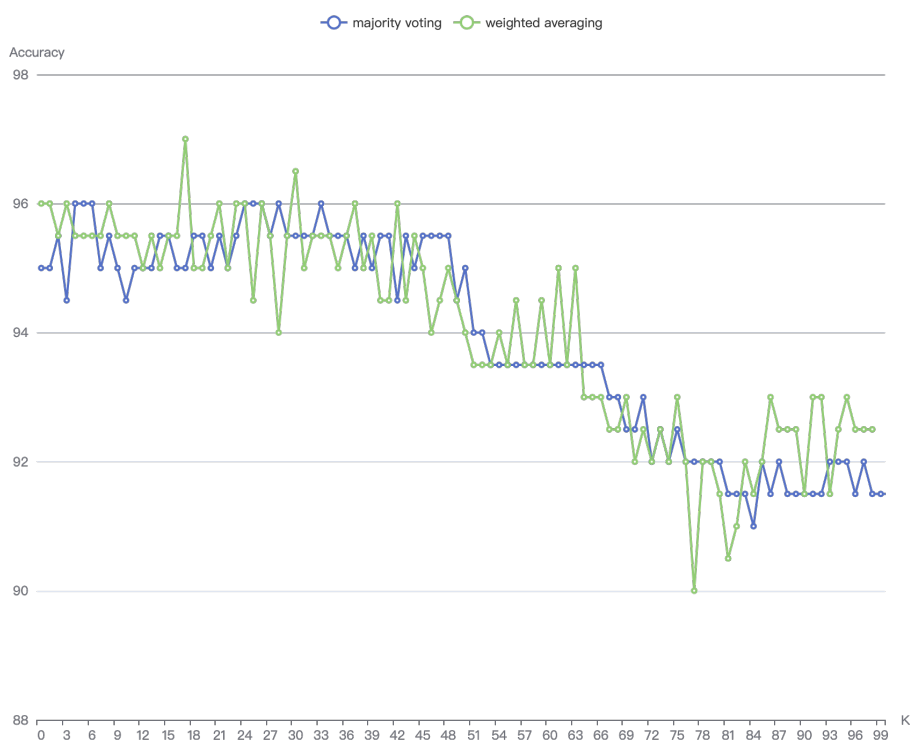


图 5.2 多数表决法与加权平均法在不同 K 上的表现

实验结果如图5.2所示，多数表决法和加权平均法不同 K 上的平均表现相近，但加权平均法的方差较大，这可能是因为 K 的微小变化会造成样本权值的变化。

同时  $K$  较大时加权平均法表现较佳，这可能是因为已经选取了足够多的、足以对抗强干扰样本的正确样本进入集合。

事实上，由于强干扰样本在图片识别中经常出现，多数表决议牺牲了极少的准确率换得了较加权平均法更强的鲁棒性和抗干扰能力。因此，在分类算法中多数表决议使用的更多，而加权平均法在回归计算中使用更多。 $K=5$  时，多数表决议表现更佳，因此我们选择多数表决议作为最终的决策方法。

## 6 实验结果验证

经过前文的实验，我们选取的 KNN 三个超参数最终为：

距离评估方式	L2 距离（欧式距离）
K 值	K=5
决策方式	多数表决法

我们将三个超参数放入模型，并将训练集调大至 60,000，在大小为 10,000 的测试集上验证最终结果。

最终得到的测试准确率如下表所示：

1,000 images	2,000 images	3,000 images	4,000 image	5,000 images
96.29%	95.84%	95.56%	95.62%	95.57%
6,000 images	7,000 images	8,000 images	9,000 image	10,000 images
95.98%	96.31%	96.66%	96.91%	96.93%

表 6.1 最终模型在测试集中的准确率

可以看到，最终模型在 10,000 张测试集上的准确率为 96.93%，结果较为令人满意，说明超参数的选取是较合理的。

## 7 实验弊端与改进方法分析

### 7.1 KNN 算法优劣性分析

首先, KNN 算法原理简单易懂, 实现起来也不难, 对新手很友好。其次, KNN 是一种惰性学习算法, 这意味着它不需要训练, 只需要将训练图片存储起来, 然后在测试时将图片取出一一比对即可。然而测试要花费大量时间计算, 因为每个测试图像需要和所有存储的训练图像进行比较, 这显然是一个缺点。在实际应用中, 我们关注测试效率远远高于训练效率。其实, 其他一些机器学习算法比如卷积神经网络更符合实际应用需求: 虽然训练花费很多时间, 但是一旦训练完成, 对新的测试数据进行分类非常快。对算法进行时间复杂度分析, 容易发现 KNN 算法的训练时间复杂度为  $O(1)$ , 而分类的时间复杂度为  $O(n)$ , 当训练集数据库非常庞大时, 这样的复杂度显然是无法接受的——以人脸识别为例, 我们当然不希望每次打开手机, 系统都需要遍历 14 亿人脸, 然后从中找到和我们最相似的结果。事实上, 在最后的实验结果验证环节, 即使使用了 Google 提供的 GPU 加速服务, 也耗费了将近两个小时时间才跑出最终的结果。

其次, KNN 算法对于距离评估函数的要求极高: 只有合理的距离评估函数才能有较为喜人的分类准确率。然而在实际应用问题中要找到一个合适的评估函数是极为困难的——尤其在数据维度较高 (不仅是大小, 还包含多个颜色通道)、初始数据大小不一等情况下, 高维度向量之间的距离通常是反直觉的。同时, 距离评估函数的白盒性也使得构造对抗样本极为容易。例如下面的图7.2展示了和原始数据在视觉上有极大差异, 但欧式距离相同的三张混淆图片。很显然, 基于像素比较的相似和感官上以及语义上的相似是不同的。

另外一个证明是, Laurens van der Maaten 等人的研究 [5] 指出, 单纯使用像素差异来比较图像是不够的。图7.3是他们的研究成果, 它将 CIFAR-10 中的图片按照二维方式排布, 这样能很好展示图片之间的像素差异值。在这张图片中, 排

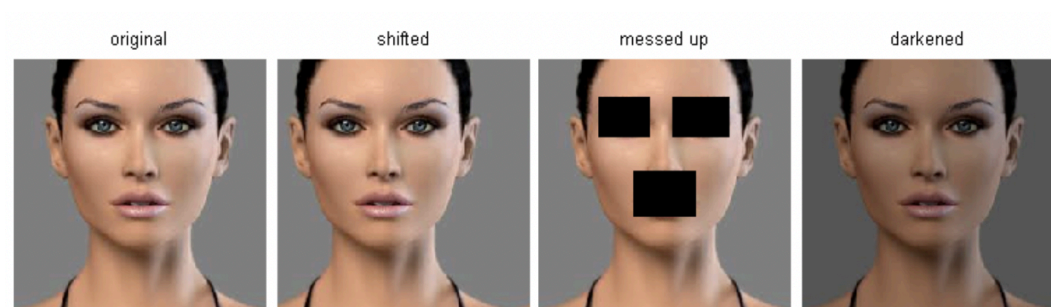


图 7.1 三张和原始图片 L2 距离相同的混淆图片

列相邻的图片 L2 距离就小。可以看出，图片的排列是被背景主导而不是图片语义内容本身主导。事实上，在 CIFAR-10 等多维彩色数据集上，无论使用何种距离评估函数、如何调整超参数，KNN 分类器的最高准确率也只有 40% 左右。这也说明 KNN 算法存在缺陷。

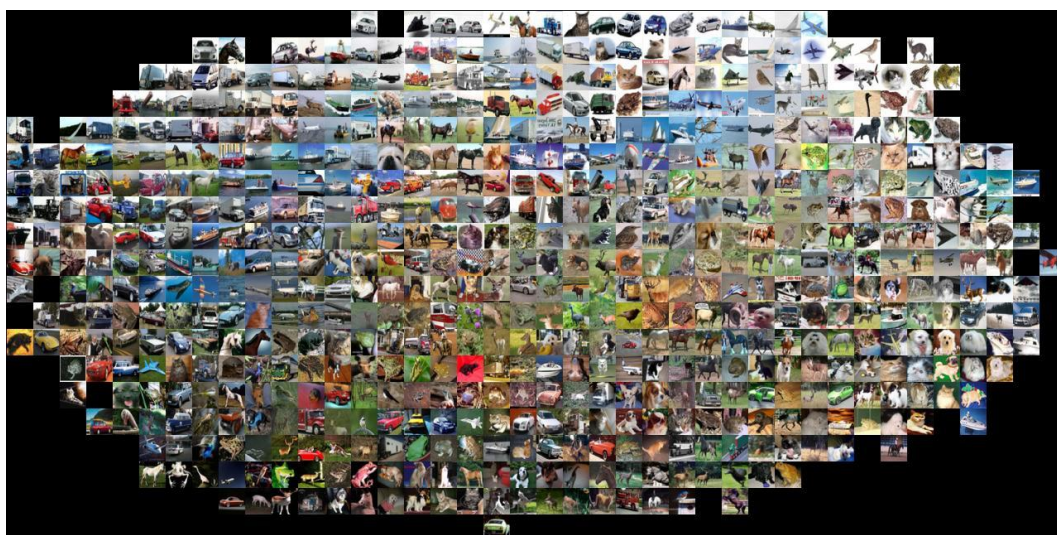


图 7.2 t-SNE, 图中越相近的图片 L2 距离越近

同样，由于距离评估函数和 K 值这两大超参数都和具体内容有极大的联系，因此需要对每个参数 K 和距离评估函数都进行尝试，而 KNN 的每次测试都需要消耗极大的时间，这是令人无法接受的。



## 7.2 实验过程中的缺陷

### 7.2.1 在测试集上验证

可以发现，在前面的实验过程中，笔者没有将训练集按比例划分为训练集和验证集，而是在训练集上训练、在测试集上验证。这可能会出现过拟合的问题：由于超参数都是基于测试集调整的，自然得到的是表现好的结果。然而真正在未知数据集上的表现未知。这就好比一个学生没有学习知识，而是将所有练习题和测试题都背了下来，由于平时模拟题都是测试题，该学生可能会有较好的表现，但当真正到高考考场面对没有原题的试卷时，该学生的表现将极为糟糕。

但是在实验过程中，笔者实际的操作是将测试集的前 200 张图片作为验证集使用，而测试集后 9,800 张图片没有在训练中使用。这也能在一定程度上解释在最终验证实验中，为什么在前 1,000 张图片上的表现最好，而后续的准确率出现了不同程度的下降，就是出现了过拟合的问题。

但值得提及的是，测试集的结果仅仅用于调试超参数，而并未加入测试集中。因此对验证的影响不算很大。

这启示我们以后一定要将训练集分为训练集和验证集两部分，训练集用于学习训练，验证集用来验证结果和调整超参数，测试集仅用于测试最终结果，不能作为调试超参数的依据，更不能放入训练集中，否则未来仍会出现过拟合的问题。

### 7.2.2 没有交叉验证

交叉验证是指，假设在划分测试集时将测试集分为等量的 5 份，4 份用于训练 1 份用于验证，那么交叉验证需要循环取 5 份数据中的 4 份，最后取五次训练的平均值作为最终的训练结果，如下图所示。

交叉验证能极大降低数据偶然性和噪音的影响，从而选取出最为合理的超参数。但交叉验证需要消耗更大的算力和更多的时间，而本实验中为节省时间没有使用。

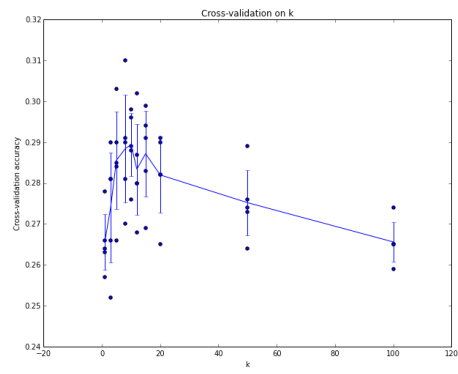


图 7.3 交叉验证的例子

## 参考文献

- [1] 赵卫东等. 基于 KNN 算法的手写数字识别研究 [J]. 成都大学学报 (自然科学版) , 2017, 第 36 期, 第 6 版.
- [2] 胡君萍等. 基于改进 KNN 算法的手写数字识别研究 [J]. 武汉理工大学学报 (信息与管理工程版). 2019, 第 41 卷, 第 1 期.
- [3] Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K., 2003. KNN Model-Based Approach in Classification, in: Lecture Notes in Computer Science. Lecture Notes in Computer Science, pp.
- [4] Divas Grover, MNIST Dataset Classification Utilizing k-NNClassifier with Modified Sliding-window Metric。
- [5] Laurens van der Maaten, <http://lvdmaaten.github.io/tsne/>.