گزارش تمرین چهارم بینایی ماشین

نام تهیه‌کننده: ملیکا نوبختیان

شماره دانشجویی: ۹۷۵۲۲۰۹۴

نسخه: ۱

۱- سوال اول

۲۲ خرداد
12 یکشنبه
Sun.
Jun. 2022
۱۲ ذی القعده ۱۴۴۳

2	3
1	4

$u=0 \quad v=0 \quad f(x,y)$

$u=1 \quad v=0 \quad f(x,y)\, e^{-\pi j x}$

$u=0 \quad v=1 \quad f(x,y)\, e^{-\pi j y}$

$u=1 \quad v=1 \quad f(x,y)\, e^{-2\pi j \left(\frac{x}{2}, \frac{y}{2}\right)}$

$$\Rightarrow F(u=0, v=0) = \frac{10}{4} = 2,5$$

$$F(u=1, v=0) = \frac{1}{4} \times 5\, e^{-\pi j} = 1,25$$

$$F(u=0, v=1) = \frac{1}{4} \times 7\, e^{-\pi j} = 1,75$$

$$F(u=1, v=1) = \frac{1}{4} \times \left(4 e^{-\pi j} + 4 e^{-2\pi j}\right)$$

$$= 1 + j$$

۲- سوال دوم

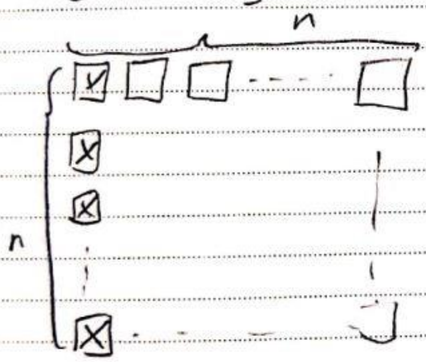
-۲

الف) فرمول تبدیل فوریه برای تصویر n×n به شکل زیر است:

$$F(u,v) = \frac{1}{n^2}\sum_{x=0}^{n-1}\sum_{y=0}^{n-1} f(x,y)\, e^{+2j\pi\left(\frac{ux+vy}{n}\right)}$$

$$e^{\frac{2j\pi xu}{n}} \times e^{\frac{2j\pi vy}{n}}$$

با توجه این قسمت می‌توانیم ببینیم که با داشتن مقادیر x از ۰ تا n-1 و y از ۰ تا n-1 بقیه عنصرها را می‌توانیم از ضرب این موارد به دست آوریم و عناصر مشخص ماتریس ما به شکل زیر خواهند بود.



که در این صورت تعداد مقادیر آزاد ما برابر 2n-1 خواهد بود

ب) در u=0, v=0 تبدیل فوریه به شکل زیر است:

$$F(0,0) = \frac{1}{n^2}\sum_{x=0}^{n-1}\sum_{y=0}^{n-1} f(x,y) = \frac{\sum f(x,y)}{n^2}$$

مشاهده می‌شود که فرمول بالا در واقع نشان‌گر میانگین تصویر است پس F(0,0) میانگین تصویر را نشان خواهد داد.

۳- سوال سوم

در بخش اول برای اینکه بتوانیم یک فیلتر را روی تصویر موردنظر اعمال کنیم، باید اول مقدار padding موردنیاز در عرض و ارتفاع تصویر را انجام دهیم. چون فیلترمان هر بار به اندازه یک حرکت می‌دهیم چه سطری و چه ستونی، مقدار padding کلی لازم برای تصویر، برابر طول فیلتر منهای یک و عرض فیلتر منهای یک خواهد بود. اگر طول یا عرض فیلتر برابر ۱ باشد، دیگر در آن بعد نیاز به padding نخواهیم داشت:

```
padding_height = max((filter_h - 1), 0)
padding_width = max((filter_w - 1), 0)
```

برای اینکه مقدار padding در چپ و راست و پایین و بالا تصویر را به دست آوریم به این صورت عمل می‌کنیم که مقدار padding کلی را تقسیم بر دو می‌کنیم و با این مقدار در یک سمت padding خواهیم داشت و مقدار padding کلی منهای این مقدار، مقدار padding در سمت دیگر خواهد بود:

```
pad_top = padding_height // 2
pad_bottom = padding_height - pad_top
pad_left = padding_width // 2
pad_right = padding_width - pad_left
```

حالا با استفاده از این مقادیر padding را روی تصویر اعمال خواهیم کرد:

```
padded_image = np.zeros((img_h + padding_height, img_w + padding_width))
padded_image[pad_top:-pad_bottom, pad_left:-pad_right] = image
```

حالا می‌توانیم فیلتر را روی تصویر اعمال کنیم و با پیمایش روی پیکسل‌ها و ضرب کردن فیلتر با قسمت موردنظر تصویر به نتیجه نهایی خواهیم رسید:

```
for x in range(img_w):
    for y in range(img_h):
        result[y, x] = (kernel * padded_image[y :y  + filter_h, x :x + fil
ter_w]).sum()
```

در بخش بعد برای ساختن فیلتر میانگین‌گیر با ابعاد داده‌شده ابتدا یک فیلتر مربعی با اعضا برابر یک می‌سازیم. چون فیلتر میانگین‌گیر است و باید مجموع برابر یک باشد، همه اعضای ماتریکس را بر سایز آن تقسیم خواهیم کرد و در نهایت به فیلتر موردنظر خواهیم رسید:

```
result = np.ones((size, size))
kernel_count = size * size
result = result / kernel_count
```

تصویر اصلی بدون اعمال فیلتر به شکل زیر خواهد بود:

و با اعمال فیلتر میانگین گیر به شکل زیر خواهد شد:

پیاده‌سازی فیلتر میانه‌گیر مانند حالت قبل خواهد بود با این تغییر که در این حالت هنگام اعمال فیلتر به قسمت موردنظر تصویر، میانه آن قسمت از تصویر را پیدا می‌کنیم و آن مقدار را به عنوان مقدار جدید جایگزین می‌کنیم:

```
for x in range(img_w):
        for y in range(img_h):
            median = np.median(padded_image[y :y  + size, x :x + size])
            result[y, x] = median
```

تصویر قبل از اعمال فیلتر میانه‌گیر به شکل زیر است:

و پس از اعمال فیلتر میانه‌گیر نتیجه به صورت زیر خواهد بود:

در اینجا می‌بینیم که فیلتر میانه‌گیر نتیجه بهتری روی نویز نمک و فلفل داشته‌است. نویز نمک و فلفل از نوع جمع‌شونده نیست و به همه پیکسل‌ها یک مقدار اضافه نمی‌کند بلکه صرفا روی بعضی از پیکسل‌ها اعمال می‌شود. از این رو فیلتر میانگین‌گیر نتیجه خوبی روی این تصاویر نخواهد داشت زیرا در هر قسمت از تصویر به سراغ میانگین گرفتن می‌رود در حالی که در تصاویری که نویز نمک و فلفل داریم این رویه درست نیست. برای مثال در قسمتی از تصویر که همه پیکسل‌ها صفر بوده‌اند و یک نویز ۲۵۵ اضافه شده‌است میانگین‌گرفتن باعث خرابی پیکسل‌ها و عوض شدن مقدار خواهد شد در حالی که میانه‌گیری ما را به مقدار درست خواهد رساند.

اما برعکس این فیلتر روی تصاویر با نویز جمع شونده عملکرد خوبی نخواهد داشت.

در بخش بعدی برای محاسبه مشتق عمودی یا افقی لازم است یک فیلتر روی تصویر اعمال کنیم. من فیلتر افقی را روی تصویر اعمال کردم و به همین دلیل در فیلتر تنها $x - 1$ و $x + 1$ مقدار خواهند داشت که برابر ضرایبشان در فرمول مشتق خواهد بود:

```
derivative_kernel = np.array([
    [0, 0, 0],
    [0, 0, 0],
    [0, 0, 0],
], dtype='float64')

derivative_kernel[1, 0] , derivative_kernel[1, 2] = -1 / 2 , 1 / 2
```

نتیجه اعمال آن بر تصویر به صورت زیر خواهد بود:

منبع:

https://mmuratarat.github.io/2019-01-17/implementing-padding-schemes-of-tensorflow-in-python