

媒体与认知 第二次作业

无 81 马啸阳 2018011054

2020 年 4 月 3 日

1 选择题

1. D

2. E, `result[0, 1, 2] = b[1] + (W[1] * x[0, :, 2*stride:2*stride + kernel_size]).sum()`

3. A

4. D

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial y_1} \frac{\partial y_1}{\partial w_2} + \frac{\partial L}{\partial y_2} \frac{\partial y_2}{\partial w_2} + \frac{\partial L}{\partial y_3} \frac{\partial y_3}{\partial w_2} = \frac{\partial L}{\partial y_1} x_2 + \frac{\partial L}{\partial y_2} x_3 + \frac{\partial L}{\partial y_3} x_4$$
$$\frac{\partial L}{\partial x_2} = \frac{\partial L}{\partial y_1} \frac{\partial y_1}{\partial x_2} + \frac{\partial L}{\partial y_2} \frac{\partial y_2}{\partial x_2} + \frac{\partial L}{\partial y_3} \frac{\partial y_3}{\partial x_2} = \frac{\partial L}{\partial y_1} w_2 + \frac{\partial L}{\partial y_2} w_1$$

5. B

2 实验结果

自动评判程序的结果如图 1 所示。

3 实验总结

本次实验较为基础简单，实现简单的一维卷积神经网络。其中卷积操作是通过多层感知机共享权值实现的。实验中主要部分在于计算各层的输出维度，以及卷积网络的反向传播。

值得注意的是 Sigmoid 函数的数值稳定写法。我原先采用的写法为：`self.state = np.where(x > 0, 1.0 / (1.0 + np.exp(-x)), np.exp(x) / (1.0 + np.exp(x)))`。这种写法在 numpy 1.16 版本下可以正确运行不溢出，但最新版本的 numpy 中，`numpy.where` 会先计算最后两个 array，使

```

-----
Step I - Convolutional Layer
Step I - Forward
Conv1D Forward: PASS
Step I - Backward
Conv1D dX: PASS
Conv1D dW: PASS
Conv1D db: PASS
-----

Step II - CNN as a Simple Scanning MLP
Scanning MLP: PASS
-----

Step II - CNN as a Distributed Scanning MLP
Distributed MLP: PASS
-----

Step III - CNN Complete Model
Conv1D Model Forward: PASS
Conv1D Model dX: PASS
Conv1D Model dW: PASS
Conv1D Model db: PASS

```

图 1: 自动评判程序运行结果

得计算时先产生溢出，但不影响最后结果的正确性。为了修复这个问题，我后来采用掩码的方式解决了这一问题，也即：

```

self.state = np.zeros_like(x)
self.state[x >= 0] = 1.0 / (1.0 + np.exp(-x[x >= 0]))
self.state[x < 0] = np.exp(x[x < 0]) / (1.0 + np.exp(x[x < 0]))

```

另外，现在的打包 shell 脚本缺失 exclude.txt，会产生错误，希望能修复这一问题。