

手册

2017 年 10 月 2 日

目录

1 平台硬件搭建	3
1.1 硬件清单	3
1.2 硬件平台照片	3
2 软件平台描述	6
3 实验过程	6
3.1 数据准备与预处理	6
3.1.1 数据预处理	6
3.1.2 数据可视化	6
3.2 使用 BP 神经网络进行数据拟合	7
3.3 使用自动编码器改进模型	7
3.4 使用降噪自编码与 DropOut 改进自编码	10

1 平台硬件搭建

1.1 硬件清单

硬件平台包含五台服务器，其中 2 台为 GPU 计算服务器，3 台为 CPU 计算服务器。详细清单如下表：

表 1：硬件清单

设备	型号	数量
GPU 计算服务器	AMAX	2
CPU 计算服务器	MIWIN	3
GPU	NVIDIA 1080	4
GPU	NVIDIA TITAN X	4
CPU	Intel Xeon E5-2698 v4	5
RAM	64 GB 2,133 MHz DDR4 LRDIMM	2
RAM	128 GB 2,133 MHz DDR4 LRDIMM	3
存储	1 TB HDD	4

1.2 硬件平台照片



图 1: 硬件设备概览



图 2: GPU 服务器机柜图



图 3: CPU 服务器图 1



图 4: CPU 服务器图 2

2 软件平台描述

软件平台： 操作系统为 Ubuntu Mate 16.04，使用的编程语言为 64 位的 Python，版本为 3.5.3，基于 Amaconda4.4.0 发行版。使用的第三方软件库包括 numpy，版本为 1.12.1，matplotlib，版本为 2.0.2，tensorflow，版本为 0.10，如图 5、6 所示。

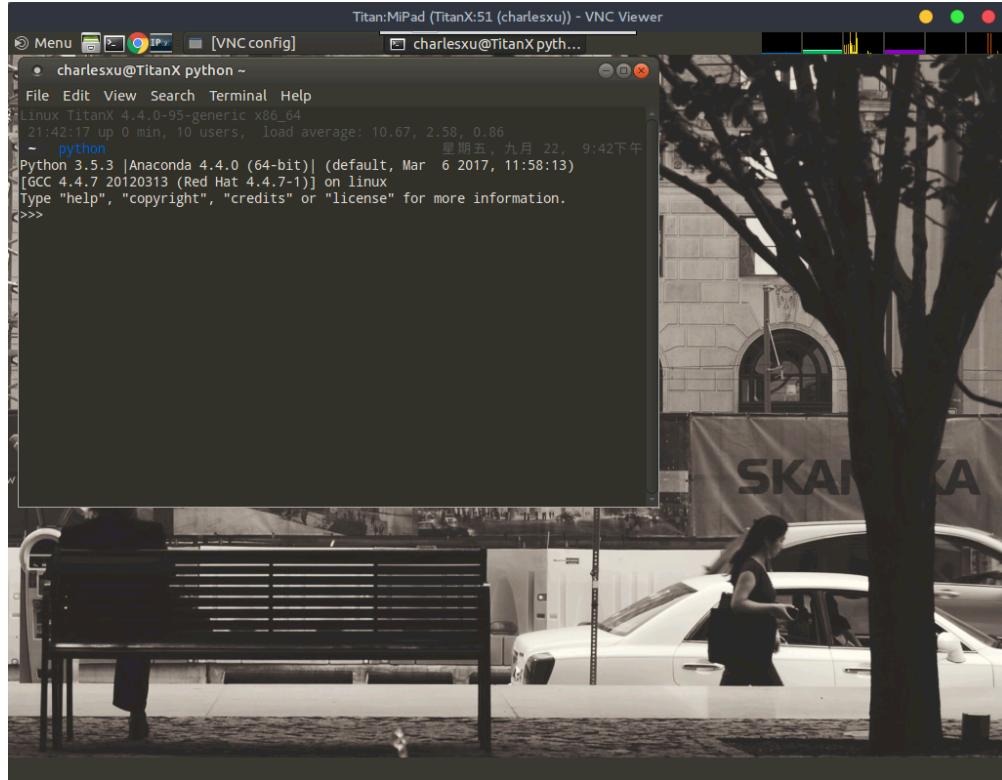


图 5：操作系统

3 实验过程

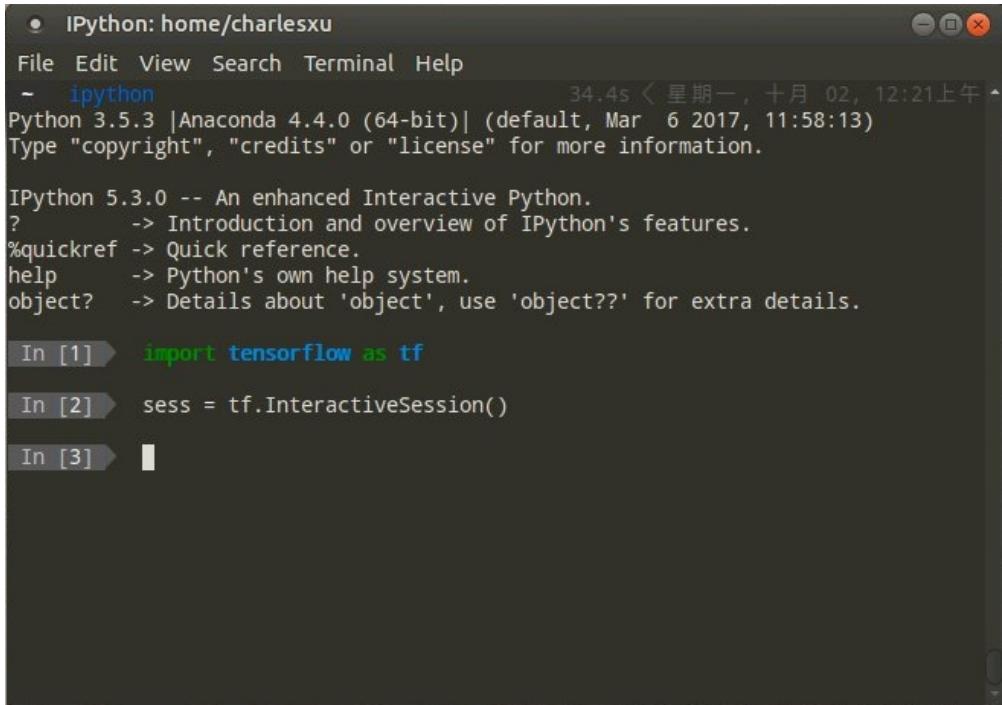
3.1 数据准备与预处理

3.1.1 数据预处理

由于本项目为单分类问题，将数据进行归一化，并将标签进行 One-Hot 编码。进行特征选择，筛选掉无效（即不变）特征，得到数据集，其中每条数据都是特征 → 标签的映射对，以便之后的分析。

3.1.2 数据可视化

首先对数据进行可视化分析，查看数据之间的关系，分析数据的特征。如图 7，通过数据可视化，显示出在第二维和第三维特征下的不同类别数据分布图。

A screenshot of an IPython notebook window titled "IPython: home/charlesxu". The window has a dark theme with light-colored text. At the top, there's a menu bar with File, Edit, View, Search, Terminal, and Help. Below the menu, it says "ipython" and shows the Python version: "Python 3.5.3 |Anaconda 4.4.0 (64-bit)| (default, Mar 6 2017, 11:58:13)". It also shows the date and time: "34.4s < 星期一, 十月 02, 12:21上午 >". A message at the bottom says "Type "copyright", "credits" or "license" for more information." Below this, there's a help section for IPython 5.3.0. The notebook has three cells: In [1] contains "import tensorflow as tf"; In [2] contains "sess = tf.InteractiveSession()"; and In [3] is currently empty. The status bar at the bottom right shows "In [3]: █".

```
File Edit View Search Terminal Help
- ipython
Python 3.5.3 |Anaconda 4.4.0 (64-bit)| (default, Mar 6 2017, 11:58:13)
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?           -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.

In [1]: import tensorflow as tf
In [2]: sess = tf.InteractiveSession()
In [3]: █
```

图 6: Python 界面

3.2 使用 BP 神经网络进行数据拟合

将数据输入到神经网络中，进行初步训练，并将神经元网络的效果作为分类效果的参考标准。基于神经元网络的特点，设计如图 8

为使神经网络训练达到最好的效果，使用“批量训练”的方法进行梯度下降优化；从数据中抽取一定量的数据作为测试数据集选择神经网络最优参数，并最终训练得到在当前数据下泛化性能最优的神经网络模型。参数选择的方法为：先采取最小网络规模，对神经网络进行训练，并在每一轮训练后加大网络模型的广度（即每层网络中神经元数量）直到网络误差不再减小为止，增加网络的深度，继续进行搜索，直到获得最优规模的神经网络。

3.3 使用自动编码器改进模型

为了解决传统的神经网络到一定层数之后误差非降反升的问题，采取更加先进的自动编码器对数据进行降维、编码，以使神经网络的训练更加容易。

首先使用自动编码器进行数据降维，为了检验信息损失程度，将子编码还原的图像与原图像进行对比，如图 9所示。使用自动编码器编码后得到编码特征如图 10所示，所编码得到的特征比图 7的原始特征更加显著，对于分类器的训练起到促进作用。

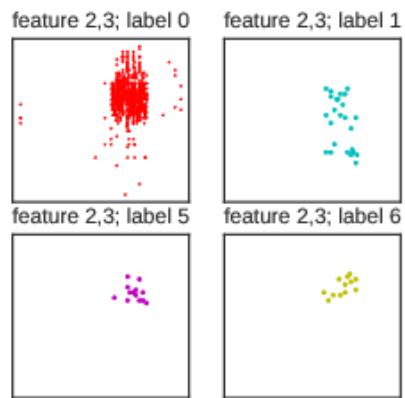


图 7: 数据特征可视化

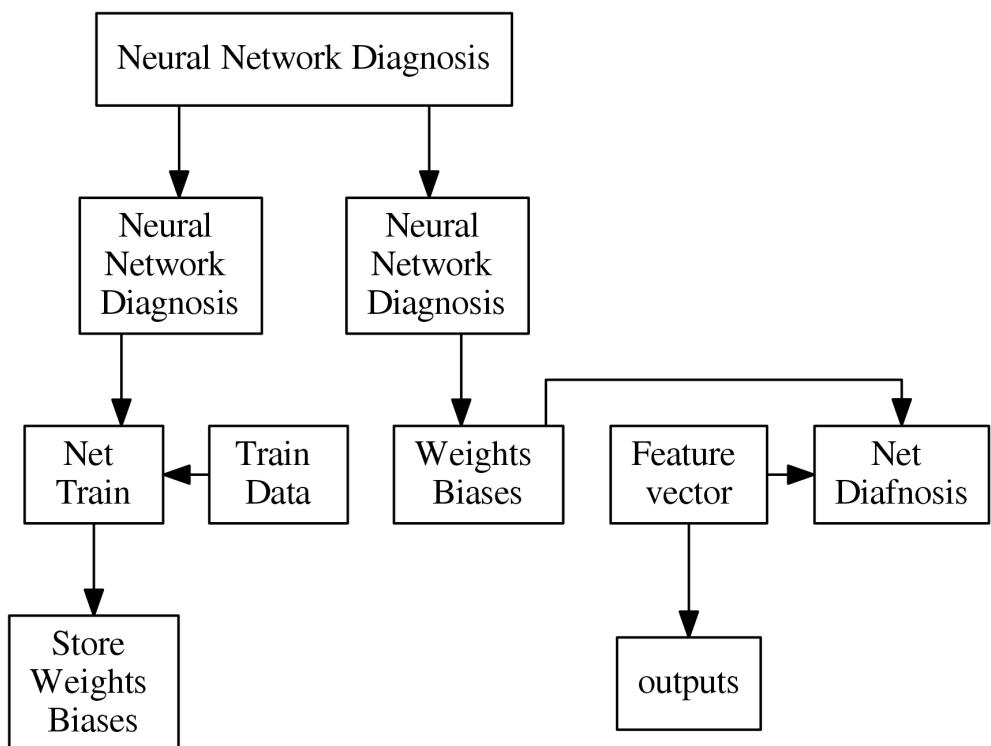


图 8: 神经网络故障诊断流程图

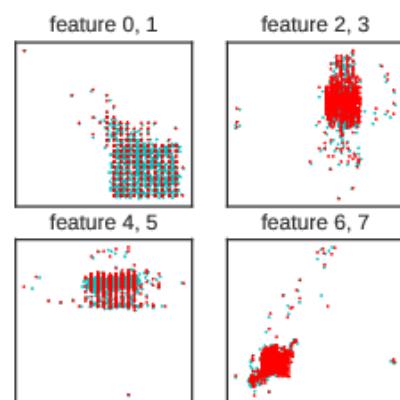


图 9: 自编码还原效果

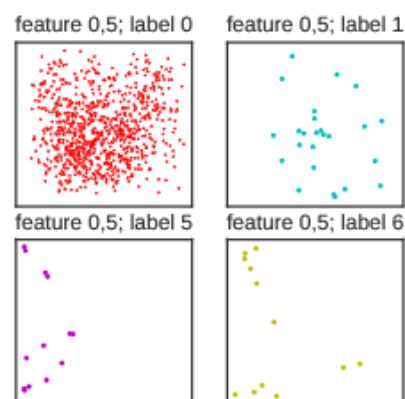


图 10: 自编码编码后的特征图

3.4 使用降噪自编码与 DropOut 改进自编码

自动编码器虽然在增加网络层数的时候对训练集的拟合比神经网络要好，在训练时却更加难以训练，因为经常出现过拟合而导致模型训练失败。为了解决这个问题，加入了对自编码的两个改进，同时使用降噪自编码和 DropOut 对自编码容易进入过拟合的问题进行改善。实验证明，分类效果具有显著提升，训练时也不易进入过拟合。