

AI 引论第九次大班课作业

信息科学技术学院 施朱鸣 1800011723

2020 年 4 月 11 日

第 1 题

进行 1 个单位宽度的 0-边界填充后的图像如下

0	0	0	0	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	2	2	0	0
0	1	1	1	1	0
0	0	0	0	0	0

以 1 个单元为步长，对每个点做卷积

$$y = \sum_i x_i \mathbf{w}_i$$

得到如下结果，即为 I' 各个位置对应的像素值

-2	-2	2	2
-3	-3	3	3
-3	-2	2	3
-1	0	0	1

第 2 题

对于每个像素点，其 $\hat{I} - I'$ 对应的矩阵如下

0	0	0	0
-1	-1	1	1
-1	-1	1	1
-2	-2	2	2

用 \mathcal{L} 表示均方差损失函数，则

$$\begin{aligned}
\mathcal{L} &= \frac{1}{16} \sum_{(m,n)} (\hat{x}_{m,n} - x'_{m,n})^2 \\
&= \frac{1}{16} (0^2 + 0^2 + 0^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 2^2 + 2^2 + 2^2 + 2^2) \\
&= \frac{3}{2} \\
&= 1.5
\end{aligned} \tag{1}$$

假设 $K_{i,j}$ 中的 i, j 都从 0 开始取值，则

$$\mathcal{L} = \frac{1}{16} \sum_{1 \leq m, n \leq 4} (\hat{x}_{m,n} - x'_{m,n})^2 \tag{2}$$

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial K_{i,j}} &= \frac{1}{16} \sum_{1 \leq m, n \leq 4} \frac{\partial (\hat{x}_{m,n} - x'_{m,n})^2}{\partial K_{i,j}} \\
&= \frac{1}{16} \times 2 \sum_{1 \leq m, n \leq 4} \frac{\partial (\hat{x}_{m,n} - x'_{m,n})^2}{\partial (\hat{x}_{m,n} - x'_{m,n})} \frac{\partial (\hat{x}_{m,n} - x'_{m,n})}{\partial K_{i,j}} \\
&= \frac{1}{8} \sum_{1 \leq m, n \leq 4} (\hat{x}_{m,n} - x'_{m,n}) \times (-1) \frac{\partial x'_{m,n}}{\partial K_{i,j}} \\
&= \frac{1}{8} \sum_{1 \leq m, n \leq 4} (\hat{x}_{m,n} - x'_{m,n}) \times (-1) \frac{\partial \sum_{0 \leq p, q \leq 2} x_{m-1+p, m-1+q} K_{p,q}}{\partial K_{i,j}} \\
&= \frac{1}{8} \sum_{1 \leq m, n \leq 4} (\hat{x}_{m,n} - x'_{m,n}) \times (-1) \times x_{m-1+i, m-1+j}
\end{aligned} \tag{3}$$

用上面的计算方法得出卷积核各点的 $\frac{\partial \mathcal{L}}{\partial K_{i,j}}$ 如下矩阵

-1.5	0.0	1.5
-1.0	0.0	1.0
-0.625	0.0	0.625

第3题

对于已经求得的梯度矩阵，利用如下式子更新卷积核参数

$$\hat{K}_{i,j} = K_{i,j} - \eta \frac{\partial \mathcal{L}}{\partial K_{i,j}}$$

当 $\eta = 0.01$ 时更新后的卷积核参数矩阵如下

0.015	0	-0.015
1.01	0	-1.01
1.00625	0	-1.00625

再次卷积得到的输出结果如下矩阵

-2.01625	-2.01625	2.01625	2.01625
-3.0375	-3.0375	3.0375	3.0375
-3.04125	-2.035	2.035	3.04125
-1.04	-0.03	0.03	1.04

用 $\hat{\mathcal{L}}$ 表示更新后的均方差损失函数，则

$$\begin{aligned}\hat{\mathcal{L}} &= \sum_{(m,n)} (\hat{x}_{m,n} - x'_{m,n})^2 \\ &= 1.428\end{aligned}\tag{4}$$

附：实现以上计算的 Python 代码

```
1 # To add a new cell , type '# %%'
2 # To add a new markdown cell , type '# %% [markdown] '
3 # %%
4 kernel = [
5     [0, 0, 0],
6     [1, 0, -1],
7     [1, 0, -1]]
8
9 # %%
10 I = [
11     [0, 0, 0, 0, 0, 0],
```

```

12         [0, 0, 1, 1, 0, 0],
13         [0, 0, 1, 1, 0, 0],
14         [0, 0, 2, 2, 0, 0],
15         [0, 1, 1, 1, 1, 0],
16         [0, 0, 0, 0, 0, 0]]
17
18 # %%
19 targetI = [
20     [-2, -2, 2, 2],
21     [-4, -4, 4, 4],
22     [-4, -3, 3, 4],
23     [-3, -2, 2, 3]]
24
25 # %%
26 outputI = [
27     [0, 0, 0, 0],
28     [0, 0, 0, 0],
29     [0, 0, 0, 0],
30     [0, 0, 0, 0]]
31
32 # %%
33 grad = [
34     [0, 0, 0],
35     [0, 0, 0],
36     [0, 0, 0]]
37
38 # %%
39 # 卷积
40 for i in range(1, 5):
41     for j in range(1, 5):
42         outputI[i-1][j-1] = 0
43         for k in range(3):
44             for l in range(3):

```

```

45         outputI[i-1][j-1] += I[i-1+k][j-1+l] * kernel[k][l]
46     print(outputI)
47
48     # %%
49     # 计算均方差
50     MSE = 0
51     for i in range(4):
52         for j in range(4):
53             MSE += (targetI[i][j] - outputI[i][j])**2
54     MSE = float(MSE)/16.0
55     print(MSE)
56
57     # %%
58     for i in range(3):
59         for j in range(3):
60             grad[i][j] = 0
61             for k in range(4):
62                 for l in range(4):
63                     grad[i][j] += ((targetI[k][l] - outputI[k][l]))*(-1)*(I
64     print(grad)
65
66     # %%
67     eta = 0.01
68     for i in range(3):
69         for j in range(3):
70             kernel[i][j] -= eta * grad[i][j]
71     print(kernel)
72
73     # %%
74     for i in range(1, 5):
75         for j in range(1, 5):
76             outputI[i-1][j-1] = 0
77             for k in range(3):

```

```

78         for l in range(3):
79             outputI[i-1][j-1] += I[i-1+k][j-1+l] * kernel[k][l]
80     print(outputI)
81
82     # %%
83     MSE = 0
84     for i in range(4):
85         for j in range(4):
86             MSE += (targetI[i][j] - outputI[i][j])**2
87     MSE = float(MSE)/16.0
88     print(MSE)

```