به نام خدا

قسمت الف و ب:

برای انجام rotation pretext هر تصویر بدون بر چسب را به تابعی به اسم rotate_image دادم تا آن را به مقداری که خواسته بچرخاند. هر تصویر با زوایای 0و 90و 180و 270 ساخته و برچسب زده شد.

	Train accuracy	Test accuracy
Classification without weight initialization	55٪	38٪
Rotation pretext	77٪	80٪
Classification with weight initialization	70٪	44٪

قسمت پ:

در این برای انجام rotation pretext فقط یکبار آن را با یکی از زوایای 0و 90و 180و 270 به صورت رندوم چرخاندم و دلیل آن جلوگیری از 4 برابر شدن داده های تست بود.

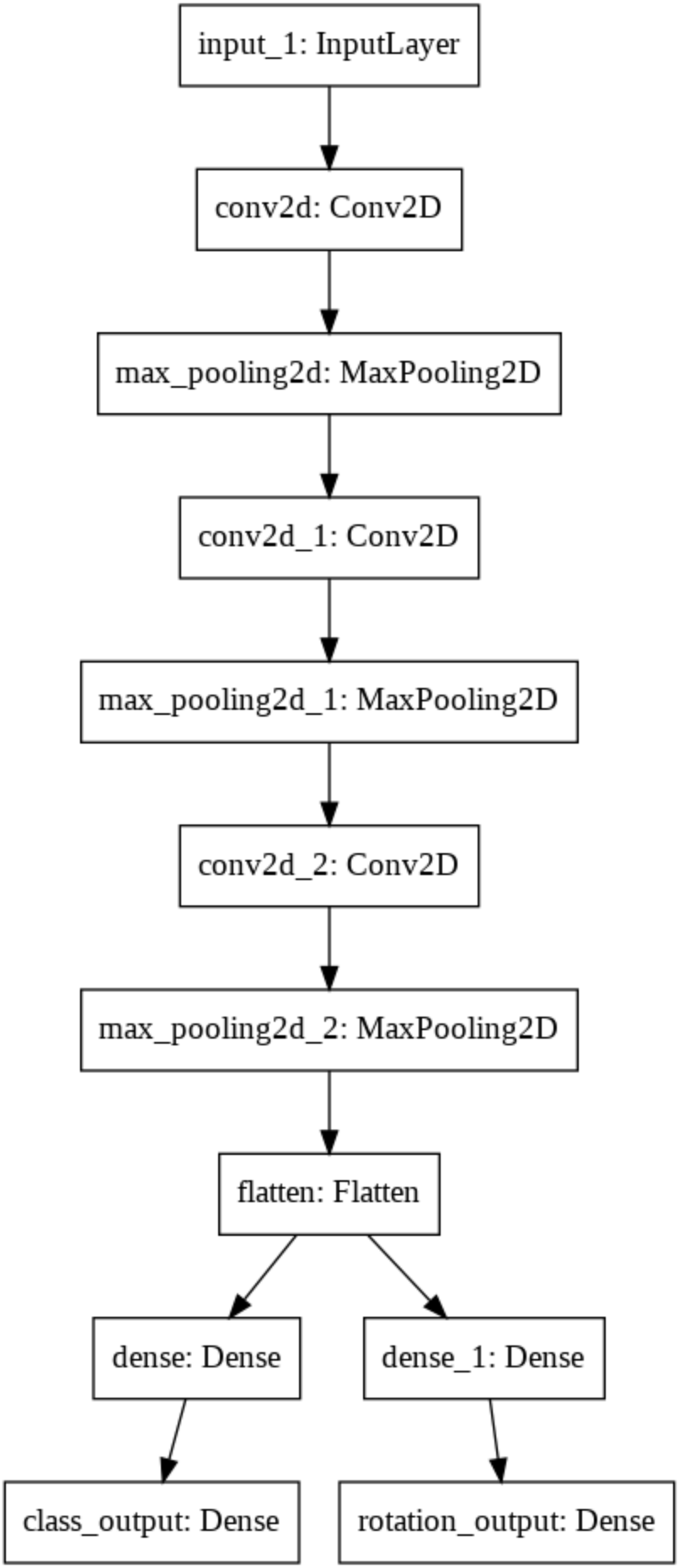
Loss_weights = {class_output, rotation_output}	Train class acc	Train rotation acc	Test class acc	Test rotation acc
1e + 6, 1e + 3	11.22%	34.58%	10.79%	35.07%
1e + 3, 1e + 6	12.13%	30.39%	12.49%	32.49%
1, 1	9.92%	29.85%	11.63%	36.72%
800, 200	11.71%	34.33%	11.11%	38.10%
200, 800	11.6%	33.58%	11.11%	29.65%
0.001, 5	8.79%	38.68%	12.42%	40.04%
5, 0.002	10.73%	34.22%	10.78%	34.94%
1, 0.3	13.11%	29.20%	10.5%	35.91%
1, 0.5	7.84%	33.81%	11.58%	31.50%

بهترین نتیجه ای که برای دسته بندی به دست آمد 12.49% بود که ضریب تابع ضرر خروجی دسته بندی برابر با 1e + 3 و ضریب تابع ضرر خروجی زاویه 1e + 6 میباشد. البته برای زاویه نتایج بهتری هم به دست آمد اما متناظر با دقت های ذکر شده برای خروجی دسته بند در جدول بالا نبودند.

در کل برای تمام حالت های بهبود چشمگیری حاصل نشد. با اینکه انتظار داشتم با بزرگتر دادن ضریب تابع ضرر دسته بندی نتیجه بهتری بگیرم. بنظر میرسد این امر دو علت میتواند داشته باشد. 1- حالت هایی که ضریب ضرر دسته بندی کوچکتر است: اثر پیش بینی های اشتباه دسته بندی در شبکه کم است و وزن ها نمیتوانند براساس آن اپدیت شوند. 2- حالت هایی که ضریب ضرر دسته بندی بزرگتر است: جریمه زیاد است اما چون فقط 200 داده هستند که برچسب دسته بندی را دارند نمیتوانند اثر زیادی روی شبکه بگذارند.

قسمت ت: مسئله pretext من تشخیص کانال های رنگی بود اما به علت کمبود وقت موفق به اجرای کامل آن نشدم.

مدلی که برای قسمت پ استفاده کردم در صفحه بعد نشان داده شده است که بخش convent آن مشترک است و لایه های dense متفاوت هستند:


input_1: InputLayer
conv2d: Conv2D
max_pooling2d: MaxPooling2D
conv2d_1: Conv2D
max_pooling2d_1: MaxPooling2D
conv2d_2: Conv2D
max_pooling2d_2: MaxPooling2D
flatten: Flatten
dense: Dense
dense_1: Dense
class_output: Dense
rotation_output: Dense