神经网络以及卷积神经网络

12330238 马毅辉

第一阶段的神经网络是最基本的,我主要分为以下几个步骤完成:

- 1 理论理解
- 2 编码
- 3 数据预处理
- 4 参数设置
- 5 矩阵实现
- 6 文件存取

1. 理论理解

在理论的方面,难理解的主要是反向传播的理解,以及公式推导。这里的推导是比较好的,附上链接:

http://blog.csdn.net/acdreamers/article/details/44657439

2. 编码

我实现的网络结构如图:

网络结构(类图)

net		
-firstnerou -col		
-h_hide	:int	
-rate -e hide <v< th=""><td>int ector> :Layer</td><td></td></v<>	int ector> :Layer	
tfilename:		
+Compute +Back(i:in	e(i:int) :int t) :void	
+Check(ti +CTest():v +Train(i;in +Save(i:in +date() :v +Load(t:in	vold t) :vold t) :bool old	

	Layer	
-numofbefore :int		
-numofsize	:int	
-output	:matrix	
-W	:matrix	
-bias	:matrix	
-residual	:matrix	
+Layer()		
+Layer(nof	before:int,nofsize);	
+SetStart()		
+SetTStart		
+GetSize()		
+GetOutPu		
+GetBias()		
+GetW():m		
+Getresidu	al:matrix();	
+Computer	(date:matrix):void	
	(date:matrix):void ofend(label:matrix,frontout:matrix):void	
+UpdateW		
+UpdateW +UpdateW	ofend(label:matrix,frontout:matrix):void	

在编码中遇到的主要问题是反向传播的残差写错,后面改正了就好了,一开始使用 for 循环写的运行比较慢,后面用到了一个 armadillo 矩阵库,效果还不错,我没用过其他的矩阵库,所以只能说比 for 循环运行快。

3 数据预处理

这一步是有必要的,有必要归一化,只要你一步一步计算就可以知道, 当输入网络的数据过大的时候,累加起来进过激活函数,网络的输出结果就 会趋于1,所以又必要归一化。

4参数设置

网络的参数设置最好不要过大,如果过大也会出现所有结果趋于一个常数的情况,虽然后面可以收敛回来,但是需要更多的时间来收敛,这是没有必要的。

5. 矩阵实现

我用到了一个名叫 armadillo 的 c++矩阵库,相对于 for 循环,其操作起来 更简洁,程序运行速度也更快。用到矩阵库的内容并不多,只需要掌握几个 基本的操作就可以了。

6.文件存取

这里那些参数我存取的是矩阵,因为 armadillo 矩阵库自带矩阵的存取, 所有非常方便,直接 matrix.save(filename)就可以存好了,并且载入只需 要 load 就行,所有非常方便。

卷积神经网络的实现

卷积神经网络就是在全连接神经网络前面再加一层卷积层。

- 1. 理论研究
- 2. 编码
- 3. 参数调整
- 4. 文件存取

1. 理论研究

这里用比较多的时间来看理论的东西,因为靠谱的文章并不多,这里我犯了一个错误,就是急于编码,然后编完后由于没有扎实的理论作为基础,将一些程序正常的情况视为 bug,调了很多并不存在的bug。浪费了大量的时间。

2. 编码

由于调试,编码编了很久。对于一些正常的现象,事前没有分析清楚导致调了很多并不存在的 bug。

3. 参数设置

最好设置参数时有正有负,如果一开始就将参数设置不正确,那么当 你调试的时候,看到结果迟迟不收敛则会很容易认为是自己的编码出 错。

4. 文件存取

也是存取的是矩阵。