**机器学习研究报告**

关于PCA、Kernel-PCA以及VAE在MNIST学习集上的表现

硕8065 鲁德镐 3118106022

### MNIST数据集简介

MNIST数据库是一个通常用于训练各种图像处理系统的大型手写数字数据库，该数据库还广泛用于机器学习领域的培训和测试。MNIST数据库包含60,000个训练图像和10,000个测试图像。每张黑白图像都被归一化以引入了灰度等级来适应28x28像素的边界，训练集的一半和测试集的一半来自NIST(由来自人口普查局和高中生构成的数据集)的训练数据集，而训练集的另一半和测试集的另一半来自NIST的测试数据集。如今有许多的机器学习方法都在MNIST上面运行过，是一个可靠性很高，可以进行对比的数据集。下图展现了MNIST中手写数学数据的一小部分



图 1‑1 来自于测试集中的一些简单例子图像[1]

MNIST数据集是从官方网址[2]下载的，其中包含四个文件：

* **Training set images**: train-images-idx3-ubyte.gz (9.9 MB, 解压后 47 MB, 包含 60,000 个样本)
* **Training set labels**: train-labels-idx1-ubyte.gz (29 KB, 解压后 60 KB, 包含 60,000 个标签)
* **Test set images:** t10k-images-idx3-ubyte.gz (1.6 MB, 解压后 7.8 MB, 包含 10,000 个样本)
* **Test set labels:** t10k-labels-idx1-ubyte.gz (5KB, 解压后 10 KB, 包含 10,000 个标签)

将这四个文件读入到我本次的实验平台：Python之中，并保存为字典的形式，其训练数据的维度为60000x784，其中784是将28x28的图像展开成一维向量的形式，而测试集数据的维度为10000x784

### 线性降维方法：PCA、Kernel-PCA

每一个训练样本含有784个向量，这相当于每一张手写数字图像是定位在784维空间中的一个点，处理这么大的维度，当然会延缓程序的运行速度，所以我们需要考虑到降维的问题，当然，降到多少维度可以对原始空间尽可能的表征还原，也是我们所应该考虑到的问题。

#### PCA

PCA的本质是：低维(K)空间上的目标数据Y是由高维(N)空间上m个样本数X通过与变换矩阵P进行如下变换得到的：

为了能够让数据在主轴上更好的表征，我们需要让数据在主轴v上的投影上的方差最大化，我们可以先考虑投影的表达形式，如式2所示，其中代表着与主轴单位向量的夹角，向量之间的内积也可以用矩阵相乘的形式进行表示。

则方差可以用式3表示(假设我们的数据已经经过预处理：均值归零化了)：

其中矩阵C正好代表着X的协方差均值，我们再使用拉格朗日待定系数法进行求解，得：

分别对上式的两个变量和进行求导，可得：

式3所提到的方差可以转化为，即数据在主轴v上的方差大小完全取决于其协方差矩阵C的特征值的大小，即将协方差矩阵C的特征值从大到小进行排列，前k个所代表着的特征向量所组成的矩阵即为变换矩阵。

另一方面，这k个特征向量需要相互线性无关，且均为单位向量，即变换之后的低维矩阵的协方差矩阵D应该为对角阵，这是由C是实对称矩阵所决定的。

关于低维空间的维度，在UFLDL教程[3]中，Andrew Ng 提到使用选取特征值得和占总特征值的百分比来反映保留的方差所占总方差的百分比，如下所示：

表 2‑1 PCA算法流程

|  |  |  |
| --- | --- | --- |
| **输入**: MNIST训练集和测试集，保留百分比 | | |
|  | **1.**将载入，组装成m行n列的矩阵(60000X784) |  |
|  | **2.**对每一行进行均一标准化 |  |
|  | **3.**求的协方差矩阵 |  |
|  | **4.**求矩阵*C*的特征值和特征向量，并按从大到小排列特征值和对应向量 |  |
|  | **5.**通过，选取*k*维的特征向量组装成矩阵 |  |
|  | **6.**得到降维过后的训练矩阵,测试数据矩阵同理 |  |
| **输出**: PCA后的MNIST训练集和测试集 | | |

#### Kernel-PCA

对于带有核函数的PCA算法，本质上式利用了我们无法得到目标空间函数的映射，但是我们可以通过原始空间中数据之间的内积来表达目标空间中的内积，这所相对应的映射，叫做核函数，我们用符号K啦表示，有：

和PCA方法同理，我们考虑目标空间中的协方差矩阵*C*：

我们进行如下推导，目前是无法直接得知目标空间协方差矩阵的特征值和特征向量，我们通过核函数矩阵进行推导：

从上式不难看出，目标空间中的协方差矩阵*C*和核空间矩阵具有一样的特征值，只是特征向量并不相同，这是便可以得到目标空间中在主方向（需要单位化）上的投影了。

值得注意的是，我们在PCA 中提前使用了均一化使得数据的均值为0，但是在目标空间之中，由于我们并不知道映射，需要对核函数矩阵K进行操作，即：

其中是元素为的均值；*N*为矩阵*K*的维度

表 2‑2 Kernel-PCA算法流程

|  |  |  |
| --- | --- | --- |
| **输入**: MNIST训练集和测试集，保留百分比 | | |
|  | **1.**将载入，组装成m行n列的矩阵() |  |
|  | **2.**对每一行进行均一标准化 |  |
|  | **3.**求的核函数均值 ，并将矩阵K()归一化 |  |
|  | **4.**求矩阵*K*的特征值和向量，并按从大到小排列特征值和对应向量（单位化） |  |
|  | **5.**通过，选取*k*维的特征向量组装成矩阵 |  |
|  | **6.**得到降维过后的训练矩阵,测试数据矩阵同理 |  |
| **输出**: PCA后的MNIST训练集和测试集 | | |

### 非线性降维方法：VAE

VAE，即变分自编码器（Variational Auto-Encoder），是基于自编码器的一个变种，想要知道VAE的原理，首先得先了解自编码器。

自编码器自2006年以来由G.Hittion[4]所提出，代表着一种新的理解：假设有一个网络，它将N维数据输入，在其中进行了多层的变换，最后又输出了相同维度的数据，那么，我们是否可以将该网络的隐藏层（M维）蕴含着的某些特点看做是输入数据的特点呢，因为我们根据隐藏层的变换，最后又重新输出了重建的原始数据。

首先我们考虑解码（decoder）与编码（encoder）的问题，我们假设解码（decoder）代表的问题往往具有以下特征：

1. 输入m维，输出n维的神经网络；
2. 该神经网络可以是任意的结构，但需要保证输入数据与输出数据处于同一区间下。

为了训练我们的解码器（decoder），我们需要一个辅助的编码器（encoder），通过对原输入数据进行编码，我们可以获取到数据中所蕴含着的隐藏特征。当然，获取特征的手段并没有限制。

为了将这两个结构对接起来，我们需要对编码器的输出进行相应的处理以符合解码器的输入，我们假设其输出是两个值：M维高斯分布的均值和方差的对数，通过这两个输出，我们可以构建解码器的输入层：既是一个高斯分布，这里通过对在上进行采样，然后使用来构建可以进行梯度优化的高斯分布，并对输入的似然进行优化，我们的目标函数是由输入与重建数据之间的交叉熵和对于编码器输出的均值与对数方差的KL散度之和，如下所示：

其中和分别代表着输入量和重建量；和分别是编码层的输出。

### 实验结果

#### PCA与Kernel-PCA

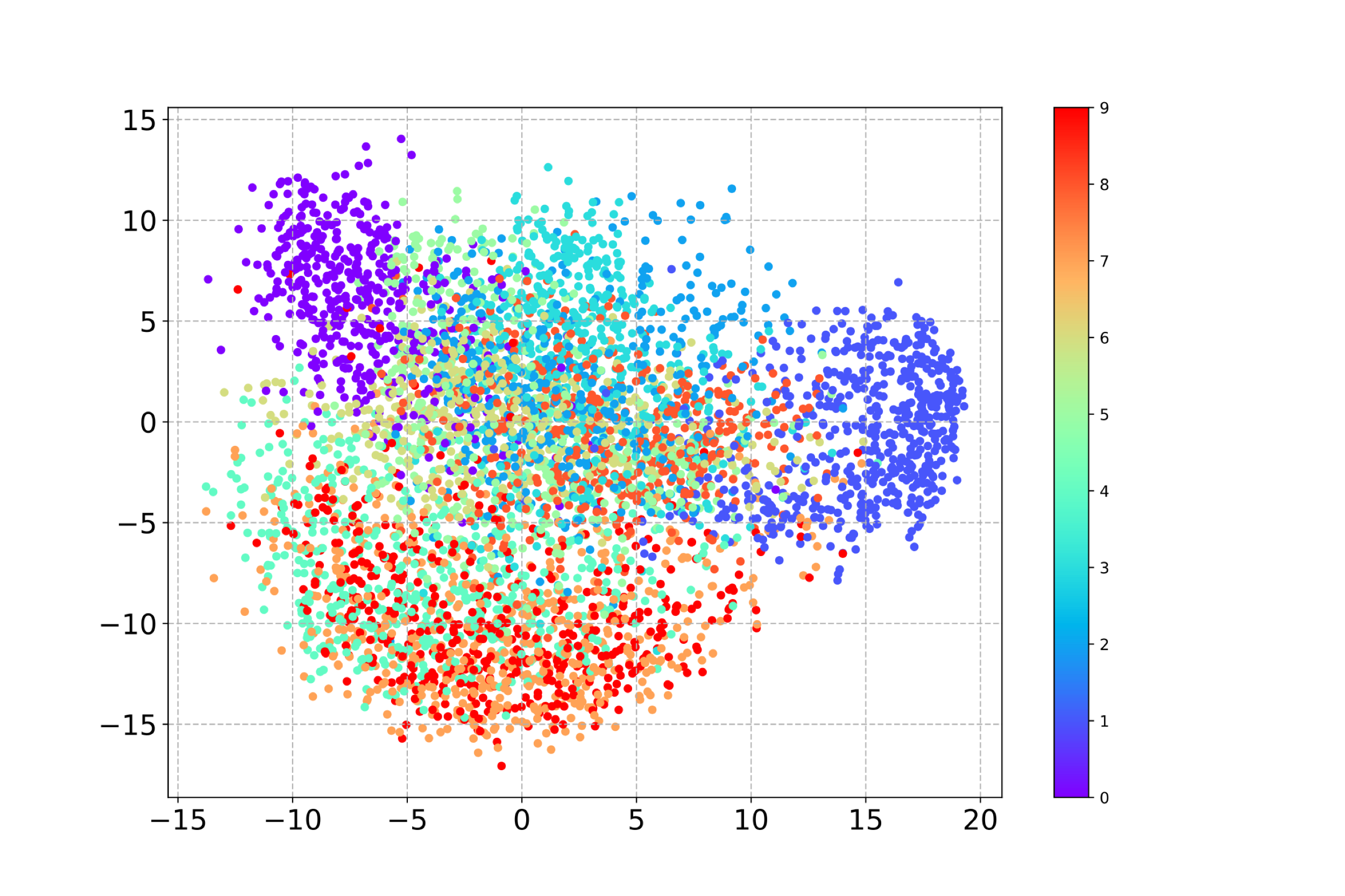
首先我使用Python中的Numpy库搭建了一个3层神经网络用于MNIST数据集的识别，隐藏层的数量分别为800与400，使用Mini-Batch对MNIST数据集进行每块128的划分，激活函数选择ReLu，输出层使用Softmax回归函数，反向传播使用Adam优化方法对目标函数进行梯度下降的优化以减小损失函数下降时的振荡，能够更快的达到最小值，其中的超参数为：

其中代表着学习率；分别是Adam优化方法中的相关参数。

首先在没有进行PCA降维的情况下进行训练，并同时对训练集和测试集进行预测，最终画出损失函数下降图以及训练集、测试集准确率图，并在此规定，只取前100次的迭代数据进行分析

通过普通的100次迭代分析我们可以知道，大约在第n次迭代处训练开始收敛，其对应的训练集与测试集的准确率分别如图所示：

对MNIST数据进行了PCA之后，我们重新观察目标函数的收敛性，我们可以发现，PCA之后的数据收敛的更快，而且达到相同的准确率所用的迭代次数也更少，这说明了PCA方法降维过程中在依然保留了图像原有的特征之中，还能够大幅缩小图像所占用的特征



### 附录：代码

六、参考

1. <https://en.wikipedia.org/wiki/MNIST_database#/media/File:MnistExamples.png>
2. <http://yann.lecun.com/exdb/mnist/>
3. http://ufldl.stanford.edu/wiki/index.php/