Home Work No 3

Report

Yazdan Zandiye Vakili

610397193

1. Pre-processing

در این قسمت دیتاست را باز کرده و برچسب های داده های برچسب دار را جدا کردم و برای بخش پیش پردازش دو کار مطمئن شدن از عدم وجود NULL or NaN را انجام دادم، اما به صورت شهودی و چون دیتاست یک دیتاست بسیار مشهور بود و در کتاب خانه Keras به همین شهودی دیدن بسنده کردم. درباره نرمالایز کردن داده ها هم در ابتدا با استفاده از تابع zscore در کتاب خانه scipy این کار را انجام داده بودم ولی به دلیل اینکه در ادامه روال نیاز به اعداد صحیح داشتیم و این رند شدن خیلی ضرر به روند کار می زد و همچنین چون کل اعداد در بازه 0 تا 255 بودند، پس نیازی در واقع به این کار نبود.

2. Build the Auto Encoder

در این قسمت با استفاده از کتاب خانه TensorFlow یک Auto Encoder با سه لایه مخفی، به ترتیب 128 ، 16و 128 ساختم تا در نهایت بتوانیم داده ها را ابعادشان را از 728 به 16 برسانم. برای Compile کردن Auto Encoder از تابع SGD به عنوان Optimizer و از تابع MeanSquareError (MSE) به عنوان Loss Function استفاده کردم که نتیجه بهتری به من در طی روند داد (البته نتیجه کلی خوب نبود و به دلیل بودن تمرین در بازه امتحانات وقت برای بهینه تر کردن تمرین پیدا نکردم) و برای استفاده از Encoder ، دو ساختار Encoder و Decoder را جداگانه ساختم و در کنار هم قرار دادم و با هم Auto Encoder را پیاده سازی کردم و بر روی داده های آموزش برچسب دار، Fit کردم و بعد برای کمتر شدن زمان اجرا این Encoding را روی داده های تست برچسب دار و همچنین داده های آموزش بدون برچسب هم

پیاده سـازی کردم تا در روند Clustering به زمان کمتری برای اجرا نیاز داشـته باشـم (ولی خب همانطور که می بینیم دقت پایین آمد)

3. Clustring (K-Means)

برای خوشـه بندی داده های بدون برچسـب از الگوریتم K-Means اسـتفاده کردم چون دقیقا به 10 برچسـب نیاز داشـتم و با الگوریتم DBSCAN به Post-processing هم برای رسـاندن تعداد خوشـه ها به این عدد نیاز داشـتم ولی با این الگوریتم خیر (هر چند که ممکن بود دقت بیشـتری بدهد)

4. Post-processing on Clustering

حال همانطور که گفته شده بود در ویدیو توضیح پروژه در گروه، به صورت شهودی تعداد نمونه از هر خوشـه برداشـتم و با نگاه کردن گرفتن اکثریت عدد متناظر با هر برچسـب را تعیین کردم و برچسـب هر خوشـه را به عدد متناظرش تغییر دادم. (در پوشـه Sample_Images ، 100 نمونه ای که گرفته بودم و با اسـتفاده از کتاب خانه Pillow از روی ماتریس موجود بازسـازی کرده بودم را می توانید مشاهده کنید ولی، همانطور که در خود پروژه هم ذکر کردم این نمونه 100 تایی - 10 تا از هر خوشـه – نتوانسـت کامل باشـد و مجبور شـدم تعداد بیشـتر نمونه بگیرم و نگاه کنم ولی برای ارسـال فایل فقط همان 100 نمونه را ارسـال کردم. همچنین یکبار برای مشـخص کردن عدد متناظر به هر خوشـه، مرکز هر خوشـه را گرفتم و با استفاده از Decoder به ابعاد 784 بردم و در نهایت تصـویر را بازسـازی کردم ولی به دلیل دقت پایین Auto Encoder نتیجه اصـلا رضـایت بخش نبود و نمی شـد به آن اتکا کرد و با توجه به این موضـوع، همان نمونه هارا بیشـتر کردم و به بررسی آن ها پرداختم و برای کمتر نشـدن دقت از دیتاسـت ای از داده های بدون برچسـب که همچنان آن را حفظ کرده بودم اسـتفاده کردم و برحسـب آن تضـاویر را بازسـازی کردم) در خود پروژه اعداد متناظر با هر خوشه را ذکر کردم و در پایین تصویر آن را می گذارم.

```
label 0 --> number 8
label 1 --> number 4
label 2 --> number 3
label 3 --> number 7
label 4 --> number 0
label 5 --> number 2
label 6 --> number 6
label 7 --> number 9
label 8 --> number 1
label 9 --> number 5
```

5. Combining DataSets

در این قسمت برچسب های بدست آمده را بعد از تغییر به عدد متناظرشان به دیتاست برچسب های موجود از داده های آموزش اضافه کردم و یک دیتاست از برچسب های داده های آموزش ساختم و همین کار را برای داده های آموزش انجام دادم و در نهایت با ترکیب داده های آموزش برچسب دار و بدون برچسب، یک مجموعه داده آموزش را ساختم.

6. Implement MLP Classifier with GridSearch

با استفاده از کتاب خانه Sci-kit Learn یک MLP Classifier ساختم و با استفاده از روش 5 Folds Cross-Validation این Classifier را بر روی دیتاست های ترکیب شده Fit کردم و با دادن مجموعه ای از پارامتر ها به این Classifier ، بهترین پارامتر ها را بدست آوردم. در پایین می توانید مجموعه پارامتر هایی که برای MLP Classifier در نظر گرفته شده بود را مشاهده کنید.

```
# Define a range of choices for each Hyper-Parameter
param_grid = {
    'hidden_layer_sizes': [(100, 100), (50, 100)],
    'activation': ['relu', 'logistic'],
    'learning_rate_init': [0.001, 0.01],
    'batch_size': [32, 64],
    'max_iter': [10, 20],}
```

بهترین پارامتر ها در نهایت به این شکل شدند :

1. Hidden layer sizes : (100, 100)
2. Activation Function : Relu
3. Learning Rate Init : 0.001
4. Batch Size : 64
5. Max Iteration : 20

و در نهایت نتایج آموزش این Classifier بر روی این دیتاست ترکیب شده از دو دیتاست قبلی بدین شکل شد :

1. Accuracy : 23.58 %
2. Precision : 25.20 %
3. Recall : 23.57 %
4. F1 Score : 23.26 %
5. Fitting Time : 41.46 Mins
6. Confusion Matrix :

```
[[367   7  80   8  69 147 135 148  14   5]
 [  0 420  28 650   0   0  32   3   2   0]
 [ 24  33 549  29 126  25  98 133  10   5]
 [ 15 114  95 109 107  94 373  16  49  38]
 [  6 330 107  68  27  31 110  36  75 192]
 [ 36  98  86  52  14 202 326  30  31  17]
 [ 13  10 459  28   1   5 146 292   1   3]
 [  0 270  11  50   1   4  12  95 268 317]
 [ 26 120  39  18   5 245 317  20 168  16]
 [  1 308  12  35   2  24   7  13 332 275]]
```

پی نوشـــت : در نهایت باز هم به دلیل دقت پایین تمرین عذرخواهی می کنم و باز هم ذکر می کنم به دلیل انجام تمرین در بازه امتحانات وقت کافی برای بهینه کردن آن را نداشتم.