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

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

假设 $K_{i,j}$ 中的 i,j 都从 0 开始取值，则

$$
\mathcal{L} = \sum_{1 \le m,n \le 4} \left(\hat{x}_{m,n} - {x'}_{m,n}\right)^2
\tag{2}
$$

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

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

-12	0	12
-8	0	8
-5	0	5

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

0.12	0	-0.12
1.08	0	-1.08
1.05	0	-1.05

再次卷积得到的输出结果如下矩阵

-2.13	-2.13	2.13	2.13
-3.30	-3.30	3.30	3.30
-3.33	-2.28	2.28	3.33
-1.32	-0.24	0.24	1.32

用 $\hat{\mathcal{L}}$ 表示更新后的均方差损失函数，则

$$\hat{\mathcal{L}} = \sum_{(m,n)} \left(\hat{x}_{m,n} - x'_{m,n} \right)^2 \tag{4}$$
$$= 0.988$$

附：实现以上计算的 Python 代码

```python
# To add a new cell, type '# %%'
# To add a new markdown cell, type '# %% [markdown]'
# %%
kernel = [
        [0, 0, 0],
        [1, 0, -1],
        [1, 0, -1]]


# %%
I = [
        [0, 0, 0, 0, 0, 0],
```

```python
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):
```

```python
                        outputI[i-1][j-1] += I[i-1+k][j-1+l] * kernel[k][l]
print(outputI)


# %%
# 计算均方差
MSE = 0
for i in range(4):
    for j in range(4):
        MSE += (targetI[i][j] - outputI[i][j])**2
MSE = float(MSE)/16.0
print(MSE)


# %%
for i in range(3):
    for j in range(3):
        grad[i][j] = 0
        for k in range(4):
            for l in range(4):
                grad[i][j] += (targetI[k][l] - outputI[k][l])*(-1)*(I[
print(grad)


# %%
eta = 0.01
for i in range(3):
    for j in range(3):
        kernel[i][j] -= eta * grad[i][j]
print(kernel)


# %%
for i in range(1, 5):
    for j in range(1, 5):
        outputI[i-1][j-1] = 0
        for k in range(3):
```

```python
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)
```