

به نام خدا

گزارش پروژه سیگنال و سیستم

بخش پردازش تصویر:

پیاده سازی کانولوشن

```
import numpy as np
import cv2

def apply_convolution(image, kernel, stride, padding):
    inplen = len(image[0])
    kernelLen = len(kernel)
    outlen = int((inplen-kernelLen)/stride + 1)

    output = np.zeros((outlen, outlen))

    for k in range(3):
        for i1 in range(outlen):
            if(i1*stride+kernelLen > inplen):
                break
            for j1 in range(outlen):
                if(j1*stride+kernelLen > inplen):
                    break
                sum = 0
                for i2 in range(kernelLen):
                    for j2 in range(kernelLen):
                        sum += float(image[k][i1*stride + i2][j1*stride + j2]) * (kernel[i2][j2])
                if(sum<0):
                    sum = 0
                output[i1][j1] += sum

    return output
```

همانطور که در ویدیو آموزشی توضیح داده شد تابع کانولوشن به این صورت پیاده سازی شد که یک کرنل روی آرایه عکس مورد نظر ما پیمایش میکند و ضرایب کرنل را در ضرایب عکس ضرب میکنم و خروجی را در یک آرایه دو بعدی جدید میریزد و کانولوشن ما روی سه کanal R G B انجام میشود و مقادیر آرایه های آنها باهم جمع میشود و اینگونه کانولوشن ار فیلتر گرفته میشود

حلقه اول برای هر کانال ما هست
حلقه دوم و سوم برای تعیین طول و عرض کرنل و دو حلقه بعدی برای ضرب

درایه های کرنل با درایه های متناظر رو ورودی اصلی ما است که در آخر بعد محاسبه مقدار آن روی یک آرایه خروجی ریخته میشود و خروجی هر سه کanal باهم جمع میشود

ماکس پولینگ:

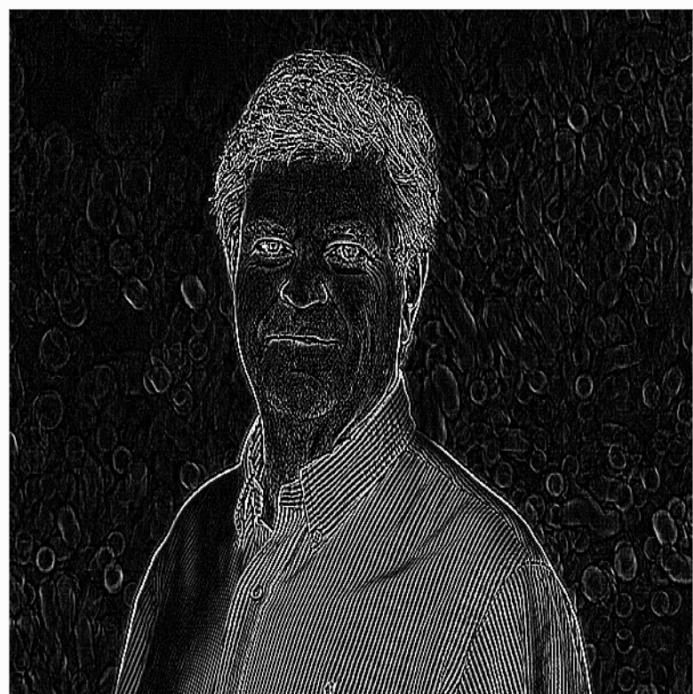
```
7 def max_pooling(image, size, stride):
8     inpLen = len(image[0])
9     outLen = int((inpLen-size)/stride + 1)
10    output = np.zeros((2, outLen, outLen))
11    print(len(output))
12
13    for k in range(2):
14        channel = np.zeros((outLen, outLen))
15        for i1 in range(outLen):
16            if(i1*stride+size > inpLen):
17                break
18            for j1 in range(outLen):
19                if(j1*stride+size > inpLen):
20                    break
21                max = -10000
22                for i2 in range(size):
23                    for j2 in range(size):
24                        if(image[k][i1*stride + i2][j1*stride + j2] > max):
25                            max = image[k][i1*stride + i2][j1*stride + j2]
26                channel[i1][j1] = max
27        output[k] = channel
28
29    return output
30
31
```

ماکس پولینگ هم مانند کانولوشن یک چیزی مانند کرنل روی آرایه ما حرکت میکند ولی مقدار ندارد فقط روی اون کرنلی که حرکت میکند بزرگ ترین مقدار درایه را پیدا میکند و روی یک آرایه دیگر می ریزد
حلقه اول برای پیمایش کanal های ورودی هست
حلقه دوم و سوم برای طول و عرض قاب یا کرنل هست
و دو حلقه بعدی برای پیدا کردن ماکسیمم در اون قاب یا کرنل هست

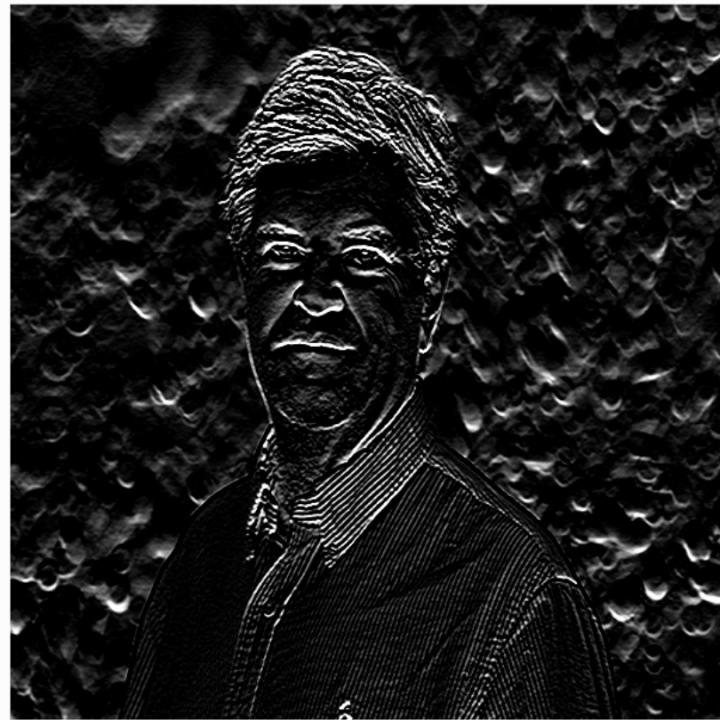
فراخوانی توابع:

```
◆ ImageProcessing.py > ...
05
06     cv2.imwrite('image.jpg', image_array)
07
08 Edge_Enhancement = [[-1,-1,-1],[-1,8,-1],[-1,-1,-1]]
09 Horizontal_Edge_Detection = [[1,1,1],[0,0,0],[-1,-1,-1]]
10 Vertical_Edge_Detection = [[1,0,-1],[1,0,-1],[1,0,-1]]
11
12 first_feature_map = []
13
14 Gaussian_Blu...
15 Large_Edge_Enhancement = [[0,0,-1,0,0],[0,0,-1,0,0],[-1,-1,8,-1,-1],[0,0,-1,0,0],[0,0,-1,0,0]]
16
17 out_Edge_Enhancement = apply_convolution(mainImage, Edge_Enhancement, 1, 1)
18 out_Horizontal_Edge_Detection = apply_convolution(mainImage, Horizontal_Edge_Detection, 1, 1)
19 out_Vertical_Edge_Detection = apply_convolution(mainImage, Vertical_Edge_Detection, 1, 1)
20
21 first_feature_map.append(out_Edge_Enhancement)
22 first_feature_map.append(out_Horizontal_Edge_Detection)
23 first_feature_map.append(out_Vertical_Edge_Detection)
24
25
26 out_Gaussian_Blu...
27 out_Large_Edge_Enhancement = apply_convolution(first_feature_map, Large_Edge_Enhancement, 1, 1)
28
29 second_feature_map = []
30
31 second_feature_map.append(out_Gaussian_Blu...
32 second_feature_map.append(out_Large_Edge_Enhancement)
33
34 final_output = max_pooling(second_feature_map,3,1)
35
36
37 cv2.imwrite('Edge_Enhancement.jpg', out_Edge_Enhancement)
38 cv2.imwrite('Horizontal_Edge_Detection.jpg', out_Horizontal_Edge_Detection)
39 cv2.imwrite('Vertical_Edge_Detection.jpg', out_Vertical_Edge_Detection)
40
41 cv2.imwrite('Gaussian_Blu...
42 cv2.imwrite('Large_Edge_Enhancement.jpg', out_Large_Edge_Enhancement)
43
44 cv2.imwrite('final1.jpg', final_output[0])
45 cv2.imwrite('final2.jpg', final_output[1])
46 |
```

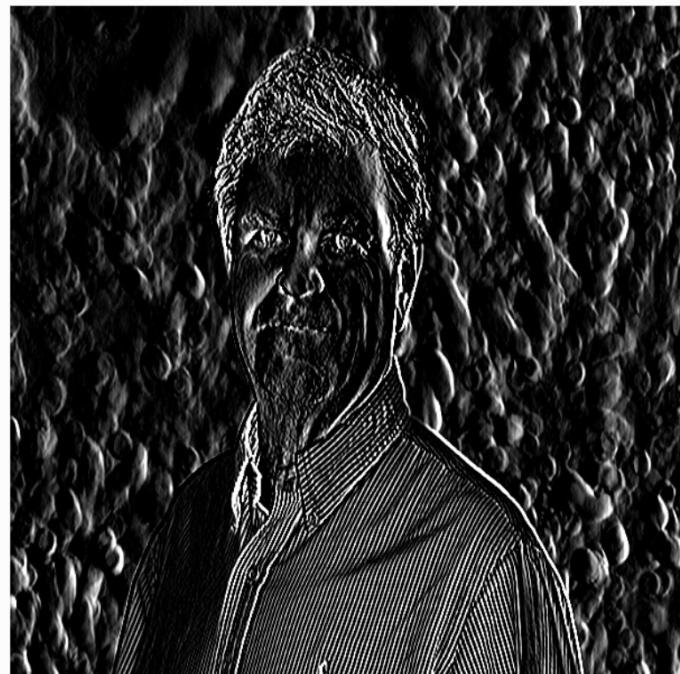
خروجی ها



خروجی با فیلتر Edge Enhancement



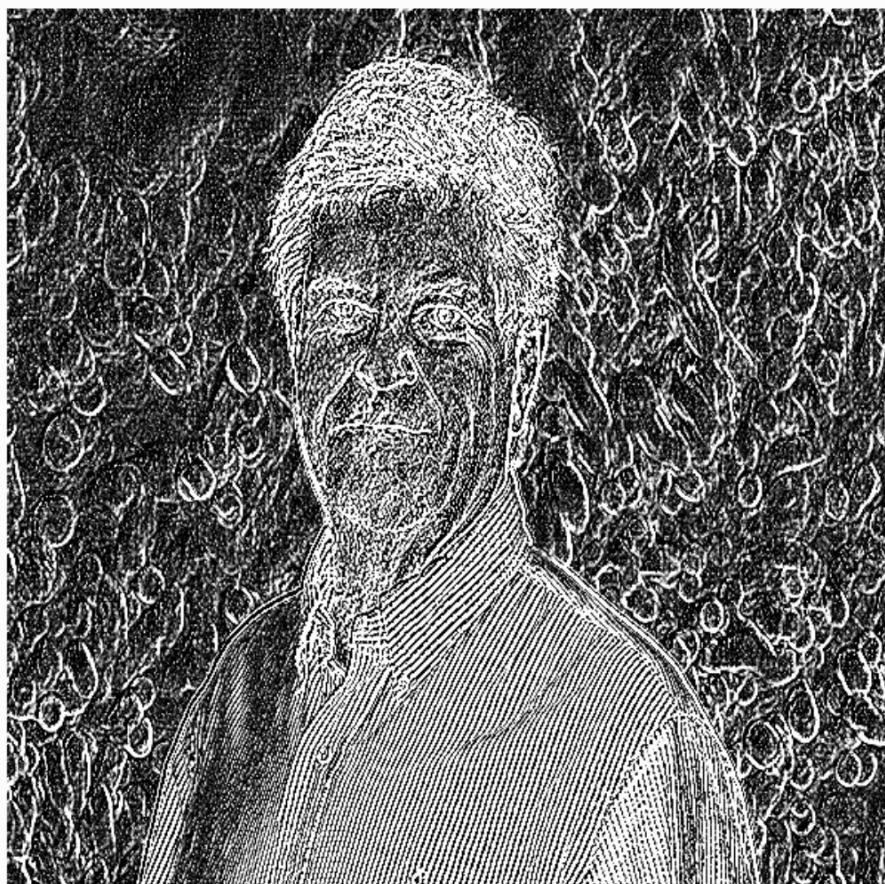
خروجی با فیلتر Horizontol Edge Detection



خروجی با فیلتر Vertical Edge Detection



خروجی کانولوشن فیچرمپ لایه اول با فیلتر Gaussian Blur



خروجی کانولوشن فیچرمپ لایه اول با فیلتر Large Edge Enhancenemt



خروجی ماکس پولینگ شده Gaussian Blur



خروجی ماکس پولینگ شده ای Large Edge Enhancement