项目实践个人报告

2017级 自动化学院 模式识别课程 第六组 “厚 大 法 考”

作者：危昊成 06111704班 1120171140

1.小组项目完成情况概述

在本次课程中，小组使用了一维卷积神经网络对问题分类，圆满完成了任务。提交次数为3，结果准确率分别为0.98，0.99，1，最终达到了100%准确率。截止到我们提交，在DC竞赛平台上排名为47名。

我们的主要思路是将轴承震动的时域信号进行傅里叶变换，转移到频域进行处理。在频域中，轴承故障的特征更加集中，不同故障的特征差距非常明显，为训练简单的一维卷积神经网络提供了可行性。

使用卷积神经网络，更方便集中提取一维信号中的特征，同时进行降维。通过五个卷积层和两个池化层，我们将6000个点的频域信号压缩为32个特征，最后使用全连接层对特征进行一步线性化处理，输出为最后预测的label。将网络预测的label与数据集给出的label进行比对，以调整各神经元的权重。反复训练10次后，准确率达到理想值。

我们小组采用Python进行编程，和其他组不同的是我们采用了PyTorch深度学习框架而非Tensorflow。因此我们在程序结构上可能略显独特。PyTorch有非常方便的建立层次网络的模型并且提供了批处理的读取数据的方式，在时间和代码量上优化了我们的工作。我们采用GPU进行计算，与CPU相比，对网络的训练速度加快了约600%，最终优化后，我们的程序可以说是非常小巧轻量的一个程序。

2.个人所完成的具体工作

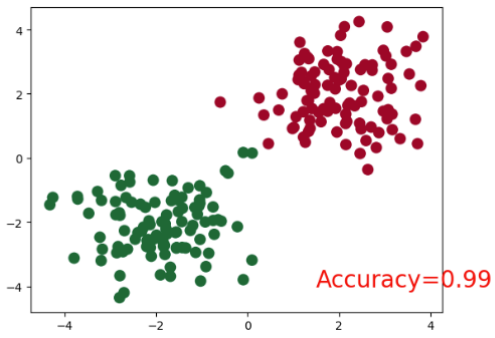
一句话概括：我在小组中承担组长的工作，协调组内成员的合作，带头完成代码编写，并主要负责了论文撰写、中期交流和结题交流的工作。

因为在开课时正好在自学PyTorch便大胆提出使用PyTorch作为我们的深度学习框架，不用传统的模式识别方式而用神经网络来完成任务。小组成员在编程上比较吃力，而任务量相比之前软件工程课程时小了很多，因此在编程部分基本是由我一个人完成的，代码贡献量占小组的90%以上。图为我们组项目文件夹中的所有文件。

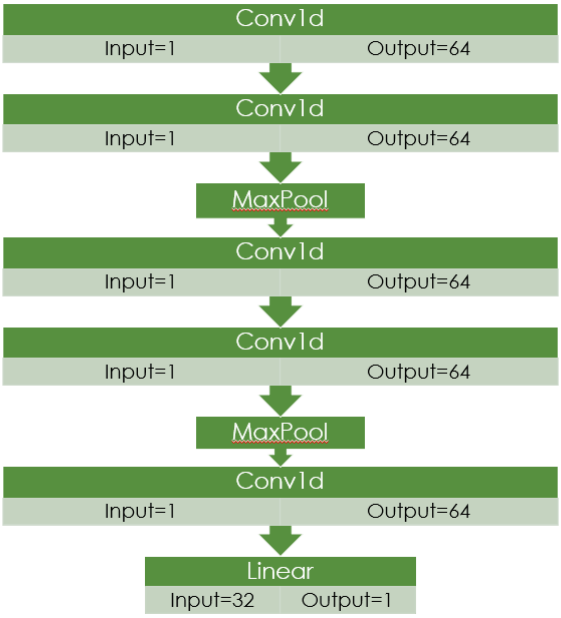
.idea	2020/5/18 15:11	文件夹	
.ipynb_checkpoints	2020/5/18 15:11	文件夹	
__pycache__	2020/5/18 15:11	文件夹	
fft.ipynb	2020/5/8 1:29	IPYNB 文件	55 KB
fftshow.py	2020/5/7 22:23	Python 源文件	2 KB
main.py	2020/5/12 22:10	Python 源文件	5 KB
main-old.py	2020/5/12 12:08	Python 源文件	4 KB
mycnn.py	2020/5/12 19:10	Python 源文件	2 KB
pred.csv	2020/5/12 22:12	Microsoft Excel ...	4 KB
test.py	2020/5/12 18:30	Python 源文件	1 KB
test_data.csv	2018/9/19 21:02	Microsoft Excel ...	62,902 KB
test_remastered.csv	2020/5/6 16:08	Microsoft Excel ...	38,152 KB
train.csv	2018/9/19 21:02	Microsoft Excel ...	94,277 KB
train_remastered.csv	2020/4/26 13:26	Microsoft Excel ...	57,195 KB
trainer-classifier.py	2020/5/6 20:28	Python 源文件	3 KB
trainer-classifier-old.py	2020/5/6 18:07	Python 源文件	4 KB
tutorial_net.py	2020/5/6 13:21	Python 源文件	1 KB
tutorial-classifier.py	2020/5/7 21:57	Python 源文件	2 KB
Untitled.ipynb	2020/5/7 19:44	IPYNB 文件	1 KB
Zhoucheng.ipynb	2020/5/12 21:20	IPYNB 文件	27 KB

其实在项目一开始，我也对Python编程不太熟练，但到现在我已经可以熟练使用Python解决大部分日常作业需要的编程环节了。上图中的三个.ipynb文件为JupyterNotebook的代码，在JupyterNotebook中，我主要进行numpy，scipy的学习，对自己python语法的精进，以及对我们竞赛给出的问题进行初步分析。

在学习时，我除了完成MOOC上的编程任务，自己也私下使用PyTorch进行了几个比较简单的分类器的训练，比如如图所示的二分类问题：



在项目开始时我只掌握了基本的BP网络，搭建了大约4层的BP网络结构并打算直接使用时域序列对网络进行暴力投喂，但结果并不理想。但随着后续项目的进行和同学之间的交流我发现我们的BP网络层次太浅了，需要更多层的网络。但当时没多想，就直接将网络结构换成了一维卷积神经网络。一维神经网络由王沛展同学提出，并最先用Tensorflow实现。后来我优化了网络结构并使用了PyTorch进行实现。网络结构如下图所示：



小组成员分别提出了三种处理数据的方案，我均给出了代码实现：

```python
def featureparam(a):
    x_avg = np.mean(a, axis=1, keepdims=True)  # 算术均值
    x_std = np.std(a, axis=1, ddof=1, keepdims=True)  # 标准差
    x_var = np.var(a, axis=1, ddof=1, keepdims=True)  # 方差
    x_ptp = np.ptp(a, axis=1, keepdims=True)  # 峰峰值
    x_rms = np.sqrt(np.mean(a ** 2, axis=1, keepdims=True))  # 有效值
    x_skw = stats.skew(a, axis=1).reshape(a.shape[0], 1)  # 偏度
    x_kur = stats.kurtosis(a, axis=1).reshape(a.shape[0], 1)  # 峰度
    feature = torch.from_numpy(np.array([x_avg, x_std, x_var,
                                         x_ptp, x_rms, x_skw, x_kur]).squeeze().T)
    return feature


def fastft(a):
    x = np.array([fft(a[i, :]) for i in range(a.shape[0])])
    y = np.array([np.abs(x[i, :]) for i in range(a.shape[0])])
    y = torch.from_numpy(y / (y.max() - y.min()))
    z = torch.from_numpy(np.array([np.angle(x[i, :]) for i in range(a.shape[0])]))

    return y, z
```

上图为特征值方法和快速傅里叶变换方法的实现代码，除此之外还有直接使用时域分析的思路，不需要额外代码。

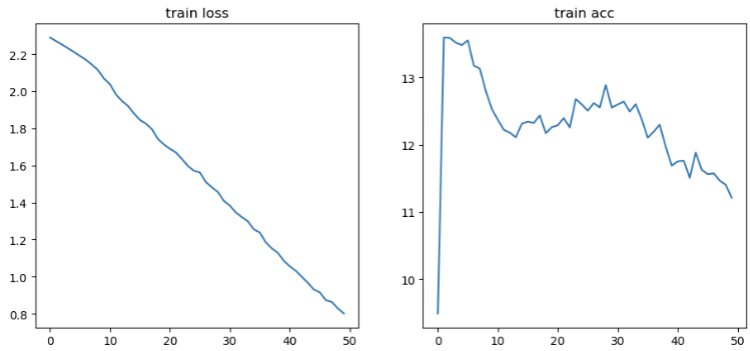
除了代码方面，我还负责了竞赛平台组队和结果的提交工作，并将代码托管到GitHub平台进行版本控制，平台上目前有三个版本的代码，目前为止收获了三颗Star，分别来自三位小组成员。

除此之外，小组的中期交流和结题展示由我负责进行PPT制作及讲解。图为结题展示PPT首页。

3.所遇到的主要困难和解决思路

3.1 网络效率太低

中期交流前使用BP网络进行训练时，由于网络过浅，得到的结果非常不理想，如下图：



当时训练的预测结果除了9就是7，正确率只有13%左右，明显是这个BP网络被前几个数据带偏了。

在之后我们就更换了一维CNN网络进行训练，同时加入了随机梯度下降优化器，使得训练数据不再从表格头部开始获取，训练的结果明显正常了许多。

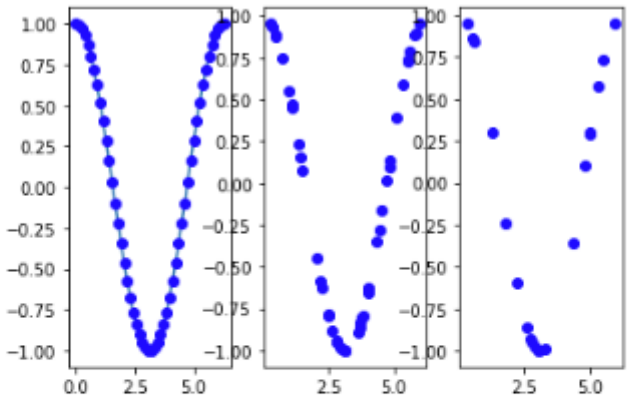
3.2 数据集编码问题

第一次加载数据集到程序时第一个字出现乱码，发现是由于数据集首部有3字节的隐藏文字，用于指示编码方式等等，然而这个小问题我尝试了很多种办法都无法使用程序手段进行解决。于是我直接使用Excel打开了CSV文件并另存了副本，解决了这个问题。

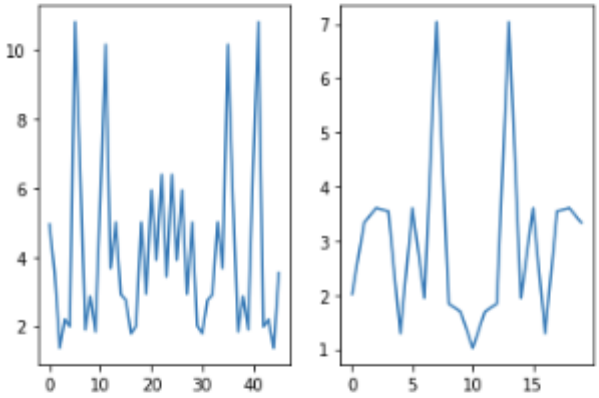
3.3 处理数据

一开始我们被竞赛官网的信息迷惑了，不知道这6000个采样是随机采样还是怎么采样的，为什么不能当作连续信号处理？于是我们模拟了随机采样的情况，并对采样结果进行FFT再进行IFFT，发现这时同一个波形经过一对傅里叶变换还原的时域信号已经难以辨认了：

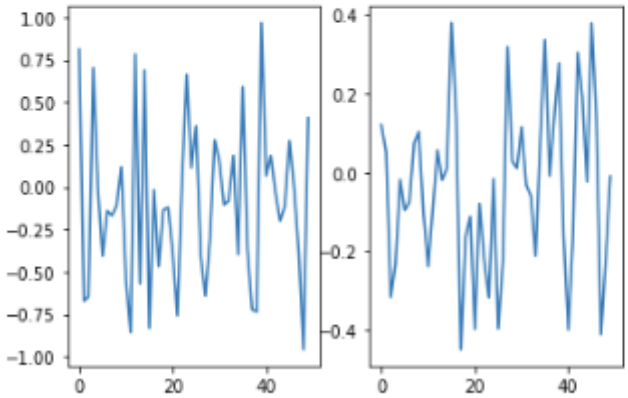
```
Out[15]: [<matplotlib.lines.Line2D at 0x2cf3f4c7888>]
```



original & different random sample



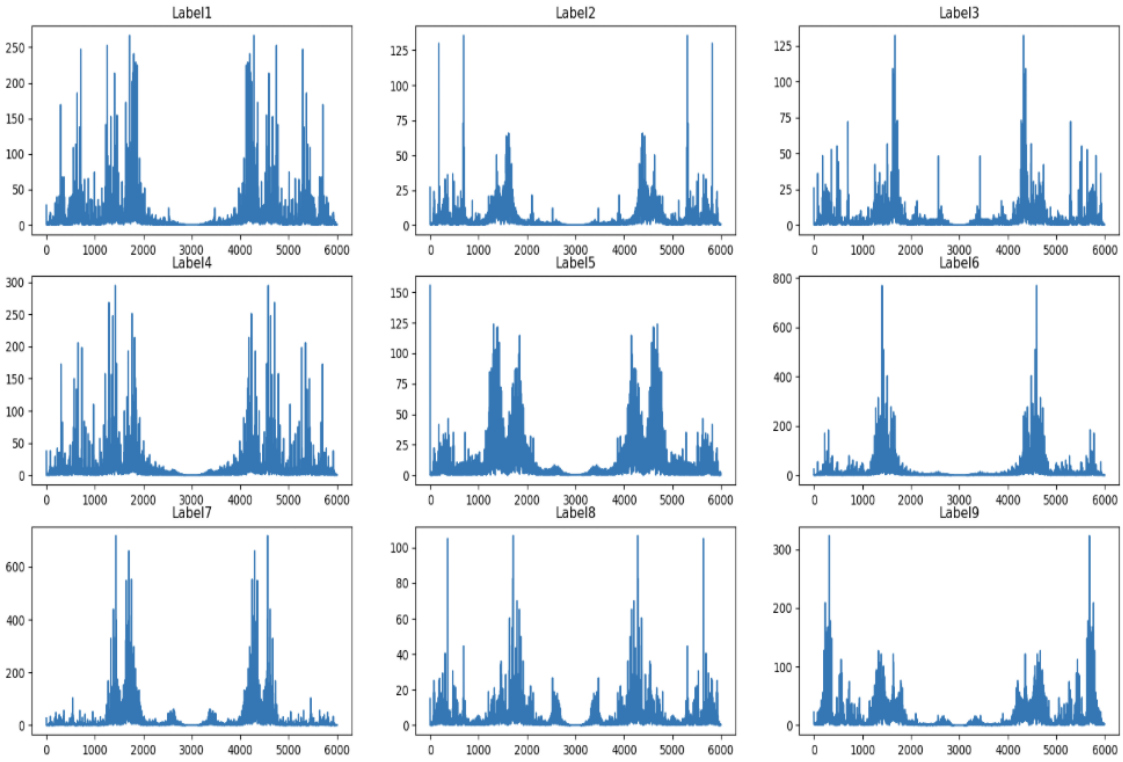
Fourier transform

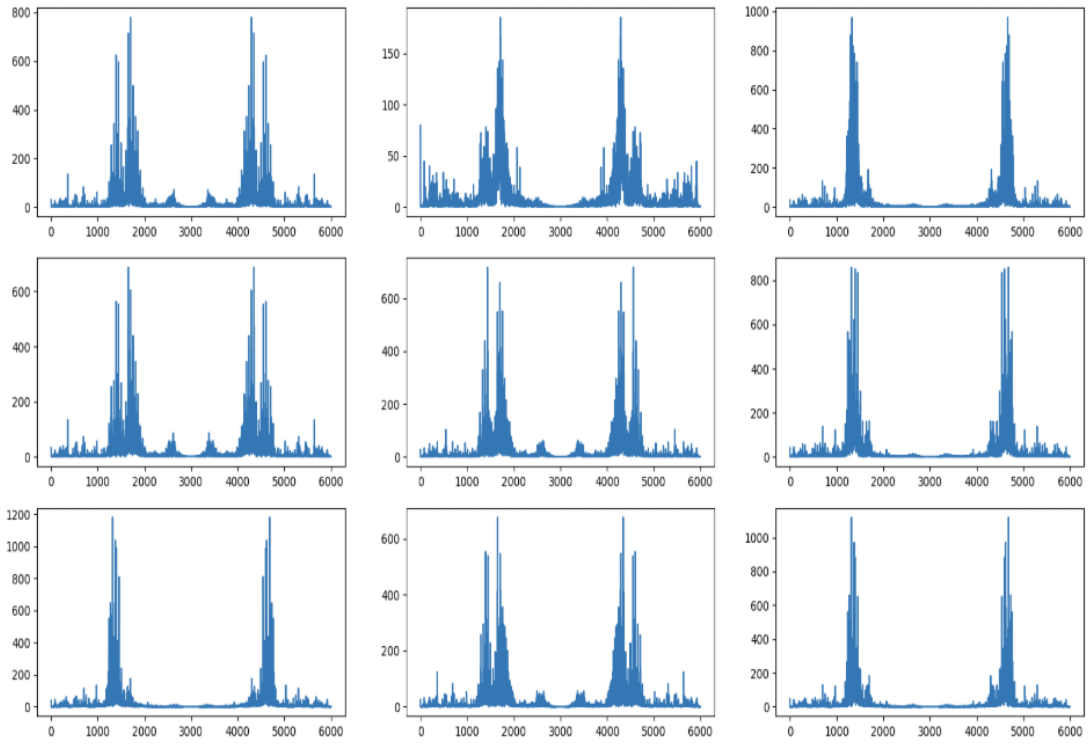


inverse Fourier transform

因此我们才发现它给出的不是随机信号，而可以解释为对同一个故障轴承的振动信号在不同时段的抽样。因为我们观察到不同样本的震动信号振幅明显不同。这也提醒我们了在读取样本后先进行一次数据的归一化，将所有数据归到（0，1）区间里再进行训练。

于是我采用了一个笨办法：把所有的样本的频域特征全给画出来，先看看我自己能不能把它分类？之后就有了下面两张中期和结题我都用到了的图：





上图为不同Label的九种信号和同一Label的九个信号。这样我就发现，通过频域分析，靠程序员自己也基本能分清不同的故障。于是最后我们放弃了复杂的小波变换，直接用快速傅里叶变换进行了分析。这两组18张图也给其他同学带去了灵感，在结课汇报时我看到其他组也有人打印出了频域的信号。

中期交流时谢春浩同学对轴承故障的分析公式给了我很多启发。在后期我逐渐对9种故障有了具象的概念，而不是把它看成人类读不懂的奇怪信号。

3.4 提交时的准确度

网络对于训练集的准确率是100但对于测试集往往会有一定误差。我们的网络就总会错那么一两个数据。5月11日第一次提交时，正确率为0.98，本来我以为这个结果已经很满意了，后来发现要按排名计算成绩！于是我在最后一天紧急重新提交了两次，每次结果准确率都略有浮动，最后终于准确率为1了，属于一个比较看运气的工作，有的组第一次提交准确率就能达到1，确实令我非常佩服！还好自己只用了3次，感觉已经远远高于我的平均运气了！

4.个人收获体会

1. 小组合作

本次小组合作非常顺利，大家积极配合，各尽所能。但在合作完成项目上我做的还不够好，因为我使用的这个库大家不一定熟悉，我就没有多花时间帮助他们上手，如果不是因为疫情我应该可以做到在完成项目的时候也教会他们用这个东西来编程。当然我的两位组员也在其他地方给了我很多帮助，所以在提交成绩分配的时候我很中肯的给了他们和我一样的成绩。

2. 课堂学习

每周五的课上时间，大家都会分享对于不同问题的见解。这是一个非常新颖的方式，一开始也非常吸引我。但是随着后来其他课业压力的加重，每周五分享会的准备时间严重不足，加上每周五要用整堂课听分享会，课下还要自己看Mooc，使得我对自己时间的分配不太从容，到后期明显有些疲惫。我还是觉得有老师带着学习比较有效率！

3. 项目开发

这个项目一开始我还是初学python的新人，如今已经可以熟练掌握python 的很多技巧了，这门语言真的很优雅！另外就是对这门课程乃至整个学科的理解。我们课上所讲的都是一些很传统的机器学习的东西，但是它们非常基础，如今让有非常多的模型建立在如SVM和感知器之类的基本结构上的，掌握这些东西对今后非常有帮助。

4. 其他体会

在其他课学枯燥知识的时候，我希望所有课都像这门课一样能够动手做项目。但真动手做项目的时候，往往因为再简单的问题卡住、以及寻找组员总令人担心不理想、以及项目占据了太多时间等等情况下而感到“啊，还是死读书考试好啊！”然而事实上在最后克服了一个又一个困难，才发现这样得到的收获远远大于死记硬背得到的收获

。