به نام خدا

امیرمحمد کمیجانی ۹۹۵۲۲۰۳۲

تمرین سری ۶

۱)

الف) خط به خط کد داده شده را جلو میبریم.

خط اول:

ابعاد خروجی = همان ابعاد ورودی است(input است) = 512*512*3

تعداد پارامتر = محاسبه نداریم برای این خط

خط دوم :

ابعاد خروجی = 256,256,32

تعداد پارامتر = 7808 = (1 + 3*9*9)*32

خط سوم :

ابعاد خروجی = 64,64,32

تعداد پارامتر = محاسبه ندارد

خط چهارم:

ابعاد خروجی = 60,60,64 برای کانولوشن منهای ۴ کردیم

تعداد پارامتر = 51264 = (1 + 32*5*5)*64

خط پنجم :

ابعاد خروجی = 30,30,64

تعداد پارامتر = محاسبه ندارد

خط ششم :

ابعاد خروجی = 28,28,128

تعداد پارامتر = (1 + 128*3*3)*128

خط هفتم:

ابعاد خروجی = 28,28,128 در اینجا padding = ‘same’

تعداد پارامتر = (1 + 128 * 3 * 3) * 128

خط هشتم:

ابعاد خروجی = 14,14,128

تعداد پارامتر = محاسبه ندارد

خط نهم :

ابعاد خروجی = 12,12,512

تعداد پارامتر =(1 + 128 * 3 * 3)*512

خط دهم :

ابعاد خروجی = 512

تعداد پارامتر = محاسبه ندارد

خط یازدهم:

ابعاد خروجی = ۱۰۲۴

تعداد پارامتر = 512*1024 + 1024 =

خط دوازدهم:

ابعاد خروجی = ۱۰

تعداد پارامتر = 10*1024+10

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 32)      7808

 max_pooling2d (MaxPooling2  (None, 64, 64, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 60, 60, 64)        51264

 average_pooling2d (Average  (None, 30, 30, 64)        0
 Pooling2D)

 conv2d_2 (Conv2D)           (None, 28, 28, 128)       73856

 conv2d_3 (Conv2D)           (None, 28, 28, 128)       147584

 max_pooling2d_1 (MaxPoolin  (None, 14, 14, 128)       0
 g2D)

 conv2d_4 (Conv2D)           (None, 12, 12, 512)       590336

 global_average_pooling2d (  (None, 512)               0
 GlobalAveragePooling2D)

 dense (Dense)               (None, 1024)              525312

 dense_1 (Dense)             (None, 10)                10250

=================================================================
Total params: 1406410 (5.37 MB)
Trainable params: 1406410 (5.37 MB)
Non-trainable params: 0 (0.00 Byte)
```

۲)

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_d}\right]^{\mathrm{T}}$$

$$f(\mathbf{x} + \boldsymbol{\epsilon}) = f(\mathbf{x}) + \boldsymbol{\epsilon}^{\mathrm{T}} \nabla f(\mathbf{x}) + \mathcal{O}(\|\boldsymbol{\epsilon}\|^2)$$

$$\mathbf{x} \leftarrow \mathbf{x} - \eta \nabla f(\mathbf{x})$$

به صورت کلی میدانیم هرچقدر مقدار learning rate بیشتر باشد نمودار loss با سرعت و شیب بیشتری نزول میکند پس اگر از این فرمول استفاده کنیم میتوانیم به مقادیری دست پیدا کنیم:

Gradient = 2x – 10

برای هر یک از مقادیر بدست می آوریم:

X = 14
gradient = 30
14 = 20 – 30*alpha => alpha = 0.2

X=19.4
gradient = 30
19.4 = 20 – 30*alpha => alpha = 0.02

X = 19.94
gradient = 30
19.94 = 20 – 30*alpha => alpha = 0.002

همچنین راه دیگری نیز وجود دارد که مقادیر L های جدید را برحسب ورودی ها بدست آوریم:مقدار ثابت e را حذف کردم.

X = 20
L = 20^2 – 10*20 = 200

X = 14
L = 14^2 – 10*14 = 56

X = 19.4

L = 182.36

X = 19.94
L = 198.2

میزان تغییرات نسبت به حالت اولیه را نیز بررسی میکنیم.
در کل هر دو روش به ما ثابت میکنند که :
نمودار a برای مقدار 19.94
نمودار b برای مقدار 14
نمودار c برای مقدار 19.4

۳) در ماژول inception تمامی لایه های padding=’same’ هستند در نتیجه در خروجی دو بعد ورودی حفظ میشود و برای بعد سوم تعداد فیلترها را قبل از concat کردن حساب میکند.که به این صورت میشود:

64 + 32 + 128 + 64 = 288

یعنی در نهایت برابر (12,12,288) میشود.
قسمت مشخص شده نیز چون قبل از مرحله concatenation نیست در ابعاد خروجی نهایی تاثیر گذار نیست.

۴)

الف)

epoch به تعداد دفعاتی میگویند که مدل کل داده های آموزشی را طی کند.
batch size برای این است که داده آموزش را به قسمت هایی بشکنیم تا در انتهای هر بَچ وزن ها را آپدیت کنیم.

حال برای مقادیر گفته شده همچین فرمولی داریم:

Weight updates = e * (t/b)

ب)

در batchGD به این صورت است که در هر epoch میانگین تمام گرادیان ها را یک جا محاسبه میکند و به صورت کلی در هر epoch یکبار این عملیات انجام میشود و نمودار آن حالت smoothی دارد و با شیب ناگهانی و بیشتری کاهش را مشاهده خواهیم کرد.

در حالت mini-batch Gd روشی است که ما تمام دیتاست را به یکباره طی نمیکنیم و یک مقدار کوچکتر داریم و بعد از آن شروع به آپدیت کردن میکند و در هر epoch بیشتر از یکبار آپدیت انجام میشود.نمودار این حالت به صورت نوسانی است و با شیب کمتر و آهسته تری تغییر میکند.

با توجه به توضیحات داده شده نمودار باید مربوط به mini-batch GD باشد.

ج)

curve A : دیتای آموزش نتایج خوبی دارد و ارور آن کم است و از طرفی فاصله زیادی با دیتای validation دارد در نتیجه نشان دهنده پدیده overfitting میباشد.برای حل مشکل **overfitting** باید **داده افزایی و کاهش تعدادی ویژگی های ورودی** را انجام دهیم.

curve B : دیتای آموزش و validation نزدیک به هم هستند اما هر دو نتایج خوبی ندارند که این پدیده نشان دهنده **underfitting** میباشد. برای برطرف کردن آن میتوانیم **تعداد لایه های شبکه را افزایش دهیم**.

۵)

الف) برای پیکسل هایی که بزرگتر مساوی پیکسل مرکزی هستند مقدار ۱ قرار میدهیم و پیکسل هایی که مقادیر کمتری دارند صفر.به صورت ساعتگرد نیز حرکت میکنیم.

00000111 = 7	00000001 = 1	00000000 = 0
10000001 = 129	11000011 = 195	01100100 = 100
11000000 = 192	11110001 = 241	00000000 = 0

ب) اگر عدد ثابت c را به هر یک از پیکسل ها اضافه کنیم تغییری حاصل نخواهد شد چون به تمام پیکسل های اطراف ۱۰۰ مقداری را اضافه کردیم که به خود ۱۰۰ هم

اضافه کردیم و چون نوع مقایسه ما بزرگ و کوچک بودن است تغییری نخواهیم داشت. معادله ریاضی آن به این صورت میشود:

100 + c (vs) x + c => minus c from both sides => 100 vs x
x = one of the pixels around 100

در حالتی که یک عدد ثابت مثل c را در تمام پیکسل ها ضرب کنیم چون مقدار c>0 است باز هم تغییری نخواهیم داشت.(فرض بر این است که مقادیر پیکسل ها میتوانند بیشتر از ۲۵۰ باشند و ما نیازی به مپ کردن دوباره نداریم)

در هر دو حالت تغییری حاصل نمیشود.

ج)

هرچقدر هیستوگرام LBP دارای کد های بیشتری باشد یعنی الگوهای بیشتری یافت شده است و یعنی در تصویر اصلی تغییرات بیشتری داریم.

تصویر سوم دارای تغییرات نسبتا زیادی میباشد و هیستوگرام A برای آن مناسب تر است.

تصویر دوم دارای یکنواختی نسبتا زیادی است و الگوهای آن اکثرا مشابه هستند و هیستوگرام c برای آن مناسب تر است.

تصویر اول نه به اندازه تصویر سوم پر تغییرات است و نه به اندازه تصویر تصویر دوم یکنواخت در نتیجه هیستوگرام B برای آن مناسب است.

سوالات عملی:

۶)

الف)

ابتدا تصاویر سگ و گربه را برای دیتای آموزش و تست از دیتاست تنسورفلو لود میکنیم.
در مرحله بعد که باید preprocessing انجام دهیم.به این صورت که تصاویر را به شکل ۱۲۸ در ۱۲۸ تغییر سایز میدهیم و با تابع cast دیتاتایپ را مناسب مقادیر کرده و نورمالیزیشن را انجام میدهیم. سپس مقادیر داده های آموزش و تست را عملیات های preprocessing را انجام میدهیم.با استفاده از تابع shuffle ۱۰۰۰ تای آنها را برای آموزش انتخاب کرده و batch size را برابر ۱۶ قرار میدهیم.
در مرحله بعدی مدل خود را میسازیم.نکاتی که وجود دارد این است که برای لایه های نهان از تابع فعالساز relu استفاده کرده ایم و برای خروجی با توجه به اینکه تسک ما به نوعی classification است و در خروجی فقط یک نورون در نظر داریم از sigmoid استفاده میکنیم.

سپس مدل را با بهینه ساز آدام و تابع ضرر binary cross entropy کامپایل میکنیم و در نهایت یک summary از نوع لایه های کانولوشنی و در کل تمامی لایه های مدل ارائه میدهیم.

سپس مدل را با epoch = 10 آموزش میدهیم.
در قسمت های بعدی با استفاده از تابع evaluate و داده تست مدل را تست میکنیم نتیجه تست را نمایش میدهیم و در قسمت بعد پلات های خواسته شده را رسم میکنیم.

ب)

موارد اولیه شبیه به قسمت قبل است.
پلات از ۱۲ تصویر سگ و گربه نمایش میدهیم و تشخیص مدل را در بالای آن نمایش میدهیم.مدل inceptionV3 را مقداردهی کرده و یک summary از آن را نمایش میدهیم.این

مدل از وزن های imagenet استفاده میکند که حاوی تصاویر طبقه بندی شده است و از قبل آموزش داده شده است.
در قسمت بعد خروجی مدل inceptionV3 را ابتدا flatten میکنیم یعنی به صورت یک بعدی نمایش میدهیم سپس این مقادیر که x نامیده میشوند را به activation func خود یا sigmoid میدهیم که گفتیم برای مسائل classification مناسب است میدهیم.سپس با استفاده از تابع dense همان عملیات w*x+b را انجام میدهیم و مقدار پیشبینی شده را محاسبه میکنیم.سپس مدل را فریز میکنیم یعنی دست به وزن های آن به صورت کلی نمیزنیم و فقط با لایه ای که ساختیم کار میکند.
سپس مدل را کامپایل کرده و summary از آن نشان میدهیم.
در مراحل بعد مدل را آموزش داده و سپس تست میکنیم و نتایج را نمایش میدهیم.

۷)

ابتدای دیتای آموزش و تست را از mnist لود میکنیم.
در مرحله بعد خواسته سوال که گفته شده یک زیرمجموعه از دیتای ۰و۱و۲ فقط داشته باشیم را با استفاده از تابع isin انجام میدهیم.سپس یک پلات از ۵ تا از داده ها به صورت Gray نمایش میدهیم.
در مرحله بعد با استفاده از تابع normalize عملیات نورمالیزیشن که در صورت سوال خواسته شده را انجام میدهیم.
میدانیم برای استفاده از Humoments تصاویر باید به صورت باینری باشند پس در مرحله اول ابتدا تصاویر را به صورت باینری ذخیره میکنیم سپس با استفاده از تابع moments ، descriptor خود را پیاده سازی میکنیم و سپس خروجی آنرا به تابع HuMoments میدهیم و به صورت flatten تبدیل میکنیم و در لیست خود ذخیره

میکنیم و در نهایت خروجی آن لیست را به آرایه نامپای تبدیل کرده و خروجی میدهیم.

سپس در مرحله بعد این تابع را برروی داده های تست و آموزش فراخوانی میکنیم و برای Scale کردن از standardScaler استفاده میکنیم تا میانگین ۰ و واریانس ۱ برسیم.سپس برای داده های آموزش از fit_transform و برای داده های تست از transform استفاده میکنیم که هر دو را به یک scale برسانند.

برای آموزش از svm استفاده میکنیم و البته یک کرنل خطی و مدل را آموزش میدهیم.

در مرحله بعد بر روی دیتای تست پیش بینی و سپس دقت مدل را میسنجیم.و در نهایت خروجی هایی از مدل را نمایش میدهیم.