name: amir mohammad arghavany
email: amir.arghavoon@gmail.com
dataset: CIFAR10

MAKTABKHOONEH
season 4 - project 2
deep learning

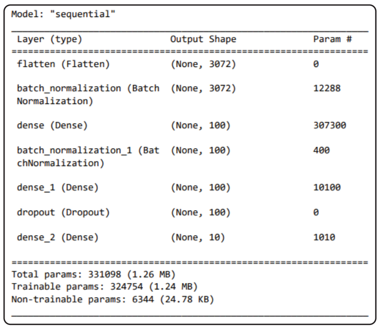
ANN - CNN
SGD -> exponantial vs 1cycle
pooling layer
stride 1 vs 2 vs 4
pretrained learning

شبکه کانولوشنال عمیقی برای پیش‌بینی کلاس‌ها در دیتاست CIFAR۱۰ بسازید. ۱۵ درصد دیتاست را برای تست در نظر بگیرید. عملکرد این شبکه را با بهترین شبکه فصل قبل مقایسه کنید.

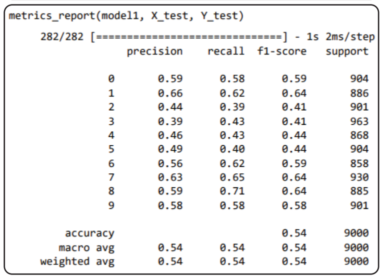
- تعداد لایه‌ها و تعداد و ابعاد کرنل‌های در نظر گرفته شده را گزارش کنید.
- مدت زمان آموزش شبکه را گزارش کرده و با شبکه Dense فصل قبل مقایسه کنید و accuracy پیش‌بینی را روی داده‌های تست گزارش کنید.

```
test size: 15.0 %
validation size 8.5 %
train size: 76.5 %
```

ANN



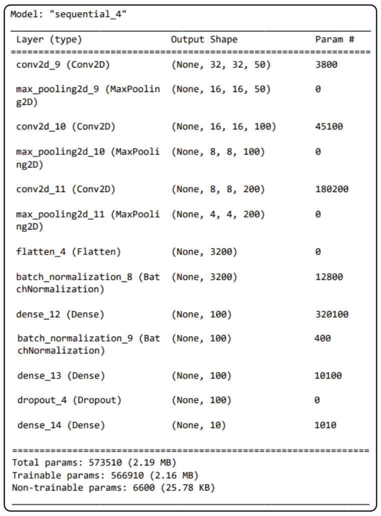
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 3072)              0

 batch_normalization (Batch  (None, 3072)              12288
 Normalization)

 dense (Dense)               (None, 100)               307300

 batch_normalization_1 (Bat  (None, 100)               400
 chNormalization)

 dense_1 (Dense)             (None, 100)               10100

 dropout (Dropout)           (None, 100)               0

 dense_2 (Dense)             (None, 10)                1010

=================================================================
Total params: 331098 (1.26 MB)
Trainable params: 324754 (1.24 MB)
Non-trainable params: 6344 (24.78 KB)
_________________________________________________________________
```



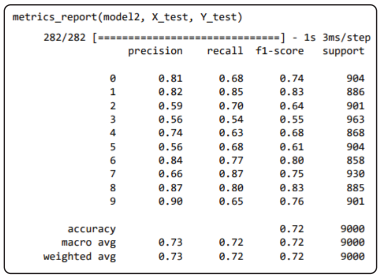
```
metrics_report(model1, X_test, Y_test)

     282/282 [==============================] - 1s 2ms/step
                  precision    recall  f1-score   support

               0       0.59      0.58      0.59       904
               1       0.66      0.62      0.64       886
               2       0.44      0.39      0.41       901
               3       0.39      0.43      0.41       963
               4       0.46      0.43      0.44       868
               5       0.49      0.40      0.44       904
               6       0.56      0.62      0.59       858
               7       0.63      0.65      0.64       930
               8       0.59      0.71      0.64       885
               9       0.58      0.58      0.58       901

        accuracy                           0.54      9000
       macro avg       0.54      0.54      0.54      9000
    weighted avg       0.54      0.54      0.54      9000
```

learning runtime: 167 sec

CNN



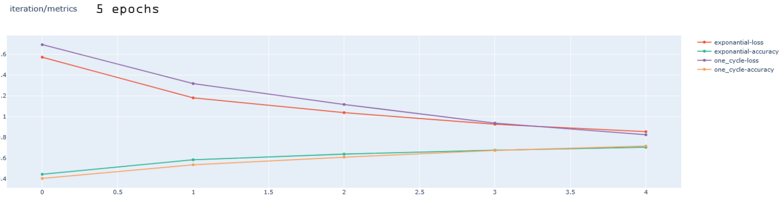
```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_9 (Conv2D)           (None, 32, 32, 50)        3800

 max_pooling2d_9 (MaxPoolin  (None, 16, 16, 50)        0
 g2D)

 conv2d_10 (Conv2D)          (None, 16, 16, 100)       45100

 max_pooling2d_10 (MaxPooli  (None, 8, 8, 100)         0
 ng2D)

 conv2d_11 (Conv2D)          (None, 8, 8, 200)         180200

 max_pooling2d_11 (MaxPooli  (None, 4, 4, 200)         0
 ng2D)

 flatten_4 (Flatten)         (None, 3200)              0

 batch_normalization_8 (Bat  (None, 3200)              12800
 chNormalization)

 dense_12 (Dense)            (None, 100)               320100

 batch_normalization_9 (Bat  (None, 100)               400
 chNormalization)

 dense_13 (Dense)            (None, 100)               10100

 dropout_4 (Dropout)         (None, 100)               0

 dense_14 (Dense)            (None, 10)                1010

=================================================================
Total params: 573510 (2.19 MB)
Trainable params: 566910 (2.16 MB)
Non-trainable params: 6600 (25.78 KB)
_________________________________________________________________
```



```
metrics_report(model2, X_test, Y_test)

     282/282 [==============================] - 1s 3ms/step
                  precision    recall  f1-score   support

               0       0.81      0.68      0.74       904
               1       0.82      0.85      0.83       886
               2       0.59      0.70      0.64       901
               3       0.56      0.54      0.55       963
               4       0.74      0.63      0.68       868
               5       0.56      0.68      0.61       904
               6       0.84      0.77      0.80       858
               7       0.66      0.87      0.75       930
               8       0.87      0.80      0.83       885
               9       0.90      0.65      0.76       901

        accuracy                           0.72      9000
       macro avg       0.73      0.72      0.72      9000
    weighted avg       0.73      0.72      0.72      9000
```

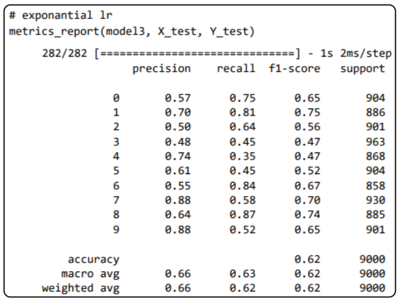
accuracy: CNN>ANN

learning runtime: 184 sec

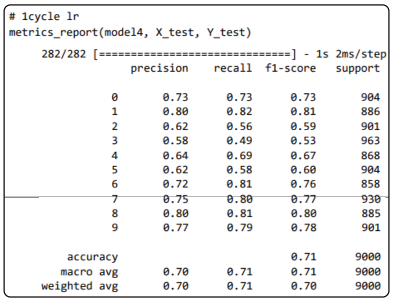


exponantial < 1cycle

learning runtime: 84 sec



```
# exponantial lr
metrics_report(model3, X_test, Y_test)

     282/282 [==============================] - 1s 2ms/step
                  precision    recall  f1-score   support

               0       0.57      0.75      0.65       904
               1       0.70      0.81      0.75       886
               2       0.50      0.64      0.56       901
               3       0.48      0.45      0.47       963
               4       0.74      0.35      0.47       868
               5       0.61      0.45      0.52       904
               6       0.55      0.84      0.67       858
               7       0.88      0.58      0.70       930
               8       0.64      0.87      0.74       885
               9       0.88      0.52      0.65       901

        accuracy                           0.62      9000
       macro avg       0.66      0.63      0.62      9000
    weighted avg       0.66      0.62      0.62      9000
```

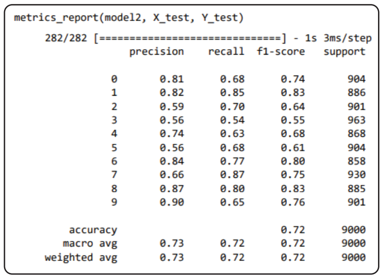


```
# 1cycle lr
metrics_report(model4, X_test, Y_test)

     282/282 [==============================] - 1s 2ms/step
                  precision    recall  f1-score   support

               0       0.73      0.73      0.73       904
               1       0.80      0.82      0.81       886
               2       0.62      0.56      0.59       901
               3       0.58      0.49      0.53       963
               4       0.64      0.69      0.67       868
               5       0.62      0.58      0.60       904
               6       0.72      0.81      0.76       858
               7       0.75      0.80      0.77       930
               8       0.80      0.81      0.80       885
               9       0.77      0.79      0.78       901

        accuracy                           0.71      9000
       macro avg       0.70      0.71      0.71      9000
    weighted avg       0.70      0.71      0.70      9000
```

accuracy: one_cycle_lr>exponantial_lr

- تفاوت حضور و عدم حضور Pooling layer روی شبکه را برای ۵۰ epoch بررسی کنید. مدت زمان آموزش را گزارش کنید و accuracy و loss را نمایش دهید.

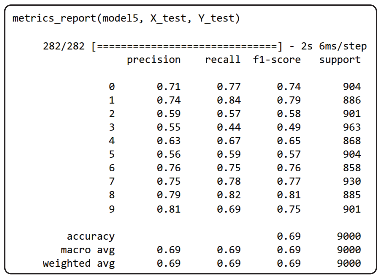
CNN
with pooling layers



```
metrics_report(model2, X_test, Y_test)

     282/282 [==============================] - 1s 3ms/step
                  precision    recall  f1-score   support

               0       0.81      0.68      0.74       904
               1       0.82      0.85      0.83       886
               2       0.59      0.70      0.64       901
               3       0.56      0.54      0.55       963
               4       0.74      0.63      0.68       868
               5       0.56      0.68      0.61       904
               6       0.84      0.77      0.80       858
               7       0.66      0.87      0.75       930
               8       0.87      0.80      0.83       885
               9       0.90      0.65      0.76       901

        accuracy                           0.72      9000
       macro avg       0.73      0.72      0.72      9000
    weighted avg       0.73      0.72      0.72      9000
```

learning runtime: 184 sec

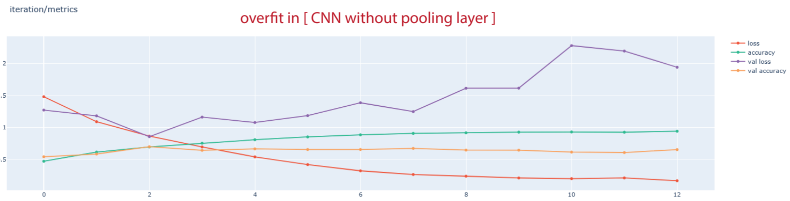
CNN
without pooling layers



```
metrics_report(model5, X_test, Y_test)

     282/282 [==============================] - 2s 6ms/step
                  precision    recall  f1-score   support

               0       0.71      0.77      0.74       904
               1       0.74      0.84      0.79       886
               2       0.59      0.57      0.58       901
               3       0.55      0.44      0.49       963
               4       0.63      0.67      0.65       868
               5       0.56      0.59      0.57       904
               6       0.76      0.75      0.76       858
               7       0.75      0.78      0.77       930
               8       0.79      0.82      0.81       885
               9       0.81      0.69      0.75       901

        accuracy                           0.69      9000
       macro avg       0.69      0.69      0.69      9000
    weighted avg       0.69      0.69      0.69      9000
```

learning runtime: 491 sec

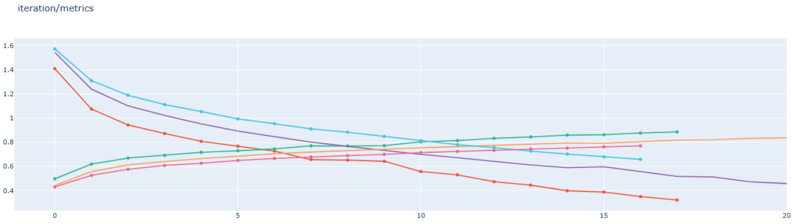
overfit in [CNN without pooling layer]



accuracy: CNN+pooling_layer>CNN (without pooling layer)

- روی لایه Pooling اثر strideهای ۲ و ۴ را برای ۵۰ epoch بررسی کنید و accuracy و loss را نمایش دهید.

stride=1 learning runtime: 184 sec
stride=2 learning runtime: 203 sec
stride=4 learning runtime: 157 sec

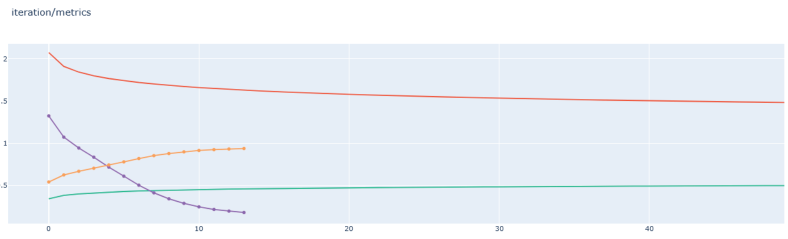


learning runtime stride2>stride1 -> because of early stopping
in theory -> stride1>stride2>stride4

accuracy: stride1>stride2>stride4

- [illegible] شبکه‌های قدرتمندی که در این فصل دیدید حل کنید. یک بار تمام شبکه را فریز کرده و بار دیگر یک لایه از آن [illegible] ه را برای ۵۰ epoch انجام دهید و accuracy و loss هر دو حالت را در یک گراف نمایش دهید.

1 activated layer learning runtime: 711 sec
2 activated layer learning runtime: 265 sec



accuracy: 2activated_layer>1activated_1layer