محمّدصادق کلامی یزدی ۴۰۲۸۱۱۰۶۸

تمرین ۱– با مطالعه‌ی بخش ۱ و ۲ مرجع "classification metrics.pdf" و بررسی نحوه‌ی محاسبه‌ی Accuracy, Percision, Recall, F-1 Score از روی Confusion Matrix، این مقادیر را برای ماتریس زیر که برای طبقه‌بندی ۵-کلاسه است، محاسبه کنید. (حدالامکان محاسبات کامل نوشته شوند)

		True Class				
		A	B	C	D	E
Predicted Class	A	100	20	30	40	10
	B	20	200	50	10	30
	C	10	20	250	50	30
	D	10	10	20	300	50
	E	10	20	30	40	300

Predicted classes \ True classes	A	B	C	D	E
A	۱۰۰	۲۰	۳۰	۴۰	۱۰
B	۲۰	۲۰۰	۵۰	۱۰	۳۰
C	۱۰	۲۰	۲۵۰	۵۰	۴۰
D	۱۰	۱۰	۲۰	۴۰۰	۵۰
E	۱۰	۲۰	۳۰	۴۰	۴۰۰

عدد دانشجویی ۴۰۲۱۱۱۰۶۸

$$PPV = Precision = \frac{TP}{FP + TP} \qquad TPR = Recall = \frac{TP}{TP + FP}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad F1\text{-}score = 2 \cdot \frac{Precision \times Recall}{Precision + Recall}$$

1

Precision:

$$PPV_A = \frac{100}{(20+30+40+10)+100} = 0.5$$

$$Precision_B = \frac{200}{(20+50+10+40)+200} \Rightarrow$$

$$Precision_B = 0,4285714286$$

2

Precision C: $\frac{۲۵۰}{(۱۰+۲۰+۵۰+۵۰)+۲۵۰} = \frac{۲۵۰}{۳۶۰} = ۰/۶۹۴$

Precision D: $\frac{۳۰۰}{(۱۰+۱۰+۲۰+۵۰)+۳۰۰} = \frac{۳۰۰}{۴۹۰} = ۰/۷۶۹۲۳۰۷۷$

Precision E: $\frac{۳۰۰}{(۱۰+۲۰+۵۰+۴۰)+۳۰۰} = \frac{۵۰۰}{۶۰۰} = ۰/۷۵$

Recall:

Recall A: $\frac{۱۰۰}{۱۰۰+۵۰} = ۰/۶۶۷$

Recall B: $\frac{۲۰۰}{۲۰۰+(۲۰+۲۰+۱۰+۲۰)} = ۰/۷۴۱$

Recall C: $\frac{۲۵۰}{۲۵۰+(۵۰+۵۰+۲۰+۵۰)} = ۰/۶۵۸$

Recall D: $\frac{۵۰۰}{۵۰۰+(۴۰+۱۰+۵۰+۴۰)} = ۰/۸۱۲$

Recall E: $\frac{۳۰۰}{۳۰۰+(۱۰+۵۰+۵۰+۵۰)} = ۰/۷۱۴$

F1-Score: ۲

F1-A: $۲ \times \frac{۱۵ \times ۰/۹۹۷}{۱۵ + ۰/۹۹۷} = ۰/۵۷۱۵$

F1-B: $۲ \times \frac{۱}{\frac{۱}{۰/۹۶۵} + \frac{۱}{۱/۰۴۱}} = ۰/۹۸۹۹$

F1-C: $۲ \times \frac{۱}{\frac{۱}{۰/۹۹۴} + \frac{۱}{۰/۹۵۸}} = ۰/۹۷۵۵$

F1-D: $۲ \times \frac{۱}{\frac{۱}{۰/۷۹۹} + \frac{۱}{۰/۹۸۲}} = ۰/۷۲۳$

F1-E: $۲ \times \frac{۱}{\frac{۱}{۰/۱۵} + \frac{۱}{۰/۱۱۴}} = ۰/۱۷۵۵۷$

(۴) Accuracy:

$$\text{Total Accuracy} = \frac{\text{Correctly classified Values}}{\text{Total number of Values}}$$

$$= \frac{۱۵۵ + ۲۵۵ + ۲۵۰ + ۵۵۰ + ۵۰}{۱۵۰ + ۲۷۰ + ۵۸۰ + ۴۴۰ + ۴۲۰} = \frac{۱۱۵۰}{۱۴۹۰} = ۰/۹۹۲۷۱۰۸$$

تمرین ۲- یکی از روش‌ها مقابله با over-fitting، استفاده از لایه‌ی Dropout است. این کار را می‌توان برای هر یک از لایه‌های تعریف‌شده در مدل Sequential در Keras، با استفاده از دستور model.add(keras.layers.Dropout(rate)) پس از تعریف لایه‌ی مد نظر انجام داد. برای مثال کد:

```
model.add(keras.layers.Conv2D(64, kernel_size=(3, 3), activation="relu"))
model.add(keras.layers.Dropout(0.2))
```

یک لایه‌ی کانولوشنی حاوی Dropout با احتمال حذف نورون ٪۲۰ را به مدل Sequential اضافه می‌کند.
در کد "CNN(CIFAR-10).ipynb" با اعمال لایه‌ی Dropout پس از لایه‌های Conv و Dense با نرخ‌های حذف نورون[1](آرگومان rate) متفاوت نتیجه‌ی حاصله روی عملکرد مدل را تحلیل کنید. (کد با نتیجه حاصله با بهترین نرخ‌های Dropout را بصورت یک فایل ipynb. ارسال نمایید.)

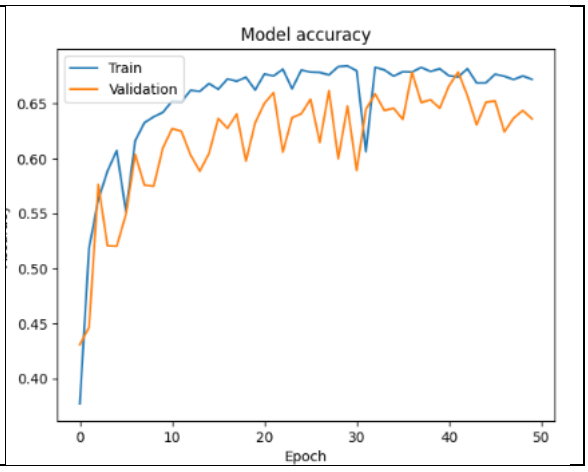
تمامی کدها در Kaggle ران شده‌اند و کدهای آن‌ها به Colab انتقال داده شده است. در صورت نیاز به خود Kaggle اطلاع بدید تا به شما دسترسی بدهم.

در تمرین ۲ با استفاده از Keras-Tuner بهترین مقادیر Dropout برای هر لایه بدست آورده شده ولی کماکان مدل مشکلی که در تمرین ۳ گفتید را داشت، پس به صورت گام به گام موارد زیر اضافه شد تا اثر هر کدام دیده و گزارش شود. در نهایت بهترین مدل، مدلی است که از EarlyStopping، LR-Schecular، Dropout، Batchnorm و L2-Regularization استفاده می‌کند و مقدار بهینه‌تمامی این پارامترها توی هر بخش گزارش شده است.

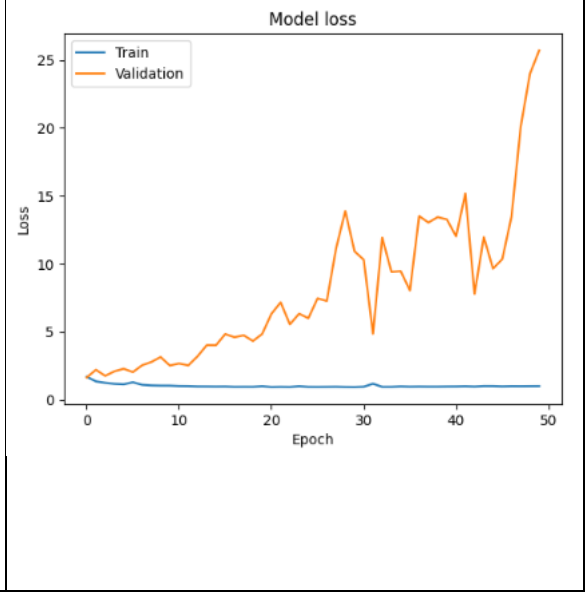
- L2 Regularization
- Dropout
- EarlyStopping and Learning Rate Scheduler
- Batchnormalization

۱. مدل بهینه با استفاده از Dropout

```
Trial 44 Complete [00h 00m 48s]
val_accuracy: 0.2855771481990814

Best val_accuracy So Far: 0.627909779548645
Total elapsed time: 00h 34m 21s

The hyperparameter search is complete. The optimal dropout rates are:
    - Dropout 1: 0.30000000000000004
    - Dropout 2: 0.1
    - Dropout 3: 0.2
    - Dropout 4: 0.7000000000000001
```



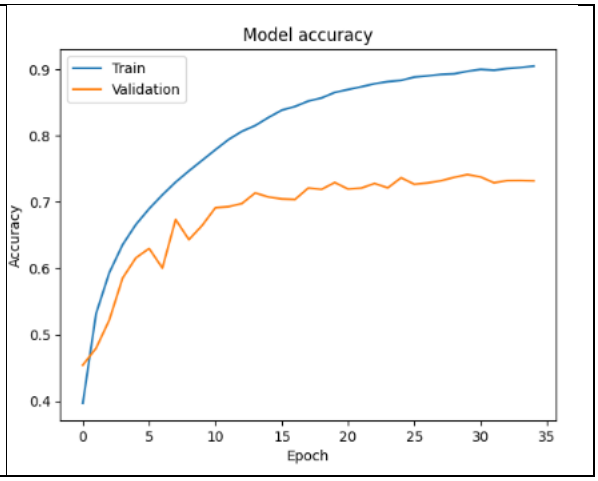
```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.6424 - loss: 1.7851
Test Loss: 1.7520262002944946
Test Accuracy: 0.6381000280380249
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
              precision    recall  f1-score   support

           0       0.65      0.65      0.65      1000
           1       0.92      0.70      0.79      1000
           2       0.53      0.55      0.54      1000
           3       0.40      0.55      0.46      1000
           4       0.55      0.73      0.62      1000
           5       0.66      0.50      0.57      1000
           6       0.84      0.63      0.72      1000
           7       0.86      0.63      0.73      1000
           8       0.55      0.93      0.69      1000
           9       0.91      0.51      0.65      1000

    accuracy                           0.64     10000
   macro avg       0.69      0.64      0.64     10000
weighted avg       0.69      0.64      0.64     10000

[[654   2  47  24  22   3   2   6 239   1]
 [ 50 697   7  44   9   7   6   2 131  47]
 [ 90   1 548 101 136  36  31  15  42   0]
 [ 38   2  95 550 106 105  57  13  33   1]
 [ 24   0  84  73 728  15  13  28  35   0]
 [  9   0 100 271  72 499   8  21  20   0]
 [ 12   1  68 129 104  23 634   3  25   1]
 [ 14   1  62  97 122  58   1 630  15   0]
 [ 29   2  11  14   7   3   2   0 930   2]
 [ 85  52  15  70  29  10   3  14 211 511]]
```



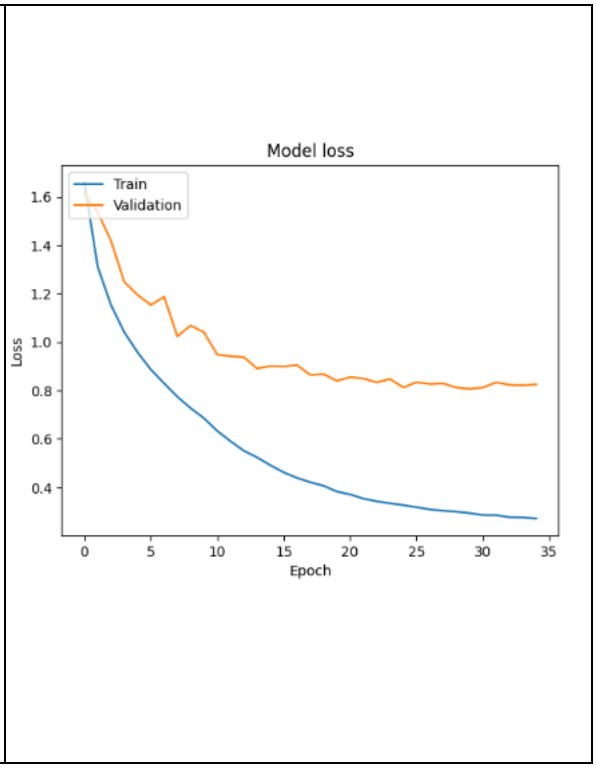
---

۲. مدل بهینه با استفاده از EarlyStopping، LR-Schecular و Dropout

```
Trial 44 Complete [00h 01m 13s]
val_accuracy: 0.4695224463939667

Best val_accuracy So Far: 0.7354211807250977
Total elapsed time: 00h 38m 47s

The hyperparameter search is complete. The optimal parameters are:
    - Dropout 1: 0.30000000000000004
    - Dropout 2: 0.1
    - Dropout 3: 0.7000000000000001
    - Dropout 4: 0.1
    - Learning Rate: 0.00015904925418362462
```

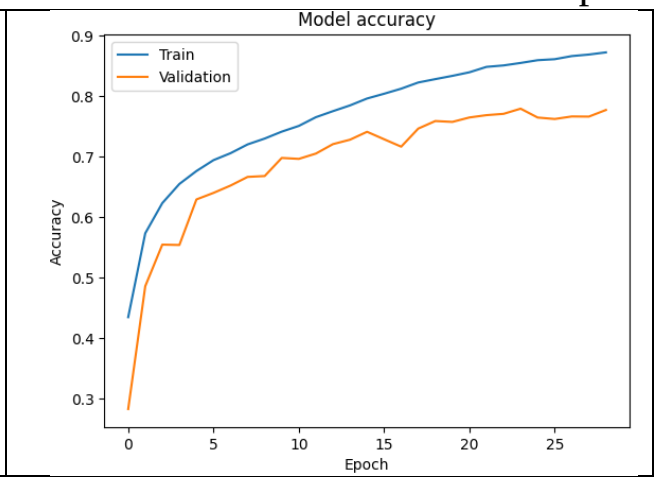


```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.7365 - loss: 0.7837
Test Loss: 0.7841531038284302
Test Accuracy: 0.7347999811172485
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
              precision    recall  f1-score   support

           0       0.67      0.87      0.76      1000
           1       0.87      0.84      0.86      1000
           2       0.70      0.61      0.65      1000
           3       0.48      0.66      0.56      1000
           4       0.77      0.65      0.70      1000
           5       0.56      0.70      0.63      1000
           6       0.95      0.67      0.78      1000
           7       0.82      0.76      0.79      1000
           8       0.89      0.80      0.84      1000
           9       0.89      0.78      0.83      1000

    accuracy                           0.73     10000
   macro avg       0.76      0.73      0.74     10000
weighted avg       0.76      0.73      0.74     10000

[[874  11  25  26   6   8   1   3  30  16]
 [ 40 839   8  28   5  13   2   4  18  43]
 [101   2 608 104  58  82  10  24   7   4]
 [ 29   1  39 663  35 191   8  23   6   5]
 [ 36   2  63 105 651  66  10  59   7   1]
 [ 24   2  25 196  17 704   2  24   2   4]
 [ 18   6  58 128  40  68 665  10   4   3]
 [ 24   2  18  65  32  91   1 763   1   3]
 [101  29  11  26   1  11   0   5 803  13]
 [ 52  67   8  29   5  17   1  16  27 778]]
```

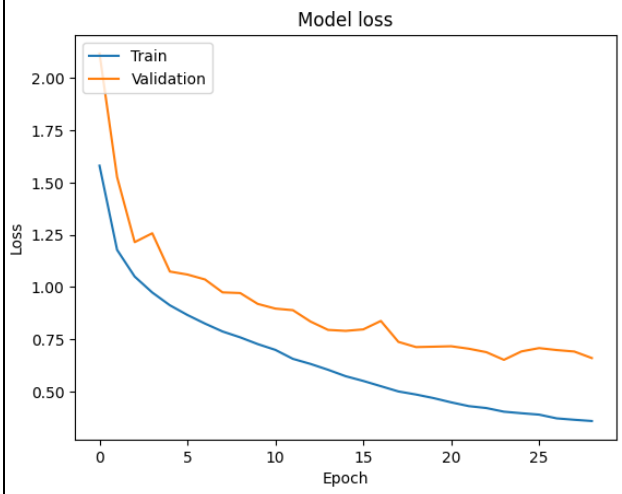


۳. مدل بهینه با استفاده از EarlyStopping، LR-Schecular، Batchnorm و Dropout

```
Trial 44 Complete [00h 00m 51s]
val_accuracy: 0.25053995847702026

Best val_accuracy So Far: 0.7678185701370239
Total elapsed time: 00h 42m 05s

The hyperparameter search is complete. The optimal parameters are:
    - Dropout 1: 0.30000000000000004
    - Dropout 2: 0.30000000000000004
    - Dropout 3: 0.4
    - Dropout 4: 0.1
    - Learning Rate: 0.0010637278760861717
```

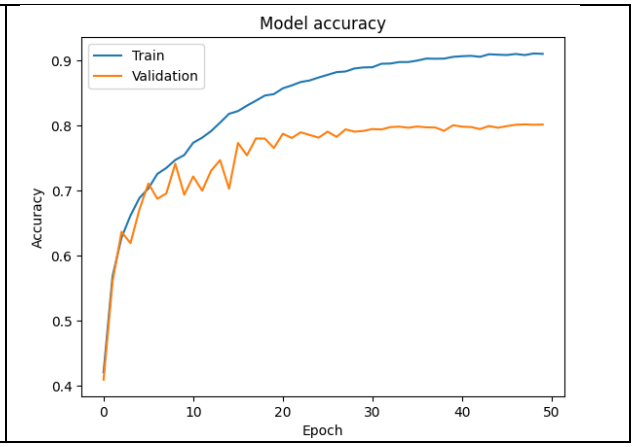


```
Test Loss: 0.6622140407562256
Test Accuracy: 0.7781999707221985
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
              precision    recall  f1-score   support

           0       0.85      0.79      0.82      1000
           1       0.93      0.87      0.90      1000
           2       0.73      0.68      0.70      1000
           3       0.58      0.65      0.61      1000
           4       0.62      0.86      0.72      1000
           5       0.72      0.66      0.69      1000
           6       0.76      0.89      0.82      1000
           7       0.93      0.68      0.78      1000
           8       0.92      0.87      0.89      1000
           9       0.90      0.84      0.87      1000

    accuracy                           0.78     10000
   macro avg       0.79      0.78      0.78     10000
weighted avg       0.79      0.78      0.78     10000

[[791  10  44  38  37   4  16   7  33  20]
 [  7 869   7   8   9   6  17   1  16  60]
 [ 46   2 676  50 100  45  66  10   4   1]
 [  8   1  56 654  88 111  73   6   1   2]
 [  3   0  34  38 864   7  44   7   2   1]
 [  1   1  37 200  70 657  24   9   1   0]
 [  1   0  29  36  35   7 888   1   2   1]
 [  8   0  25  46 173  61   9 677   0   1]
 [ 39  10  13  27   9   4  21   1 866  10]
 [ 23  41  11  32  13   7  10   7  16 840]]
```

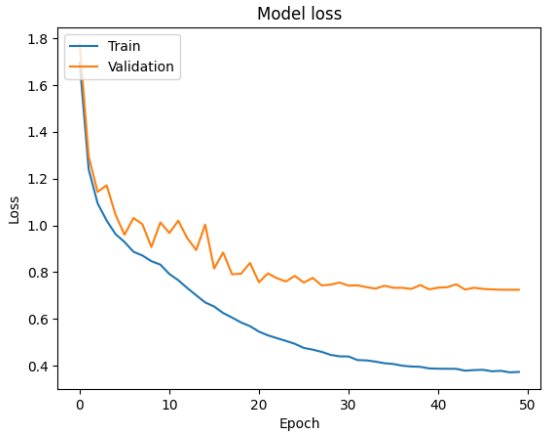


۴. مدل بهینه با استفاده از EarlyStopping، LR-Schecular، Batchnorm، L2 regularization و Dropout

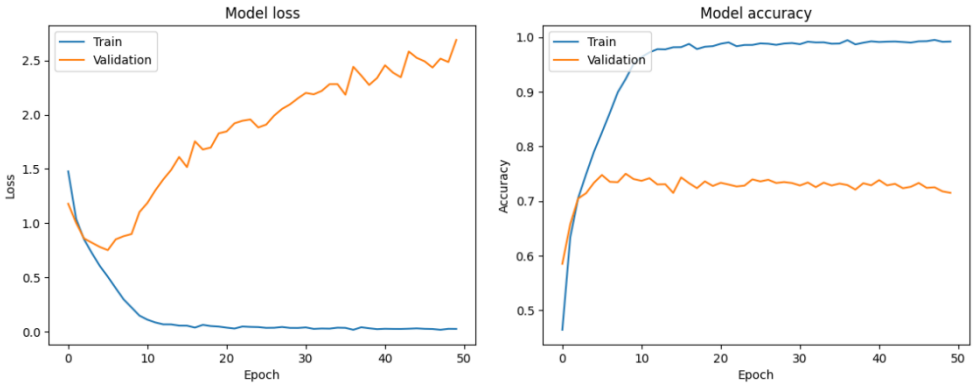
```
Trial 161 Complete [00h 02m 07s]
val_accuracy: 0.6280297636985779

Best val_accuracy So Far: 0.8000959753990173
Total elapsed time: 02h 32m 54s

The hyperparameter search is complete. The optimal parameters are:
    - Dropout 1: 0.2
    - Dropout 2: 0.2
    - Dropout 3: 0.30000000000000004
    - Dropout 4: 0.1
    - Learning Rate: 0.0014251629213975 99
    - Learning Rate: 6.9973906021894e-05
```



```
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 5ms/step - accuracy: 0.8059 - loss: 0.7169
Test Loss: 0.7348937988281250
Test Accuracy: 0.8000000119209290
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
              precision    recall  f1-score   support

           0       0.83      0.81      0.82      1000
           1       0.89      0.91      0.90      1000
           2       0.73      0.70      0.71      1000
           3       0.59      0.72      0.65      1000
           4       0.73      0.81      0.77      1000
           5       0.75      0.66      0.70      1000
           6       0.82      0.88      0.85      1000
           7       0.90      0.79      0.84      1000
           8       0.91      0.88      0.89      1000
           9       0.90      0.84      0.87      1000

    accuracy                           0.80     10000
   macro avg       0.81      0.80      0.80     10000
weighted avg       0.81      0.80      0.80     10000

[[810  15  47  23  32   3   9   5  36  20]
 [ 12 910   2  12   2   2   5   0  11  44]
 [ 44   1 699  74  72  34  56  12   5   3]
 [ 11   4  56 722  47  91  49  11   4   5]
 [  9   1  49  47 810  24  31  23   5   1]
 [  7   4  28 203  48 663  19  27   1   0]
 [  5   0  37  45  17   9 884   1   2   0]
 [ 10   2  26  47  70  49   7 785   1   3]
 [ 48  17   7  20   3   2   9   1 875  18]
 [ 25  69   7  22   3   4   4   7  17 842]]
```



تمرین ۳- در اجرای کد دفترچه‌ی "CNN(CIFAR-10).ipynb" (و "ANN(CIFAR-10).ipynb") می‌بینیم که مشابه شکل زیر پس از اجرای تعدادی دوره آموزش، Validation-Accuracy مدل ثابت می‌ماند و تغییری نمی‌کند. با این حال Validation-Loss مدل افزایش می‌یابد و حتی از مقدار اولیه خود نیز بیشتر می‌شود. چگونه این پدیده را تحلیل می‌کنید. آیا نباید با افزایش Loss، مقدار Accuracy کاهش یابد؟



نتیجه‌گیری:

با اضافه کردن هر کدام از این پارامترها (در سوال۲) به مدل، از Overfitting جلوگیری کردم که در نهایت مدل هم کاهش loss بسزایی دارد و هم accuracy بالاتری، یعنی در عین افزایش دقت، اطمینان (Confidance) مدل هم افزایش داده شده است.

بررسی بالانس بود:

```
Class distribution in training set:
[[   0 4164]
 [   1 4207]
 [   2 4161]
 [   3 4154]
 [   4 4153]
 [   5 4155]
 [   6 4144]
 [   7 4144]
 [   8 4195]
 [   9 4189]]
```

چون مدل می‌تواند دقت تقریباً یکسانی داشته باشد اما Confidanceهای متفاوتی داشته باشد که هرچقدر Confidance کمتر شود Loss بیشتر می‌شود.

این اتفاق به این دلیل می‌افتد که ما در آخرین لایه شبکه از **softmax** استفاده کرده‌ایم. softmax به تنها چیزی که نگاه می‌کند این است که در آخرین لایه کدام کلاس احتمال بیشتری دارد و آن به عنوان کلاس خروجی انتخاب می‌کند و از این رو اگر Confidance آنقدر کم شود که کلاس را اشتباه تشخیص ندهد (کلاس درست بیشترین احتمال را بین مابقی کلاس‌ها داشته باشد) و فقط اطمینان (Confidance) از کلاس انتخابی کاهش یابد، در نتیجه روی Accuracy اثری ندارد و مقدار Loss افزایش می‌یابد.

$$Multi-Classes:$$

$$loss = -\sum_{i=1}^{classes} y_i \cdot \log \hat{y}_i$$

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$$