امیرحسین احمدی ۹۷۵۲۲۲۹۲ – تمرین نهم

۱- الف) در 2D Convolution فیلر ما یک ماتریس دو بعدی است یعنی تنها دو بعد از داده ورودی را در نظر میگیرد. البته تعداد چنل ها در ورودی های با توجه به بعد سوم، تغییر میکند ولی نمیتواند مانند 3D عمل کند. در نقطه‌ی مقابل فیلترهای 3D امکان در نظر گرفتن عمق ورودی را نیز دارند و جزعیات بیشتری را در نظر میگیرند. از مزایای 2D میتوان به حجم محاسباتی کمتر و نمایش آسان تر آن اشاره کرد ولی در عوض دقت کمتری نسبت به 3D دارد و به همه جزعیات توجه نمیکند. از کاربردهای 2D میتوان به استفاده از آن در اپلیکیشن تشخیص لبه (Sobel Edge Filter) و تز 3D نیز میتوان به کاربرد آن در video ها و تشخیص عمق اجسام در عکس اشاره کرد.

ب) از آنجایی که معمولا جهت پیدا کردن الگو در لایه ها اهمیتی ندارد از فیلتر مربعی استفاده میشود که متقارن باشد و جهتی نسبت به جهت دیگر برتری نداشته باشد. برای انتخاب اندازه فیلتر باید در نظر داشت که با افزایش آن تعداد پارامترها به سرعت افزایش پیدا میکند و باید به دقت آن را انتخاب کرد. البته میتوان لایه های ابتدایی را با سایزهای بالاتر در نظر گرفت.

ج) Pooling ها به چهار نوع Avrage, Minimum, Maximum و Global تقسیم می‌شوند. لایه pooling در خروجی شبکه در یک موقعیت مشخص را با یک مشخصه آماری از مقادیر همسایگی آن جایگزین میکند که این کار باعث میشود شبکه نست به تغییرات کوچک حساسیت کمتری داشته باشد. برای مثال Max خروجی را با ماکسیموم مقدار همسایگی آن جایگذاری میکند، Min و Avrage هم به همین تریتیب کار میکنند. Global Pooling خود به دو دسته GlobalAvragePooling و GlobalMaxPooling تقسیم میشود. مثلا GlobalMaxPooling ماکس پولینگ را روی تمام بعدها انجام داده تا تمام بعدها یک شوند.

۲- در این سوال میخواهیم تفاوت فیلترهای مختلف را روی تصاویر با استفاده از لایه های کانولوشنی ببینیم. فیلتر اول یکی از فیلترهای معروف برای محو کردن تصاویر است، که آن را یک بار با فیلتر ۳ در ۳ و یک بار با لایه ۵ در ۵ امتحان کردم که نتیجه را در زیر میبینیم(تصویر سمت راست با فیلتر ۳ در ۳ است):


فیلتر بعدی برای نمایان کردن لبه های عکس به کار میرود:

دو فیلتر بعدی نیز مانند فیلتر بالا برای نمایان کردن لبه های عکس به کار میروند، با این تفاوت که اولی لبه های افقی و دیگری لبه های عمودی عکس را نشان میدهند:


۳- الف) Keras Tuner ابزاری برای کنترل و بهبود دادن هایپر پارامترهای شبکه است. استفاده از آن بسیار آسان است. کافیست تابعی تعریف کرده و هایپرپارامترهایی که میخواهیم توسط tuner بهبود بیابند را با سینتکسی خاص مشخص کنیم. در tuner میتوان مشخص کرد که هایپرپارامتر مورد نظر چه تایپی باشد (int, float, ...)، میتوان برای آن حداقل و حداکثر تعریف کرد و حتی میتوان از بین چندین گزینه انتخاب کرد. (مانند انتخاب اوپتیمایز بهینه)

ب) در حال حاظر tuner های RandomSearch، BayesianOptimization، Hyperband در Keras Tuner موجود است. رندوم سرچ همانطور که پیداست در هر مرحله به صورت رندوم هایپرپارامترها را انتخاب کرده و در نهایت بهترینشان را خروجی میدهد که طبیعتن روش بهینه ای نیست. hyperband تقریبا همان رندوم است ولی به این صورت که ابتدا مثلا ۱۰۰ مجموعه رندوم از هایپر پارامترها در نظر گرفته و تا ۱۰۰ مرحله اموزش میدهد. نصفه ای که بدترین عملکرد را داشتند کنار میگذارد و بقیه را ۱۰۰ مرحله دیگر آموزش میدهد و این کار را تکرار میکند. مزیت این روش نسبت به random سرعت بیشتر آن است که لازم نیست همه مجموعه ها را به طور کامل آموزش دهیم. ولی روش bayesian کاملن فرق میکند. به این صورت که مقادیر اولیه را به صورت رندوم انتخاب کرده و سپس با روش گوسی آن ها را بهینه میکند. به همین دلیل ما در این سوال از روش bayesian استفاده کرده ایم.

ج) برای این سوال ابتدا یک تابع build تعریف کرده که مدل مان را بسازیم. مراحل ساخت مدل مانند همیشه است با ایت تفاوت که هایپرپارامتر هایی که میخواهیم بهینه کنیم را با استفاده از ورودی تابع محدودیت هایش را اعمال میکنیم تا tuner بتواند آن ها را بهینه کند. ابتدا یم لایه کانولوشنی با کرنل سایز ۵ در ۵ اضافه کرده و مقدار فیلتر آن تعریف میکنیم که بین ۳۲ و ۲۵۶ باشد و tuner آن را پیدا کند. برای لایه های کانولوشنی بعدی یک هایپر پارامتر تعریف میکنیم که برابر با تعداد لایه های conv باشد که حداقل آن را صفر و حداکثر را ۴ در نظر میگیریم. بعد از لایه های کانولوشنی یک لایه فلتن اضافه کرده و سپس مانند لایه های کانولوشنی ۰ تا ۵ لایه دنس نیز اضافه میکنیم که تعداد نورون های آن ها را بین ۳۲ تا ۲۵۶ میخواهیم بهینه کنیم. در آخر نیز یک لایه ۱۰ نورون برای تشخیص کلاس خروجی با softmax اضافه میکنیم. لرنینگ ریت را یک عدد فلوت از ۰.۰۰۱ تا ۰.۰۰۰۱ در نظر میگیریم و اوپتیمایزر میگذاریم tuner را از بین adam و sgd انتخاب کند.

سپس تابع را به bayesian optimizer داده و آبجکتیو فانکشن را برابر دقت ولیدیشن قرار داده تا آن را بهینه کند. با دستور tuner.search_space_summary میتوانیم داده هایی که قرار است توسط tuner بهینه شوند و مشخصات آن ها را ببینیم. سپس سرچ را شروع کرده و بعد از آن مدل

بهینه بدست آمده و مقدار هر پارامتر را در خروجی میبینیم. در آخر هم یک با مدل بهینه را ترین کرده و میبینیم که دقت ۶۳ درصدی را برای داده ولیدیشن بدست میاورد. مقدار نهایی هایپرپارامتر ها را میتوانیم در زیر ببینیم:

```
Hyperparameters:
conv_layer_1_filter_size: 64
conv_layers_num: 3
dense_layers_num: 5
lr: 0.0003861988397057912
optimizer: adam
conv_layer_2_filters: 32
conv_layer_3_filters: 32
conv_layer_4_filters: 32
dense_layer_1_units: 32
dense_layer_2_units: 32
dense_layer_3_units: 32
dense_layer_4_units: 32
dense_layer_5_units: 32
Score: 0.6351000070571899
```

۴- لایه اول یک لایه کانولوشنی دو بعدی میباشد که ۲۰ فیلتر دارد که سایز هر کدام (7,7,1) میباشد. (تصویر ورودی یک کاناله است.) پس تعداد پارامترهای این لایه برابر است با (1 + 1 * 7 * 7) * 20 که یک بایاس هم داریم:

$$20 * (7 * 7 * 1 + 1) = 1000$$

سایز خروجی لایه بالا (22, 22, 20) است.

لایه دوم یک لایه ماکس پولینگ است که هیچ پارامتری ندارد و سایز خروجی آن (11, 11, 20) است.

لایه سوم دوباره یک لایه کانولوشنی دو بعدیست که شامل ۱۰ فیلتر به سایز (5,5,20) میباشد.(به دلیل این که لایه قبلی ۲۰ فیلتر داشت، عمق کرنل ما برابر با ۲۰ میشود.) پس تعداد پارامتر های این لایه برابر است با:

$$10 * (5 * 5 * 20 + 1) = 5010$$

خروجی لایه بالا برابر با (7, 7, 10) است.

لایه بعدی یک لایه لوکالی کانکتد است که تفاوتش با کانولوشنی در متفاوت بودن وزن های هر فیلتر است. در این لایه دو فیلتر به سایز (3,3,10) داریم، پس تعداد کرنل ها برابر با 5 * 5 است. بنابراین تعداد پارامترهای آن به شکل زیر محاسبه میشود:

$$2 * \big(5 * 5 * (3 * 3 * 10 + 1)\big) = 4550$$

خروجی این لایه سایزی برابر با (*5, 5, 2*) دارد.

لایه پنجم فلتن است که پارامتری ندارد و فقط خروجی را *50 = 2 * 5 * 5* میکند.

در نهایت لایه آخر ۱۰ نورون دارد که پارامترهای آن برابر میشود با:

$$(50 + 1) * 10 = 550$$

بنابراین مجموع تعداد پارامترهای قابل آموزش برابر میشود با:

$$1000 + 5010 + 4550 + 550 = 11070$$

و همانطور که مشاهده میکنید برابر است با مقدار محاسبه شده در کراس:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 22, 22, 20)        1000

 max_pooling2d (MaxPooling2D  (None, 11, 11, 20)       0
 )

 conv2d_1 (Conv2D)           (None, 7, 7, 10)          5010

 locally_connected2d (Locall  (None, 5, 5, 2)          4550
 yConnected2D)

 flatten (Flatten)           (None, 50)                0

 dense (Dense)               (None, 10)                510

=================================================================
Total params: 11,070
Trainable params: 11,070
Non-trainable params: 0
```