# 实验8《自编码器》

实验学时： 2 实验地点： 二综204 实验日期： 2019/12/05

1. **问题描述**

自编码器是一种数据的压缩算法，其中数据的压缩和解压缩函数有如下几个特点：

1. 数据相关的
2. 有损的
3. 从样本中自动学习。

在大部分提到的自动编码器的场合，压缩和解压缩的函数是通过神经网络实现的。

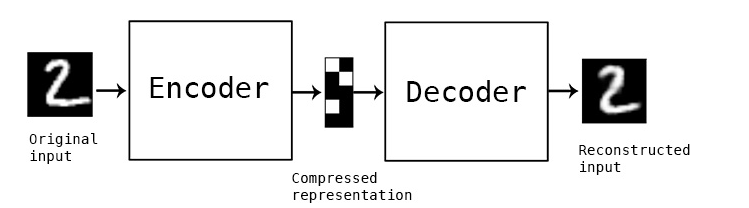
搭建一个自动编码器需要完成下面三洋工作：搭建编码器，搭建解码器，设定一个损失函数，用以衡量由于压缩而损失掉的信息。本实验会通过搭建一个简单的自编码器观测数据信息，并再搭建一个卷积自编码器作为对比，并学会使用自编码器进行降噪。

**二、 设计简要描述**

1．自编码器

#### 1.1 自编码器简介

自编码器是一类尝试使用反向传播算法重新创建输入数据作为输出的神经网络。自编码器包含两部分：编码器和解码器。编码器读取输入并把它压缩成紧凑表示，解码器则读取紧凑表示并用其重建输入。如下图所示：



#### 1.2 搭建简单的自编码器模型

#### 1.3 导入数据集

#### 1.4 拟合模型

#### 1.5 查看重构的输出与原来的输出对比

2. 卷积自编码器

#### 2.1 搭建卷积自编码器

当输入是图像时，使用卷积神经网络基本上总是有意义的。在现实中，用于处理图像的自动编码器几乎都是卷积自动编码器——又简单又快

卷积自编码器的编码器部分由卷积层和MaxPooling层构成，MaxPooling负责空域下采样。而解码器由卷积层和上采样层构成。

#### 2.2 加载数据并拟合模型

#### 2.3 查看重构的输出与原来的输出对比

3. 自编码器的应用

#### 3.1 介绍

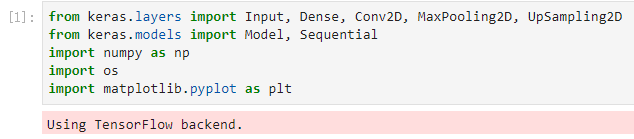
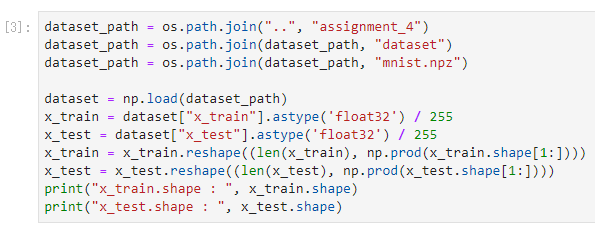
我们把训练样本用噪声污染，然后使用解码器解码出干净的照片，以获得去噪自动编码器。

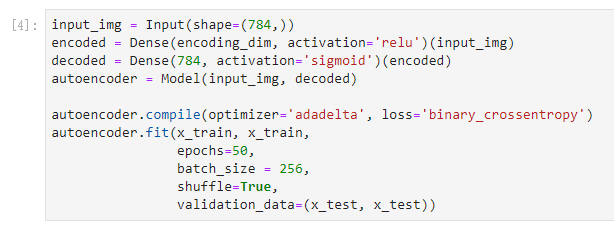
#### 3.2 把原图片加入高斯噪声

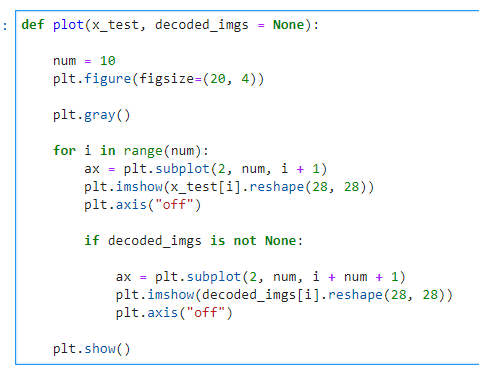
#### 3.3 构建自编码器

1. 设置input\_img参数
2. 搭建编码器，添加Conv2D层，输入是input\_img，输出维度32，卷积核大小3×3，激活函数’‘relu’,padding设置为‘same’，返回给x
3. 添加Maxpooling2D层，大小是（2，2），padding设置为‘same’
4. 添加Conv2D层，输出维度32，卷积核大小3×3，激活函数’‘relu’,padding设置为‘same’
5. 添加Maxpooling2D层，大小是（2，2），padding设置为‘same’，返回给encoded
6. 设置解码器，添加Conv2D层，输入是encoded，输出维度32，卷积核大小3×3，激活函数’‘relu’,padding设置为‘same’，返回给x
7. 添加上采样层，大小是（2，2）
8. 添加Conv2D层，输出维度32，卷积核大小3×3，激活函数’‘relu’,padding设置为‘same’
9. 添加上采样层，大小是（2，2）
10. 添加Conv2D层，输入是encoded，输出维度1，卷积核大小3×3，激活函数’‘sigmoid’,padding设置为‘same’，返回给decoded
11. 设置自编码器，输入是input\_img,decoded
12. 编译自编码器，参数不变。
13. 拟合数据，参数与2.1节一致

#### 3.3 查看重构的输出与原来的输出对比

1. **程序清单**
2. **导入相应工具包**
3. **读取数据**
4. **定义简单nn模型并训练**

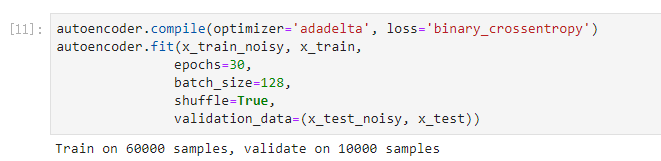
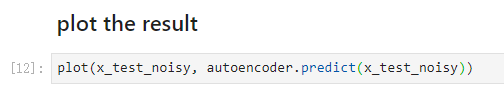
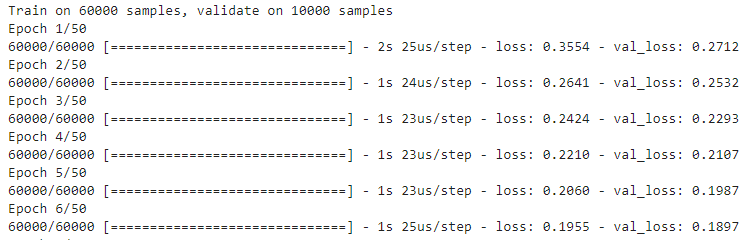
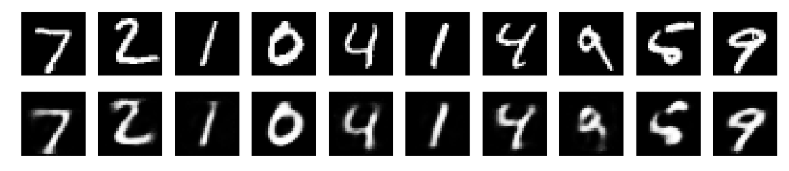
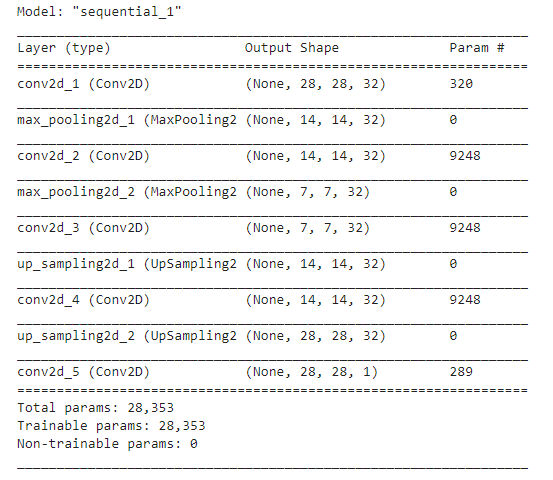
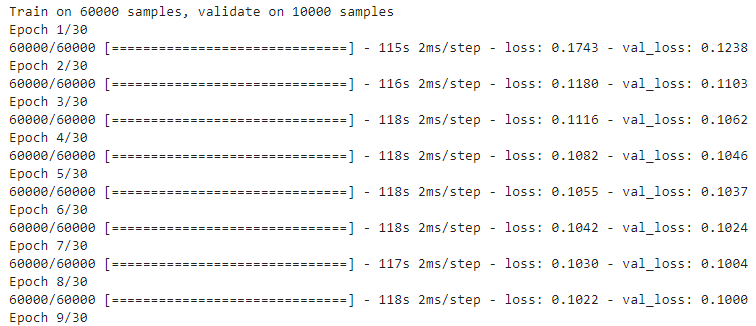
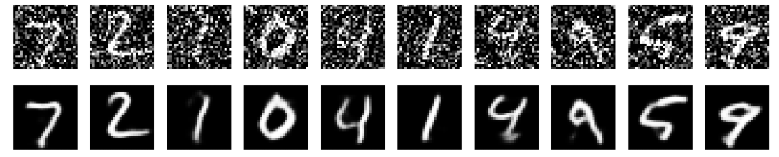


1. **定义画图函数**
2. **画出自编码器效果图**



1. **画出噪声图像**
2. **定义卷积神经网络**



1. **训练模型**
2. **查看去噪效果**
3. **结果分析**
4. **导入相应工具包**
5. **读取数据**
6. **定义简单nn模型并训练**
7. **定义画图函数**
8. **画出自编码器效果图**
9. **画出噪声图像**
10. **定义卷积神经网络**
11. **训练模型**
12. **查看去噪效果**
13. **调试报告**
14. **所遇到的问题**

和上次实验一样，基本没遇到问题，一些小问题通过报错信息也很快就解决了。

1. **经验和体会**

个人感觉自编码器是这八次实验里最炫酷的了。感觉自编码器的应用是非常广泛的，特别是自编码器的思想，不过我感觉自编码器的思想也可能是取自别的学科。比如在通信领域，编码和解码的操作非常频繁。但我们将自编码器应用到模型上会发生什么呢？将已经训练好的多个model放入autoencoder中进行训练，获得解码器参数，这样我们就可以生成无数多参数正常的model，但前提是我们训练的model是可以使用的。用这些生成的model组合为融合模型，效果会怎么样呢？

源码链接：

<https://github.com/sunnyswag/ml_dl_assignments/blob/master/assignment_8/assignment_8.ipynb>