```
from google.colab import drive
drive.mount('/content/drive')

     Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
%cd /content/drive/MyDrive/ml_projects/my_model

import sys
import time
import pickle

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from keras.datasets import mnist, cifar10
import sklearn.datasets
from tqdm import tqdm

from dataset import Dataset
from layers import Dense, Conv2D, Flatten, MaxPooling, AvgPooling, Dropout
from model import Model

     /content/drive/MyDrive/ml_projects/my_model
```

```
(train_x, train_y), (test_x, test_y) = mnist.load_data()
# (train_x, train_y), (test_x, test_y) = cifar10.load_data()
# train_x, train_y = sklearn.datasets.make_circles(n_samples=100, shuffle=False, factor=0.3, noise=0.1)
# train_x, train_y = sklearn.datasets.load_iris(return_X_y=True)
```

```
# train_x = train_x[:5000]
# train_y = train_y[:5000]
# test_x = test_x[:500]
# test_y = test_y[:500]
```

```
train_data = Dataset(train_x, train_y)
train_data.one_hot()
# min_val, max_val = train_data.normalize()

test_data = Dataset(test_x, test_y)
test_data.one_hot()
# test_data.normalize(min_val, max_val)
```

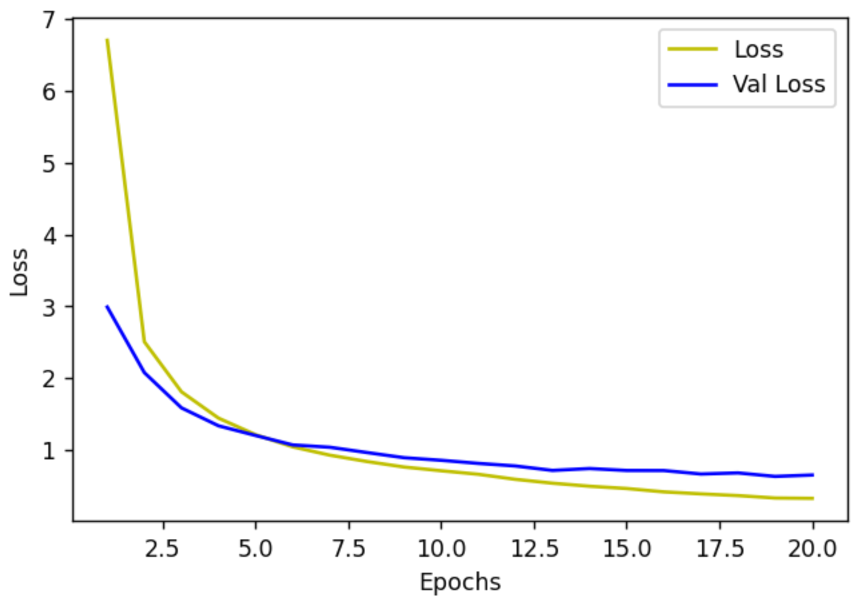
```
layers = [Conv2D(16, 3, activation='relu',input_shape=train_data.get_shape()),
          MaxPooling(),
          Conv2D(32, 3, activation='relu'),
          MaxPooling(),
          Conv2D(64, 3, activation='relu'),
          Flatten(),
          # Dropout(0.2),
          Dense(64, activation='relu'),
          # Dropout(0.2),
          Dense(train_data.num_classes, activation='softmax')]
# layers = [Flatten(input_shape=train_data.get_shape()),
#           # Dense(256, activation='relu'),
#           Dense(128, activation='relu'),
#           Dense(train_data.num_classes, activation='softmax')]


model = Model(layers)
model.compile(optim='adam',metrics='accuracy', loss='crossentropy')
model.summary()
```

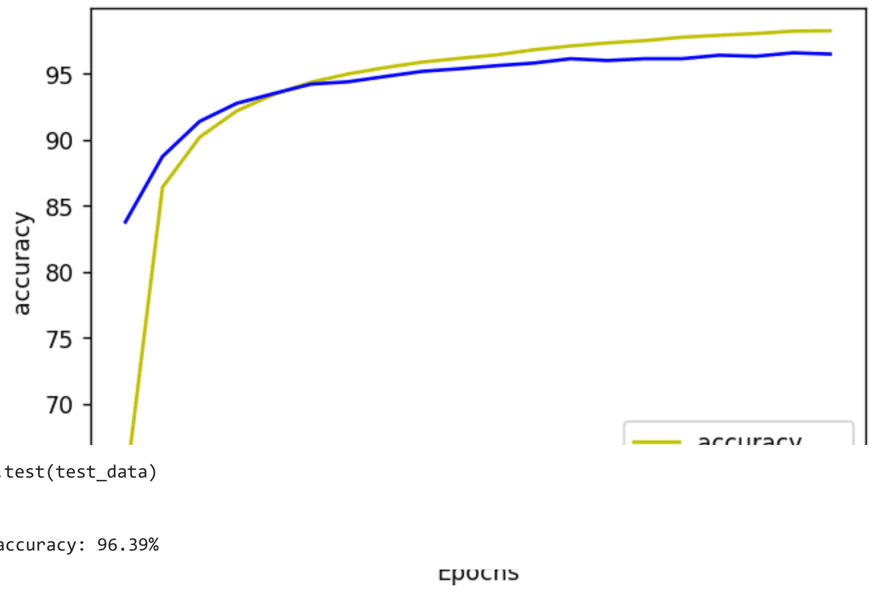
```
    Model Summary
    conv2d_0: 16 x 3 x 3 Activation: relu Parameters:160
    max_pool_0: Activation:None Parameters:0
    conv2d_1: 32 x 3 x 3 Activation: relu Parameters:4640
    max_pool_1: Activation:None Parameters:0
    conv2d_2: 64 x 3 x 3 Activation: relu Parameters:18496
    flatten_0: Activation:None Parameters:0
    dense_0: 576 x 64 Activation: relu Parameters:36928
    dense_1: 64 x 10 Activation: softmax Parameters:650
    Total Parameters:60874
```

```
results = model.train(train_data,
                        batch_size = 256,
                        lr = 1e-3,
                        epochs = 20,
                        val_data = test_data)
```

```
    Epoch 1: 235 batches [07:31,  1.92s/ batches, loss=6.71, val_loss=2.99, acc=63.6, val_acc=83.8]
    Epoch 2: 235 batches [07:29,  1.91s/ batches, loss=2.51, val_loss=2.08, acc=86.4, val_acc=88.7]
    Epoch 3: 235 batches [07:26,  1.90s/ batches, loss=1.81, val_loss=1.59, acc=90.2, val_acc=91.4]
    Epoch 4: 235 batches [07:28,  1.91s/ batches, loss=1.44, val_loss=1.33, acc=92.2, val_acc=92.7]
    Epoch 5: 235 batches [07:27,  1.90s/ batches, loss=1.21, val_loss=1.2, acc=93.4, val_acc=93.5]
    Epoch 6: 235 batches [07:26,  1.90s/ batches, loss=1.04, val_loss=1.07, acc=94.3, val_acc=94.2]
    Epoch 7: 235 batches [07:29,  1.91s/ batches, loss=0.926, val_loss=1.04, acc=95, val_acc=94.4]
    Epoch 8: 235 batches [07:30,  1.92s/ batches, loss=0.837, val_loss=0.962, acc=95.4, val_acc=94.8]
    Epoch 9: 235 batches [07:29,  1.91s/ batches, loss=0.761, val_loss=0.89, acc=95.9, val_acc=95.2]
    Epoch 10: 235 batches [07:28,  1.91s/ batches, loss=0.709, val_loss=0.854, acc=96.1, val_acc=95.4]
    Epoch 11: 235 batches [07:26,  1.90s/ batches, loss=0.66, val_loss=0.811, acc=96.4, val_acc=95.6]
    Epoch 12: 235 batches [07:28,  1.91s/ batches, loss=0.589, val_loss=0.775, acc=96.8, val_acc=95.8]
    Epoch 13: 235 batches [07:29,  1.91s/ batches, loss=0.535, val_loss=0.713, acc=97.1, val_acc=96.1]
    Epoch 14: 235 batches [07:30,  1.91s/ batches, loss=0.493, val_loss=0.739, acc=97.3, val_acc=96]
    Epoch 15: 235 batches [07:26,  1.90s/ batches, loss=0.461, val_loss=0.713, acc=97.5, val_acc=96.1]
    Epoch 16: 235 batches [07:32,  1.93s/ batches, loss=0.414, val_loss=0.712, acc=97.7, val_acc=96.1]
    Epoch 17: 235 batches [07:30,  1.91s/ batches, loss=0.387, val_loss=0.663, acc=97.9, val_acc=96.4]
    Epoch 18: 235 batches [07:29,  1.91s/ batches, loss=0.362, val_loss=0.678, acc=98, val_acc=96.3]
    Epoch 19: 235 batches [07:32,  1.93s/ batches, loss=0.329, val_loss=0.63, acc=98.2, val_acc=96.6]
    Epoch 20: 235 batches [07:27,  1.90s/ batches, loss=0.324, val_loss=0.649, acc=98.2, val_acc=96.5]
```

```
model.plot_results()
```





```
model.test(test_data)
```

```
    accuracy: 96.39%
```

```
model.save_state('mnist_cnn_adam.pickle')
```

✓ 0s completed at 5:34 PM