**بسمه تعالی**
**دانشگاه صنعتی شریف**
**دانشکده علوم کامپیوتر**
**Image processing – دکتر مصطفی کمالی**
**HW4 q3 report**

## سوال 3 – SLIC oversegmentation:

در ابتدا عکس را لود کرده و اولا یک ورژن در color space lab از آن درست میکنیم ، سپس آنرا به gray space برده و یک بار در جهت x و بار دیگر در جهت y از آن مشتق میگیریم ( با توجه به کرنلی که در کد آمده ) سپس این دو ماتریس را به توان دو رسانده و جمع میکنیم ولی sqrt نمیگیریم زیرا میخواهیم صرفا بعدا مقادیر درایه ها را با هم مقایسه کنیم و sqrt عملا کار اضافه ایست.

فقط برای یک K (k=64) توضیح میدهیم و بقیه مثل همین است که با یک for انجام شده )

حال میخواهیم cluster center های اولیه را تعریف کنیم ، میدانیم میخواهیم تصویر را به k قسمت تقسیم کنیم ، لذا مساحت تصویر را تقسیم بر k میکنیم که به ما S را میدهد که برابر فاصله دو مرکز خوشه مجاور است . حال یک آرایه برای X نقاط و یک آرایه برای Y نقاط میسازیم به اینصورت که X از مقدار S//2 تا H با قدم های S تایی و Y هم از مقدار S//2 تا W با قدم های S تایی میرود ، سپس با استفاده از np.meshgrid و دو آرایه X,Y نقاط بدست می آیند .

سپس میخواهیم محل نقاط را در یک پنجره 5×5 طوری تغییر دهیم که در کمترین گرادیان قرار گیرد ، در این پنجره ممکن است چندین محل با گرادیان کمینه باشد ، برای همین نقطه ی کمینه ای که در local minima این پنجره قرار دارد را انتخاب میکنیم ، به اینصورت که مجاور های این کاندید را با هم جمع کرده و تقسیم بر تعداد میکنیم ، حال کاندیدی که این مقدار برای آن کمتر از بقیه است به عنوان مرکز جدید انتخاب میشود ، این روش نتیجه ای اندکی بهتر از انتخاب رندم بین کاندید های مجاز میدهد .

در قدم بعدی میخواهیم برای نقاط feature vector تعریف کنیم ، به اینصورت که برای هر نقطه یک بردار 5 تایی داریم که 3 مولفه اول، رنگ آن در فضای LAB و دو مولفه دیگر x,y آن در تصویر اولیه هستند ، لذا در نهایت به یک ماتریس با بعد (H,W,5) میرسیم که feature vector پیکسل (i,j) برابر feature[i,j,:] است .

همچنین یک ماتریس dist_mat به اندازه عکس در ابتدا تعریف میکنیم که درایه های آن در ابتدا مقادیر بسیار بزرگ دارند همچنین یک ماتریس label هم به اندازه عکس داریم که label پیکسل مربوطه را نشان میدهد که در ابتدا مقدار 1- دارند .

در این مرحله ، کل کارهای زیر در یک while تکرار میشود تا زمانی که متغیر flag در ابتدای while برابر true قرار داده میشود ، تا آخر iteration از while ، trueبماند و گرنه حلقه ادامه میابد .

*Start of while*

حال برای هر مرکز خوشه یک همسایگی به اندازه 2S×2S حول آن در نظر میگیریم ، به صورت زیر :

```
pnts=feature[x_range[0]:x_range[1],y_range[0]:y_range[1],:]
```

که y_range , x_range از مقدار S قبل و بعد مختصات مرکز را نشان میدهند به اینصورت که به این شرط هم در نظر گرفته میشود که از مرزهای عکس خارج نشویم .

سپس فاصله مقدار lab برای هر پیکسل داخل pnts و مرکز خوشه اندازه گرفته میشود که ماتریس $D_{lab}$ میشود ، همچنین اختلاف مختصات نقاط داخل pnts و مرکز خوشه هم در ماتریس $D_{xy}$ ذخیره میشود ، سپس این دو ماتریس را با هم جمع میکنیم به ولی ابتدا ماتریس $D_{xy}$ در یک $\alpha$ ضرب میشود که مقدار آن را برابر با 0.25 در نظر گرفتیم که حاصل را D مینامیم . سپس در ماتریس distance بررسی میشود که در این پنجره نقاطی که D بدست آمده آنها از مقدار موجود در distance کمتر باشد ، مقدار D آنها در ماتریس dist_mat بروز میشود و label متناظر هم برابر با مرکز خوشه فعلی قرار داده مشود ، برای افزایش سرعت محاسبات کلا به صورت ماتریسی انجام شده لذا برای اینکه بدانیم کدام نقاط نیاز به بروز رسانی دارند از mask استفاده کردیم که جزییات پیاده سازی در کد مشخص است .

حال باید مرکز خوشه را با توجه به نقاط جدید که برای آن بدست آمده بروز کنیم ، به اینصورت که در در پنجره حول مرکز خوشه که برابر با pnts بود ، میانگین نقاطی که label آنها برابر با همین مرکز خوشه شد را بدست میاوریم که یک بردار به نام delta به ما میدهد ؛ حال جمع delta با مرکز خوشه ، مرکز بروز شده را به ما میدهد . حال اگر تفاضل نسبی مرکز جدید و قبلی بزرگتر از 0.001% باشد ، مرکز بروز شده و flag هم false میشود

تنها زمانی flag برابر True میماند و از while خارج میشویم که مرکز هیچکدام از خوشه ها جابجایی بیش از 0.001% نداشته باشد . در اینصورت خوشه ها همگرا شده و label ها بدست آمده اند .

*End of while*

در قدم آخر می خواهیم که نواحی را اجبار کنیم که به هم متصل باشند یا به عبارتی گام enforce connectivity در مرحله آخر الگوریتم ، این کار را با استفاده از تابع skimage.morphology.closing و با کرنل مربعی 30×30 انجام میدهیم .

سپس میخواهیم مرزها را در عکس اصلی رسم کنیم ، برای اینکار از ماتریس label بدست آمده لاپلاسین میگیریم و به بازه 0 تا 255 میبریم و محل هایی که مقدار بزرگتر از 1 دارند را به عنوان مرز انتخاب میکنیم ، میتوان مقادیر دیگری به جای یک داشت که باعث کمرنگ یا پررنگ شدن مرزهای نهایی میشوند که حاصل این یک ماسک به نام boundaries است که در محل هایی که مرز ناحیه ها هستند مقدار 1 و بقیه جاها 0 است .

سپس این ماسک را not میکنیم که در مرز ها صفر شود و بقیه جاها یک و سپس در عکس اصلی به صورت pairwise ضرب میکنیم که باعث میشود حاصل در مرز ها مقدار صفر پیدا کند ، سپس نتیجه را ذخیره میکنیم.