دانشکده مهندسی کامپیوتر

مباحث ویژه ۱ (یادگیری عمیق)

تمرین سری هشتم

علی صداقی

۹۷۵۲۱۳۷۸

۱ سوال اول

الف) هنگامی که شبکه دچار Overfit شده است می‌توان از لایه Dropout که نوعی منظم‌ساز است استفاده کنیم. از این لایه نباید در مواقعی که مدل Underfit است یا مشکل Overfit را ندارد استفاده کنیم. در واقع لایه Dropout اجازه می‌دهد مدل را زمان طولانی تری آموزش دهیم. این لایه پارامتر آموزشی ندارد و تنها یک هایپرپارامتر ورودی می‌گیرد که مشخص می‌کند نرخ غیر فعال شدن نورون‌ها چقدر باشد. مثلا اگر ۱۰۰ نورون داشته باشیم و از Dropout با نرخ 0.6 استفاده کنیم ۶۰ تا نورون‌ها در زمان آموزش غیر فعال می‌شوند. این لایه فقط در زمان آموزش (Train) نورون‌ها را غیر فعال می‌کند و در فاز ارزیابی (Validation) و Test دیگر نورونی را خاموش نمی‌کند. برای اینکه شبکه اثر نرخ Dropout را خنثی کند، خروجی لایه را در عبارت $1/(1-\text{rate})$ ضرب می‌کنیم. (یک روش Inverse هم وجود دارد.)

مقدار دهی پارامتر: هر چه نرخ دراپ کردن بیشتر باشد قدرت منظم‌سازی بیشتر می‌شود و شبکه کمتر دچار Overfit می‌شود. پس اگر همچنان مشکل Overfit وجود داشت نرخ دراپ را بیشتر می‌کنیم.

بهتر است در لایه Classifier مثل سافتمکس از Dropout استفاده نکنیم زیرا اگر یک نورون را دراپ کنیم در واقع انگار یکی از کلاس‌های مسئله را ندید گرفته‌ایم.

این لایه شبکه را Robust تر نیز می‌کند زیرا شبکه به خاموش شدن برخی نورون‌ها که دارای ویژگی‌هایی هستند عادت می‌کند و در صورتی که نورونی دچار مشکل شد خطای زیادی رخ نمی‌دهد.

دلیل تاثیر این لایه این است که شبکه دیگر روی هیچ ویژگی ای حساب ۱۰۰ درصد باز نمی‌کند و با تغییر زیادی که در ویژگی‌ها ایجاد می‌شود شبکه امکان حفظ کردن یک الگوی خاص را از دست می‌دهد.

همچنین با حذف شدن تعدادی نورون شبکه Sparse تر می‌شود و این ویژگی مهمی است.

ب) افزایش نرخ دراپ باعث می‌شود در هر تکرار نورون‌های بیشتری حذف شوند، ظرفیت یک شبکه هم با تعداد نورون‌های لایه‌های میانی رابطه مستقیم دارد، بنابراین نرخ دراپ با ظرفیت شبکه رابطه عکس دارد یعنی هرچه نرخ دراپ کمتر باشد ظرفیت شبکه بیشتر است و برعکس. برای مثال اگر نرخ دراپ برابر ۱ باشد تمامی نورون‌های شبکه خاموش می‌باشند و شبکه هیچ ظرفیتی ندارد. اگر نرخ دراپ برابر ۰ باشد همه نورون‌ها فعال می‌باشند و شبکه در بیشترین ظرفیت ممکن خود به سر می‌برد.

۲ سوال دوم

لایه Fully Connected: در این لایه هر خروجی لایه قبل به تمامی نورون‌های این لایه متصل است، بنابراین تعداد اتصالات و متناسب با آن تعداد وزن‌ها بسیار زیاد است. این لایه به صورت کامل و جامع به ورودی نگاه می‌کند و می‌تواند در ترکیب ویژگی‌هایی که دارای اهمیت مکانی (Spatial) یا زمانی (Temporal) نیستند بسیار مناسب باشد. معمولا از این لایه در لایه‌های آخر شبکه و در قسمت Classification برای ترکیب ویژگی‌های سطح بالا استفاده می‌کنیم. معمولا آن را در لایه‌های اولیه که ورودی اندازه بزرگی دارد و ویژگی‌ها سطح پایین هستند استفاده نمی‌کنیم زیرا هم تعداد پارامترها زیاد می‌شود هم ویژگی‌ها محلی هستند نه عمومی.

لایه Locally Connected: این لایه بسیار مشابه لایه Conv است. تنها تفاوت در اشتراک وزن هاست. در لایه Conv از یک فیلتر با وزن‌های ثابت در قسمت‌های مختلف ورودی استفاده می‌کنیم. اما در لایه Locally Connected در هر محل (Location) از یک فیلتر جدید با وزن‌های جدید استفاده می‌کنیم. این امر باعث می‌شود تعداد پارامترهای این لایه بسیار زیاد شود. مزیت این لایه این است که می‌توانیم در هر محل (Location) ویژگی منحصر به فرد و جدیدی را تشخیص دهیم. مثلا در کاربرد تحلیل چهره می‌توان در نقاط اساسی چهره مانند چشم، لب، بینی و ... فیلترهایی محلی با وزن‌های متمایز استفاده کنیم و اطلاعات متفاوتی را از هر ناحیه صورت استخراج کنیم.

لایه Convolutional: در این لایه اتصالات به صورت تنک است، یعنی در هر لحظه فقط تعداد کمی از نقاط ورودی به این لایه متصل هستند. این امر باعث Parameter sharing می‌شود و تعداد پارامترها را بسیار کم می‌کند. با حرکت دادن (عملیات Conv) از همان وزن‌های مشترک می‌توانیم در قسمت‌های مختلف ورودی استفاده کنیم. این لایه برای استخراج ویژگی‌های محلی (Local) مناسب است. مثلا در تصویر ما می‌خواهیم یک کار مشخص (مثلا تشخیص لبه) را در قسمت‌های مختلف عکس انجام بدهیم پس یک فیلتر با تعداد وزن کم که مناسب تشخیص لبه هست را روی عکس حرکت می‌دهیم و در هر محل (Local) از عکس، لبه‌ها را پیدا می‌کنیم. نکته مثبت دیگر این لایه Sparse بودن اتصالات است. نکته مثبت دیگر این لایه بازنمایی‌های هم‌تغییر (equivariant) است. از این لایه برای تشخیص ویژگی‌های محلی زمانی (Temporal) و مکانی (Spatial) استفاده می‌شود. در لایه‌های ابتدایی که ورودی اندازه بزرگی دارد (عکس، صوت و ...) کاربرد زیادی دارد و به خوبی می‌تواند ویژگی‌های سطح پایین را استخراج کند.

Fully Connected	*Convolution*	*Locally Connected*
اتصالات کامل	اتصالات محلی	اتصالات محلی
اشتراک وزن نداریم	اشتراک وزن داریم	اشتراک وزن نداریم
تعداد پارامتر زیاد	تعداد پارامتر کم	تعداد پارامتر زیاد
مناسب ویژگی‌های سطح بالا	مناسب ویژگی‌های سطح پایین	مناسب ویژگی‌های سطح پایین و بالا
Global	*Temporal, Spatial*	*Temporal, Spatial*
لایه‌های انتهایی	لایه‌های ابتدایی	لایه‌های ابتدایی و انتهایی
Fully Connected	*Sparsity of Connection*	*Sparsity of Connection*

۳ سوال سوم

الف) یک تابع get_generators پیاده‌سازی شد که با توجه به فعال بودن یا نبودن Augmentation عملکر متفاوتی دارد. در حالت بدون داده‌افزایی عکس‌های آموزش و تست را فقط تقسیم بر ۲۵۵ می‌کنیم. در حالت فعال بودن داده افزایی داده‌های تست تنها rescale می‌شوند زیرا داده‌افزایی روی داده آموزش صورت می‌گیرد. برای داده آموزش داریم:

```
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
rescale=1./255,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest'
```

تصاویر را به صورت Random در یک بازه ۴۰ درجه دوران می‌دهیم، همچنین از طول و عرض شیفت می‌دهیم، مقادیر را تقسیم بر ۲۵۵ می‌کنیم، با استفاده از shear range تصویر را از دید زوایای مختلف افزایی می‌کنیم، همچنین روی تصاویر زوم می‌کنیم، به صورت افقی معکوس می‌کنیم و ...

اندازه خروجی (target size) را به صورت ۱۵۰ * ۱۵۰ در نظر می‌گیریم. اندازه هر بسته (batch size) نیز برابر ۱۶ است.

هایپرپارامترهای شبکه به این صورت است:

```
LOSS = 'categorical_crossentropy'
OPTIMIZER = Adam(learning_rate=0.001)
EPOCHS = 50
HEIGHT = 150
WIDTH = 150
BATCH_SIZE = 16
```

ساختار مدل بکار رفته در قسمت‌های ب، پ، ت یکی و به صورت زیر می‌باشد.

لایه‌های Dropout توسط ورودی تابع Wrapper می‌تواند غیرفعال یا فعال شود.

همانطور که مشاهده می‌کنید در این مدل ۴ لایه Conv2D، دو لایه Dense و سه MaxPool داریم.

- ✓ برای شلوغ نشدن نوتبوک، خروجی قسمت Fit را نمایش ندادیم. (Verbose = 0)
- ✓ بخش‌های د، ه در هر سه بخش ب، پ، ت انجام شده است.

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
conv2d (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 17, 17, 64)	0
conv2d_3 (Conv2D)	(None, 15, 15, 64)	36928
flatten (Flatten)	(None, 14400)	0
dense (Dense)	(None, 64)	921664
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 5)	325

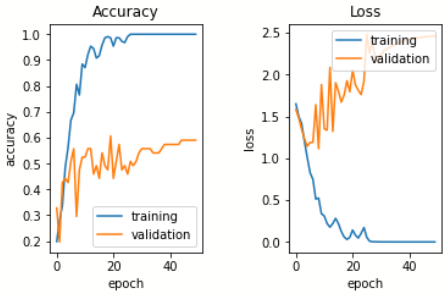
Total params: 987,557
Trainable params: 987,557
Non-trainable params: 0

ب) ارزیابی مدل بدون داده‌افزایی

Best Epoch: 20

Train Loss: 0.0209 Train Acc: 1.0000

Test Loss: 1.7940 Test Acc: 0.6066



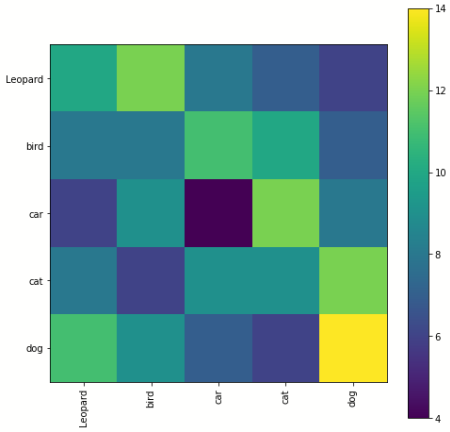
همانطور که مشاهده می‌شود در این مدل دچار Overfit شدید هستیم پس باید از داده‌افزایی و Dropout استفاده کنیم.

```
Classification Report Train
              precision    recall  f1-score   support

     Leopard       0.23      0.23      0.23        43
        bird       0.18      0.18      0.18        44
         car       0.10      0.10      0.10        39
         cat       0.20      0.20      0.20        44
         dog       0.30      0.30      0.30        47

    accuracy                           0.21       217
   macro avg       0.20      0.20      0.20       217
weighted avg       0.21      0.21      0.21       217

Confusion Matrix Train
[[10 12  8  7  6]
 [ 8  8 11 10  7]
 [ 6  9  4 12  8]
 [ 8  6  9  9 12]
 [11  9  7  6 14]]
```

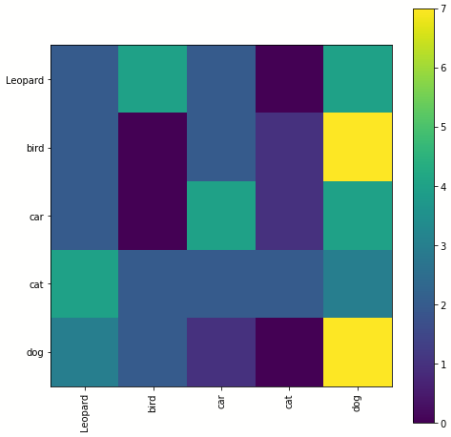


Classification Report Test

	precision	recall	f1-score	support
Leopard	0.15	0.17	0.16	12
bird	0.00	0.00	0.00	12
car	0.36	0.36	0.36	11
cat	0.50	0.15	0.24	13
dog	0.28	0.54	0.37	13
accuracy			0.25	61
macro avg	0.26	0.24	0.23	61
weighted avg	0.26	0.25	0.23	61

Confusion Matrix Test

```
[[2 4 2 0 4]
 [2 0 2 1 7]
 [2 0 4 1 4]
 [4 2 2 2 3]
 [3 2 1 0 7]]
```

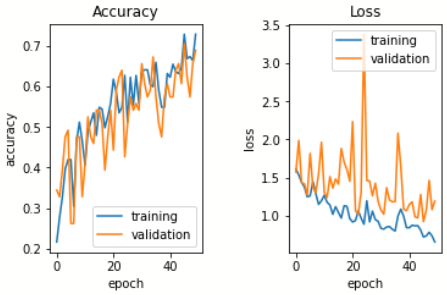


پ) داده‌افزایی‌های متفاوتی را استفاده کردیم که در قسمت الف به توضیح آن پرداختم، نتایج به‌صورت زیر است:

Best Epoch: 46

Train Loss: 0.7311 Train Acc: 0.6912

Test Loss: 0.9255 Test Acc: 0.7049



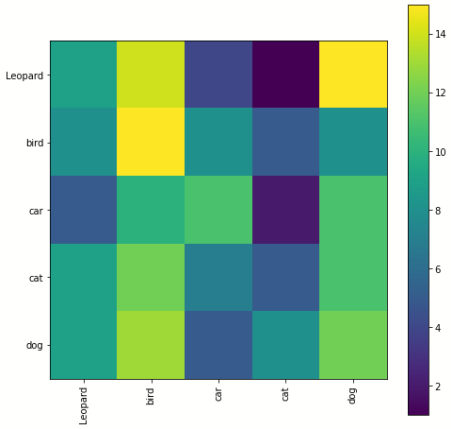
همانطور که مشاهده می‌شود این مدل دیگر مشکل Overfit را ندارد و دقت در حالت آموزش و ارزیابی تقریبا برابر شده است. بسته به اینکه خطای انسان (Human error) در این مسئله چقدر است می‌توان در باره Underfit شدن نظر داد.

```
Classification Report Train
              precision    recall  f1-score   support

     Leopard       0.23      0.21      0.22        43
        bird       0.23      0.34      0.28        44
         car       0.31      0.28      0.30        39
         cat       0.24      0.11      0.15        44
         dog       0.21      0.26      0.23        47

    accuracy                           0.24       217
   macro avg       0.24      0.24      0.24       217
weighted avg       0.24      0.24      0.23       217

Confusion Matrix Train
[[ 9 14  4  1 15]
 [ 8 15  8  5  8]
 [ 5 10 11  2 11]
 [ 9 12  7  5 11]
 [ 9 13  5  8 12]]
```

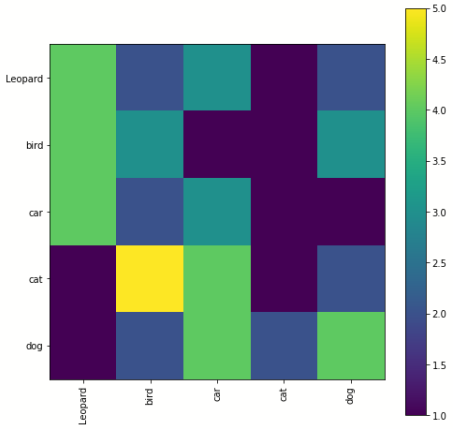


```
Classification Report Test
              precision    recall  f1-score   support

     Leopard       0.29      0.33      0.31        12
        bird       0.21      0.25      0.23        12
         car       0.20      0.27      0.23        11
         cat       0.17      0.08      0.11        13
         dog       0.33      0.31      0.32        13

    accuracy                           0.25        61
   macro avg       0.24      0.25      0.24        61
weighted avg       0.24      0.25      0.24        61

Confusion Matrix Test
[[4 2 3 1 2]
 [4 3 1 1 3]
 [4 2 3 1 1]
 [1 5 4 1 2]
 [1 2 4 2 4]]
```



احتمال اینکه با پیچیده تر کردن شبکه و افزایش زمان آموزش به دقت بالاتر برسیم ممکن بود، اما چون هایپرپارامترها را برای بررسی مساوی همه حالات در نظر گرفتیم این کار را نکردیم.

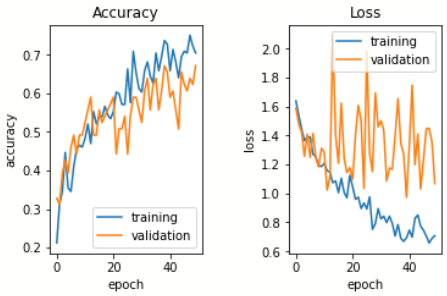
ت) سه حالت ۰/۲، ۰/۵ و ۰/۸ برای Dropout استفاده می‌کنیم، نتایج به‌صورت زیر است:

Dropout = 0.2

Best Epoch: 39

Train Loss: 0.6547 Train Acc: 0.7235

Test Loss: 1.2735 Test Acc: 0.6721

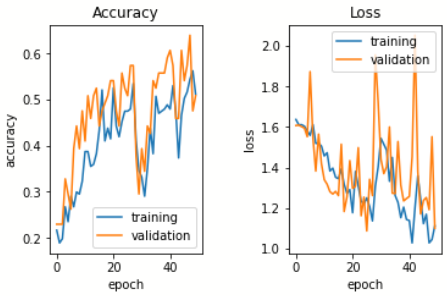


Dropout = 0.8

Best Epoch: 48

Train Loss: 0.9128 Train Acc: 0.6175

Test Loss: 1.1917 Test Acc: 0.6393

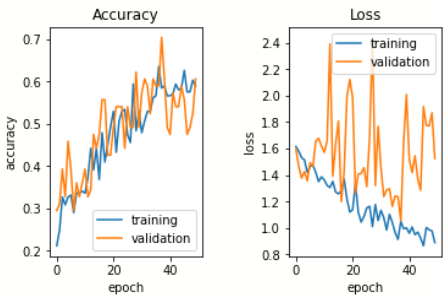


Dropout = 0.5

Best Epoch: 38

Train Loss: 0.8870 Train Acc: 0.6313

Test Loss: 1.0550 Test Acc: 0.7049



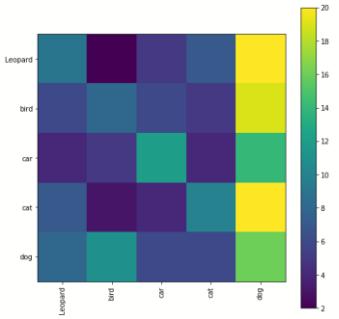
صفحه بعدی درباره حالت 0.5 می‌باشد.

Classification Report Train

	precision	recall	f1-score	support
Leopard	0.26	0.21	0.23	43
bird	0.28	0.18	0.22	44
car	0.36	0.31	0.33	39
cat	0.31	0.23	0.26	44
dog	0.18	0.34	0.24	47
accuracy			0.25	217
macro avg	0.28	0.25	0.26	217
weighted avg	0.28	0.25	0.25	217

Confusion Matrix Train

```
[[ 9  2  5  7 20]
 [ 6  8  6  5 19]
 [ 4  5 12  4 14]
 [ 7  3  4 10 20]
 [ 8 11  6  6 16]]
```

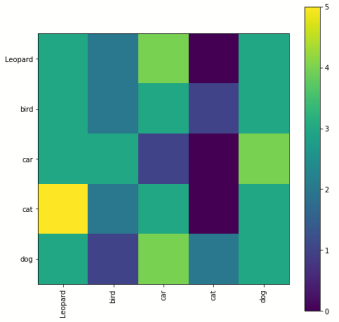


Classification Report Test

	precision	recall	f1-score	support
Leopard	0.18	0.25	0.21	12
bird	0.20	0.17	0.18	12
car	0.07	0.09	0.08	11
cat	0.00	0.00	0.00	13
dog	0.19	0.23	0.21	13
accuracy			0.15	61
macro avg	0.13	0.15	0.13	61
weighted avg	0.13	0.15	0.13	61

Confusion Matrix Test

```
[[3 2 4 0 3]
 [3 2 3 1 3]
 [3 3 1 0 4]
 [5 2 3 0 3]
 [3 1 4 2 3]]
```



مقایسه نتایج:

Model	Best Epoch	Train Best	Test Best	Condition
No Aug No Drop	20	loss: 0.0209 acc: 1.0000	loss: 1.7940 acc: 0.6066	Overfit
Aug No Drop	46	loss: 0.7311 acc: 0.6912	loss: 0.9255 acc: 0.7049	Perfect
Aug Drop = 0.2	39	loss: 0.6547 acc: 0.7235	loss: 1.2735 acc: 0.6721	Good
Aug Drop = 0.8	48	loss: 0.9128 acc: 0.6175	loss: 1.1917 acc: 0.6393	Underfit
Aug Drop = 0.5	38	loss: 0.8870 acc: 0.6313	loss: 1.0550 acc: 0.7049	Strange Underfit

بهترین حالت مدلی است که داده ازایی داریم و از Dropout استفاده نکردیم. در واقع هرچه نرخ دراپ را بیشتر می‌کنیم عملکرد مدل بدتر می‌شود و به سمت Underfit می‌رود. این نشان می‌دهد داده‌افزایی مناسبی استفاده کردیم که مشکل Overfit را کامل حل کرده است.

د) در هر قسمت از سوال مقادیر f1-score، recall، precision و support محاسبه شده است و در گزارش نیز آورده شده است. در این قسمت مفهوم هر یک را بررسی می‌کنیم:

Precision: نسبت True Positive به مجموع True Positive و False Positive است. در واقع نسبت True Positive به تمامی پیشبینی‌های Positive. مفهوم آن این است که مدل در پیشبینی‌هایی که گفته Positive است چقدر دقیق بوده است. این معیار برای زمانی مناسب است که هزینه False Positive زیاد است. مثلا نباید ایمیلی که اسپم نیست را اسپم پیشبینی کنیم.

Recall: نسبت True Positive به مجموع حالت‌هایی که واقعا True هستند می‌گویند. در واقع بیانگر این است که مدل ما چه تعداد از داده‌هایی را که واقعا Positive هستند را درست Positive پیشبینی کرده است. این معیار برای زمانی مناسب است که که هزینه False Negative زیاد است. مثلا اگر یک فرد بیمار را سالم تشخیص بدهیم بسیار بد است.

F1 Score: دو معیار Precision و Recall را با هم ترکیب می‌کند:

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

این معیار زمانی مناسب است که می‌خواهیم یک Balance میان دو معیار بالا برقرار کنیم. همچنین این معیار برای زمانی که که توزیع داده‌ها در کلاس‌ها نا برابر است مناسب است.

منبع:

https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

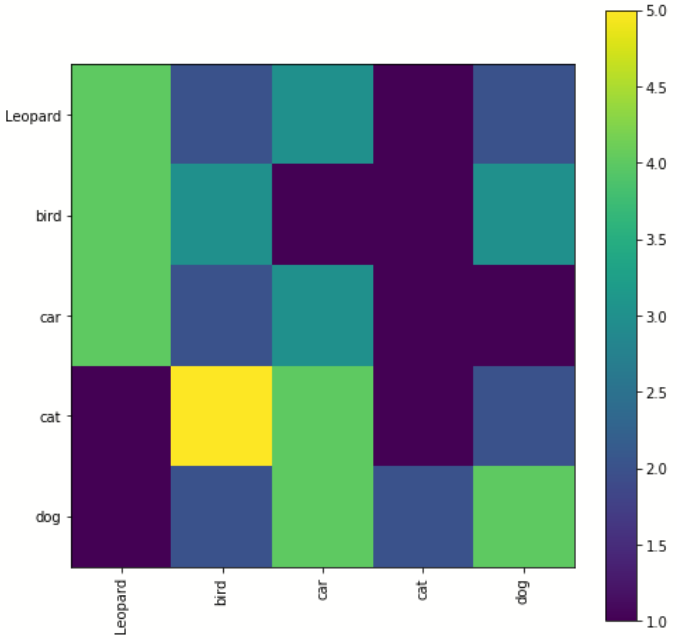
ه) این معیار هم برای تمامی قسمت‌های این سوال محاسبه و رسم شده است. در این قسمت مفهوم آن در این مسئله را بررسی خواهیم کرد:

در واقع یک ماتریس N*N است (N تعداد کلاس) و هر درایه مانند i,j از این ماتریس بیان می‌کند که چه تعداد از داده‌های دسته i به اشتباه به عنوان دسته j تشخیص داده شده است.

باید توجه کرد که این ماتریس متقارن نیست مثلا در دیتاست MNIST ممکن است تعدادی زیادی از ۴ها را ۹ ببینیم ولی هیچ ۹ایی را ۴ نبینیم.

در حالت پ (داده افزایی بدون دراپ) به بررسی این ماتریس می‌پردازیم:

```
Confusion Matrix Test
[[4 2 3 1 2]
 [4 3 1 1 3]
 [4 2 3 1 1]
 [1 5 4 1 2]
 [1 2 4 2 4]]
```



هر چه طیف رنگی بالاتر می‌رود نشان این است که این دو کلاس با هم بیشتر به اشتباه گرفته شده اند. برای مثال تعداد ۵ گربه را به اشتباه پرنده تشخیص داده ایم. اما فقط ۱ پرنده را گربه تشخیص داده ایم.