Neural Network Deep Learning

Assignment – 7

Name: Kishor Kumar Andekar

Student ID: 700744713

Github Link : https://github.com/kishorreyansh/Neural-Network-Deep-Learning/tree/main/Assignment-7

Lesson Overview:

In this lesson, we are going to discuss Image classification with CNN.

Use Case Description:

LeNet5, AlexNet, Vgg16, Vgg19

1. Training the model

2. Evaluating the model

Programming elements:

1. About CNN

2. Hyperparameters of CNN

3. Image classification with CNN

In class programming:

1. Tune hyperparameter and make necessary addition to the baseline model to improve validation accuracy

and reduce validation loss.

2. Provide logical description of which steps lead to improved response and what was its impact on

architecture behavior.

3. Create at least two more visualizations using matplotlib (Other than provided in the source file)

4. Use dataset of your own choice and implement baseline models provided.

5. Apply modified architecture to your own selected dataset and train it.

6. Evaluate your model on testing set.

7. Save the improved model and use it for prediction on testing data

8. Provide plot of confusion matric

9. Provide Training and testing Loss and accuracy plots in one plot using subplot command and history object.

10. Provide at least two more visualizations reflecting your solution.

11. Provide logical description of which steps lead to improved response for new dataset when compared with

baseline model and enhance architecture and what was its impact on architecture behavior

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

T4 RAM Disk Colab AI

```
x_train.shape
```

```
(50000, 32, 32, 3)
```

```
import tensorflow as tf
from tensorflow.keras import layers, models

lenet = models.Sequential([
    layers.Conv2D(6, kernel_size=5, strides=1, activation='relu', input_shape=(32,32,3), padding='same'), #C1
    layers.AveragePooling2D(pool_size=(2, 2)), #S1
    layers.Conv2D(16, kernel_size=5, strides=1, activation='relu', padding='valid'), #C2
    layers.AveragePooling2D(pool_size=(2, 2)), #S2
    layers.Conv2D(120, kernel_size=5, strides=1, activation='relu', padding='valid'), #C3
    layers.Flatten(), #Flatten
    layers.Dense(84, activation='relu'), #F1
    layers.Dense(10, activation='softmax') #Output layer
])
```

colab.research.google.com/drive/1VEZJamCiIQ1PZzlYkIEQhR9Vq04wWI00?userstoinvite=mxg93480%40ucmo.edu&sharingaction=manageaccess&role=writer

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

T4 RAM Disk Colab AI

```
lenet.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 6)         456

 average_pooling2d (Average  (None, 16, 16, 6)         0
 Pooling2D)

 conv2d_1 (Conv2D)           (None, 12, 12, 16)        2416

 average_pooling2d_1 (Avera  (None, 6, 6, 16)          0
 gePooling2D)

 conv2d_2 (Conv2D)           (None, 2, 2, 120)         48120

 flatten (Flatten)           (None, 480)               0

 dense (Dense)               (None, 84)                40404

 dense_1 (Dense)             (None, 10)                850

=================================================================
Total params: 92246 (360.34 KB)
Trainable params: 92246 (360.34 KB)
Non-trainable params: 0 (0.00 Byte)
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

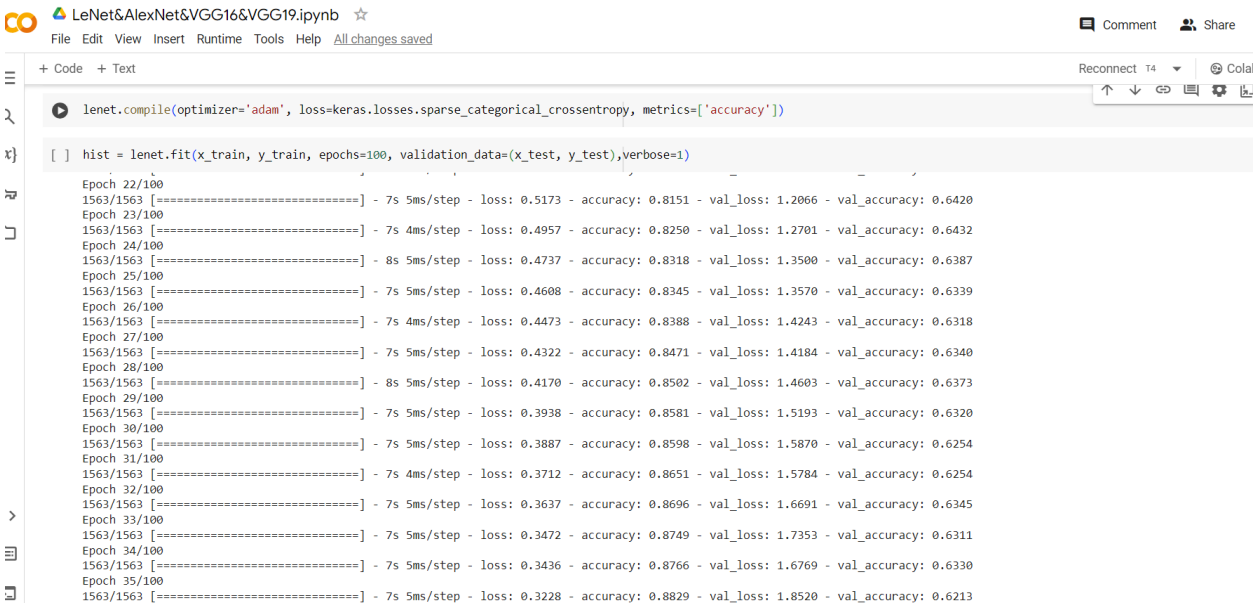
+ Code + Text

T4 RAM Disk Colab AI

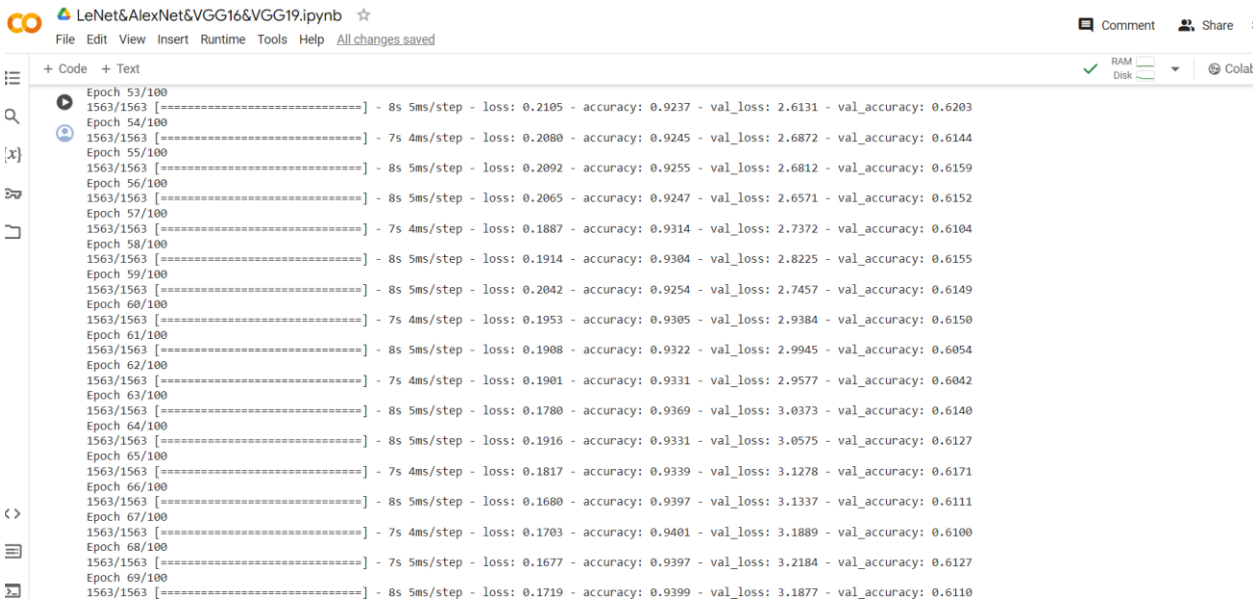
```
lenet.compile(optimizer='adam', loss=keras.losses.sparse_categorical_crossentropy, metrics=['accuracy'])
```

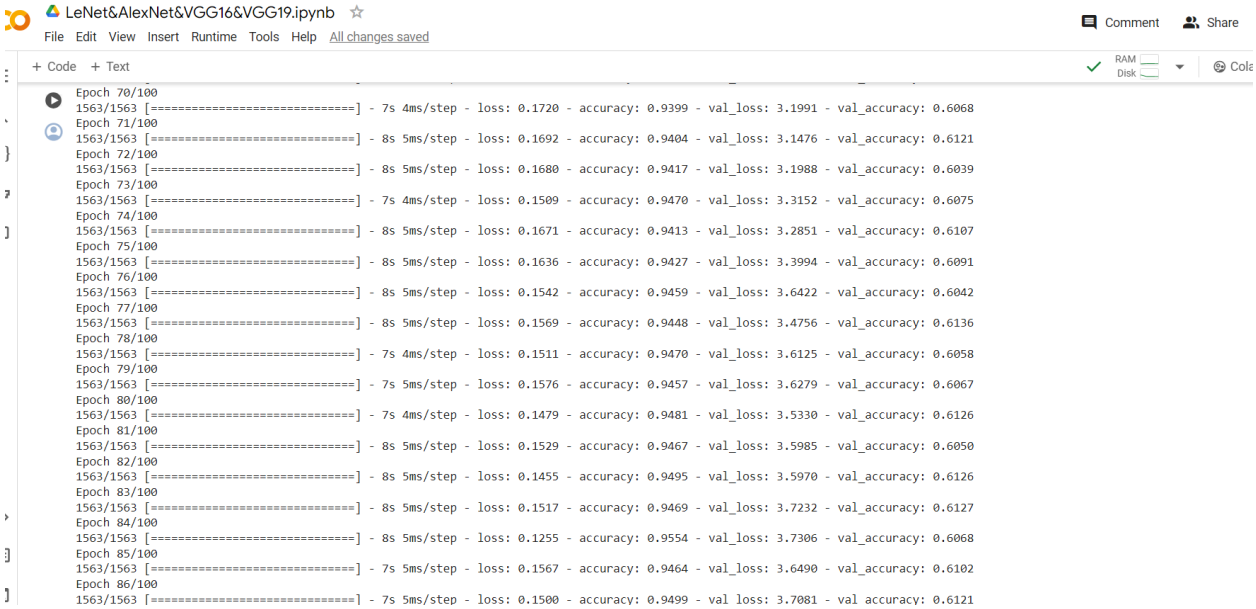
```
hist = lenet.fit(x_train, y_train, epochs=100, validation_data=(x_test, y_test),verbose=1)
```

```
1563/1563 [==============================] - 8s 5ms/step - loss: 0.9483 - accuracy: 0.6633 - val_loss: 1.0637 - val_accuracy: 0.6285
Epoch 8/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.9056 - accuracy: 0.6775 - val_loss: 1.0436 - val_accuracy: 0.6381
Epoch 9/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.8644 - accuracy: 0.6930 - val_loss: 1.0821 - val_accuracy: 0.6260
Epoch 10/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.8233 - accuracy: 0.7086 - val_loss: 1.0242 - val_accuracy: 0.6482
Epoch 11/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.7933 - accuracy: 0.7186 - val_loss: 1.0355 - val_accuracy: 0.6453
Epoch 12/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.7588 - accuracy: 0.7311 - val_loss: 1.0554 - val_accuracy: 0.6455
Epoch 13/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.7300 - accuracy: 0.7419 - val_loss: 1.0362 - val_accuracy: 0.6495
Epoch 14/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.6984 - accuracy: 0.7531 - val_loss: 1.0579 - val_accuracy: 0.6600
Epoch 15/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.6713 - accuracy: 0.7604 - val_loss: 1.0926 - val_accuracy: 0.6483
Epoch 16/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.6468 - accuracy: 0.7702 - val_loss: 1.0843 - val_accuracy: 0.6503
Epoch 17/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.6208 - accuracy: 0.7787 - val_loss: 1.1190 - val_accuracy: 0.6460
Epoch 18/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.5974 - accuracy: 0.7893 - val_loss: 1.1425 - val_accuracy: 0.6403
Epoch 19/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.5784 - accuracy: 0.7947 - val_loss: 1.1248 - val_accuracy: 0.6542
Epoch 20/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.5514 - accuracy: 0.8034 - val_loss: 1.1669 - val_accuracy: 0.6470
Epoch 21/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.5339 - accuracy: 0.8092 - val_loss: 1.2164 - val_accuracy: 0.6478
```


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Reconnect
lenet.compile(optimizer='adam', loss=keras.losses.sparse_categorical_crossentropy, metrics=['accuracy'])
[ ] hist = lenet.fit(x_train, y_train, epochs=100, validation_data=(x_test, y_test),verbose=1)
Epoch 22/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.5173 - accuracy: 0.8151 - val_loss: 1.2066 - val_accuracy: 0.6420
Epoch 23/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.4957 - accuracy: 0.8250 - val_loss: 1.2701 - val_accuracy: 0.6432
Epoch 24/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.4737 - accuracy: 0.8318 - val_loss: 1.3500 - val_accuracy: 0.6387
Epoch 25/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.4608 - accuracy: 0.8345 - val_loss: 1.3570 - val_accuracy: 0.6339
Epoch 26/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.4473 - accuracy: 0.8388 - val_loss: 1.4243 - val_accuracy: 0.6318
Epoch 27/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.4322 - accuracy: 0.8471 - val_loss: 1.4184 - val_accuracy: 0.6340
Epoch 28/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.4170 - accuracy: 0.8502 - val_loss: 1.4603 - val_accuracy: 0.6373
Epoch 29/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3938 - accuracy: 0.8581 - val_loss: 1.5193 - val_accuracy: 0.6320
Epoch 30/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3887 - accuracy: 0.8598 - val_loss: 1.5870 - val_accuracy: 0.6254
Epoch 31/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.3712 - accuracy: 0.8651 - val_loss: 1.5784 - val_accuracy: 0.6254
Epoch 32/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3637 - accuracy: 0.8696 - val_loss: 1.6691 - val_accuracy: 0.6345
Epoch 33/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3472 - accuracy: 0.8749 - val_loss: 1.7353 - val_accuracy: 0.6311
Epoch 34/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3436 - accuracy: 0.8766 - val_loss: 1.6769 - val_accuracy: 0.6330
Epoch 35/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.3228 - accuracy: 0.8829 - val_loss: 1.8520 - val_accuracy: 0.6213


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
+ Code + Text
RAM
Disk
Epoch 36/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.3212 - accuracy: 0.8827 - val_loss: 1.8635 - val_accuracy: 0.6222
Epoch 37/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.3143 - accuracy: 0.8873 - val_loss: 1.9245 - val_accuracy: 0.6203
Epoch 38/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2983 - accuracy: 0.8913 - val_loss: 1.9481 - val_accuracy: 0.6238
Epoch 39/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.2883 - accuracy: 0.8959 - val_loss: 1.9932 - val_accuracy: 0.6286
Epoch 40/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2885 - accuracy: 0.8966 - val_loss: 1.9895 - val_accuracy: 0.6231
Epoch 41/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2815 - accuracy: 0.8979 - val_loss: 2.0099 - val_accuracy: 0.6209
Epoch 42/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.2634 - accuracy: 0.9046 - val_loss: 2.0627 - val_accuracy: 0.6207
Epoch 43/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.2619 - accuracy: 0.9049 - val_loss: 2.1620 - val_accuracy: 0.6254
Epoch 44/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.2487 - accuracy: 0.9099 - val_loss: 2.1874 - val_accuracy: 0.6238
Epoch 45/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2543 - accuracy: 0.9087 - val_loss: 2.3490 - val_accuracy: 0.6173
Epoch 46/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.2427 - accuracy: 0.9116 - val_loss: 2.2998 - val_accuracy: 0.6225
Epoch 47/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2363 - accuracy: 0.9145 - val_loss: 2.3125 - val_accuracy: 0.6155
Epoch 48/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2396 - accuracy: 0.9117 - val_loss: 2.4779 - val_accuracy: 0.6140
Epoch 49/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.2282 - accuracy: 0.9181 - val_loss: 2.5237 - val_accuracy: 0.6082
Epoch 50/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2234 - accuracy: 0.9198 - val_loss: 2.4989 - val_accuracy: 0.6151
Epoch 51/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.2183 - accuracy: 0.9211 - val_loss: 2.5813 - val_accuracy: 0.6112
Epoch 52/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.2235 - accuracy: 0.9191 - val_loss: 2.4816 - val_accuracy: 0.6167


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Epoch 53/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.2105 - accuracy: 0.9237 - val_loss: 2.6131 - val_accuracy: 0.6203
Epoch 54/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.2080 - accuracy: 0.9245 - val_loss: 2.6872 - val_accuracy: 0.6144
Epoch 55/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.2092 - accuracy: 0.9255 - val_loss: 2.6812 - val_accuracy: 0.6159
Epoch 56/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.2065 - accuracy: 0.9247 - val_loss: 2.6571 - val_accuracy: 0.6152
Epoch 57/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1887 - accuracy: 0.9314 - val_loss: 2.7372 - val_accuracy: 0.6104
Epoch 58/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1914 - accuracy: 0.9304 - val_loss: 2.8225 - val_accuracy: 0.6155
Epoch 59/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.2042 - accuracy: 0.9254 - val_loss: 2.7457 - val_accuracy: 0.6149
Epoch 60/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1953 - accuracy: 0.9305 - val_loss: 2.9384 - val_accuracy: 0.6150
Epoch 61/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1908 - accuracy: 0.9322 - val_loss: 2.9945 - val_accuracy: 0.6054
Epoch 62/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1901 - accuracy: 0.9331 - val_loss: 2.9577 - val_accuracy: 0.6042
Epoch 63/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1780 - accuracy: 0.9369 - val_loss: 3.0373 - val_accuracy: 0.6140
Epoch 64/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1916 - accuracy: 0.9331 - val_loss: 3.0575 - val_accuracy: 0.6127
Epoch 65/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1817 - accuracy: 0.9339 - val_loss: 3.1278 - val_accuracy: 0.6171
Epoch 66/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1680 - accuracy: 0.9397 - val_loss: 3.1337 - val_accuracy: 0.6111
Epoch 67/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1703 - accuracy: 0.9401 - val_loss: 3.1889 - val_accuracy: 0.6100
Epoch 68/100
1563/1563 [==========] - 7s 5ms/step - loss: 0.1677 - accuracy: 0.9397 - val_loss: 3.2184 - val_accuracy: 0.6127
Epoch 69/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1719 - accuracy: 0.9399 - val_loss: 3.1877 - val_accuracy: 0.6110


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Epoch 70/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1720 - accuracy: 0.9399 - val_loss: 3.1991 - val_accuracy: 0.6068
Epoch 71/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1692 - accuracy: 0.9404 - val_loss: 3.1476 - val_accuracy: 0.6121
Epoch 72/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1680 - accuracy: 0.9417 - val_loss: 3.1988 - val_accuracy: 0.6039
Epoch 73/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1509 - accuracy: 0.9470 - val_loss: 3.3152 - val_accuracy: 0.6075
Epoch 74/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1671 - accuracy: 0.9413 - val_loss: 3.2851 - val_accuracy: 0.6107
Epoch 75/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1636 - accuracy: 0.9427 - val_loss: 3.3994 - val_accuracy: 0.6091
Epoch 76/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1542 - accuracy: 0.9459 - val_loss: 3.6422 - val_accuracy: 0.6042
Epoch 77/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1569 - accuracy: 0.9448 - val_loss: 3.4756 - val_accuracy: 0.6136
Epoch 78/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1511 - accuracy: 0.9470 - val_loss: 3.6125 - val_accuracy: 0.6058
Epoch 79/100
1563/1563 [==========] - 7s 5ms/step - loss: 0.1576 - accuracy: 0.9457 - val_loss: 3.6279 - val_accuracy: 0.6067
Epoch 80/100
1563/1563 [==========] - 7s 4ms/step - loss: 0.1479 - accuracy: 0.9481 - val_loss: 3.5330 - val_accuracy: 0.6126
Epoch 81/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1529 - accuracy: 0.9467 - val_loss: 3.5985 - val_accuracy: 0.6050
Epoch 82/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1455 - accuracy: 0.9495 - val_loss: 3.5970 - val_accuracy: 0.6126
Epoch 83/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1517 - accuracy: 0.9469 - val_loss: 3.7232 - val_accuracy: 0.6127
Epoch 84/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.1255 - accuracy: 0.9554 - val_loss: 3.7306 - val_accuracy: 0.6068
Epoch 85/100
1563/1563 [==========] - 7s 5ms/step - loss: 0.1567 - accuracy: 0.9464 - val_loss: 3.6490 - val_accuracy: 0.6102
Epoch 86/100
1563/1563 [==========] - 7s 5ms/step - loss: 0.1500 - accuracy: 0.9499 - val_loss: 3.7081 - val_accuracy: 0.6121

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
Epoch 87/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1435 - accuracy: 0.9515 - val_loss: 3.7734 - val_accuracy: 0.6077
Epoch 88/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.1366 - accuracy: 0.9513 - val_loss: 3.8923 - val_accuracy: 0.5998
Epoch 89/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1409 - accuracy: 0.9512 - val_loss: 3.8005 - val_accuracy: 0.6069
Epoch 90/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1512 - accuracy: 0.9481 - val_loss: 3.8571 - val_accuracy: 0.6044
Epoch 91/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.1262 - accuracy: 0.9568 - val_loss: 3.9459 - val_accuracy: 0.6093
Epoch 92/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.1413 - accuracy: 0.9527 - val_loss: 3.9523 - val_accuracy: 0.6078
Epoch 93/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1371 - accuracy: 0.9547 - val_loss: 3.9131 - val_accuracy: 0.5917
Epoch 94/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1308 - accuracy: 0.9553 - val_loss: 4.0008 - val_accuracy: 0.6081
Epoch 95/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1352 - accuracy: 0.9544 - val_loss: 4.1549 - val_accuracy: 0.6075
Epoch 96/100
1563/1563 [==============================] - 7s 4ms/step - loss: 0.1265 - accuracy: 0.9573 - val_loss: 3.9909 - val_accuracy: 0.6077
Epoch 97/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1411 - accuracy: 0.9517 - val_loss: 3.9561 - val_accuracy: 0.6089
Epoch 98/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.1355 - accuracy: 0.9549 - val_loss: 3.9746 - val_accuracy: 0.6127
Epoch 99/100
1563/1563 [==============================] - 7s 5ms/step - loss: 0.1365 - accuracy: 0.9546 - val_loss: 4.1249 - val_accuracy: 0.6087
Epoch 100/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.1244 - accuracy: 0.9573 - val_loss: 4.0572 - val_accuracy: 0.5995
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
import numpy as np
```

```
# fix random seed for reproducibility
seed = 7
np.random.seed(seed)
```

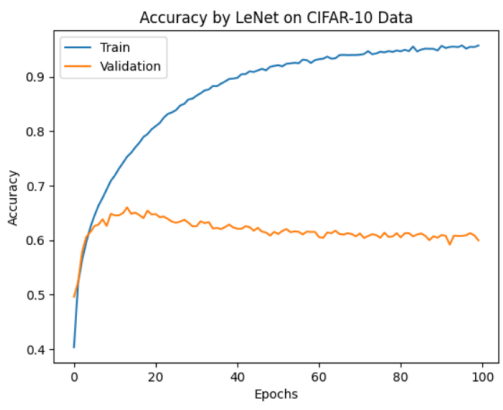
```
# summarize history for accuracy
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title("Accuracy by LeNet on CIFAR-10 Data")
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Loss by LeNet on CIFAR-10 Data')
plt.ylabel('Loss')
plt.xlabel('Epochs')
plt.legend(['Train', 'Validation'])
plt.show()
```

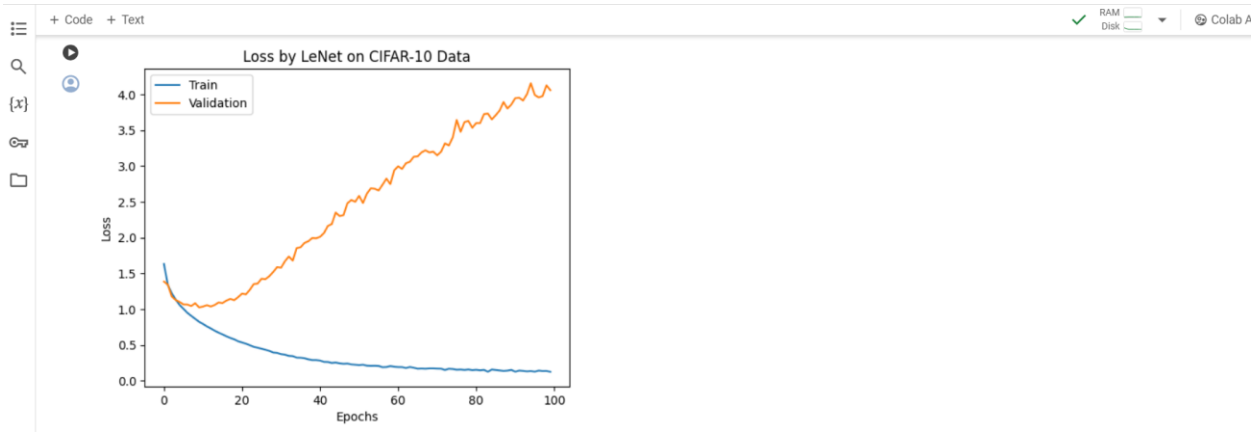
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Sh

+ Code + Text





LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab AI

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
y_predictions= lenet.predict(x_test)
y_predictions.reshape(-1,)
y_predictions= np.argmax(y_predictions, axis=1)

confusion_matrix(y_test, y_predictions)
```

```
313/313 [==============================] - 1s 2ms/step
array([[619,  15,  91,  40,  22,  14,  21,  11, 113,  54],
       [ 23, 731,  21,  16,   6,  12,  19,   5,  48, 119],
       [ 57,   6, 531,  97,  86,  87,  71,  30,  21,  14],
       [ 27,  12, 106, 439,  65, 179,  92,  32,  17,  31],
       [ 33,   3, 143,  95, 493,  73,  66,  64,  22,   8],
       [ 13,   8, 131, 211,  67, 453,  41,  50,  11,  15],
       [  8,   3,  92, 106,  48,  36, 666,  17,   9,  15],
       [ 19,   6,  50,  73, 110,  78,  17, 613,   6,  28],
       [ 80,  30,  30,  27,  19,   6,  21,   4, 748,  35],
       [ 43,  90,  34,  31,   7,  14,  18,  14,  47, 702]])
```

```
[ ] # confusion matrix and accuracy
from sklearn.metrics import confusion_matrix, accuracy_score
plt.figure(figsize=(7, 6))
plt.title('Confusion matrix', fontsize=16)
plt.imshow(confusion_matrix(y_test, y_predictions))
plt.xticks(np.arange(10), classes, rotation=45, fontsize=12)
plt.yticks(np.arange(10), classes, fontsize=12)
plt.colorbar()
plt.show()
```

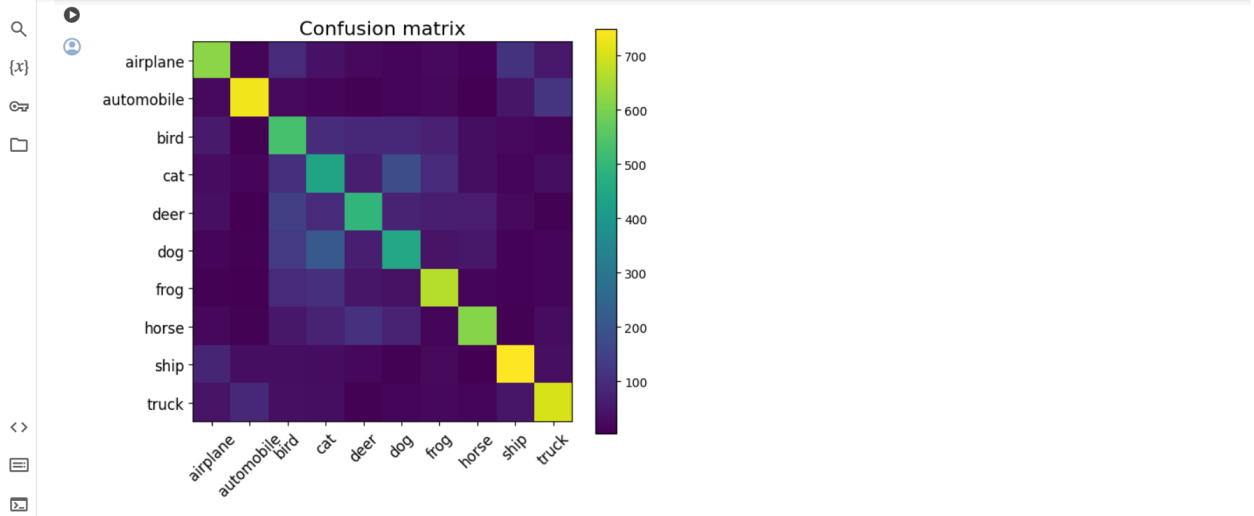
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk



LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab AI

```
print("Test accuracy:", accuracy_score(y_test, y_predictions))
```

```
Test accuracy: 0.5995
```

```
[ ] L = 8
    W = 8
    fig, axes = plt.subplots(L, W, figsize = (20,20))
    axes = axes.ravel() #

    for i in np.arange(0, L * W):
        axes[i].imshow(x_test[i])
        axes[i].set_title("Predicted = {}\n Actual  = {}".format(classes[y_predictions[i]], classes[y_test[i]]))
        axes[i].axis('off')

    plt.subplots_adjust(wspace=1)
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab AI

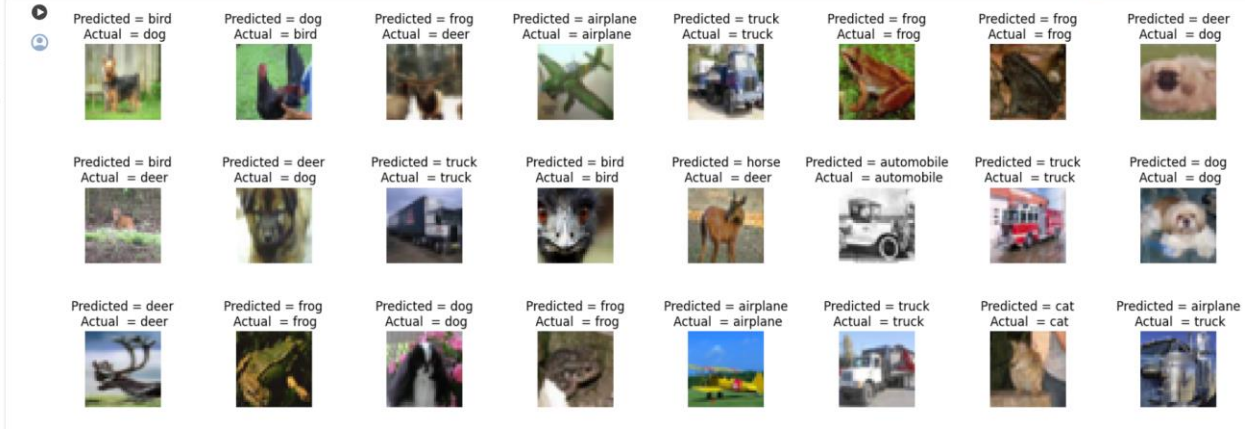
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab AI



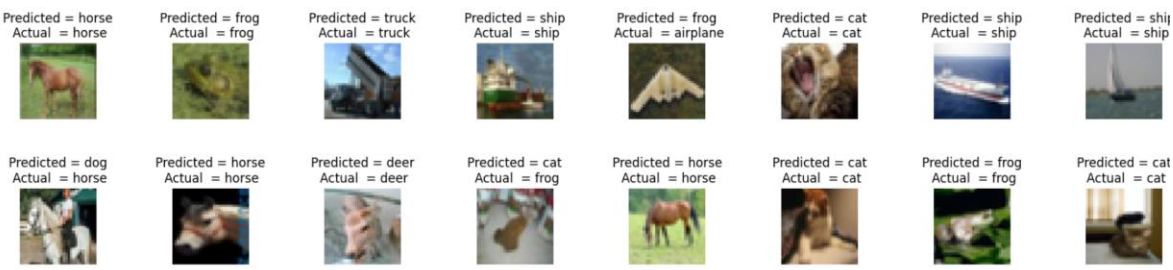
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab AI




```
[ ] from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten
    from tensorflow.keras.optimizers import SGD, Adam
    from tensorflow.keras.layers import Convolution2D as Conv2D
    from tensorflow.keras.layers import MaxPooling2D
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Colab A

```
#Define Alexnet Model
AlexNet = Sequential()
AlexNet.add(Conv2D(filters=16,kernel_size=(3,3),strides=(4,4),input_shape=(32,32,3), activation='relu'))
AlexNet.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
AlexNet.add(Conv2D(60,(5,5),padding='same',activation='relu'))
AlexNet.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
AlexNet.add(Conv2D(60,(3,3),padding='same',activation='relu'))
AlexNet.add(Conv2D(30,(3,3),padding='same',activation='relu'))
AlexNet.add(Conv2D(20,(3,3),padding='same',activation='relu'))
AlexNet.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
AlexNet.add(Flatten())
AlexNet.add(Dense(200, activation='relu'))
AlexNet.add(Dropout(0.1))
AlexNet.add(Dense(200, activation='relu'))
AlexNet.add(Dropout(0.1))
AlexNet.add(Dense(10,activation='softmax'))

AlexNet.compile(optimizer='SGD', loss=keras.losses.sparse_categorical_crossentropy, metrics=['accuracy'])
AlexNet.summary()
```

LeNet&AlexNet&VGG16&VGG19.ipynb

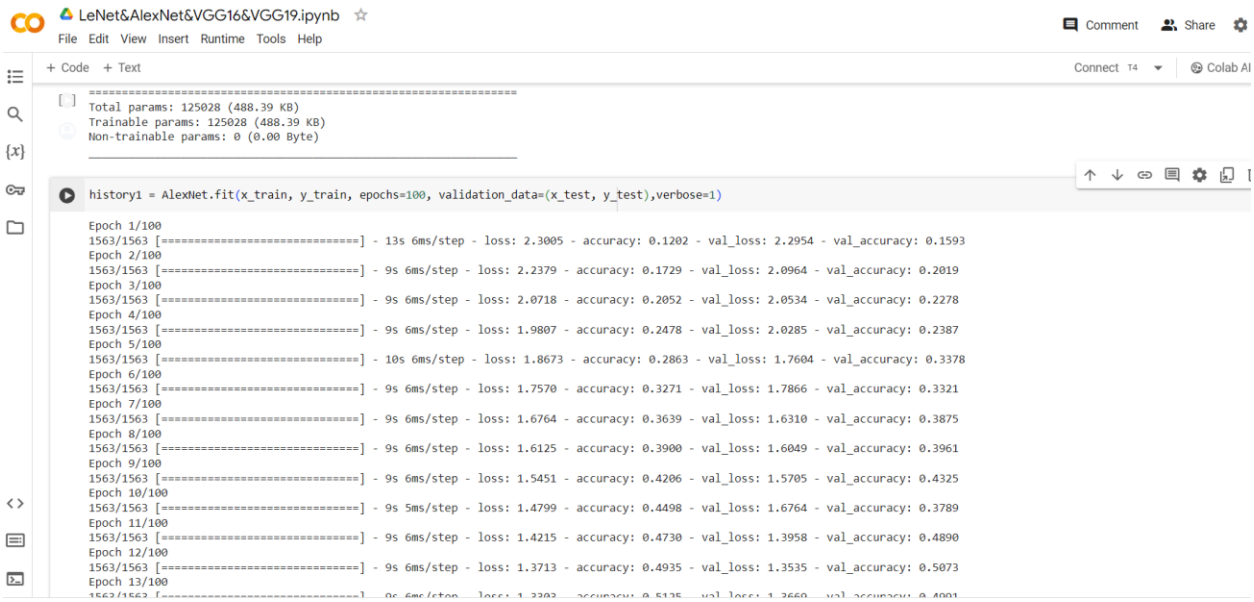
File Edit View Insert Runtime Tools Help All changes saved

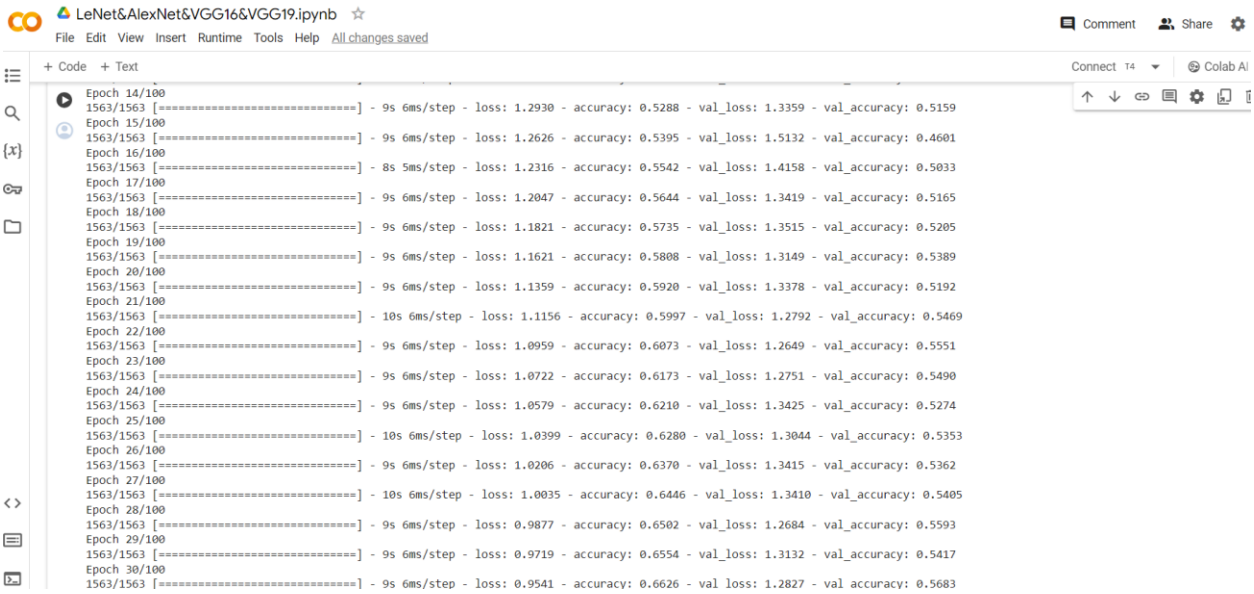
Comment Share

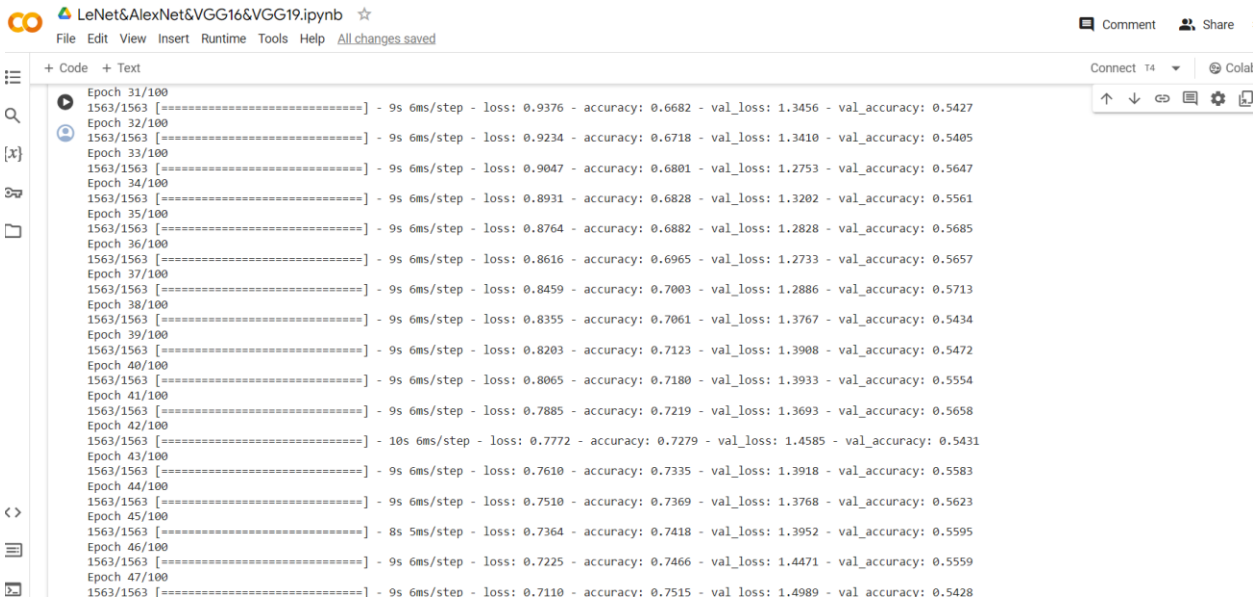
+ Code + Text

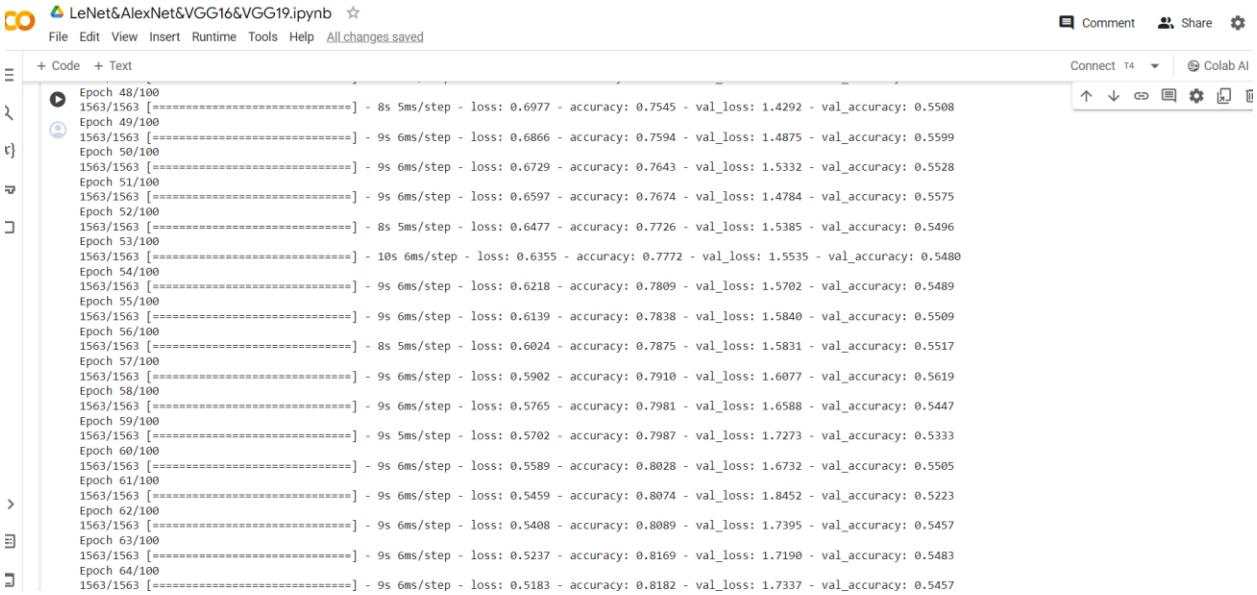
RAM Disk Colab AI

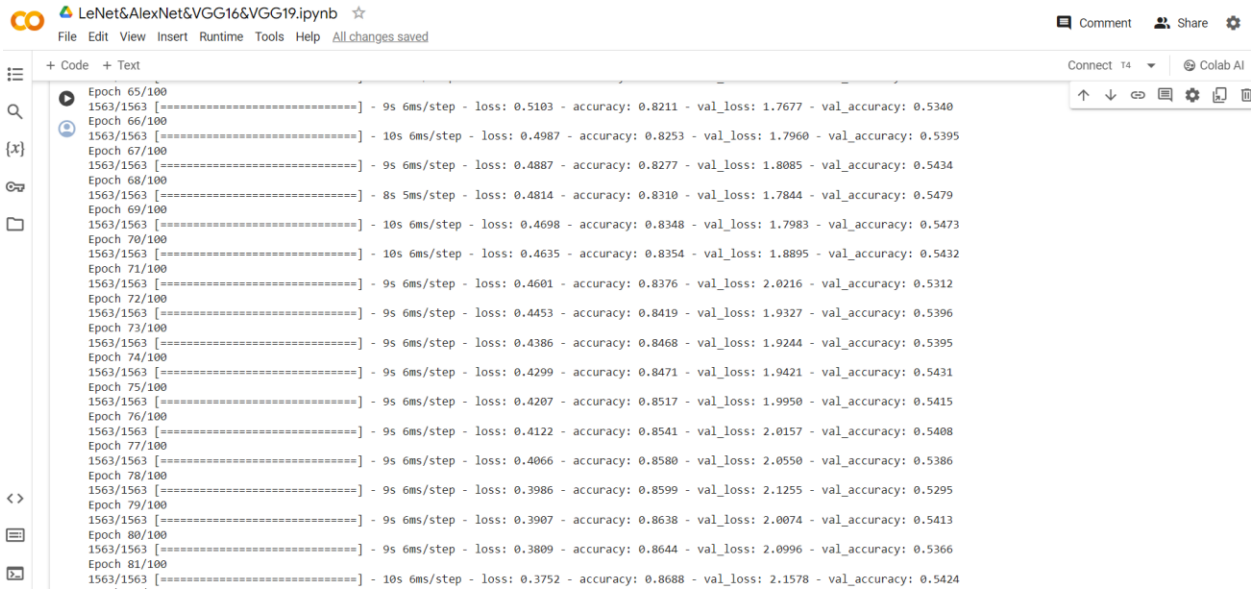
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 8, 8, 16)          448

max_pooling2d (MaxPooling2   (None, 4, 4, 16)          0
D)

conv2d_4 (Conv2D)            (None, 4, 4, 60)          24060

max_pooling2d_1 (MaxPoolin   (None, 2, 2, 60)          0
g2D)

conv2d_5 (Conv2D)            (None, 2, 2, 60)          32460

conv2d_6 (Conv2D)            (None, 2, 2, 30)          16230

conv2d_7 (Conv2D)            (None, 2, 2, 20)          5420

max_pooling2d_2 (MaxPoolin   (None, 1, 1, 20)          0
g2D)

flatten_1 (Flatten)          (None, 20)                0

dense_2 (Dense)              (None, 200)               4200

dropout (Dropout)            (None, 200)               0

dense_3 (Dense)              (None, 200)               40200

dropout_1 (Dropout)          (None, 200)               0

dense_4 (Dense)              (None, 10)                2010
```

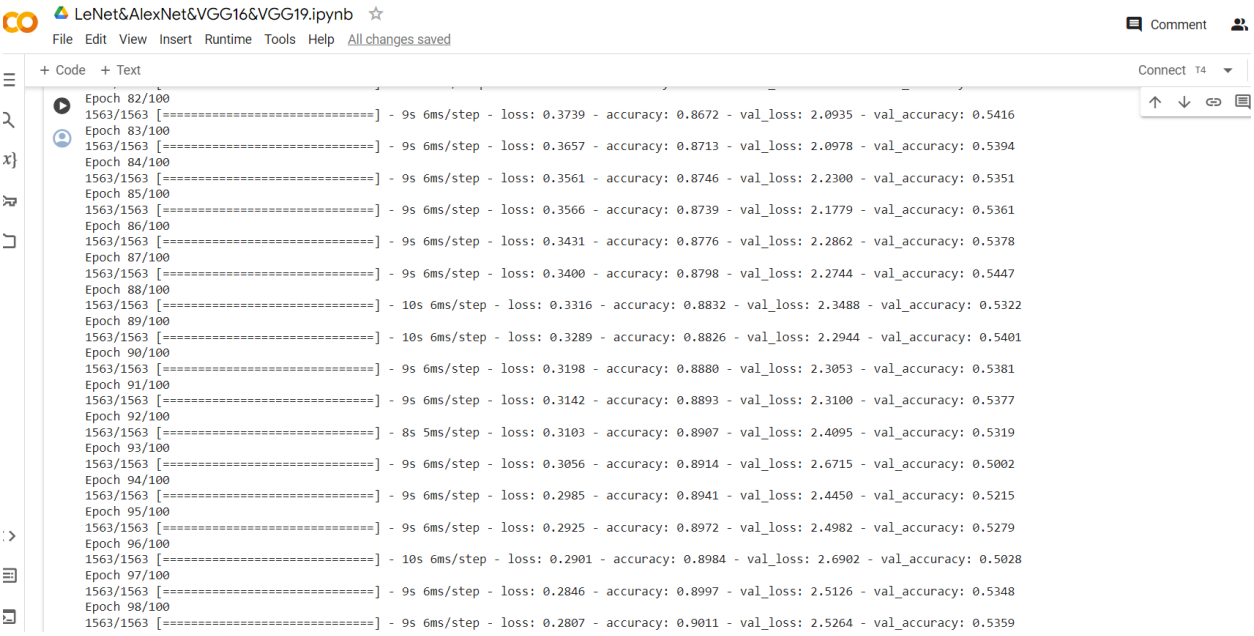

LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
+ Code + Text
Connect T4
Colab AI
Total params: 125028 (488.39 KB)
Trainable params: 125028 (488.39 KB)
Non-trainable params: 0 (0.00 Byte)
history1 = AlexNet.fit(x_train, y_train, epochs=100, validation_data=(x_test, y_test),verbose=1)
Epoch 1/100
1563/1563 [==========] - 13s 6ms/step - loss: 2.3005 - accuracy: 0.1202 - val_loss: 2.2954 - val_accuracy: 0.1593
Epoch 2/100
1563/1563 [==========] - 9s 6ms/step - loss: 2.2379 - accuracy: 0.1729 - val_loss: 2.0964 - val_accuracy: 0.2019
Epoch 3/100
1563/1563 [==========] - 9s 6ms/step - loss: 2.0718 - accuracy: 0.2052 - val_loss: 2.0534 - val_accuracy: 0.2278
Epoch 4/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.9807 - accuracy: 0.2478 - val_loss: 2.0285 - val_accuracy: 0.2387
Epoch 5/100
1563/1563 [==========] - 10s 6ms/step - loss: 1.8673 - accuracy: 0.2863 - val_loss: 1.7604 - val_accuracy: 0.3378
Epoch 6/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.7570 - accuracy: 0.3271 - val_loss: 1.7866 - val_accuracy: 0.3321
Epoch 7/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.6764 - accuracy: 0.3639 - val_loss: 1.6310 - val_accuracy: 0.3875
Epoch 8/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.6125 - accuracy: 0.3900 - val_loss: 1.6049 - val_accuracy: 0.3961
Epoch 9/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.5451 - accuracy: 0.4206 - val_loss: 1.5705 - val_accuracy: 0.4325
Epoch 10/100
1563/1563 [==========] - 9s 5ms/step - loss: 1.4799 - accuracy: 0.4498 - val_loss: 1.6764 - val_accuracy: 0.3789
Epoch 11/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.4215 - accuracy: 0.4730 - val_loss: 1.3958 - val_accuracy: 0.4890
Epoch 12/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.3713 - accuracy: 0.4935 - val_loss: 1.3535 - val_accuracy: 0.5073
Epoch 13/100


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Colab AI
Epoch 14/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.2930 - accuracy: 0.5288 - val_loss: 1.3359 - val_accuracy: 0.5159
Epoch 15/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.2626 - accuracy: 0.5395 - val_loss: 1.5132 - val_accuracy: 0.4601
Epoch 16/100
1563/1563 [==========] - 8s 5ms/step - loss: 1.2316 - accuracy: 0.5542 - val_loss: 1.4158 - val_accuracy: 0.5033
Epoch 17/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.2047 - accuracy: 0.5644 - val_loss: 1.3419 - val_accuracy: 0.5165
Epoch 18/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.1821 - accuracy: 0.5735 - val_loss: 1.3515 - val_accuracy: 0.5205
Epoch 19/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.1621 - accuracy: 0.5808 - val_loss: 1.3149 - val_accuracy: 0.5389
Epoch 20/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.1359 - accuracy: 0.5920 - val_loss: 1.3378 - val_accuracy: 0.5192
Epoch 21/100
1563/1563 [==========] - 10s 6ms/step - loss: 1.1156 - accuracy: 0.5997 - val_loss: 1.2792 - val_accuracy: 0.5469
Epoch 22/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.0959 - accuracy: 0.6073 - val_loss: 1.2649 - val_accuracy: 0.5551
Epoch 23/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.0722 - accuracy: 0.6173 - val_loss: 1.2751 - val_accuracy: 0.5490
Epoch 24/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.0579 - accuracy: 0.6210 - val_loss: 1.3425 - val_accuracy: 0.5274
Epoch 25/100
1563/1563 [==========] - 10s 6ms/step - loss: 1.0399 - accuracy: 0.6280 - val_loss: 1.3044 - val_accuracy: 0.5353
Epoch 26/100
1563/1563 [==========] - 9s 6ms/step - loss: 1.0206 - accuracy: 0.6370 - val_loss: 1.3415 - val_accuracy: 0.5362
Epoch 27/100
1563/1563 [==========] - 10s 6ms/step - loss: 1.0035 - accuracy: 0.6446 - val_loss: 1.3410 - val_accuracy: 0.5405
Epoch 28/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9877 - accuracy: 0.6502 - val_loss: 1.2684 - val_accuracy: 0.5593
Epoch 29/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9719 - accuracy: 0.6554 - val_loss: 1.3132 - val_accuracy: 0.5417
Epoch 30/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9541 - accuracy: 0.6626 - val_loss: 1.2827 - val_accuracy: 0.5683


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Epoch 31/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9376 - accuracy: 0.6682 - val_loss: 1.3456 - val_accuracy: 0.5427
Epoch 32/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9234 - accuracy: 0.6718 - val_loss: 1.3410 - val_accuracy: 0.5405
Epoch 33/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.9047 - accuracy: 0.6801 - val_loss: 1.2753 - val_accuracy: 0.5647
Epoch 34/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8931 - accuracy: 0.6828 - val_loss: 1.3202 - val_accuracy: 0.5561
Epoch 35/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8764 - accuracy: 0.6882 - val_loss: 1.2828 - val_accuracy: 0.5685
Epoch 36/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8616 - accuracy: 0.6965 - val_loss: 1.2733 - val_accuracy: 0.5657
Epoch 37/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8459 - accuracy: 0.7003 - val_loss: 1.2886 - val_accuracy: 0.5713
Epoch 38/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8355 - accuracy: 0.7061 - val_loss: 1.3767 - val_accuracy: 0.5434
Epoch 39/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8203 - accuracy: 0.7123 - val_loss: 1.3908 - val_accuracy: 0.5472
Epoch 40/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.8065 - accuracy: 0.7180 - val_loss: 1.3933 - val_accuracy: 0.5554
Epoch 41/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.7885 - accuracy: 0.7219 - val_loss: 1.3693 - val_accuracy: 0.5658
Epoch 42/100
1563/1563 [==========] - 10s 6ms/step - loss: 0.7772 - accuracy: 0.7279 - val_loss: 1.4585 - val_accuracy: 0.5431
Epoch 43/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.7610 - accuracy: 0.7335 - val_loss: 1.3918 - val_accuracy: 0.5583
Epoch 44/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.7510 - accuracy: 0.7369 - val_loss: 1.3768 - val_accuracy: 0.5623
Epoch 45/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.7364 - accuracy: 0.7418 - val_loss: 1.3952 - val_accuracy: 0.5595
Epoch 46/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.7225 - accuracy: 0.7466 - val_loss: 1.4471 - val_accuracy: 0.5559
Epoch 47/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.7110 - accuracy: 0.7515 - val_loss: 1.4989 - val_accuracy: 0.5428


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Colab AI
Epoch 48/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.6977 - accuracy: 0.7545 - val_loss: 1.4292 - val_accuracy: 0.5508
Epoch 49/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.6866 - accuracy: 0.7594 - val_loss: 1.4875 - val_accuracy: 0.5599
Epoch 50/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.6729 - accuracy: 0.7643 - val_loss: 1.5332 - val_accuracy: 0.5528
Epoch 51/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.6597 - accuracy: 0.7674 - val_loss: 1.4784 - val_accuracy: 0.5575
Epoch 52/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.6477 - accuracy: 0.7726 - val_loss: 1.5385 - val_accuracy: 0.5496
Epoch 53/100
1563/1563 [==========] - 10s 6ms/step - loss: 0.6355 - accuracy: 0.7772 - val_loss: 1.5535 - val_accuracy: 0.5480
Epoch 54/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.6218 - accuracy: 0.7809 - val_loss: 1.5702 - val_accuracy: 0.5489
Epoch 55/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.6139 - accuracy: 0.7838 - val_loss: 1.5840 - val_accuracy: 0.5509
Epoch 56/100
1563/1563 [==========] - 8s 5ms/step - loss: 0.6024 - accuracy: 0.7875 - val_loss: 1.5831 - val_accuracy: 0.5517
Epoch 57/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5902 - accuracy: 0.7910 - val_loss: 1.6077 - val_accuracy: 0.5619
Epoch 58/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5765 - accuracy: 0.7981 - val_loss: 1.6588 - val_accuracy: 0.5447
Epoch 59/100
1563/1563 [==========] - 9s 5ms/step - loss: 0.5702 - accuracy: 0.7987 - val_loss: 1.7273 - val_accuracy: 0.5333
Epoch 60/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5589 - accuracy: 0.8028 - val_loss: 1.6732 - val_accuracy: 0.5505
Epoch 61/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5459 - accuracy: 0.8074 - val_loss: 1.8452 - val_accuracy: 0.5223
Epoch 62/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5408 - accuracy: 0.8089 - val_loss: 1.7395 - val_accuracy: 0.5457
Epoch 63/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5237 - accuracy: 0.8169 - val_loss: 1.7190 - val_accuracy: 0.5483
Epoch 64/100
1563/1563 [==========] - 9s 6ms/step - loss: 0.5183 - accuracy: 0.8182 - val_loss: 1.7337 - val_accuracy: 0.5457


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Colab AI
Epoch 65/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.5103 - accuracy: 0.8211 - val_loss: 1.7677 - val_accuracy: 0.5340
Epoch 66/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.4987 - accuracy: 0.8253 - val_loss: 1.7960 - val_accuracy: 0.5395
Epoch 67/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4887 - accuracy: 0.8277 - val_loss: 1.8085 - val_accuracy: 0.5434
Epoch 68/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.4814 - accuracy: 0.8310 - val_loss: 1.7844 - val_accuracy: 0.5479
Epoch 69/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.4698 - accuracy: 0.8348 - val_loss: 1.7983 - val_accuracy: 0.5473
Epoch 70/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.4635 - accuracy: 0.8354 - val_loss: 1.8895 - val_accuracy: 0.5432
Epoch 71/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4601 - accuracy: 0.8376 - val_loss: 2.0216 - val_accuracy: 0.5312
Epoch 72/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4453 - accuracy: 0.8419 - val_loss: 1.9327 - val_accuracy: 0.5396
Epoch 73/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4386 - accuracy: 0.8468 - val_loss: 1.9244 - val_accuracy: 0.5395
Epoch 74/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4299 - accuracy: 0.8471 - val_loss: 1.9421 - val_accuracy: 0.5431
Epoch 75/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4207 - accuracy: 0.8517 - val_loss: 1.9950 - val_accuracy: 0.5415
Epoch 76/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4122 - accuracy: 0.8541 - val_loss: 2.0157 - val_accuracy: 0.5408
Epoch 77/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.4066 - accuracy: 0.8580 - val_loss: 2.0550 - val_accuracy: 0.5386
Epoch 78/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3986 - accuracy: 0.8599 - val_loss: 2.1255 - val_accuracy: 0.5295
Epoch 79/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3907 - accuracy: 0.8638 - val_loss: 2.0074 - val_accuracy: 0.5413
Epoch 80/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3809 - accuracy: 0.8644 - val_loss: 2.0996 - val_accuracy: 0.5366
Epoch 81/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.3752 - accuracy: 0.8688 - val_loss: 2.1578 - val_accuracy: 0.5424


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
+ Code + Text
Connect T4
Epoch 82/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3739 - accuracy: 0.8672 - val_loss: 2.0935 - val_accuracy: 0.5416
Epoch 83/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3657 - accuracy: 0.8713 - val_loss: 2.0978 - val_accuracy: 0.5394
Epoch 84/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3561 - accuracy: 0.8746 - val_loss: 2.2300 - val_accuracy: 0.5351
Epoch 85/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3566 - accuracy: 0.8739 - val_loss: 2.1779 - val_accuracy: 0.5361
Epoch 86/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3431 - accuracy: 0.8776 - val_loss: 2.2862 - val_accuracy: 0.5378
Epoch 87/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3400 - accuracy: 0.8798 - val_loss: 2.2744 - val_accuracy: 0.5447
Epoch 88/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.3316 - accuracy: 0.8832 - val_loss: 2.3488 - val_accuracy: 0.5322
Epoch 89/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.3289 - accuracy: 0.8826 - val_loss: 2.2944 - val_accuracy: 0.5401
Epoch 90/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3198 - accuracy: 0.8880 - val_loss: 2.3053 - val_accuracy: 0.5381
Epoch 91/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3142 - accuracy: 0.8893 - val_loss: 2.3100 - val_accuracy: 0.5377
Epoch 92/100
1563/1563 [==============================] - 8s 5ms/step - loss: 0.3103 - accuracy: 0.8907 - val_loss: 2.4095 - val_accuracy: 0.5319
Epoch 93/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.3056 - accuracy: 0.8914 - val_loss: 2.6715 - val_accuracy: 0.5002
Epoch 94/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2985 - accuracy: 0.8941 - val_loss: 2.4450 - val_accuracy: 0.5215
Epoch 95/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2925 - accuracy: 0.8972 - val_loss: 2.4982 - val_accuracy: 0.5279
Epoch 96/100
1563/1563 [==============================] - 10s 6ms/step - loss: 0.2901 - accuracy: 0.8984 - val_loss: 2.6902 - val_accuracy: 0.5028
Epoch 97/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2846 - accuracy: 0.8997 - val_loss: 2.5126 - val_accuracy: 0.5348
Epoch 98/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2807 - accuracy: 0.9011 - val_loss: 2.5264 - val_accuracy: 0.5359

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4

```
Epoch 99/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2739 - accuracy: 0.9039 - val_loss: 2.5997 - val_accuracy: 0.5227
Epoch 100/100
1563/1563 [==============================] - 9s 6ms/step - loss: 0.2681 - accuracy: 0.9057 - val_loss: 2.5864 - val_accuracy: 0.5327
```

```
[ ] # summarize history for accuracy
    plt.plot(history1.history['accuracy'])
    plt.plot(history1.history['val_accuracy'])
    plt.title("Accuracy by AlexNet on CIFAR-10 Data")
    plt.ylabel('Accuracy')
    plt.xlabel('Epochs')
    plt.legend(['Train', 'Validation'], loc='upper left')
    plt.show()
    # summarize history for loss
    plt.plot(history1.history['loss'])
    plt.plot(history1.history['val_loss'])
    plt.title('Loss by AlexNet on CIFAR-10 Data')
    plt.ylabel('Loss')
    plt.xlabel('Epochs')
    plt.legend(['Train', 'Validation'])
    plt.show()
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4

```
[ ] y_predictions1 = AlexNet.predict(x_test)
    y_predictions1.reshape(-1,)
    y_predictions1= np.argmax(y_predictions1, axis=1)

    confusion_matrix(y_test, y_predictions1)
```

```
313/313 [==============================] - 1s 2ms/step
array([[636,  32,  69,  48,  32,  13,   9,  41,  66,  54],
       [ 57, 585,  10,  28,   8,  11,  15,  31,  50, 205],
       [ 90,   4, 381, 106, 169,  71,  70,  66,  21,  22],
       [ 33,  16,  73, 449,  75, 202,  70,  53,  10,  19],
       [ 32,   8,  93,  83, 519,  59,  82, 105,  12,   7],
       [ 14,   6,  64, 257,  75, 429,  40,  95,   4,  16],
       [  7,  12,  78, 136, 146,  53, 515,  16,  11,  26],
       [ 31,  14,  54,  65, 107,  82,  17, 582,   6,  42],
       [147,  52,  26,  42,  30,  15,  13,   9, 617,  49],
       [ 65, 124,  24,  37,   9,  16,  12,  62,  37, 614]])
```

```
[ ] # confusion matrix and accuracy
    plt.figure(figsize=(7, 6))
    plt.title('Confusion matrix of AlexNet Model', fontsize=16)
    plt.imshow(confusion_matrix(y_test, y_predictions1))
    plt.xticks(np.arange(10), classes, rotation=45, fontsize=12)
    plt.yticks(np.arange(10), classes, fontsize=12)
    plt.colorbar()
    plt.show()
```

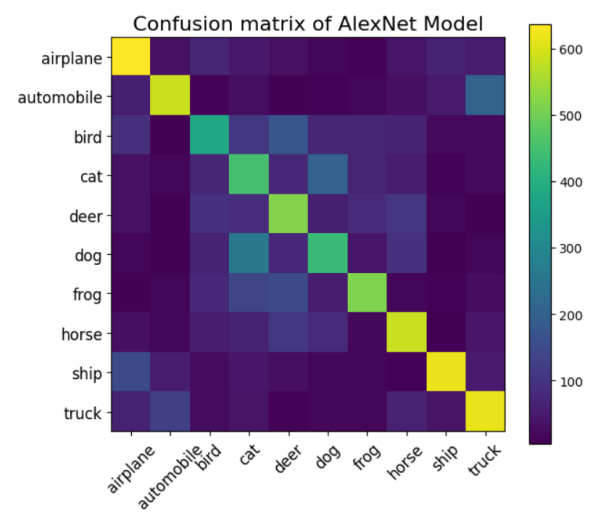
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

Connect T4



LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4

```
print("Test accuracy by AlexNet:", accuracy_score(y_test, y_predictions))
```

```
Test accuracy by AlexNet: 0.5995
```

```
[ ] L = 8
    W = 8
    fig, axes = plt.subplots(L, W, figsize = (20,20))
    axes = axes.ravel() #

    for i in np.arange(0, L * W):
        axes[i].imshow(x_test[i])
        axes[i].set_title("Predicted = {}\n Actual  = {}".format(classes[y_predictions[i]], classes[y_test[i]]))
        axes[i].axis('off')

    plt.subplots_adjust(wspace=1)
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4 Colab AI

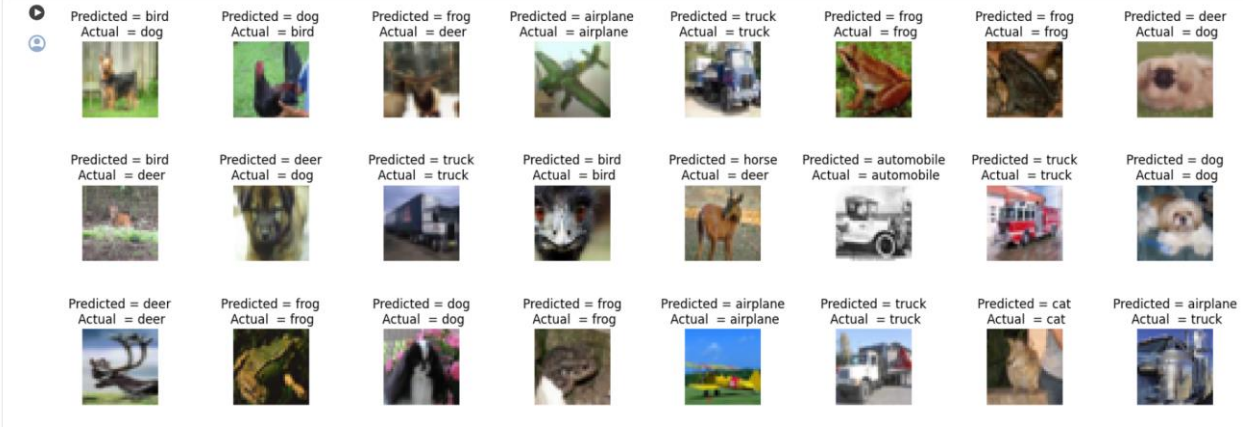
LeNet&AlexNet&VGG16&VGG19.ipynb

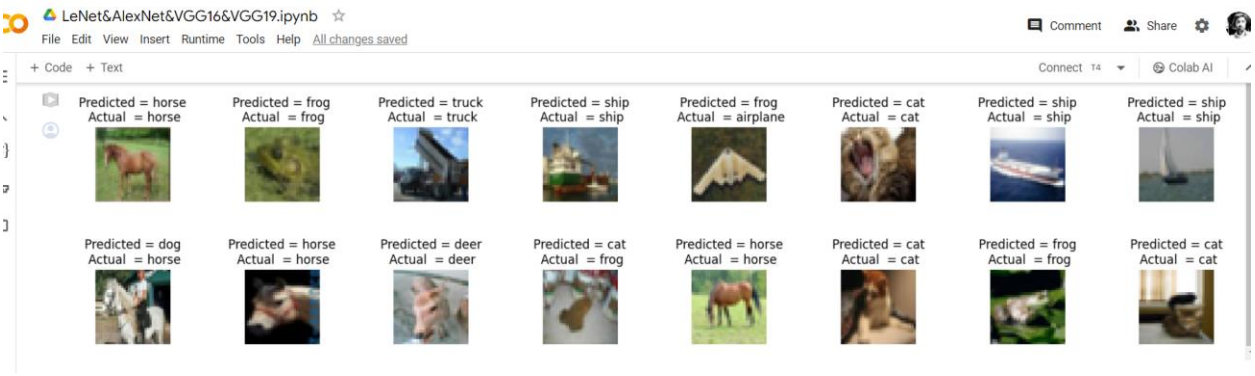
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4 Colab AI




LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Colab AI
Predicted = horse Actual = horse
Predicted = frog Actual = frog
Predicted = truck Actual = truck
Predicted = ship Actual = ship
Predicted = frog Actual = airplane
Predicted = cat Actual = cat
Predicted = ship Actual = ship
Predicted = ship Actual = ship
Predicted = dog Actual = horse
Predicted = horse Actual = horse
Predicted = deer Actual = deer
Predicted = cat Actual = frog
Predicted = horse Actual = horse
Predicted = cat Actual = cat
Predicted = frog Actual = frog
Predicted = cat Actual = cat


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
+ Code + Text
Connect T4
Colab AI
3.VGG 16
[ ] import keras
from keras.models import Sequential
from keras.layers import Activation,Dense,Dropout,Conv2D,Flatten,MaxPooling2D
from keras.datasets import cifar10
from keras import optimizers
from matplotlib import pyplot as plt
[ ] # generate cifar10 data
(x_train,y_train),(x_test,y_test) = cifar10.load_data()
[ ] # config parameters
num_classes = 10
input_shape = x_train.shape[1:4]
optimizer = optimizers.Adam(lr=0.0003)
WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.Adam.
[ ] # convert label to one-hot
one_hot_y_train = keras.utils.to_categorical(y_train,num_classes=num_classes)
one_hot_y_test = keras.utils.to_categorical(y_test,num_classes=num_classes)

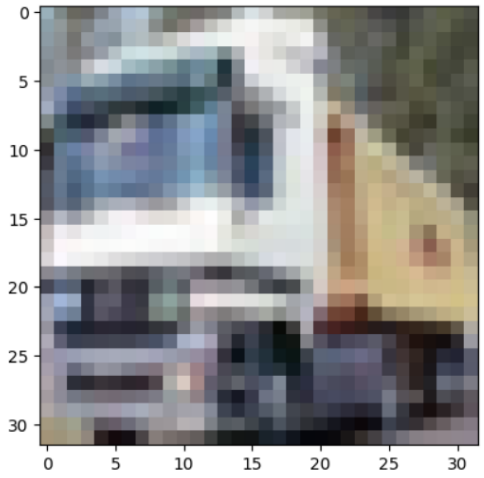
LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
# check data
plt.imshow(x_train[1])
print(x_train[1].shape)
```

```
(32, 32, 3)
```



LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
# build model(similar to VGG16, only change the input and output shape)
model = Sequential()
model.add(Conv2D(64,(3,3),activation='relu',input_shape=input_shape,padding='same'))
model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(256,(3,3),activation='relu',padding='same'))
model.add(Conv2D(256,(3,3),activation='relu',padding='same'))
model.add(Conv2D(256,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Flatten())
model.add(Dense(4096,activation='relu'))
model.add(Dense(4096,activation='relu'))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
```

```
[ ] # config optimizer,loss,metrics
model.compile(optimizer=optimizer,loss='categorical_crossentropy',metrics=['accuracy'])
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
# check model
model.summary()
```

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_8 (Conv2D)            (None, 32, 32, 64)        1792

conv2d_9 (Conv2D)            (None, 32, 32, 64)        36928

max_pooling2d_3 (MaxPoolin   (None, 16, 16, 64)        0
g2D)

conv2d_10 (Conv2D)           (None, 16, 16, 128)       73856

conv2d_11 (Conv2D)           (None, 16, 16, 128)       147584

max_pooling2d_4 (MaxPoolin   (None, 8, 8, 128)         0
g2D)

conv2d_12 (Conv2D)           (None, 8, 8, 256)         295168

conv2d_13 (Conv2D)           (None, 8, 8, 256)         590080

conv2d_14 (Conv2D)           (None, 8, 8, 256)         590080

max_pooling2d_5 (MaxPoolin   (None, 4, 4, 256)         0
g2D)

conv2d_15 (Conv2D)           (None, 4, 4, 512)         1180160

conv2d_16 (Conv2D)           (None, 4, 4, 512)         2359808

conv2d_17 (Conv2D)           (None, 4, 4, 512)         2359808
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
max_pooling2d_6 (MaxPoolin   (None, 2, 2, 512)         0
g2D)

conv2d_18 (Conv2D)           (None, 2, 2, 512)         2359808

conv2d_19 (Conv2D)           (None, 2, 2, 512)         2359808

conv2d_20 (Conv2D)           (None, 2, 2, 512)         2359808

max_pooling2d_7 (MaxPoolin   (None, 1, 1, 512)         0
g2D)

flatten_2 (Flatten)          (None, 512)               0

dense_5 (Dense)              (None, 4096)              2101248

dense_6 (Dense)              (None, 4096)              16781312

dense_7 (Dense)              (None, 10)                40970

activation (Activation)      (None, 10)                0

=================================================================
Total params: 33638218 (128.32 MB)
Trainable params: 33638218 (128.32 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
# train
model.fit(x=x_train,y=one_hot_y_train,batch_size=128,epochs=10)
```

```
Epoch 1/10
391/391 [==============================] - 36s 68ms/step - loss: 2.3356 - accuracy: 0.0983
Epoch 2/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0959
Epoch 3/10
391/391 [==============================] - 23s 58ms/step - loss: 2.3027 - accuracy: 0.0961
Epoch 4/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0975
Epoch 5/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0969
Epoch 6/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0957
Epoch 7/10
391/391 [==============================] - 23s 58ms/step - loss: 2.3027 - accuracy: 0.0967
Epoch 8/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0959
Epoch 9/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0982
Epoch 10/10
391/391 [==============================] - 23s 59ms/step - loss: 2.3027 - accuracy: 0.0981
<keras.src.callbacks.History at 0x7b47ad77feb0>
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

Connect T4

```
# evaluate
print(model.metrics_names)
model.evaluate(x=x_test,y=one_hot_y_test,batch_size=512)
```

```
['loss', 'accuracy']
20/20 [==============================] - 7s 162ms/step - loss: 2.3026 - accuracy: 0.1000
[2.3025965690612793, 0.10000000149011612]
```

```
[ ] # predict
    plt.imshow(x_test[1000])

    result = model.predict(x_test[1000:1001]).tolist()
    predict = 0
    expect = y_test[1000][0]
    for i,_ in enumerate(result[0]):
        if result[0][i] > result[0][predict]:
            predict = i
    print("predict class:",predict)
    print("expected class:",expect)
```

LeNet&AlexNet&VGG16&VGG19.ipynb

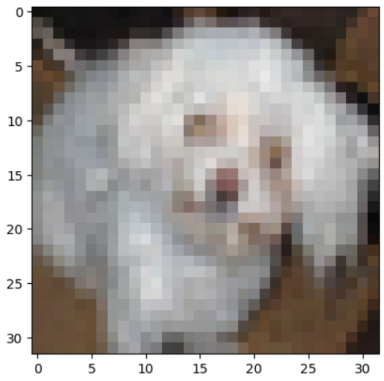
File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

Connect T4

```
1/1 [==============================] - 1s 640ms/step
predict class: 9
expected class: 5
```



```
[ ] # save model
    model.save("keras-VGG16-cifar10.h5")
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

Connect T4

4.VGG19Model

```
[ ] import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    import tensorflow as tf
    import keras

    from keras.datasets import cifar10
    from tensorflow.keras.optimizers import RMSprop
    from keras.preprocessing import image
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dropout, BatchNormalization

    %matplotlib inline
```

Extract data and train and test dataset

```
[ ] cifar100 = tf.keras.datasets.cifar100
    (X_train,Y_train) , (X_test,Y_test) = cifar10.load_data()
```

```
[ ] classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4 Colab

```
# X_train = X_train.astype('float32')
# X_test = X_test.astype('float32')
# X_train = X_train / 255.0
# X_test = X_test / 255.0
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Sha

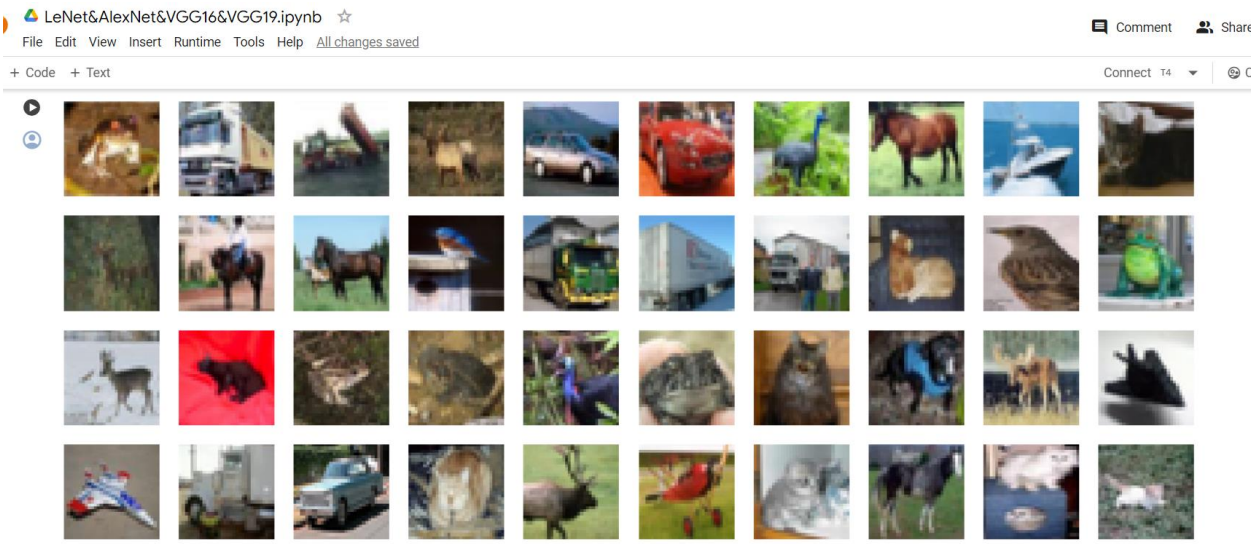
+ Code + Text

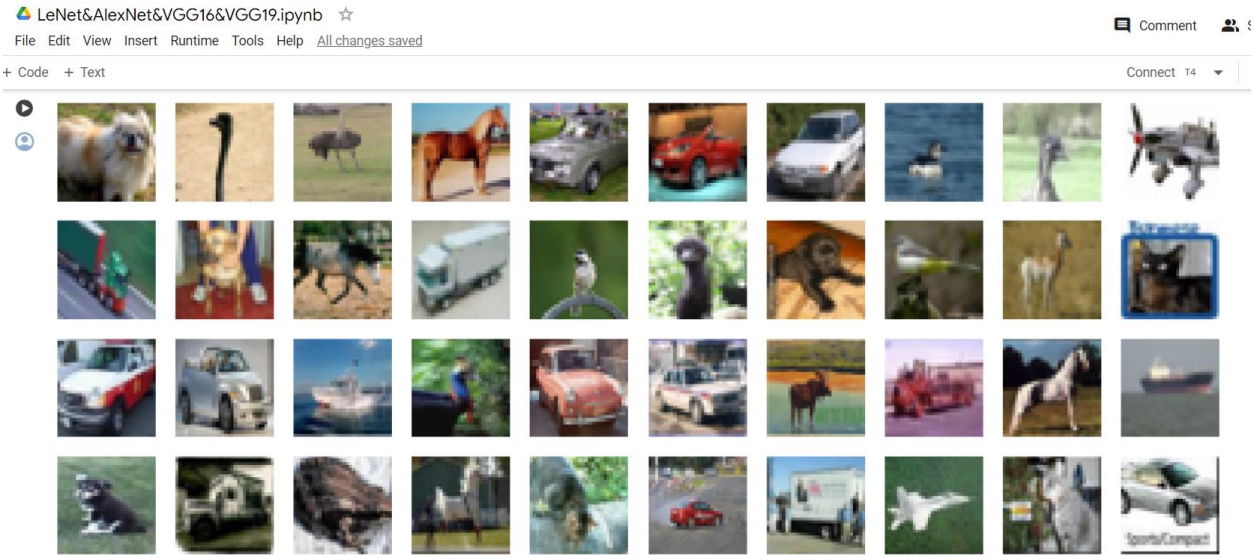
Connect T4

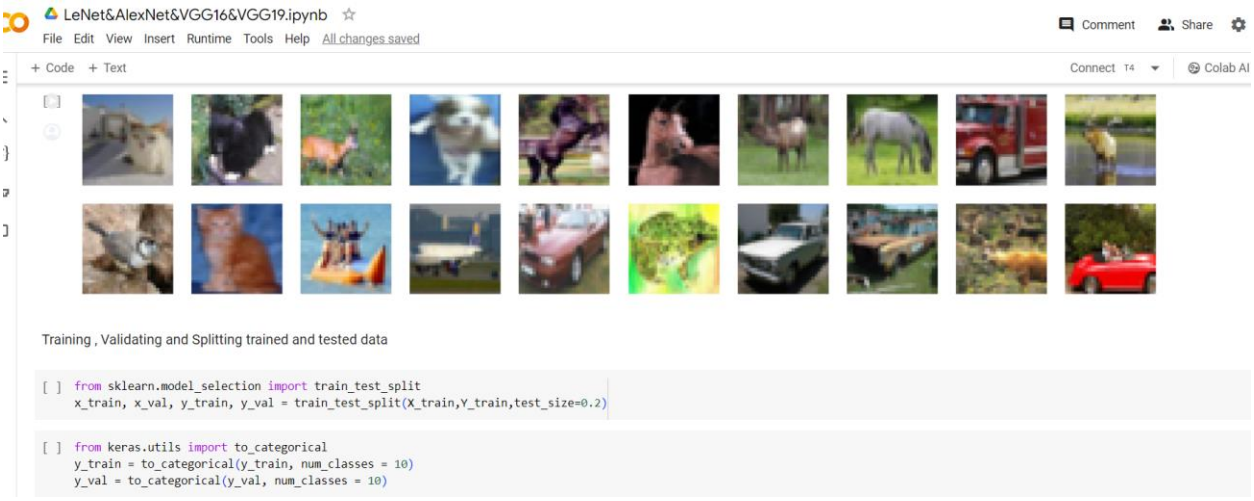
let's look into the dataset images

+ Code + Text

```
[ ] plt.figure(figsize = (16,16))
    for i in range(100):
      plt.subplot(10,10,1+i)
      plt.axis('off')
      plt.imshow(X_train[i], cmap = 'gray')
```


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
+ Code + Text
Connect T4


LeNet&AlexNet&VGG16&VGG19.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Connect T4
Colab AI
Training , Validating and Splitting trained and tested data
[ ] from sklearn.model_selection import train_test_split
x_train, x_val, y_train, y_val = train_test_split(X_train,Y_train,test_size=0.2)
[ ] from keras.utils import to_categorical
y_train = to_categorical(y_train, num_classes = 10)
y_val = to_categorical(y_val, num_classes = 10)

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4

```
print(x_train.shape)
print(y_train.shape)
print(x_val.shape)
print(y_val.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(40000, 32, 32, 3)
(40000, 10)
(10000, 32, 32, 3)
(10000, 10)
(10000, 32, 32, 3)
(10000, 1)
```

```
[ ] train_datagen = ImageDataGenerator(
        preprocessing_function = tf.keras.applications.vgg19.preprocess_input,
        rotation_range=10,
        zoom_range = 0.1,
        width_shift_range = 0.1,
        height_shift_range = 0.1,
        shear_range = 0.1,
        horizontal_flip = True
    )
    train_datagen.fit(x_train)

    val_datagen = ImageDataGenerator(preprocessing_function = tf.keras.applications.vgg19.preprocess_input)
    val_datagen.fit(x_val)
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comm

+ Code + Text

Connect

```
from keras.callbacks import ReduceLROnPlateau
learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy',
                                            patience=3,
                                            verbose=1,
                                            factor=0.5,
                                            min_lr=0.00001)
```

We have used only 16 layers out of 19 layers in the CNN

```
[ ] vgg_model = tf.keras.applications.VGG19(
        include_top=False,
        weights="imagenet",
        input_shape=(32,32,3),
    )

    vgg_model.summary()
```

```
Model: "vgg19"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 32, 32, 3)]       0

 block1_conv1 (Conv2D)       (None, 32, 32, 64)        1792

 block1_conv2 (Conv2D)       (None, 32, 32, 64)        36928

 block1_pool (MaxPooling2D)  (None, 16, 16, 64)        0

 block2_conv1 (Conv2D)       (None, 16, 16, 128)       73856
```

LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
block2_conv1 (Conv2D)       (None, 16, 16, 128)       73856
block2_conv2 (Conv2D)       (None, 16, 16, 128)       147584
block2_pool (MaxPooling2D)  (None, 8, 8, 128)         0
block3_conv1 (Conv2D)       (None, 8, 8, 256)         295168
block3_conv2 (Conv2D)       (None, 8, 8, 256)         590080
block3_conv3 (Conv2D)       (None, 8, 8, 256)         590080
block3_conv4 (Conv2D)       (None, 8, 8, 256)         590080
block3_pool (MaxPooling2D)  (None, 4, 4, 256)         0
block4_conv1 (Conv2D)       (None, 4, 4, 512)         1180160
block4_conv2 (Conv2D)       (None, 4, 4, 512)         2359808
block4_conv3 (Conv2D)       (None, 4, 4, 512)         2359808
block4_conv4 (Conv2D)       (None, 4, 4, 512)         2359808
block4_pool (MaxPooling2D)  (None, 2, 2, 512)         0
block5_conv1 (Conv2D)       (None, 2, 2, 512)         2359808
block5_conv2 (Conv2D)       (None, 2, 2, 512)         2359808
block5_conv3 (Conv2D)       (None, 2, 2, 512)         2359808
block5_conv4 (Conv2D)       (None, 2, 2, 512)         2359808
```

LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
block5_conv4 (Conv2D)       (None, 2, 2, 512)         2359808
block5_pool (MaxPooling2D)  (None, 1, 1, 512)         0
=================================================================
Total params: 20024384 (76.39 MB)
Trainable params: 20024384 (76.39 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
[ ] model = tf.keras.Sequential()
    model.add(vgg_model)
    model.add(Flatten())
    model.add(Dense(1024, activation = 'relu'))
    model.add(Dense(1024, activation = 'relu'))
    model.add(Dense(256, activation = 'sigmoid'))
    model.add(Dense(10, activation = 'softmax'))

    model.summary()
```

LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 1, 1, 512)         20024384

 flatten_3 (Flatten)         (None, 512)               0

 dense_10 (Dense)            (None, 1024)              525312

 dense_11 (Dense)            (None, 1024)              1049600

 dense_12 (Dense)            (None, 256)               262400

 dense_13 (Dense)            (None, 10)                2570

=================================================================
Total params: 21864266 (83.41 MB)
Trainable params: 21864266 (83.41 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
[ ] optimizer = tf.keras.optimizers.SGD(learning_rate = 0.001, momentum = 0.9)
    model.compile(optimizer= optimizer,
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
```

LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect T4 Cola

```
history = model.fit(
    train_datagen.flow(x_train, y_train, batch_size = 128),
    validation_data = val_datagen.flow(x_val,y_val, batch_size = 128),
    epochs = 10,
    verbose = 1,
    callbacks = [learning_rate_reduction]
)
```

```
313/313 [==============================] - 38s 112ms/step - loss: 1.1032 - accuracy: 0.6178 - val_loss: 0.7488 - val_accuracy: 0.7486 - lr: 0.0010
Epoch 2/10
313/313 [==============================] - 34s 107ms/step - loss: 0.7297 - accuracy: 0.7541 - val_loss: 0.6326 - val_accuracy: 0.7842 - lr: 0.0010
Epoch 3/10
313/313 [==============================] - 34s 107ms/step - loss: 0.6273 - accuracy: 0.7884 - val_loss: 0.6150 - val_accuracy: 0.7950 - lr: 0.0010
Epoch 4/10
313/313 [==============================] - 34s 107ms/step - loss: 0.5704 - accuracy: 0.8069 - val_loss: 0.5969 - val_accuracy: 0.7970 - lr: 0.0010
Epoch 5/10
313/313 [==============================] - 34s 108ms/step - loss: 0.5160 - accuracy: 0.8231 - val_loss: 0.5143 - val_accuracy: 0.8258 - lr: 0.0010
Epoch 6/10
313/313 [==============================] - 34s 109ms/step - loss: 0.4794 - accuracy: 0.8351 - val_loss: 0.5276 - val_accuracy: 0.8218 - lr: 0.0010
Epoch 7/10
313/313 [==============================] - 35s 112ms/step - loss: 0.4492 - accuracy: 0.8448 - val_loss: 0.5124 - val_accuracy: 0.8297 - lr: 0.0010
Epoch 8/10
313/313 [==============================] - 33s 106ms/step - loss: 0.4173 - accuracy: 0.8559 - val_loss: 0.4892 - val_accuracy: 0.8359 - lr: 0.0010
Epoch 9/10
313/313 [==============================] - 33s 104ms/step - loss: 0.3871 - accuracy: 0.8678 - val_loss: 0.4836 - val_accuracy: 0.8373 - lr: 0.0010
Epoch 10/10
313/313 [==============================] - 33s 106ms/step - loss: 0.3677 - accuracy: 0.8744 - val_loss: 0.4372 - val_accuracy: 0.8516 - lr: 0.0010
```

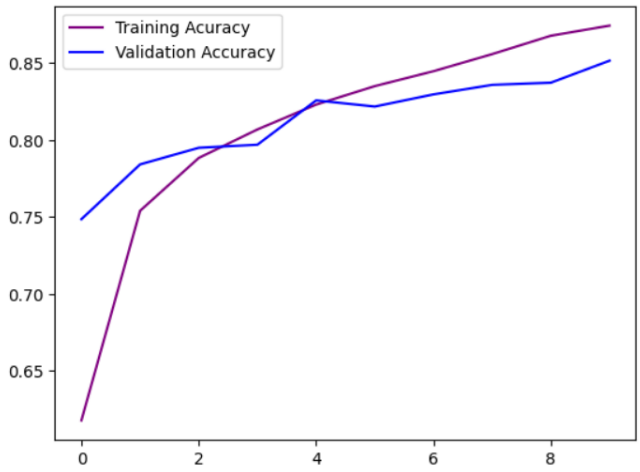
LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

plt.figure()
plt.plot(acc,color = 'purple',label = 'Training Acuracy')
plt.plot(val_acc,color = 'blue',label = 'Validation Accuracy')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f9ed4129930>
```



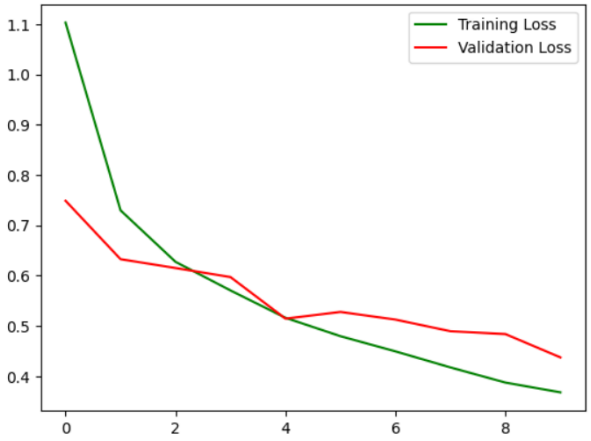
LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
loss = history.history['loss']
val_loss = history.history['val_loss']

plt.figure()
plt.plot(loss,color = 'green',label = 'Training Loss')
plt.plot(val_loss,color = 'red',label = 'Validation Loss')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f9ed55d68f0>
```



LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
x_test = tf.keras.applications.vgg19.preprocess_input(X_test)
y_pred = np.argmax(model.predict(x_test), axis=-1)
y_pred[:10]
```

```
313/313 [==============================] - 3s 9ms/step
array([3, 8, 8, 0, 6, 6, 1, 6, 3, 1])
```

```
[ ] cm = confusion_matrix(Y_test, y_pred)
    cm
```

```
array([[892,   4,  19,   5,  10,   2,   2,  12,  36,  18],
       [ 12, 885,   1,   1,   0,   2,   3,   2,  11,  83],
       [ 35,   1, 826,  19,  48,  19,  35,   9,   5,   3],
       [  5,   3,  36, 700,  32, 145,  39,  19,   6,  15],
       [ 12,   1,  37,  27, 834,  15,  29,  40,   4,   1],
       [  3,   1,  28, 105,  22, 793,  11,  34,   1,   2],
       [ 10,   2,  28,  38,  15,  11, 884,   1,   7,   4],
       [  5,   0,  13,  21,  25,  30,   1, 898,   2,   5],
       [ 29,  15,   5,   1,   3,   1,   0,   0, 923,  23],
       [ 15,  16,   3,  15,   1,   1,   1,   7,  12, 929]])
```

```
[ ] from sklearn.metrics import confusion_matrix, accuracy_score
    print('Testing Accuarcy : ', accuracy_score(Y_test, y_pred))
```

```
Testing Accuarcy :  0.8564
```

LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

Connect T4

```
import itertools
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Greens):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=30)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    #print(cm)

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
            horizontalalignment="center",
            color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[ ]
        plt.tight_layout()
        plt.ylabel('True label')
        plt.xlabel('Predicted label')
```

+ Code + Text

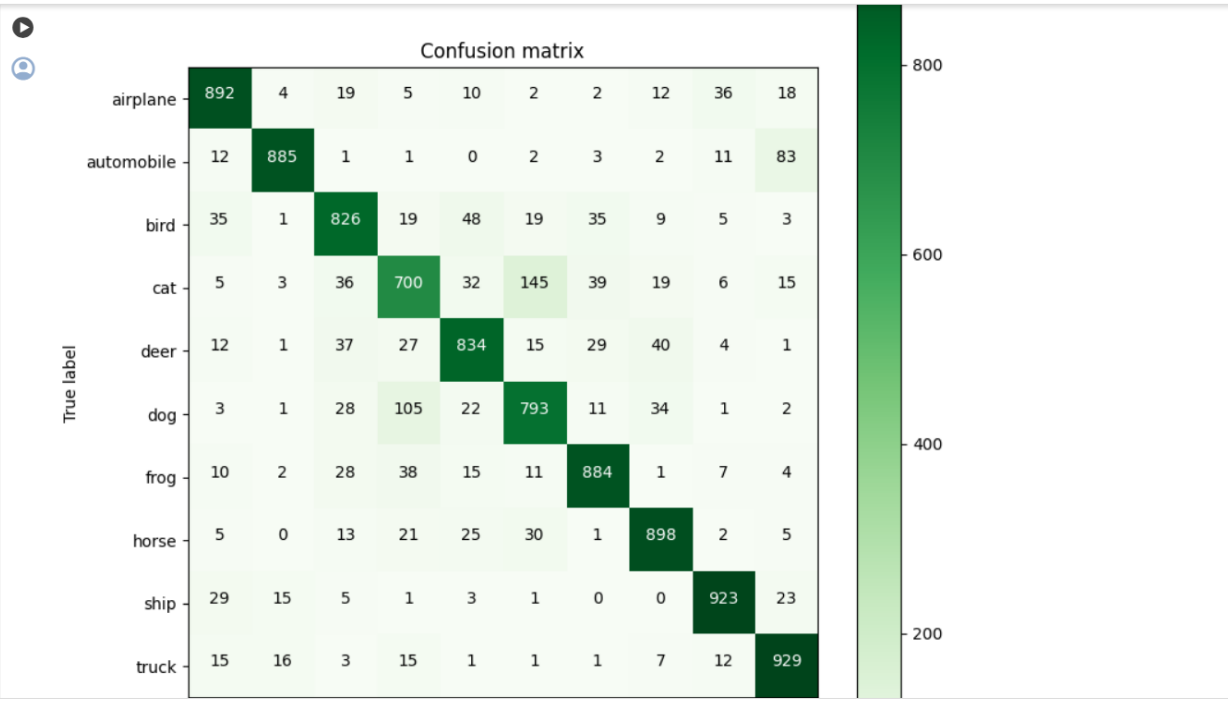
```
plt.figure(figsize=(8,8))
plot_confusion_matrix(cm,classes)
```

```
Confusion matrix, without normalization
```

LeNet&AlexNet&VGG16&VGG19.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text



LeNet&AlexNet&VGG16&VGG19.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text