

به نام خدا



تمرین سری دهم درس یادگیری عمیق

دکتر محمدی

محمد یارمقدم

۹۶۴۶۲۱۰۴

## سوال اول)

به دلیل مشکل کمبود **usage** در محیط کولب تعداد **epoch** ها در اندازه نتایج نیست و در حد امکان انجام شد.

## الف)

در قسمت اول این تمرین هدف، آموزش شبکه با وزن های تصادفی است. برای این کار به جای استفاده از شبکه از پیش آموزش داده شده، خودمان شبکه را آموزش می دهیم و این کار را با استفاده از **false** قرار دادن آرگومان **pretrained** در ساخت مدل **resnet** انجام می دهیم. در نهایت این شبکه را در ده **epoch** آموزش داده و نتایج زیر را کسب کردم:

```
Epoch 3/9
-----
Iterating through data...
train Loss: 137.0372 Acc: 6.5319
Iterating through data...
val Loss: 143.2909 Acc: 5.4688

Epoch 4/9
-----
Iterating through data...
train Loss: 132.1725 Acc: 8.0147
Iterating through data...
val Loss: 139.5121 Acc: 7.1181

Epoch 5/9
-----
Iterating through data...
train Loss: 125.5628 Acc: 10.5025
Iterating through data...
val Loss: 138.2287 Acc: 7.1925

Epoch 6/9
-----
Iterating through data...
train Loss: 108.6612 Acc: 18.4436
Iterating through data...
val Loss: 124.9041 Acc: 13.1076

Epoch 7/9
-----
Iterating through data...
train Loss: 101.8361 Acc: 21.7157
Iterating through data...
val Loss: 124.2823 Acc: 13.1448

Epoch 8/9
-----
Iterating through data...
train Loss: 97.3640 Acc: 24.7181
Iterating through data...
val Loss: 123.9796 Acc: 13.6657

Epoch 9/9
-----
Iterating through data...
train Loss: 93.4284 Acc: 26.7157
Iterating through data...
val Loss: 123.6141 Acc: 14.3105

Training complete in 58m 39s
Best val Acc: 14.310516
=> saving checkpoint
```

(ب)

در این قسمت هدف استفاده از شبکه از پیش آموزش دیده شده **resnet** است. در این قسمت با استفاده از **feature extractor** نتایج لایه های شبکه را بهبود بخشیدیم و روند را سرعت بخشیدیم. در شبکه های **pretrained** لایه های اولیه قسمت های کلی و عمومی را آموزش می بینند و لایه های انتهایی جزئیات را فرا می گیرند. به همین دلیل در این قسمت همه لایه های شبکه را به جز لایه آخر فریز می کنیم تا فقط همین لایه آموزش ببیند. در اینجا با این حرکت تعداد پارامتر های قابل آموزش شبکه از ۲۴ میلیون به ۱,۲ میلیون کاهش یافت که باعث میشود روند آموزش و پیشرفت شبکه بسیار سریع تر بشود.

برای این کار در پارامتر های شبکه **resnet** متغیر **requires\_grad** را برای همه لایه ها جز لایه آخر **false** قرار می دهیم. در نهایت برای این قسمت نیز نتایج زیر حاصل شد:

```
Epoch 3/9
-----
Iterating through data...
train Loss: 167.9426 Acc: 0.9069
Iterating through data...
val Loss: 167.4703 Acc: 1.7733

Epoch 4/9
-----
Iterating through data...
train Loss: 167.6606 Acc: 1.0294
Iterating through data...
val Loss: 167.1106 Acc: 2.1701

Epoch 5/9
-----
Iterating through data...
train Loss: 167.2608 Acc: 1.1520
Iterating through data...
val Loss: 166.6097 Acc: 2.5918

Epoch 6/9
-----
Iterating through data...
train Loss: 166.9853 Acc: 1.6176
Iterating through data...
val Loss: 166.5626 Acc: 2.6042

Epoch 7/9
-----
Iterating through data...
train Loss: 166.8475 Acc: 1.3725
Iterating through data...
val Loss: 166.4663 Acc: 2.6166

Epoch 8/9
-----
Iterating through data...
train Loss: 166.7750 Acc: 1.6299
Iterating through data...
val Loss: 166.3799 Acc: 2.6414

Epoch 9/9
-----
Iterating through data...
train Loss: 166.7974 Acc: 1.3848
Iterating through data...
val Loss: 166.3475 Acc: 2.6662

Training complete in 40m 45s
Best val Acc: 2.666171
```

روند همگرایی شبکه بهبود یافته است.

(پ)

در این قسمت تمامی مراحل همانند مرحله قبل است با این تفاوت که از تابع SVM به جای Linear به عنوان classifier استفاده کردیم. در این تابع نیاز به x\_train و y\_train داشتیم تا به عنوان ورودی تابع استفاده کنیم. برای این کار inputs را به شبکه resnet داده و خروجی آن را x در نظر میگیریم و label های آنها را y در نظر میگیریم. برای این کار متد جدیدی نوشتیم و از بدنه متد train\_model استفاده کردم تا برنه آن را تکمیل کنم و input و label را بدست بیاورم. سپس آنها را به svm دادم تا عمل دسته بندی انجام شود. در اولین ران این قسمت با ده epoch نتایج زیر حاصل شد:

```
Epoch 1/10
-----
train Acc: 1.1002
Epoch 2/10
-----
train Acc: 1.7421
Epoch 3/10
-----
train Acc: 1.6531
Epoch 4/10
-----
train Acc: 3.1298
Epoch 5/10
-----
train Acc: 3.8015
Epoch 6/10
-----
train Acc: 6.3519
Epoch 7/10
-----
train Acc: 10.0212
Epoch 8/10
-----
train Acc: 15.5478
Epoch 9/10
-----
train Acc: 18.5476
epoch 10/10
-----
train Acc: 16.4563
```

اما در ران دوم با پانزده epoch به علت نامعلوم دقت تحت هر شرایطی صفر چاپ میشد. حالت های مختلفی را تست و امتحان کردم ولی متوجه نشدم چرا در هر حالت صفر چاپ میشد. از آن به بعد تحت هر شرایطی این عدد صفر چاپ شد.

```
Epoch 1/10
-----
train Acc: 1.1002
Epoch 2/10
-----
train Acc: 1.7421
Epoch 3/10
-----
train Acc: 1.6531
Epoch 4/10
-----
train Acc: 3.1298
Epoch 5/10
-----
train Acc: 3.8015
Epoch 6/10
-----
train Acc: 6.3519
Epoch 7/10
-----
train Acc: 10.0212
Epoch 8/10
-----
train Acc: 15.5478
Epoch 9/10
-----
train Acc: 18.5476
epoch 10/10
-----
train Acc: 16.4563
```

مقایسه:

طبیعتاً به علت مزیت تابع SVC نسبت به دسته بند های خطی می بایست دقت در فاز آموزش بهبود می یافت. هم میزان اولیه و هم سرعت رشد دقت بهبود می یابد.

## ت)

در این قسمت هدف این است که چند لایه آخر شبکه را از حالت فریز خارج کنیم تا در فرآیند آموزش همراهی داشته باشند و متغیرهای یادگیری شبکه افزایش داشته باشد. برای این کار از متغیر `children()`. تمامی لایه های شبکه را محاسبه کرده و یک کانتر برای آن قرار میدهیم. سپس به جای حرکت روی کل شبکه و فریز کردن تمامی لایه ها، تعداد موردنظر را فریز می کنیم. همچنین برای بهبود نتیجه از دو لایه `apadtiveavgpooling` و `batchnorm` استفاده کردم تا کمی روند بهبود یابد. نتایج زیر حاصل این شبکه است:

```
Epoch 2/9
-----
Iterating through data...
train Loss: 165.4959 Acc: 2.0833
Iterating through data...
val Loss: 164.3366 Acc: 5.0967
```

```
Epoch 3/9
-----
Iterating through data...
train Loss: 163.4222 Acc: 3.3946
Iterating through data...
val Loss: 162.0901 Acc: 6.4484
```

```
Epoch 4/9
-----
Iterating through data...
train Loss: 160.9887 Acc: 4.2770
Iterating through data...
val Loss: 159.1420 Acc: 7.7009
```

```
Epoch 5/9
-----
Iterating through data...
train Loss: 157.9619 Acc: 5.1961
Iterating through data...
val Loss: 155.8115 Acc: 8.7054
```

```
Epoch 6/9
-----
Iterating through data...
train Loss: 155.8177 Acc: 5.8701
Iterating through data...
val Loss: 155.3822 Acc: 8.4449
```

```
Epoch 7/9
-----
Iterating through data...
train Loss: 155.4693 Acc: 6.3971
Iterating through data...
val Loss: 154.9638 Acc: 9.2262
```

```
Epoch 8/9
-----
Iterating through data...
train Loss: 155.2535 Acc: 6.4216
Iterating through data...
val Loss: 154.6226 Acc: 9.2262
```

```
Epoch 9/9
-----
Iterating through data...
train Loss: 154.8116 Acc: 6.1642
Iterating through data...
```

(ث)

کد این قسمت زده نشد. اما مقایسه ها را طبق معلومات خود پاسخ می دهم.

در حالتی که شبکه fine tune نشده باشد، درصد سیاهی های لایه آخر به علت عدم یادگیری کامل ویژگی باید بزرگ باشد. با انجام fine tune و درآوردن چند لایه آخر از حالت فریز درصد این سیاهی ها کم شده و هرچه تعداد لایه بیشتری از حالت فریز خارج شوند، آموزش بهتر صورت گرفته و درصد سیاهی رو به کاهش میرود.

لینک های کمکی:

[https://pytorch.org/tutorials/beginner/transfer\\_learning\\_tutorial.html](https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html)

<https://www.kaggle.com/winnie19900705/feature-extraction-resnet50-svm#Part-3:-Feed-extracted-features-into-SVM-for-classification>

<https://discuss.pytorch.org/t/how-the-pytorch-freeze-network-in-some-layers-only-the-rest-of-the-training/7088/2>