**华东理工大学**

**智能信号处理大作业**

|  |  |
| --- | --- |
| **题 目** | 手写体数字识别系统设计 |
| **学 院** | 信息科学与工程学院 |
| **专 业** | 信号与信息处理 |
| **学 号** | Y30160609 |
| **学生姓名** | 何林飞 |
| **指导教师** | 万永菁 |
| **完成日期** | 2017年5月12日 |

**手写体数字识别系统设计**

1. **内容概述**

本课程设计使用计算机视觉数据集MNIST[1]作为实验数据，它包含各种手写数字图片：



它也包含每一张图片对应的标签，告诉我们这个是数字几。比如，上面这四张图片的标签分别是5，0，4，1。MNIST数据集包含60000个训练样本和10000个测试样本，每个样本由784维的向量和1个样本标签组成，向量由28x28的图像像素构成的。

实验采用多种机器学习算法对实验数据进行分类，记录每种方法的训练和测试的运行时间，测试的分类准确度。使用的分类算法有：BP，RBF，PNN，GRNN，SVM，KNN，SOFTMAX，CNN。使用的特征降维算法有：PCA，ZCA，自编码。传统神经网络的分类效果在matlab中验证，其余算法都在python下验证。

1. **算法原理**

**常用分类算法概述：**

1. BP（Backpropagation）

反向传播算法- BP网络都是多层感知机（通常都会有一个输入层、一个隐藏层及一个输出层）。为了使隐藏层能够适合所有有用的函数，多层网络必须具有用于多个层的非线性激活函数：仅用线性激活函数的多层网络会与相当于单层线性网络。常用的非线性激活函数有逻辑函数、柔性最大函数和高斯函数。

初始化网络权值（通常是小的随机值）

**do**

**for Each** 训练样本 ex

prediction = neural-net-output(network, ex) // 正向传递

actual = teacher-output(ex)

计算输出单元的误差 (prediction - actual)

计算对于所有隐藏层到输出层的权值 // 反向传递

计算

{\displaystyle \Delta w\_{i}}

对于所有输入层到隐藏层的权值 // 继续反向传递

更新网络权值 // 输入层不会被误差估计改变

**until** 所有样本正确分类或满足其他停止标准

**return** 该网络

该算法的名称意味着误差会从输出结点反向传播到输入结点。严格地讲，反向传播算法对网络的可修改权值计算了网络误差的梯度。这个梯度会在简单随机梯度下降法中经常用来求最小化误差的权重。通常“反向传播”这个词使用更一般的含义，用来指涵盖了计算梯度以及在随机梯度下降法中使用的整个过程。在适用反向传播算法的网络中，它通常可以快速收敛到令人满意的极小值。

1. RBF（Radial basis function）

径向基函数RBF一般记为：，通常取高斯函数

所以网络(图2-1)的输出为

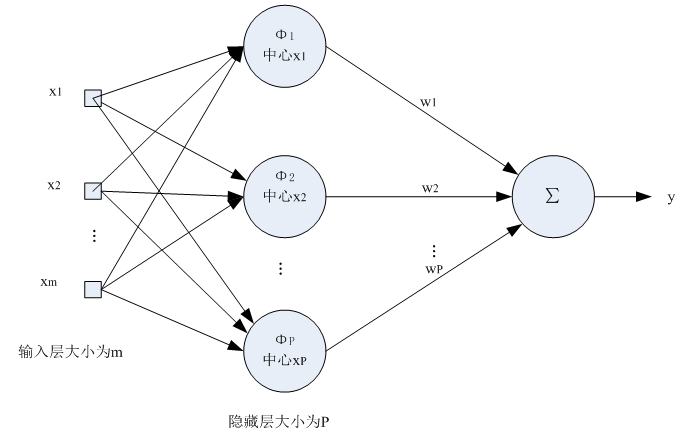


图2-1

网络中需要训练的参数有隐含层中基函数的中心，标准差，隐含层和输出层之间的权值。

正则化网络的隐含节点数等于输入训练样本个数，所以如果训练样本过大，计算量将非常巨大，所以为改进效率一般会减少隐含层节点数，即得广义网络。

基函数中心选择：1）随机选取固定中心，之后通过伪逆法求解权值；2）自组织选取中心：利用无监督聚类方法如k-means选择合适中心，之后同样用伪逆法求权值w；3）有监督选取中心：随机初始化中心和权值，用梯度下降等优化方法不断优化所有参数。

1. PNN（Probabilistic neural network）

概率神经网络（PNN）主要利用贝叶斯决策规则，即错误分类的期望风向最小，在多维输入空间内分离决策空间。吸收径向基网络与经典概率密度估计原理的有点，与传统的前馈神经网络相比，在模式识别方面具有较为显著的优势。

其网络结果如图2-2，本质还是贝叶斯决策过程，不需要训练，具体过程参考讲义。

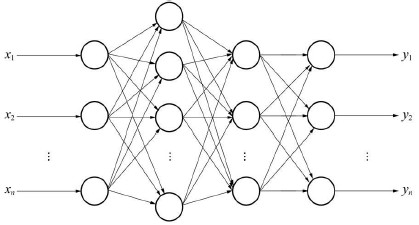


图2-2

1. GRNN（General Regression Neural Network）

广义回归神经网络（GRNN）是径向基神经网络的一种，GRNN具有很强的非线性映射能力和学习速度，比RBF具有更强的优势，网络最后收敛于样本量集聚较多的优化回归，样本数据少时，预测效果很好，网络还可以处理不稳定数据。

网络的结构如图2-3，该网络不需要训练，模式层是一个径向基函数，参数认为预设。加和层的S和都是k维向量。其中



图2-3

1. SVM（Support Vector Machine）

支持向量机（SVM）的原理是构造一个超平面，把数据分隔开来。其中两边距离超平面最近的点叫支持向量。

算法由最优间隔分类器衍生而来，通过拉格朗日对偶转换目标的优化问题，SMO算法优化寻优方法，L1正则化减小异常值的影响，核函数解决非线性问题。详见[2]

对于多分类问题，一般有两种解决方法：

一对多法（one-versus-rest,简称OVR SVMs）：训练时依次把某个类别的样本归为一类,其他剩余的样本归为另一类，这样k个类别的样本就构造出了k个SVM。分类时将未知样本分类为具有最大分类函数值的那类。

一对一法（one-versus-one,简称OVO SVMs或者pairwise）：其做法是在任意两类样本之间设计一个SVM，因此k个类别的样本就需要设计k(k-1)/2个SVM。当对一个未知样本进行分类时，最后得票最多的类别即为该未知样本的类别。这也正是libsvm[3]使用的多分类算法。

1. KNN（K-Nearest Neighbor）

最近邻法（NN）的直观理解如图2-4，两幅图像的每一个对应位置的像素求距离，并对所有距离求和。若在训练集中有一幅图像与测试图像的该距离和最小，那么测试图像认为是该图像的同类。

K最近邻法（KNN）是在这基础上的改进。取训练集中前K个与测试图像距离和最小的图像，出现最多的类别最为测试样本的类别。K为人为选择参数，可以通过交叉验证选取。

最常用的是k-fold cross-validation，意思其实就是把原始数据分成k份，轮流使用其中k-1份作为训练数据，而剩余的1份作为交叉验证数据。因此其实对于k-fold cross-validation我们会得到k个accuracy，取使其平均值最大的k作为K。

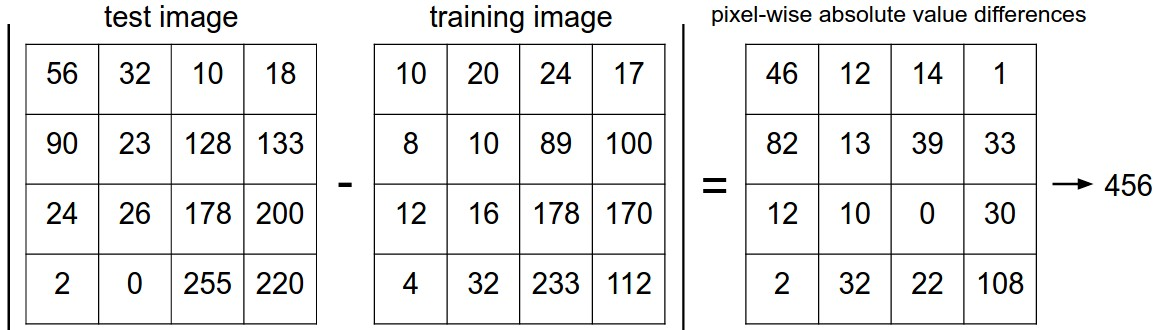


图2-4

1. SOFTMAX

Softmax回归模型是logistic回归模型在多分类问题上的推广，在多分类问题中，类标签y可以取两个以上的值。

logistic回归模型的假设函数是sigmoid，而Softmax回归模型的假设函数如下式：


\begin{align}
h_\theta(x^{(i)}) =
\begin{bmatrix}
p(y^{(i)} = 1 | x^{(i)}; \theta) \\
p(y^{(i)} = 2 | x^{(i)}; \theta) \\
\vdots \\
p(y^{(i)} = k | x^{(i)}; \theta)
\end{bmatrix}
=
\frac{1}{ \sum_{j=1}^{k}{e^{ \theta_j^T x^{(i)} }} }
\begin{bmatrix}
e^{ \theta_1^T x^{(i)} } \\
e^{ \theta_2^T x^{(i)} } \\
\vdots \\
e^{ \theta_k^T x^{(i)} } \\
\end{bmatrix}
\end{align}


含权重项的代价函数为：


\begin{align}
J(\theta) = - \frac{1}{m} \left[ \sum_{i=1}^{m} \sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^k e^{ \theta_l^T x^{(i)} }}  \right]
              + \frac{\lambda}{2} \sum_{i=1}^k \sum_{j=0}^n \theta_{ij}^2
\end{align}


可以通过梯度法，L-BFGS等优化算法找出使得代价函数的最小的参数。测试集通过求假设函数确定分类。

1. CNN（Convolutional Neural Network）

卷积神经网络主要由三种类型的层来构成：卷积层(Convolutional Layer)，抽取层(Pooling Layer)，和全连接层(Fully)；将这三种层堆叠在一起就形成了一个完整的卷积神经网络，如图2-5所示。

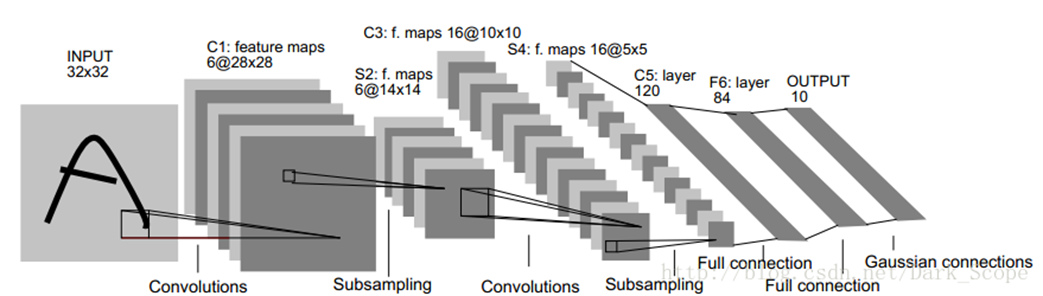


图2-5

卷积层就是从一个大尺寸图像中随机选取一小块，比如说 8x8 作为样本，并且从这个小块样本中学习到了一些特征，这时我们可以把从这个 8x8 样本中学习到的特征作为探测器，应用到这个图像的任意地方中去。特别是，我们可以用从 8x8 样本中所学习到的特征跟原本的大尺寸图像作卷积，从而对这个大尺寸图像上的任一位置获得一个不同特征的激活值，卷积过程如图2-6所示。



图2-6

在通过卷积获得了特征 (features) 之后，下一步我们希望利用这些特征去做分类。理论上讲，人们可以用所有提取得到的特征去训练分类器，例如softmax分类器，但这样做面临计算量的挑战。例如：对于一个96x96像素的图像，假设我们已经学习得到了400个定义在8x8输入上的特征，每一个特征和图像卷积都会得到一个 (96 − 8 + 1) \* (96 − 8 + 1) = 7921 维的卷积特征，由于有400个特征，所以每个样例example都会得到一个892\*400 = 3,168,400 维的卷积特征向量。学习一个拥有超过3百万特征输入的分类器十分不便，并且容易出现过拟合 (over-fitting)。

池化层就是为了解决上述问题而提出的。我们之所以决定使用卷积后的特征是因为图像具有一种“静态性”的属性，这也就意味着在一个图像区域有用的特征极有可能在另一个区域同样适用。因此，为了描述大的图像，一个很自然的想法就是对不同位置的特征进行聚合统计，例如，人们可以计算图像一个区域上的某个特定特征的平均值 (或最大值)。这些概要统计特征不仅具有低得多的维度 (相比使用所有提取得到的特征)，同时还会改善结果(不容易过拟合)。这种聚合的操作就叫做池化 (pooling)，类似于对图像的模板滤波过程[4]。

全连接层是对之前卷积层和池化层所提取出来的特征进行分类，常用的分类器有SOFTMAX，SVM，RBF等。

卷积神经网络有非常多的组合形式，其中有名的卷积神经网络有：LeNet，AlexNet，ZF Net，GoogLeNet，VGGNet。

**常用特征降维算法：**

1. PCA（Principal Components Analysis）

主成分分析（PCA）原本是一种对数据进行去相关的操作，也可以用于数据的降维。算法步骤如下：

1. 计算每个图像块的均值，并从每个图像块中减去它的均值得到x
2. 计算x的协方差矩阵
3. 求协方差矩阵的SVD分解得到U，S，V三个矩阵。那矩阵 U 将包含的特征向量（一个特征向量一列，从主向量开始排序），矩阵S对角线上的元素将包含对应的特征值（同样降序排列）。矩阵V等于U的转置，可以忽略
4. 取U的前K列，计算，y即是数据经过PCA降维后的结果。K为降维后的维度
5. ZCA

PCA白化是在PCA的基础上每个维度除以对应的标准差。

\begin{align}
x_{{\rm PCAwhite},i} = \frac{x_{{\rm rot},i} }{\sqrt{\lambda_i}}.   
\end{align}

ZCA白化，简称ZCA，在PCA白化基础上左乘特征矩阵U恢复原数据的方差。

\begin{align}
x_{\rm ZCAwhite} = U x_{\rm PCAwhite}
\end{align}

PCA白化需要保证数据各维度的方差为1，ZCA白化只需保证方差相等。PCA白化可进行降维也可以去相关性，而ZCA白化主要用于去相关性另外ZCA白化相比于PCA白化使得处理后的数据更加的接近原始数据[4]。

1. 自编码

自编码的网络结构和BP神经网络类似，如图2-5。但是自编码神经网络是一种无监督学习算法，它的目标值等于输入值，。虽然看上去不太有学习的意义，但是当我们为自编码神经网络加入某些限制，比如限定隐藏神经元的数量，我们就可以从输入数据中发现一些有趣的结构。比如网络有100个输入和输出节点，但只有50个隐含节点。那么训练过程就迫使自编码神经网络去学习输入数据的压缩表示。

如果网络的输入数据是完全随机的，比如每一个输入都是一个跟其它特征完全无关的独立同分布高斯随机变量，那么这一压缩表示将会非常难学习。但是如果输入数据中隐含着一些特定的结构，比如某些输入特征是彼此相关的，那么这一算法就可以发现输入数据中的这些相关性。事实上，这一简单的自编码神经网络通常可以学习出一个跟主元分析（PCA）结果非常相似的输入数据的低维表示。

一般隐含层的输出需要增加稀疏性的限制，这样能突出输入数据中的结构。所以含稀疏性限制的隐含层输出可以作为原始数据的特征。

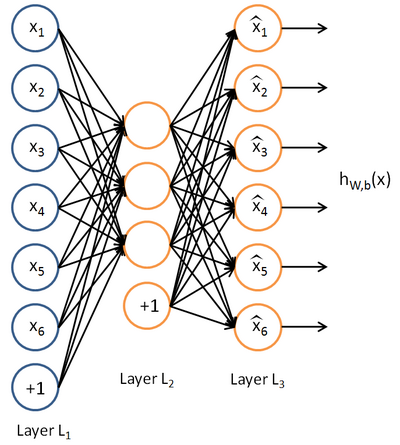


图2-5

1. **实验仿真**

实验平台：硬件Win10，64位，CPU：I5-3210M，无GPU；编程语言：matlab，python；IDE为Matlab2016和Pycharm。

1. 训练样本集和测试样本集的构成方式

数据集包含60000个训练样本和10000个测试样本，实验中使用60000x784的矩阵作为训练集输入，60000x1的矩阵表示训练标签。同理10000x784和10000x1的矩阵表示测试输入和标签。

为了方便程序debug，在一开始训练时会使用部分样本，如1000个训练，200个测试。

所有输入都归一化到[0，1]，防止出现数值问题。

1. 算法实现步骤
2. 产生小样本训练集和测试集
3. 使用小样本训练集对模型进行训练，确定模型最佳参数
4. 产生完整的训练集和测试集
5. 设置模型参数，并使用完整训练集进行训练
6. 测试集预测结果，求准确率和时间
7. 模型参数选择
8. BP神经网络使用matlab工具箱，根据经验，选取16个节点做为隐含层，20次交叉验证，‘trainlm’作为网络训练算法，训练结果如图3-1

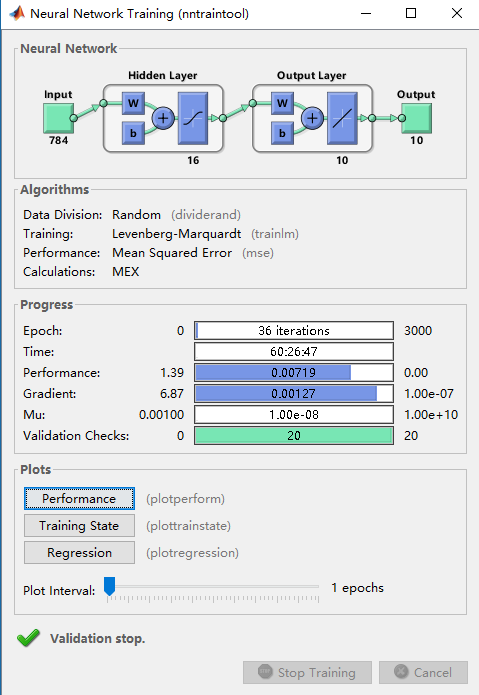


图3-1

1. RBF，PNN，GRNN在matlab中都只有一个speed参数，所以选取不同speed测试网络最佳参数，结果如图3-2，3-3，3-4。同时，在数据量很大的情况下，这些网络在会生成非常大的矩阵，超过实验平台的内存限制。所以实验采用小样本作为实验数据集，10000训练样本和2000测试样本

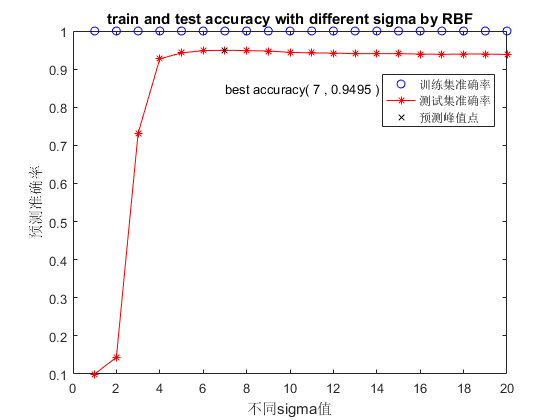


图3-2

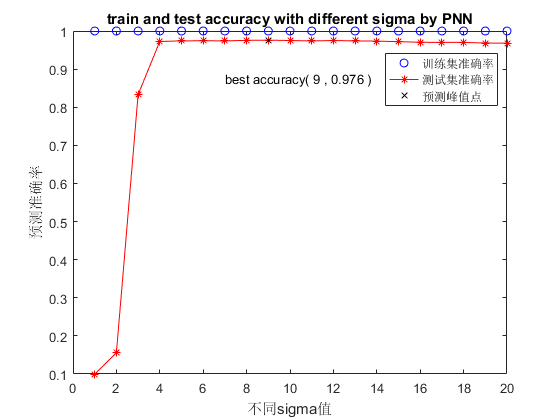


图3-3

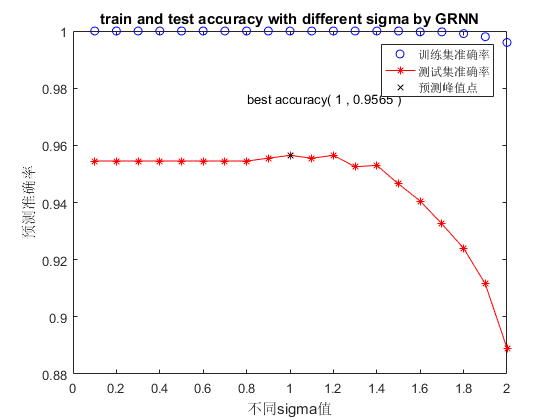


图3-4

1. SVM使用开源库libsvm[3]，使用高斯核函数，用find\_parameters函数自动通过交叉验证查找模型最佳参数c&g。计算得到c=8，g=0.03125时，模型有最好的预测效果，如图3-5

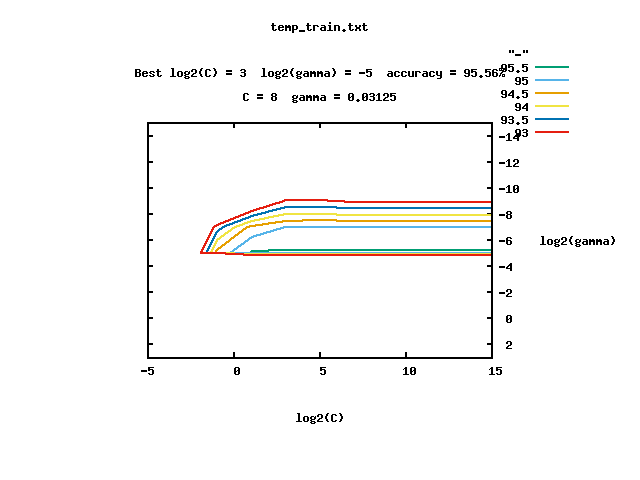
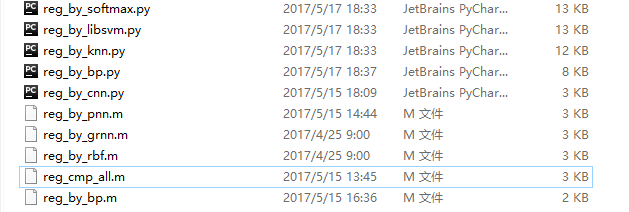


图3-5

1. KNN使用参数k=5，即最近邻的5个训练样本中出现次数最多的标签作为输出
2. Softmax使用L\_BFGS算法训练，最多迭代100次
3. CNN使用Thensorflow官方样例
4. 根据经验PCA选择保留90%~99%的成分都有较好的实验效果。所以实验保留90%的成分，训练集由784维降至345维
5. 稀疏自编码使用L\_BFGS算法训练，最多迭代400次，保留和PCA实验相同的维度，即345维，稀疏性参数取0.05
6. 实验结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 模型 | 语言 | 参数 | 训练时间(s) | 测试时间(s) | 测试准确率 |
| KNN+PCA | Python | k=5 |  | 2364.41 | 95.28 % |
| KNN+自编码 | Python | k=5 |  | 1840.47 | 97.21% |
| KNN | Python | k=5 |  | 4663.77 | 96.31 % |
| SVM+PCA | Python | c=8,g=0.03125 | 444.78 | 126.04 | 98.60% |
| SVM+自编码 | Python | c=8,g=0.03125 | 165.71 | 58.88 | 98.33% |
| SVM | Python | c=8,g=0.03125 | 789.81 | 190.58 | 98.57% |
| Softmax+PCA | Python | 见3 | 13.01 | 0.0082 | 92.33% |
| Softmax+自编码 | Python | 见3 | 13.56 | 0.0067 | 97.86 % |
| Softmax | Python | 见3 | 21.35 | 0.015 | 92.43% |
| CNN | Python | 官方样例 | 5498.74 | 87.22 | 99.24% |
| BP | Matlab | 见3 | 23207 | 0.29 | 87.18% |
| RBF | Matlab | σ=7，小样本 | 586.31 | 174.46 | 94.95% |
| PNN | Matlab | σ=9，小样本 | 593.96 | 180.71 | 97.60% |
| GRNN | Matlab | σ=1，小样本 | 0.22 | 173.48 | 95.65% |

1. 实验总结
2. 在小样本实验中，传统神经网络RBF，PNN，GRNN都表现出较好的分类能力，其中PNN正确率最高，GRNN其次，但GRNN总运行时间最短。随着参数speed的增大，RBF，PNN的泛化能力逐渐变强，最后趋于平稳；GRNN的泛化能力随speed增大而变弱，如图3-2,3-3,3-4所示
3. BP网络理论上能到95%左右的识别率，但几次实验的效果均不理想，问题应该出在训练参数的选择上
4. 从时间方面考虑，KNN的测试时间最长，不适合实时的应用；SOFTMAX测试时间最短，但识别率不高，加入自编码层后识别率上升明显，所以它适合做深度神经网络的分类层；SVM则中规中矩，高效且计算时间可以接受
5. 从降维方法看，PCA对KNN，SVM，SOFTMAX的识别准确率没有帮助，但降维能有效减少计算时间。自编码的特性则好于PCA，不仅减少计算时间，对于KNN和SOFTMAX算法识别率有提高；而对于SVM，识别率没有明显变化，但算法的训练和测试时间都减少了，这说明自编码后的样本中的支持向量减少了，加速了SVM的计算。这体现了自编码的特征提取的特性，它不仅能降维，还能突出样本的某种特征。而特征明显的样本更容易分类，对于KNN和SOFTMAX是识别率的上升，对于SVM则是支持向量数减少，类别层次更加明显。但是自编码的训练时间远长于PCA
6. CNN的识别率不出意外得最高。目前基于深度神经网络的分类算法在图像分类方面占有绝对优势，深度学习也将是机器学习发展的重要方向
7. **程序清单**



1. **参考文献**
2. Yann LeCun , Corinna Cortes, Christopher J.C. Burges, THE MNIST DATABASE of hand written digits, <http://yann.lecun.com/exdb/mnist/>
3. July，理解SVM的三层境界，<http://blog.csdn.net/v_july_v/article/details/7624837>
4. C.-C. Chang and C.-J. Lin. LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.
5. Andrew Ng，UFLDL Tutorial，<http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial>
6. 史峰等. Matlab智能算法30个案例分析[M].北京:北京航空航天大学大学出版社,2010:248-257.