# 手写数字机器学习系统设计与说明

## 数据集

本系统中我们采用的是MINST 手写字体数据，该数据库是数据库NIST的一个子集，这个数据库包含了60000个训练数据，10000个测试数据。该数据集已经经过规范化和标准化。该数据库适用于机器学习初学者设计真实世界数据的分类器，减少了学习者和研究者对于数据处理和规范化的时间。

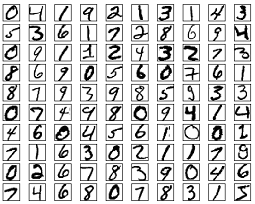


图1：MINST 手写字体数据样本实例

该数据库的下载地址是<http://yann.lecun.com/exdb/mnist/>，该数据库包含四个可用的文件：

train-images-idx3-ubyte.gz: 训练集 (9912422 bytes)

train-labels-idx1-ubyte.gz: 训练样本标签 (28881 bytes)

t10k-images-idx3-ubyte.gz: 测试集 (1648877 bytes)

t10k-labels-idx1-ubyte.gz: 测试样本标签 (4542 bytes)

训练集包含了60000个训练样本，采用十六进制的形式进行存储，每一个训练样本的代表一张28\*28大小的图片，训练样本标签代表着训练样本属于哪一类的数字，如我们所知0-9 共10类。测试集和训练集是类同的，测试集的标签主要是用作对分类器分类结果的评判。

该数据库将这些样本和标签数据采用十六进制存储，目的是减少存储空间，但是要用作分类器的输入需要对其进行解析，在后文系统设计里将详细的进行，数据库中数据的解析过程，已经系统应用过程中数据结构的设计。

## 二．系统框架



图2 ：本系统的框架结构

为了使系统的各部分功能更加明晰，我们采用模块化的设计和面向对象的设计方式实现系统。我们将系统划分成四个模块：数据解析模块、算法集成模块、计算模块、前端结果展示模块。

数据解析模块，主要的任务是将训练样本进行解析，对各个类别的样本数量进行统计，绘图，以便于使用者对样本的分布有直观了解，解决样本分布不平衡等问题，减少数据质量造成的问题对最终分类结果的影响。

算法集成模块，为了验证不同的算法的分类效果和分类结果，因此系统需要调用和实现不同的分类器。由于各个分类器对于输入输出而言，是大同小异的。除了内部实现不同，其输入和输出基本上相同的，因此可以将这些算法可以归纳集成。一方面便于代码的重复使用，减少代码量；另一方面增强系统的健壮性。

计算模块，接收数据解析模块产生的数据，生成算法集成模块实例，使用训练数据对分类器进行训练。得到模型后使用测试集数据进行测试，得到评价指标，评判分类器的好坏。

前端展示模块，主要的任务是与用户进行交互，对实验结果和数据分布进行可视化，为了让用户对手写字体数据有一个直观的认识，用户每次运行系统时，会随机的抽取10个手写数字的样本进行可视化。因为有10类的数据，为了评估分类器的性能，我们以总体的精度、召回率，fscore作为指标进行评判，同时在结果展示窗口对其进行展示。

## 三．数据解析模块与实现

训练样本标签数据train-labels-idx1-ubyte. 训练集是有60000个用例的，也就是说这个文件里面包含了60000个标签内容，每一个标签的值为0到9之间的数；我们先解析每一个属性的含义，offset代表了字节偏移量，也就是这个属性的二进制值的偏移是多少；type代表了这个属性的值的类型；value代表了这个属性的值是多少；description是说明。

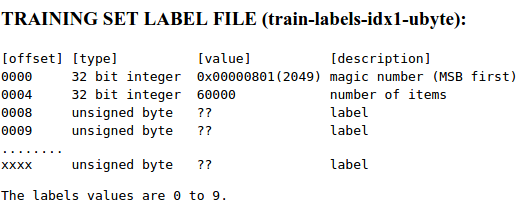


图3：训练标签数据的各字段说明

利用文本编辑器打开该文件，举例进行说明：

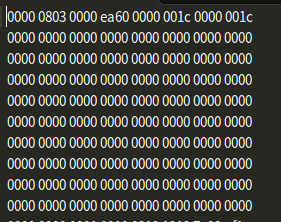


图4：训练标签数据的十六进制文件

每一个数字代表了四个位，两个数字代表了一个字节。我们首先看到偏移量为0字节处0000 0801代表了魔数，它的值为0000 0801，魔数是一个校验值，代表这个文件是否属于这个文件。偏移量为4字节处0000 ea60,我们表示容量数，也就是60000,而60000的十六进制就是ea60,；再看偏移量为8字节处05，它就表示我们的标签值了，也就是说第一个图片的标签值为5,后面的也是依此类推。

训练样本数据train-images-idx1-ubyte。各字段和上面标签数据类似如图5。

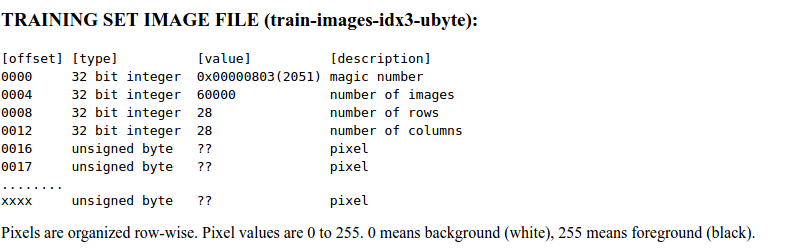


图:5：训练样本各字段简介

同样举例说明如图6所示在MNIST图片集中，所有的图片都是28×28的，也就是每个图片都有28×28个像素；我们的上述图片，其表示，我们的train-images-idx3-ubyte文件中偏移量为0字节处有一个4字节的数为0000 0803表示魔数；接下来是0000 ea60值为60000代表容量，接下来从第8个字节开始有一个4字节数，值为28也就是0000 001c，表示每个图片的行数；从第12个字节开始有一个4字节数，值也为28,也就是0000 001c表示每个图片的列数；从第16个字节开始，而且每784个字节代表一幅图片。

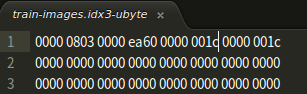


图6：训练集样本16进制数据

测试集标签和测试集样本和上面的训练集与训练集标签是一样的结构，不在赘述。

实现一个类DateUtil来实现这个解析模块。其成员函数依次为：

**parseData（datafname, dataLabels）**

参数：datafname数据文件名称；dataLabels标签文件名称；

返回值：一个字典结构，键是每张图片的ID ，值是一个字典结构，该字典有data 和Label两个字段，分别存储为28\*28的图片文件和图片的标签。

**getTrainData（）**

参数：无

返回值：成员变量字典trainDict，该函数调用了parseData，传入训练数据和标签，得到的是一个关于训练数据的字典，结构和parseData返回值结构相同。

**getTestData（）**

参数： 无

返回值：成员变量字典testDict，该函数调用了parseData,传入测试数据和标签，得到一个关于测试数据的字典，结构和parseData返回值结构相同。

**getTrainMatrix（num）**

参数：num 训练样本的数目

返回值：字典结构，键dataMartix，值为训练样本组成的矩阵；

键labelList，值为训练样本的标签组成的列表

**getTestMatrix（num）**

参数：num 测试样本的数目

返回值：字典结构，键dataMartix，值为测试样本组成的矩阵；

键labelList，值为测试样本的标签组成的列表。

**plotImage（imgId, imgType）**

参数：imgId样本的标号

imgType样本的类型（训练/测试）

返回值：无。传入合法的图片标号， 在训练/测试样本的字典中找到该标号的数据，进行展示。

## 四．算法集成模块与实现

为了研究不同的分类算法的性能，在这个模型中集成了决策树、SVM、AdaBoost、KNN、朴素贝叶斯、随机梯度下降法、随机森林等在内的7种算法。这7种算法的简单介绍和详细研究的链接如表1 所示。

表1 ：各种算法简介与比较

|  |  |  |
| --- | --- | --- |
| 算法名称 | 简介 | 链接 |
| 决策树 | 决策树算法是一种逼近离散函数值的方法。它是一种典型的分类方法，首先对数据进行处理，利用归纳算法生成可读的规则和决策树，然后使用决策对新数据进行分析。本质上决策树是通过一系列规则对数据进行分类的过程 | https://zh.wikipedia.org/wiki/决策树 |
| SVM | 支持向量机是CorinnaCortes和Vapnik等于1995年首先提出的，它在解决小样本、非线性及高维模式识别中表现出许多特有的优势，并能够推广应用到函数拟合等其他机器学习问题中 | https://zh.wikipedia.org/wiki/支持向量机 |
| AdaBoost | Adaboost是一种迭代算法，其核心思想是针对同一个训练集训练不同的分类器(弱分类器)，然后把这些弱分类器集合起来，构成一个更强的最终分类器（强分类器） | <https://zh.wikipedia.org/wiki/>  AdaBoost |
| KNN | 邻近算法，或者说K最近邻(kNN，k-NearestNeighbor)分类算法是数据挖掘分类技术中最简单的方法之一。所谓K最近邻，就是k个最近的邻居的意思，说的是每个样本都可以用它最接近的k个邻居来代表 | https://zh.wikipedia.org/wiki/最近鄰居法 |
| 朴素贝叶斯 | 贝叶斯分类是一类分类算法的总称，这类算法均以贝叶斯定理为基础，故统称为贝叶斯分类 | https://zh.wikipedia.org/wiki/朴素贝叶斯分类器 |
| 随机梯度下降法 | 梯度下降法是一个一阶最优化算法，通常也称为最速下降法。要使用梯度下降法找到一个函数的局部极小值，必须向函数上当前点对应梯度（或者是近似梯度）的反方向的规定步长距离点进行迭代搜索。如果相反地向梯度正方向迭代进行搜索，则会接近函数的局部极大值点；这个过程则被称为梯度上升法 | https://zh.wikipedia.org/wiki/梯度下降法 |
| 随机森林 | 随机森林指的是利用多棵树对样本进行训练并预测的一种分类器 | https://zh.wikipedia.org/wiki/随机森林 |

本模块通过集成7种分类器算法，实现algutil类，该类使用一个字典类型的成员变量algorithm来实现算法与名称之间的对应。其成员函数为

**classify(classifierName, trainData, testData)**

参数：classifierName分类器名称

trainData训练样本

testData测试样本

返回值：字典类型的分类报告，键为各类别标签如1，2 ..9 ,0,avg

值为字典类型，有三个字段precision，recall, fscore 。

该成员函数工作的流程传入一个算法的名称，通过成员变量algorithm获取到一个分类器对象，将训练数据和测试数据传入到分类器对象中得到分类的结果。

在本模块中我们使用Google 公司提供的 Python机器学习包sklearn 来实现以上的几个分类器算法。这个工具包由谷歌开发和提供实现了聚类、分类、回归、降维等多种功能。详情请见：http://scikit-learn.org/stable/

## 五．计算模块与前端展示模块及其实现

计算模型任务比较简单，该模块实例化了一个DataUtil和algutil。

将DataUtil生成的数据传入到algutil生成的分类器中得到分类的结果，考虑到algutil模块选择算法时需要和用户之间进行交互，DataUtil生成数据时也需要用户输入的数据，这个模块进一步和前端展示模块结合到了一起。

前端展示模块，是用户和算法进行交互的人机界面。分类器的分类效果实际上受到训练样本数目、测试样本的情况、已经分类器的影响，这些输入都需要前端界面使用合理的设计来接收用户的信息。

为了让用户对分类器的性能有直观的认识，对于实验结果的展示采用图表对于用户来说是直观的而且友好的，因此该模块也实现对于实验结果的图形化展示。初始化状态下，界面如图-7所示：



图7：系统的初始化界面

如图3 所示，图中第一行第三个矩形框是输入模块，该模块接收算法名称、训练样本数、测试样本数为输入。训练样本数、测试样本数输入到DataUtil实例中，产生相应数目的训练样本和测试样本；算法名称和训练样本、测试样本输入到algutil实例中，运行分类算法产生实验结果。

第一行第一个矩形、第一行第二个矩形接收以DataUtil产生的数据为输入，通过对训练样本和测试样本标签的统计，得到0~9类共10个类样本各自的数目，利用柱状图进行可视化。

第二行第一个矩形，随机生成10个样本的序号，将其绘制在该窗口。

第二行第一个矩形，在algutil实例的分类算法运行完毕后，统计各个类别分类的precision、recall，进行复合柱状图的绘制。

第二行第三个矩形，以一个类似于表格的形式展示了总体的精度、召回率、Fscore、算法的运行时间。

系统实际运行后如图4所示，这里采用随机森林分类器，训练样本10000个，测试样本1000个。

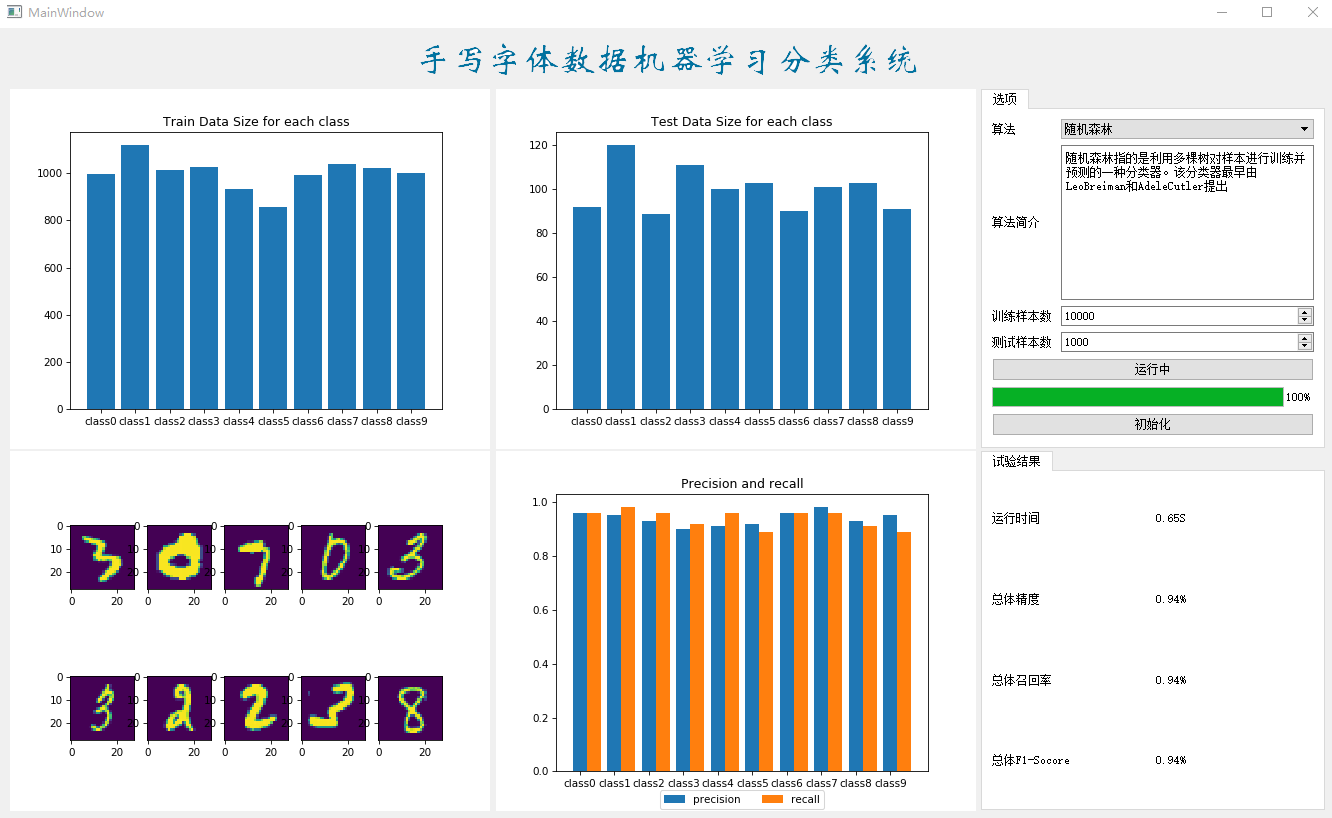


图4：随机森林分类的实验结果

计算模块与前端展示模块，通过一个类Ui\_main实现，该类的成员函数如下：

**setupUi（）**

参数：无

返回值：实例化的各个输入控件和图形展示控件

功能：初始化各个控件

**changeComobox（）**

参数： 无

返回值：无

功能：将算法名称和算法简介进行联动，当选择不同的算法时，在算法简介的展示控件中，显示算法的介绍，这些数据来自与的DataUtil的descriptData类，给类别返回了一个算法名称和简介一一对应的字典。与“算法“下拉控件绑定。

**reset（）**

参数：无；

返回值：无；

功能：将所有的空间恢复到初始化状态。与“初始化”按钮绑定。

**run（）**

参数： 无；

返回：训练样本分布、测试样本分布、抽样结果、实验结果；

功能：多线程的形式运行算法，主线程运行训练样本分布、测试样本分布、样本抽样的方法；子线程运行分类算法，得到分类结果字典，绘制分类结果图。与“运行”按钮绑定。

**algorithm(algorithm, trainData, testData)**

参数：algorithm算法名称

trainData训练数据

testData测试数据

返回：表述分类结果的字典结构，已经绘制好的分类结果的图。

六、系统的安装说明