برای انجام فرآیند preprocessing و augmentation لازم است هنگامی که یک directory را به عنوان ورودی بگیرد بتواند تمامی عکس های موجود در آن فولدر را با فرمت های مختلف بیرون بکشد . برای این کار ما تابعی به نام filter_format تعریف کرده ایم.

این تابع یک دارکتوری را به عنوان ورودی میگیرد ، در ابتدا لیستی به نام images_directory تا هر عکسی که پیدا شد داخل این لیست اضافه شود .

لیستی با نام format_list تعریف شده که فرمت هایی از عکس که مد نظرمان است درآن قرار دارد و هر فرمت دلخواهی میتوان به آن اضافه کرد .

حال با استفاده از تابع os.walk داخل دایرکتوری ورودی پیمایش میکنیم . هنگام پیمایش درفایل ها چک میکنمی اگ به امدازه طول فرمت و یک نقطه از آخرین کاراکتر های path فایل برابر با المان های داخل متغیر format_list بود ، آن path را به images_directory اضافه میکنیم ، بدین ترتیب لیستی از path تمامی عکس ها با فرمت های مورد نظرمان را بر میگردانیم .

اکنون تمامی توابع ما یک ورودی تحت عنوان dataset میگیرند .

تمامی توابع در ابتدا سعی میکنند فولدری با نام مشخصات عملیاتی که روی تصاویر انجام میدهند ، تولید کنند و اگر نامی مشابه با آن فولدر وجود داشته باشد ، متغیری به نام num یکی اضافه شده و به انتهای اسم آن دایرکتوری اضافه میشود .

تابع rotate ، 4 ورودی میگیرد اولی لیستی از path تصاویر ، دومی درجه چرخش ، سومی رنگی که قسمت های خالی با آن پر میشود و چهارمی جهت چرخش است که به صورت دیفالت ساعتگرد می باشد .

بعد از ساختن دایرکتوری برای سیو کردن عکس های پردازش شده در آن ورودی چهارم یعنی ساعتگرد و پادساعتگرد بودن چک میشود اگر ساعت گرد بود درجه قرینه میشود زیرا به صورت پیش فرض در کتابخانه PIL ، تابع rotate به صورت پادساعتگرد است .

داخل یک حلقه تمامی path های موجود در dataset باز شده عملیات rotate روی آن انجام می شود و در دایرکتوری که در ابتدای تابع ساخته شد ، تک به تک ذخیره می شود.

تابع بعدی flip می باشد که سه ورودی میگیرد ، اولی لیت آدرس ها ، دومی و سومی دو ورودی از جنس bool هستند که عمودی وافقی flip شدن عکس رو مشخص می کنند ،

نامگذاری فایل ها همانند قبل انجام میشود ، اگر vertical ، true باشد ، از تابع flip از کتابخانه PIL و اگر

Horizontal ، true باشد از تابع mirror از کتابخانه PIL استفاده می شود .

تابع resize تابع بعدی است ، که همانند قبلی ها ورودی اولش لیست آدرس ها ، و در ابتدا دایرکتوری ای با نام و شماره یکتا ایجاد میکند ، دو ورودی دیگر نیز دارد که عرض و طول می باشد و با استفاده از تابع resize از کتابخانه PIL به این اندازه ها resize میشود .

تابع crop ، 5 ورودی میگیرد ، اولی لیست آدرس ها ، دومی و سومی مختصات شروع نقطه ای که کراپ از آن جا به عرض و طول دلخواه ( ورودی چهارم وپنجم ) می باشد .

تابع crop از کتابخانه PIL یک ورودی tuple میگیرد که 4 آرگومان دارد که به ترتیب با خط های موازی ضلع چپ بالا راست و پایین عکس را کراپ میکند که همه ی آن ها نسبت به قاصله از سمت چپ و بالا سنجیده می شوند براساس coordinate system بنابر این برای به دست آوردن آرگومان سوم وچهارم tuple باید ورودی اول تابع را با عرضی که میخواهیم کراپ شود و ورودی دوم تابع را با طولی که میخواهیم کراپ شود جمع کرده ومقادیر به دست می آید . در ضمن چک میشود که اگر

از با توجه به خواسته ها از عزض و طول عکس بیرون زدیم ، عملیات انجام نشود .

تابع random_crop نیز به همین ترتیب است فقط ورودی های tuple به صورت رندم انتخاب میشود .

تابع brightness یک ورودی برای مقدار این فیلتر دریافت میکند و آن را همانند توابع قبل اعمال می کند .

تابع add_noise سه ورودی میگیرد ، اولی لیست آدرس تصاویر دومی وسومی var و mean هست که از property های mode های random_noise از کتابخانه skimage.util می باشد که نحوه خواندن تصاویر و ذخیره آن ها با قبل متفاوت است و از imwrite و imread استفاده می شود .