

计算机科学与技术学院神经网络与深度学习课程实验报告

实验题目：numpy 和 torch 的初步体验		学号：201800130086
日期：2020-9-21	班级：18 智能班	姓名：徐鹏博
Email：hsupengbo@163.com		
实验目的： <ul style="list-style-type: none">• 理解使用 numpy 向量化的优点• 学习基础的 numpy 知识		
实验软件和硬件环境： <p>Anaconda3, JupyterNotebook Inter(R) Core(TM) i5-8250 CPU @1.60GHz 1.80GHz Win10-x64</p>		
实验原理和方法： <p>NumPy 是 Python 中科学计算的基础包。它是一个 Python 库，提供多维数组对象，各种派生对象（如掩码数组和矩阵），以及用于数组快速操作的各种 API，有包括数学、逻辑、形状操作、排序、选择、输入输出、离散傅立叶变换、基本线性代数，基本统计运算和随机模拟等等。</p> <p>PyTorch 是一个基于 python 的科学计算包，主要针对两类人群：作为 NumPy 的替代品，可以利用 GPU 的性能进行计算作为一个高灵活性、速度快的深度学习平台</p> <p>Tensor(张量) 类似于 NumPy 的 ndarray，但还可以在 GPU 上使用来加速计算。</p>		
实验步骤：（不要求罗列完整源代码） <p>先对 numpy 进行初步学习，再去写本次作业涉及到的代码：</p> <p>如</p> <pre>a = np.array([[11, 12, 13, 14, 15], [16, 17, 18, 19, 20], [21, 22, 23, 24, 25], [26, 27, 28, 29, 30], [31, 32, 33, 34, 35]])</pre> <p>几种切片操作：</p> <p>In: print(a[0, 1:4]) Out: [12 13 14]</p> <p>In: print([d[1:2] for d in a]) Out: [array([12]), array([17]), array([22]), array([27]), array([32])]</p>		

```
In: print(a[::2,::2])
```

Out:

```
[[11 13 15]
 [21 23 25]
 [31 33 35]]
```

In:

```
print(a.dtype); print(a.size); print(a.shape); print(a.itemsize)
print(a.ndim); print(a.nbytes)
```

Out:int32 25 (5, 5) 4 2 100

In:

```
a = np.arange(0, 100, 10); b = np.where(a < 50); c = np.where(a >= 50)[0]
print(b)
print(c)
```

Out: (array([0, 1, 2, 3, 4], dtype=int64),
[5 6 7 8 9])

点乘使用 numpy.dot()函数

激活函数 Relu 函数将矩阵每个元素与 0 对比, >0 保留, 小于取 0

PrimeRelu 函数则是 >0 取 1, <0 取 0, 其实 PrimeRelu 就是 Relu 的导函数。

三种切片操作根据只对最后一个维度进行截取就行。

```
def Slice_fixed_point(x,leng,st):
```

```
    return [d[st:st+leng] for d in x]
```

```
    pass
```

```
def slice_last_point(x,leng):
```

```
    return [d[-leng:] for d in x]
```

```
    pass
```

```
def slice_random_point(x,l):
```

```
    return [d[np.random.randint(0,len(d)-l):][:l] for d in x]
```

```
    pass
```

Padding 意为填充, 题中两种分别需要对称填充和填充定值。

```
def pad_pattern_end(x):
```

```
    L=max(map(len,x))
```

```
    z=[]
```

```
    for d in x:
```

```
        p=(d+(d[::-1]+d)*L)[:L]
```

```
        z.append(p)
```

```
    return z
```

```
    pass
```

```
def pad_constant_central(x,v):
```

```
    L=max(map(len,x))
```

```
    M = [v]*L
```

```
    Len = max(map(len,x))
```

```
    z=[]
```

```

for d in x:
    p=(M+d+M)[Len- np.math.ceil((Len-len(d))/2) :][:Len]
    z.append(p)
return z
pass

```

Torch 中用到的张量和矩阵类似，但又有区别。

如矩阵可以视为二阶张量，矢量视为 1 阶张量，标量视为 0 阶张量。

把矩阵转换为张量可以先创建相同大小再遍历赋值，也可以利用库函数

```

def numpy2tensor(array):
    return torch.from_numpy(array)
pass
def tensor2numpy(tensor):
    return array.numpy(tensor)
pass

```

Sum-products 函数直接使用 torch.dot 函数

```

def Tensor_Sumproducts(A,B):
    return torch.dot(A,B)
pass

```

Tensor 的 ReLu and ReLu prime 函数和之前 numpy 的类似

```

def Tensor_Rel(X):
    return torch.max(torch.zeros(Xsize()),X)
pass
def Tensor_Rel_prime():
    torch.clamp(X,min=0)*torch.reciprocal(torch.clamp(X,min=1e-6))
pass

```

结论分析与体会：

Numpy 的索引方式比较多，有花式索引、布尔屏蔽、缺省索引、Where 函数。

切片多维数组可以通过对每一维分别指定一种切片操作来实现。

基本数学函数在数组上以元素方式运行，既可以作为运算符重载，也可以作为 numpy 模块中的函数。

就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟 1—3 道问答题：
 进行数据填充的时候，刚开始不知道 Map 函数，不能直接返回数组的第一维度的每组元素个数。后面通过使用 Map 内置函数实现。