به نام خدا

تمرین سری ششم - درس مبانی بینایی کامپیوتر

سید محمد علی نخاری - شماره دانشجویی: 99521496

سوال پنجم)

در ابتدا باید دیتاست گفته شده را دانلود کرده و در مسیر مورد نظر در colab آن را unzip و ذخیره کرد. این سوال را من تا بخش تعریف مدل U-Net کامل کردم و به صورت کامل پیاده سازی نشده است.

در تابع dataframe_creation با گرفتن مسیر فایل و یک نام، بر روی فایل های موجود در آن دایرکتوری پیمایش کرده و مسیر تمامی تصاویر موجود را درون لیست image_paths میریزیم. علاوه بر این باید برای هر مسیر یک آیدی هم نسبت دهیم که این آیدی همان نام تصاویر است. فقط برای تصاویر ماسک که پسوند _label در نام آنها موجود است باید برای ذخیره کردن آیدی نام _label را از آنها حذف کنیم. تابع بعدی که باید آن را پیاده سازی کنیم تابع display میباشد. این تابع با گرفتن لیستی از عکس ها به صورت tensor ابتدا آنها را به تصویر تبدیل کرده و سپس با numpy آنها را به یک آرایه تبدیل میکنیم تا با plt بتوان نمایش شان داد.

در ادامه در مسیر colab دو پوشه با نام های train, train_mask ساخته تا تصاویر آموزشی به همراه ماسک هایشان را در این دو پوشه با فرمت ,png و اندازه های مشخص ذخیره کنیم.

برای این کار باید ابتدا مسیر تصاویر موجود در پوشه دیتاست را درون آرایه هایی جداگانه بریزیم تا بتوان آنها را خواند و در مسیر دلخواه ذخیره کرد. فایل هایی که پسوند آنها .bmp و در نام آنها _lab وجود ندارد را درون آرایه images و تصاویری که پسوند آنها .bmp و در نام آنها _label وجود دارد را درون آرایه labels میریزیم. پس از این کار بر روی هر دو آرایه پیمایش کرده و تک تک فایل ها را خوانده و پسوندشان را به .png تغییر میدهیم و در مسیر مورد نظر ذخیره میکنیم.

در گام بعدی، یک دیتافریم برای تصاویر موجود در فایل train و یک دیتافریم برای تصاویر موجود در پوشه train_mask ساخته و مسیر های آنها به همراه آیدی را ذخیره میکنیم. باید توجه

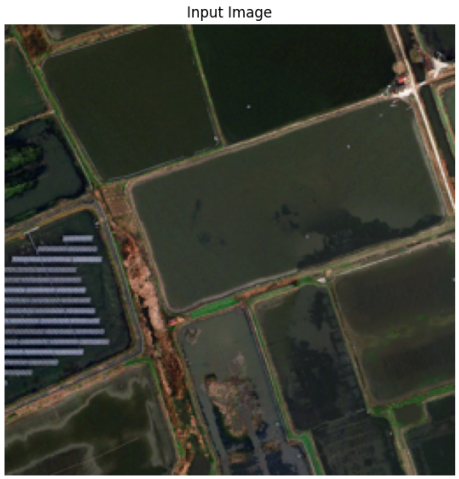
داشت برای اینکه هرعکس به صورت مناسبی به تصویر ماسک خودش نسبت داده شود باید آنها را بر اساس ستون آیدی هم مرتب کرد(هر دو دیتافریم). پس از این کار درون دیتافریمی که مربوط به تصاویر آموزشی است یک ستون با نام mask_path ایجاد میکنیم تا مسیر فایل های ماسک تصاویر را درون آن بریزیم. اگر دیتافریم img_df را در خروجی مشاهده کنیم به صورت زیر خواهد بود:

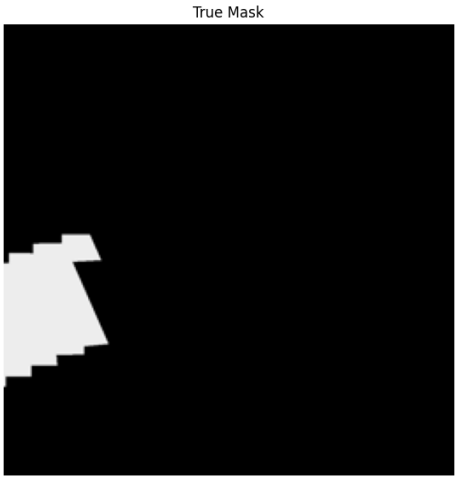
id	image_path	mask_path
0	/content/train/0.png	/content/train_masks/0_label.png
1	/content/train/1.png	/content/train_masks/1_label.png
10	/content/train/10.png	/content/train_masks/10_label.png
100	/content/train/100.png	/content/train_masks/100_label.png
101	/content/train/101.png	/content/train_masks/101_label.png
..	...	...
95	/content/train/95.png	/content/train_masks/95_label.png
96	/content/train/96.png	/content/train_masks/96_label.png
97	/content/train/97.png	/content/train_masks/97_label.png
98	/content/train/98.png	/content/train_masks/98_label.png
99	/content/train/99.png	/content/train_masks/99_label.png

[859 rows x 2 columns]

برای اینکه بتوانیم تابع create_dataset را کامل کنیم نیاز است تا دو تابع preprocessing و data_augmentation زا کامل کنیم. در تابع preprocessing مسیر تصویر به همراه ماسکش را در ورودی گرفته و آنها را با استفاده از tensorflow خوانده و با تابع decode to jpeg آنها را به صورت تصاویر سه کاناله تبدیل میکنیم. پس از این دو تصویر را resize کرده و در نهایت با تقسیم کردن مقادیر پیکسل هایشان آنها را نرمالایز میکنیم.

در تابع data_augmentation هم دو تصویر در ورودی گرفته و با استفاده از یک عدد رندوم بین صفر و یک تصاویری که به سمت چپ یا راست flip شده اند را بر میگردانیم. یک نمونه از خروجی داده های آموزشی به همراه ماسک به صورت زیر است:





در بلاک بعدی مدل ابتدایی که همان مدل mobileNet هست را خوانده و لایه های گفته شده را استخراج کرده و درون یک لیست ذخیره میکنیم.

```
base_model = tf.keras.applications.MobileNetV2(input_shape=(256, 256, 3), include_top=False, weights='imagenet')
backbone_layer = [
    base_model.get_layer('block_1_expand_relu').output,
    base_model.get_layer('block_3_expand_relu').output,
    base_model.get_layer('block_6_expand_relu').output,
    base_model.get_layer('block_13_expand_relu').output,
    base_model.get_layer('block_16_project').output
]
```

سپس یک مدل با استفاده از همین مدل mobileNet ساخته و تمامی پارامتر های لایه های مختلف آن را non-trainable میکنیم.