# Lab23

November 15, 2022

[ ]:
```python
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, BatchNormalization, Dropout, Dense,␣
↪Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.datasets import cifar10, mnist
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.optimizers import SGD, Adam
from tensorflow.keras.callbacks import EarlyStopping
import numpy as np

import matplotlib.pyplot as plt
```

[ ]:
```python
(trainX, trainy), (testX, testy) = cifar10.load_data()
```

[ ]:
```python
# summarize loaded dataset
print('Train: X=%s, y=%s' % (trainX.shape, trainy.shape))
print('Test: X=%s, y=%s' % (testX.shape, testy.shape))
```

```
Train: X=(50000, 32, 32, 3), y=(50000, 1)
Test: X=(10000, 32, 32, 3), y=(10000, 1)
```

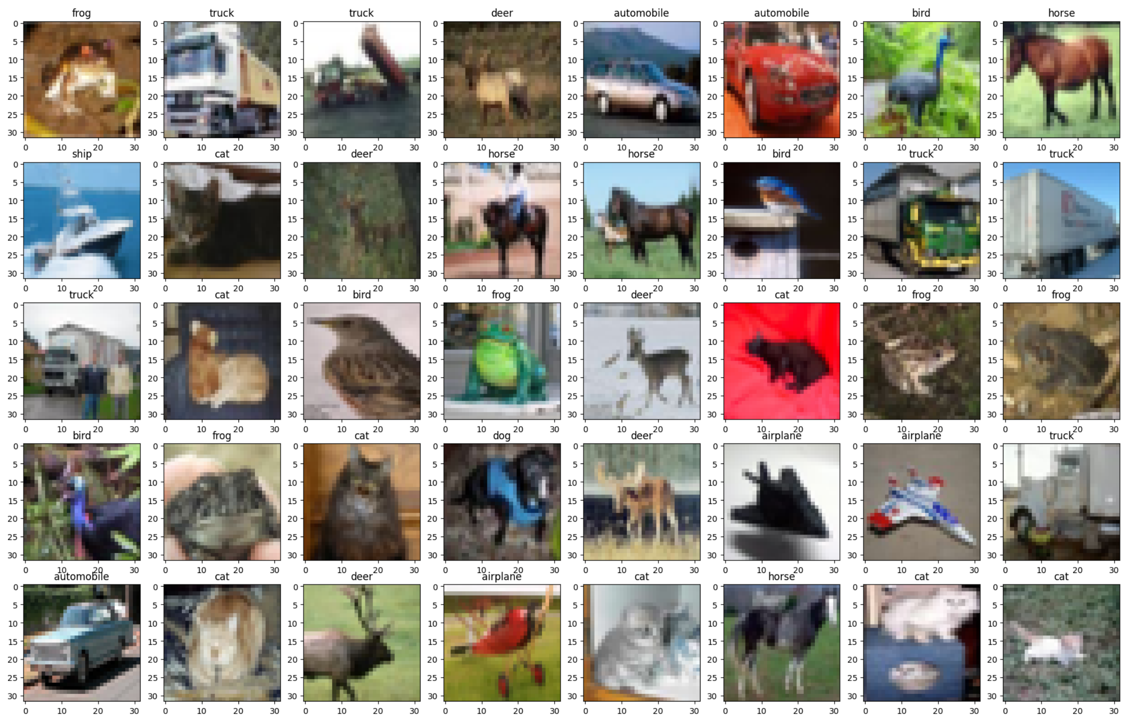
## 1 Categories

```
0: airplane
1: automobile
2: bird
3: cat
4: deer
5: dog
6: frog
7: horse
8: ship
9: truck
```

[ ]:
```python
catDict = {0: 'airplane',
           1: 'automobile',
           2: 'bird',
           3: 'cat',
           4: 'deer',
           5: 'dog',
           6: 'frog',
           7: 'horse',
           8: 'ship',
           9: 'truck'}
```

## 2 Visualize

[ ]:
```
fig, ax = plt.subplots(5,8, figsize=(24, 15))

count = 0
for i in range(5):
  for j in range(8):
    ax[i][j].imshow(trainX[count,:,:,:]);
    ax[i][j].set_title(catDict[trainy[count][0]]);
    count += 1
```



## 3 APIs

[ ]:
```
# load train and test dataset
def load_dataset(dataset = 'cifar10'):
  # load dataset
  if dataset == 'cifar10':
    (trainX, trainY), (testX, testY) = cifar10.load_data()
  elif dataset == 'mnist':
    (trainX, trainY), (testX, testY) = mnist.load_data()
  # one hot encode target values
  trainY = to_categorical(trainY)
  testY = to_categorical(testY)
  return trainX, trainY, testX, testY



# scale pixel values to [0, 1]
def prep_pixels(train, test):
  # convert from integers to floats
  train_norm = train.astype('float32')
  test_norm = test.astype('float32')
  # normalize to range 0-1
  train_norm = train_norm / 255.0
  test_norm = test_norm / 255.0
  # return normalized images
  return train_norm, test_norm

# define simple neural network model
def simpleModel(input_size, neurons=50, opt=Adam(learning_rate=0.01)):
  inputs = Input(shape=input_shape)
  flatten1 = Flatten()(inputs)
  dense1 = Dense(neurons, activation='relu')(flatten1)
  outputs = Dense(10, activation='softmax')(dense1)
  model = Model(inputs, outputs)
  model.compile(optimizer=opt, loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
  return model

def twoLayerModel(input_size, neurons1=50, neurons2=50,␣
 ↪opt=Adam(learning_rate=0.01)):
  inputs = Input(shape=input_shape)
  flatten1 = Flatten()(inputs)
  dense1 = Dense(neurons1, activation='relu')(flatten1)
  dense2 = Dense(neurons2, activation='relu')(dense1)
  outputs = Dense(10, activation='softmax')(dense2)
  model = Model(inputs, outputs)
  model.compile(optimizer=opt, loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
  return model

# plot diagnostic learning curves
def summarize_diagnostics(history):
```

```python
  fig, ax = plt.subplots(1,2, figsize=(20, 10))
  # plot loss
  ax[0].set_title('Loss Curves', fontsize=20)
  ax[0].plot(history.history['loss'], label='train')
  ax[0].plot(history.history['val_loss'], label='test')
  ax[0].set_xlabel('Epochs', fontsize=15)
  ax[0].set_ylabel('Loss', fontsize=15)
  ax[0].legend(fontsize=15)
  # plot accuracy
  ax[1].set_title('Classification Accuracy', fontsize=20)
  ax[1].plot(history.history['accuracy'], label='train')
  ax[1].plot(history.history['val_accuracy'], label='test')
  ax[1].set_xlabel('Epochs', fontsize=15)
  ax[1].set_ylabel('Accuracy', fontsize=15)
  ax[1].legend(fontsize=15)
```

## 4 Load CIFAR 10 and train

[ ]:
```python
# load dataset
trainX, trainY, testX, testY = load_dataset()
# prepare pixel data
trainX, testX = prep_pixels(trainX, testX)
# Get shape of input
input_shape = trainX[0].shape
# define model
opt = Adam(learning_rate=0.001)
model = simpleModel(input_shape, neurons=100, opt=opt)
# Print model summary
model.summary()
# fit model
history = model.fit(trainX, trainY, epochs=50, batch_size=64,␣
 ↪validation_data=(testX, testY), verbose=1)
# evaluate model
_, acc = model.evaluate(testX, testY, verbose=1)
print('> %.3f' % (acc * 100.0))
```

```
Metal device set to: Apple M1

systemMemory: 8.00 GB
maxCacheSize: 2.67 GB


2022-11-15 14:35:56.179456: I
tensorflow/core/common_runtime/pluggable_device/pluggable_device_factory.cc:305]
Could not identify NUMA node of platform GPU ID 0, defaulting to 0. Your kernel
may not have been built with NUMA support.
2022-11-15 14:35:56.187089: I
```

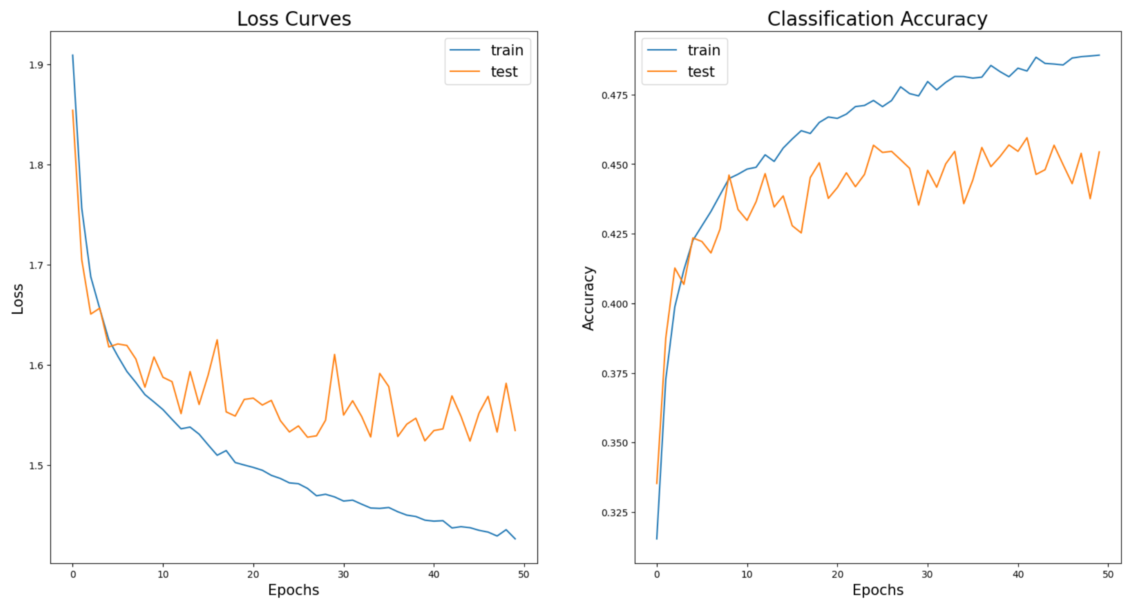
```
tensorflow/core/common_runtime/pluggable_device/pluggable_device_factory.cc:271]
Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 0
MB memory) -> physical PluggableDevice (device: 0, name: METAL, pci bus id:
<undefined>)

Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 32, 32, 3)]       0
_________________________________________________________________
flatten (Flatten)            (None, 3072)              0
_________________________________________________________________
dense (Dense)                (None, 100)               307300
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1010
=================================================================
Total params: 308,310
Trainable params: 308,310
Non-trainable params: 0
_________________________________________________________________
2022-11-15 14:35:58.766672: I
tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR
Optimization Passes are enabled (registered 2)
2022-11-15 14:35:58.808675: W
tensorflow/core/platform/profile_utils/cpu_utils.cc:128] Failed to get CPU
frequency: 0 Hz

Epoch 1/50

2022-11-15 14:35:59.349908: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

779/782 [============================>.] - ETA: 0s - loss: 1.9099 - accuracy:
0.3153

2022-11-15 14:36:07.322248: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 10s 11ms/step - loss: 1.9094 -
accuracy: 0.3154 - val_loss: 1.8543 - val_accuracy: 0.3353
Epoch 2/50
782/782 [==============================] - 8s 11ms/step - loss: 1.7568 -
accuracy: 0.3728 - val_loss: 1.7053 - val_accuracy: 0.3876
Epoch 3/50
782/782 [==============================] - 8s 10ms/step - loss: 1.6881 -
accuracy: 0.3988 - val_loss: 1.6508 - val_accuracy: 0.4127
Epoch 4/50
```

```
782/782 [==============================] - 8s 10ms/step - loss: 1.6561 -
accuracy: 0.4120 - val_loss: 1.6567 - val_accuracy: 0.4068
Epoch 5/50
782/782 [==============================] - 8s 10ms/step - loss: 1.6252 -
accuracy: 0.4226 - val_loss: 1.6180 - val_accuracy: 0.4235
Epoch 6/50
782/782 [==============================] - 8s 10ms/step - loss: 1.6087 -
accuracy: 0.4278 - val_loss: 1.6210 - val_accuracy: 0.4222
Epoch 7/50
782/782 [==============================] - 7s 10ms/step - loss: 1.5935 -
accuracy: 0.4330 - val_loss: 1.6194 - val_accuracy: 0.4181
Epoch 8/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5824 -
accuracy: 0.4389 - val_loss: 1.6058 - val_accuracy: 0.4266
Epoch 9/50
782/782 [==============================] - 7s 10ms/step - loss: 1.5704 -
accuracy: 0.4448 - val_loss: 1.5778 - val_accuracy: 0.4461
Epoch 10/50
782/782 [==============================] - 7s 9ms/step - loss: 1.5630 -
accuracy: 0.4464 - val_loss: 1.6080 - val_accuracy: 0.4337
Epoch 11/50
782/782 [==============================] - 8s 11ms/step - loss: 1.5554 -
accuracy: 0.4482 - val_loss: 1.5877 - val_accuracy: 0.4298
Epoch 12/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5456 -
accuracy: 0.4489 - val_loss: 1.5833 - val_accuracy: 0.4365
Epoch 13/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5362 -
accuracy: 0.4534 - val_loss: 1.5514 - val_accuracy: 0.4466
Epoch 14/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5379 -
accuracy: 0.4510 - val_loss: 1.5934 - val_accuracy: 0.4346
Epoch 15/50
782/782 [==============================] - 7s 9ms/step - loss: 1.5309 -
accuracy: 0.4558 - val_loss: 1.5605 - val_accuracy: 0.4386
Epoch 16/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5202 -
accuracy: 0.4590 - val_loss: 1.5899 - val_accuracy: 0.4279
Epoch 17/50
782/782 [==============================] - 7s 9ms/step - loss: 1.5098 -
accuracy: 0.4620 - val_loss: 1.6251 - val_accuracy: 0.4253
Epoch 18/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5145 -
accuracy: 0.4610 - val_loss: 1.5531 - val_accuracy: 0.4452
Epoch 19/50
782/782 [==============================] - 10s 13ms/step - loss: 1.5025 -
accuracy: 0.4650 - val_loss: 1.5489 - val_accuracy: 0.4505
Epoch 20/50
```

```
782/782 [==============================] - 8s 10ms/step - loss: 1.5000 -
accuracy: 0.4669 - val_loss: 1.5656 - val_accuracy: 0.4377
Epoch 21/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4977 -
accuracy: 0.4664 - val_loss: 1.5668 - val_accuracy: 0.4416
Epoch 22/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4948 -
accuracy: 0.4680 - val_loss: 1.5599 - val_accuracy: 0.4469
Epoch 23/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4897 -
accuracy: 0.4707 - val_loss: 1.5646 - val_accuracy: 0.4419
Epoch 24/50
782/782 [==============================] - 8s 11ms/step - loss: 1.4866 -
accuracy: 0.4711 - val_loss: 1.5444 - val_accuracy: 0.4463
Epoch 25/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4822 -
accuracy: 0.4729 - val_loss: 1.5331 - val_accuracy: 0.4568
Epoch 26/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4814 -
accuracy: 0.4707 - val_loss: 1.5391 - val_accuracy: 0.4542
Epoch 27/50
782/782 [==============================] - 9s 12ms/step - loss: 1.4767 -
accuracy: 0.4728 - val_loss: 1.5278 - val_accuracy: 0.4546
Epoch 28/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4693 -
accuracy: 0.4778 - val_loss: 1.5292 - val_accuracy: 0.4516
Epoch 29/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4708 -
accuracy: 0.4753 - val_loss: 1.5448 - val_accuracy: 0.4485
Epoch 30/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4682 -
accuracy: 0.4745 - val_loss: 1.6105 - val_accuracy: 0.4353
Epoch 31/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4641 -
accuracy: 0.4797 - val_loss: 1.5499 - val_accuracy: 0.4478
Epoch 32/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4650 -
accuracy: 0.4767 - val_loss: 1.5642 - val_accuracy: 0.4417
Epoch 33/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4609 -
accuracy: 0.4794 - val_loss: 1.5485 - val_accuracy: 0.4501
Epoch 34/50
782/782 [==============================] - 8s 11ms/step - loss: 1.4571 -
accuracy: 0.4815 - val_loss: 1.5280 - val_accuracy: 0.4546
Epoch 35/50
782/782 [==============================] - 9s 12ms/step - loss: 1.4567 -
accuracy: 0.4815 - val_loss: 1.5915 - val_accuracy: 0.4358
Epoch 36/50
```

```
782/782 [==============================] - 8s 10ms/step - loss: 1.4576 -
accuracy: 0.4809 - val_loss: 1.5784 - val_accuracy: 0.4443
Epoch 37/50
782/782 [==============================] - 9s 12ms/step - loss: 1.4534 -
accuracy: 0.4813 - val_loss: 1.5285 - val_accuracy: 0.4560
Epoch 38/50
782/782 [==============================] - 8s 11ms/step - loss: 1.4500 -
accuracy: 0.4855 - val_loss: 1.5408 - val_accuracy: 0.4491
Epoch 39/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4486 -
accuracy: 0.4833 - val_loss: 1.5468 - val_accuracy: 0.4527
Epoch 40/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4450 -
accuracy: 0.4814 - val_loss: 1.5242 - val_accuracy: 0.4569
Epoch 41/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4440 -
accuracy: 0.4845 - val_loss: 1.5345 - val_accuracy: 0.4546
Epoch 42/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4445 -
accuracy: 0.4835 - val_loss: 1.5361 - val_accuracy: 0.4595
Epoch 43/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4372 -
accuracy: 0.4884 - val_loss: 1.5691 - val_accuracy: 0.4463
Epoch 44/50
782/782 [==============================] - 10s 12ms/step - loss: 1.4385 -
accuracy: 0.4862 - val_loss: 1.5487 - val_accuracy: 0.4480
Epoch 45/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4375 -
accuracy: 0.4860 - val_loss: 1.5239 - val_accuracy: 0.4568
Epoch 46/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4348 -
accuracy: 0.4856 - val_loss: 1.5519 - val_accuracy: 0.4498
Epoch 47/50
782/782 [==============================] - 8s 10ms/step - loss: 1.4331 -
accuracy: 0.4881 - val_loss: 1.5687 - val_accuracy: 0.4430
Epoch 48/50
782/782 [==============================] - 7s 10ms/step - loss: 1.4291 -
accuracy: 0.4886 - val_loss: 1.5330 - val_accuracy: 0.4539
Epoch 49/50
782/782 [==============================] - 7s 9ms/step - loss: 1.4355 -
accuracy: 0.4889 - val_loss: 1.5817 - val_accuracy: 0.4376
Epoch 50/50
782/782 [==============================] - 7s 9ms/step - loss: 1.4263 -
accuracy: 0.4892 - val_loss: 1.5345 - val_accuracy: 0.4544
313/313 [==============================] - 2s 7ms/step - loss: 1.5345 -
accuracy: 0.4544
> 45.440
```

```
[ ]: # loss curves & accuracy
     summarize_diagnostics(history)
```



```
[ ]: # Use a slightly deeper model (2 layers)
     # 2 layers
     # define model
     opt = Adam(learning_rate=0.001)
     model = twoLayerModel(input_shape, neurons1=100, neurons2=100, opt=opt)
     # Print model summary
     model.summary()
     # fit model
     history = model.fit(trainX, trainY, epochs=50, batch_size=64,␣
      ↪validation_data=(testX, testY), verbose=1)
     # evaluate model
     _, acc = model.evaluate(testX, testY, verbose=1)
     print('> %.3f' % (acc * 100.0))

     # loss curves & accuracy
     summarize_diagnostics(history)
```

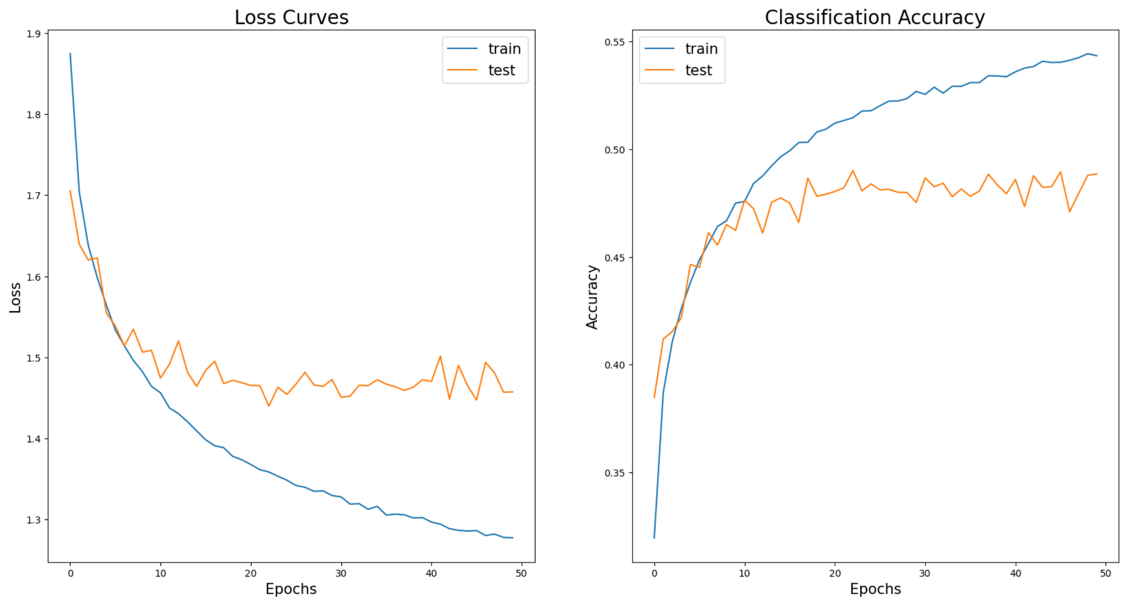
```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 32, 32, 3)]       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3072)              0
_________________________________________________________________
```

```
dense_2 (Dense)             (None, 100)              307300

_________________________________________________________________
dense_3 (Dense)             (None, 100)              10100

_________________________________________________________________
dense_4 (Dense)             (None, 10)               1010
=================================================================
Total params: 318,410
Trainable params: 318,410
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50
  1/782 […] - ETA: 11:33 - loss: 2.4068 - accuracy:
0.0938

2022-11-15 14:42:53.928057: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - ETA: 0s - loss: 1.8747 - accuracy:
0.3196

2022-11-15 14:43:01.763007: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 10s 11ms/step - loss: 1.8747 -
accuracy: 0.3196 - val_loss: 1.7055 - val_accuracy: 0.3847
Epoch 2/50
782/782 [==============================] - 8s 11ms/step - loss: 1.7046 -
accuracy: 0.3871 - val_loss: 1.6395 - val_accuracy: 0.4119
Epoch 3/50
782/782 [==============================] - 8s 11ms/step - loss: 1.6383 -
accuracy: 0.4108 - val_loss: 1.6201 - val_accuracy: 0.4154
Epoch 4/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5981 -
accuracy: 0.4261 - val_loss: 1.6227 - val_accuracy: 0.4218
Epoch 5/50
782/782 [==============================] - 8s 10ms/step - loss: 1.5647 -
accuracy: 0.4381 - val_loss: 1.5549 - val_accuracy: 0.4465
Epoch 6/50
782/782 [==============================] - 9s 11ms/step - loss: 1.5336 -
accuracy: 0.4486 - val_loss: 1.5385 - val_accuracy: 0.4451
Epoch 7/50
782/782 [==============================] - 9s 11ms/step - loss: 1.5141 -
accuracy: 0.4562 - val_loss: 1.5141 - val_accuracy: 0.4613
Epoch 8/50
782/782 [==============================] - 10s 13ms/step - loss: 1.4959 -
accuracy: 0.4641 - val_loss: 1.5345 - val_accuracy: 0.4555
Epoch 9/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4822 -
```

```
accuracy: 0.4668 - val_loss: 1.5063 - val_accuracy: 0.4650
Epoch 10/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4641 -
accuracy: 0.4750 - val_loss: 1.5084 - val_accuracy: 0.4623
Epoch 11/50
782/782 [==============================] - 10s 13ms/step - loss: 1.4561 -
accuracy: 0.4757 - val_loss: 1.4743 - val_accuracy: 0.4763
Epoch 12/50
782/782 [==============================] - 12s 15ms/step - loss: 1.4373 -
accuracy: 0.4842 - val_loss: 1.4916 - val_accuracy: 0.4724
Epoch 13/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4303 -
accuracy: 0.4876 - val_loss: 1.5202 - val_accuracy: 0.4611
Epoch 14/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4203 -
accuracy: 0.4923 - val_loss: 1.4813 - val_accuracy: 0.4754
Epoch 15/50
782/782 [==============================] - 9s 11ms/step - loss: 1.4092 -
accuracy: 0.4965 - val_loss: 1.4640 - val_accuracy: 0.4774
Epoch 16/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3981 -
accuracy: 0.4993 - val_loss: 1.4838 - val_accuracy: 0.4750
Epoch 17/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3908 -
accuracy: 0.5031 - val_loss: 1.4950 - val_accuracy: 0.4660
Epoch 18/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3883 -
accuracy: 0.5033 - val_loss: 1.4674 - val_accuracy: 0.4866
Epoch 19/50
782/782 [==============================] - 9s 12ms/step - loss: 1.3777 -
accuracy: 0.5080 - val_loss: 1.4715 - val_accuracy: 0.4782
Epoch 20/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3734 -
accuracy: 0.5093 - val_loss: 1.4686 - val_accuracy: 0.4791
Epoch 21/50
782/782 [==============================] - 10s 12ms/step - loss: 1.3676 -
accuracy: 0.5121 - val_loss: 1.4652 - val_accuracy: 0.4804
Epoch 22/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3612 -
accuracy: 0.5133 - val_loss: 1.4649 - val_accuracy: 0.4821
Epoch 23/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3583 -
accuracy: 0.5146 - val_loss: 1.4396 - val_accuracy: 0.4901
Epoch 24/50
782/782 [==============================] - 10s 12ms/step - loss: 1.3531 -
accuracy: 0.5177 - val_loss: 1.4630 - val_accuracy: 0.4807
Epoch 25/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3482 -
```

```
accuracy: 0.5179 - val_loss: 1.4542 - val_accuracy: 0.4839
Epoch 26/50
782/782 [==============================] - 9s 12ms/step - loss: 1.3416 -
accuracy: 0.5202 - val_loss: 1.4667 - val_accuracy: 0.4811
Epoch 27/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3395 -
accuracy: 0.5224 - val_loss: 1.4814 - val_accuracy: 0.4814
Epoch 28/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3345 -
accuracy: 0.5224 - val_loss: 1.4657 - val_accuracy: 0.4800
Epoch 29/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3350 -
accuracy: 0.5236 - val_loss: 1.4639 - val_accuracy: 0.4799
Epoch 30/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3293 -
accuracy: 0.5268 - val_loss: 1.4725 - val_accuracy: 0.4753
Epoch 31/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3276 -
accuracy: 0.5255 - val_loss: 1.4505 - val_accuracy: 0.4867
Epoch 32/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3187 -
accuracy: 0.5288 - val_loss: 1.4521 - val_accuracy: 0.4826
Epoch 33/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3193 -
accuracy: 0.5260 - val_loss: 1.4654 - val_accuracy: 0.4842
Epoch 34/50
782/782 [==============================] - 8s 10ms/step - loss: 1.3123 -
accuracy: 0.5293 - val_loss: 1.4649 - val_accuracy: 0.4779
Epoch 35/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3159 -
accuracy: 0.5292 - val_loss: 1.4722 - val_accuracy: 0.4816
Epoch 36/50
782/782 [==============================] - 9s 11ms/step - loss: 1.3051 -
accuracy: 0.5310 - val_loss: 1.4669 - val_accuracy: 0.4781
Epoch 37/50
782/782 [==============================] - 8s 11ms/step - loss: 1.3064 -
accuracy: 0.5309 - val_loss: 1.4635 - val_accuracy: 0.4806
Epoch 38/50
782/782 [==============================] - 9s 12ms/step - loss: 1.3054 -
accuracy: 0.5341 - val_loss: 1.4590 - val_accuracy: 0.4884
Epoch 39/50
782/782 [==============================] - 10s 12ms/step - loss: 1.3016 -
accuracy: 0.5340 - val_loss: 1.4631 - val_accuracy: 0.4834
Epoch 40/50
782/782 [==============================] - 10s 13ms/step - loss: 1.3021 -
accuracy: 0.5336 - val_loss: 1.4722 - val_accuracy: 0.4793
Epoch 41/50
782/782 [==============================] - 9s 11ms/step - loss: 1.2966 -
```

```
accuracy: 0.5360 - val_loss: 1.4700 - val_accuracy: 0.4860
Epoch 42/50
782/782 [==============================] - 9s 11ms/step - loss: 1.2940 -
accuracy: 0.5377 - val_loss: 1.5014 - val_accuracy: 0.4734
Epoch 43/50
782/782 [==============================] - 8s 11ms/step - loss: 1.2884 -
accuracy: 0.5384 - val_loss: 1.4482 - val_accuracy: 0.4877
Epoch 44/50
782/782 [==============================] - 9s 11ms/step - loss: 1.2863 -
accuracy: 0.5408 - val_loss: 1.4899 - val_accuracy: 0.4823
Epoch 45/50
782/782 [==============================] - 8s 11ms/step - loss: 1.2854 -
accuracy: 0.5403 - val_loss: 1.4650 - val_accuracy: 0.4826
Epoch 46/50
782/782 [==============================] - 8s 11ms/step - loss: 1.2860 -
accuracy: 0.5404 - val_loss: 1.4472 - val_accuracy: 0.4895
Epoch 47/50
782/782 [==============================] - 8s 11ms/step - loss: 1.2799 -
accuracy: 0.5413 - val_loss: 1.4938 - val_accuracy: 0.4710
Epoch 48/50
782/782 [==============================] - 8s 11ms/step - loss: 1.2816 -
accuracy: 0.5425 - val_loss: 1.4806 - val_accuracy: 0.4795
Epoch 49/50
782/782 [==============================] - 9s 12ms/step - loss: 1.2776 -
accuracy: 0.5444 - val_loss: 1.4568 - val_accuracy: 0.4879
Epoch 50/50
782/782 [==============================] - 9s 12ms/step - loss: 1.2771 -
accuracy: 0.5434 - val_loss: 1.4572 - val_accuracy: 0.4885
313/313 [==============================] - 2s 8ms/step - loss: 1.4572 -
accuracy: 0.4885
> 48.850
```



## 5 Load MNIST and train

```
[ ]: # load dataset
     trainX, trainY, testX, testY = load_dataset('mnist')
     # prepare pixel data
     trainX, testX = prep_pixels(trainX, testX)
     # Expand dimensions to take care of channels
     trainX = np.expand_dims(trainX, trainX.ndim)
     testX = np.expand_dims(testX, testX.ndim)
     # Get shape of input
     input_shape = trainX[0].shape
     # define model
     opt = Adam(learning_rate=0.001)
     model = simpleModel(input_shape, neurons=10, opt=opt)
     # Print model summary
     model.summary()
     # fit model
     history = model.fit(trainX, trainY, epochs=50, batch_size=64,␣
      ↪validation_data=(testX, testY), verbose=1)
     # evaluate model
     _, acc = model.evaluate(testX, testY, verbose=1)
     print('> %.3f' % (acc * 100.0))

     # loss curves & accuracy
     summarize_diagnostics(history)
```

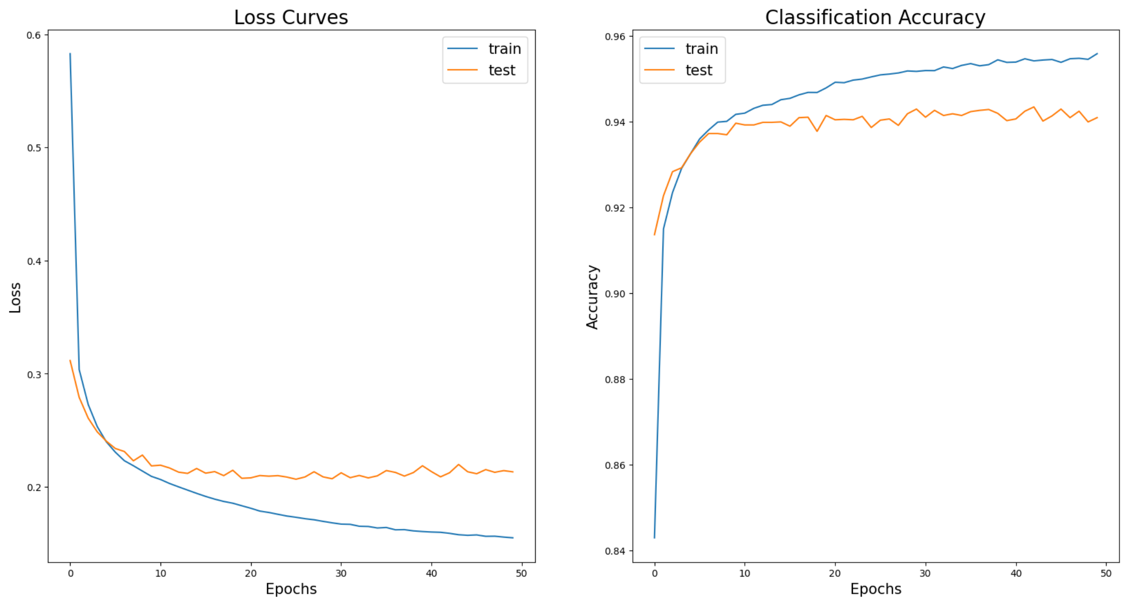
```
Model: "model_2"
```

```
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
input_3 (InputLayer)        [(None, 28, 28, 1)]       0

_________________________________________________________________
flatten_2 (Flatten)         (None, 784)               0

_________________________________________________________________
dense_5 (Dense)             (None, 10)                7850

_________________________________________________________________
dense_6 (Dense)             (None, 10)                110

=================================================================
Total params: 7,960
Trainable params: 7,960
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50

2022-11-15 14:50:17.702654: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - ETA: 0s - loss: 0.5828 - accuracy:
0.8429

2022-11-15 14:50:27.179055: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - 11s 11ms/step - loss: 0.5828 -
accuracy: 0.8429 - val_loss: 0.3116 - val_accuracy: 0.9136
Epoch 2/50
938/938 [==============================] - 10s 10ms/step - loss: 0.3037 -
accuracy: 0.9150 - val_loss: 0.2791 - val_accuracy: 0.9227
Epoch 3/50
938/938 [==============================] - 9s 10ms/step - loss: 0.2728 -
accuracy: 0.9234 - val_loss: 0.2608 - val_accuracy: 0.9283
Epoch 4/50
938/938 [==============================] - 9s 10ms/step - loss: 0.2533 -
accuracy: 0.9291 - val_loss: 0.2486 - val_accuracy: 0.9292
Epoch 5/50
938/938 [==============================] - 8s 9ms/step - loss: 0.2399 -
accuracy: 0.9325 - val_loss: 0.2404 - val_accuracy: 0.9325
Epoch 6/50
938/938 [==============================] - 8s 9ms/step - loss: 0.2307 -
accuracy: 0.9359 - val_loss: 0.2339 - val_accuracy: 0.9352
Epoch 7/50
938/938 [==============================] - 9s 9ms/step - loss: 0.2231 -
accuracy: 0.9380 - val_loss: 0.2313 - val_accuracy: 0.9372
Epoch 8/50
938/938 [==============================] - 9s 9ms/step - loss: 0.2187 -
```

```
accuracy: 0.9399 - val_loss: 0.2230 - val_accuracy: 0.9372
Epoch 9/50
938/938 [==============================] - 8s 9ms/step - loss: 0.2140 -
accuracy: 0.9400 - val_loss: 0.2281 - val_accuracy: 0.9369
Epoch 10/50
938/938 [==============================] - 10s 11ms/step - loss: 0.2093 -
accuracy: 0.9417 - val_loss: 0.2185 - val_accuracy: 0.9396
Epoch 11/50
938/938 [==============================] - 10s 11ms/step - loss: 0.2065 -
accuracy: 0.9420 - val_loss: 0.2192 - val_accuracy: 0.9392
Epoch 12/50
938/938 [==============================] - 9s 9ms/step - loss: 0.2030 -
accuracy: 0.9431 - val_loss: 0.2167 - val_accuracy: 0.9392
Epoch 13/50
938/938 [==============================] - 9s 9ms/step - loss: 0.2000 -
accuracy: 0.9438 - val_loss: 0.2130 - val_accuracy: 0.9398
Epoch 14/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1972 -
accuracy: 0.9440 - val_loss: 0.2119 - val_accuracy: 0.9398
Epoch 15/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1943 -
accuracy: 0.9451 - val_loss: 0.2163 - val_accuracy: 0.9399
Epoch 16/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1916 -
accuracy: 0.9454 - val_loss: 0.2121 - val_accuracy: 0.9389
Epoch 17/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1892 -
accuracy: 0.9462 - val_loss: 0.2135 - val_accuracy: 0.9409
Epoch 18/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1871 -
accuracy: 0.9468 - val_loss: 0.2099 - val_accuracy: 0.9410
Epoch 19/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1856 -
accuracy: 0.9468 - val_loss: 0.2147 - val_accuracy: 0.9377
Epoch 20/50
938/938 [==============================] - 9s 10ms/step - loss: 0.1833 -
accuracy: 0.9478 - val_loss: 0.2076 - val_accuracy: 0.9414
Epoch 21/50
938/938 [==============================] - 9s 10ms/step - loss: 0.1811 -
accuracy: 0.9492 - val_loss: 0.2080 - val_accuracy: 0.9404
Epoch 22/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1786 -
accuracy: 0.9491 - val_loss: 0.2100 - val_accuracy: 0.9405
Epoch 23/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1774 -
accuracy: 0.9496 - val_loss: 0.2095 - val_accuracy: 0.9404
Epoch 24/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1758 -
```

```
accuracy: 0.9499 - val_loss: 0.2099 - val_accuracy: 0.9412
Epoch 25/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1742 -
accuracy: 0.9504 - val_loss: 0.2087 - val_accuracy: 0.9386
Epoch 26/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1731 -
accuracy: 0.9509 - val_loss: 0.2068 - val_accuracy: 0.9403
Epoch 27/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1719 -
accuracy: 0.9511 - val_loss: 0.2088 - val_accuracy: 0.9406
Epoch 28/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1709 -
accuracy: 0.9513 - val_loss: 0.2134 - val_accuracy: 0.9391
Epoch 29/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1695 -
accuracy: 0.9518 - val_loss: 0.2088 - val_accuracy: 0.9418
Epoch 30/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1682 -
accuracy: 0.9517 - val_loss: 0.2072 - val_accuracy: 0.9429
Epoch 31/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1671 -
accuracy: 0.9519 - val_loss: 0.2124 - val_accuracy: 0.9410
Epoch 32/50
938/938 [==============================] - 10s 11ms/step - loss: 0.1669 -
accuracy: 0.9519 - val_loss: 0.2081 - val_accuracy: 0.9426
Epoch 33/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1652 -
accuracy: 0.9527 - val_loss: 0.2101 - val_accuracy: 0.9414
Epoch 34/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1650 -
accuracy: 0.9524 - val_loss: 0.2080 - val_accuracy: 0.9418
Epoch 35/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1637 -
accuracy: 0.9531 - val_loss: 0.2097 - val_accuracy: 0.9414
Epoch 36/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1641 -
accuracy: 0.9535 - val_loss: 0.2144 - val_accuracy: 0.9423
Epoch 37/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1621 -
accuracy: 0.9530 - val_loss: 0.2128 - val_accuracy: 0.9426
Epoch 38/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1622 -
accuracy: 0.9533 - val_loss: 0.2095 - val_accuracy: 0.9428
Epoch 39/50
938/938 [==============================] - 10s 10ms/step - loss: 0.1611 -
accuracy: 0.9544 - val_loss: 0.2127 - val_accuracy: 0.9419
Epoch 40/50
938/938 [==============================] - 9s 10ms/step - loss: 0.1605 -
```

```
accuracy: 0.9538 - val_loss: 0.2187 - val_accuracy: 0.9402
Epoch 41/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1601 -
accuracy: 0.9539 - val_loss: 0.2135 - val_accuracy: 0.9406
Epoch 42/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1599 -
accuracy: 0.9546 - val_loss: 0.2089 - val_accuracy: 0.9424
Epoch 43/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1590 -
accuracy: 0.9541 - val_loss: 0.2125 - val_accuracy: 0.9434
Epoch 44/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1577 -
accuracy: 0.9543 - val_loss: 0.2198 - val_accuracy: 0.9401
Epoch 45/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1572 -
accuracy: 0.9545 - val_loss: 0.2134 - val_accuracy: 0.9413
Epoch 46/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1576 -
accuracy: 0.9538 - val_loss: 0.2116 - val_accuracy: 0.9429
Epoch 47/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1563 -
accuracy: 0.9546 - val_loss: 0.2153 - val_accuracy: 0.9409
Epoch 48/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1564 -
accuracy: 0.9547 - val_loss: 0.2129 - val_accuracy: 0.9424
Epoch 49/50
938/938 [==============================] - 8s 9ms/step - loss: 0.1556 -
accuracy: 0.9545 - val_loss: 0.2144 - val_accuracy: 0.9399
Epoch 50/50
938/938 [==============================] - 9s 9ms/step - loss: 0.1550 -
accuracy: 0.9558 - val_loss: 0.2133 - val_accuracy: 0.9409
313/313 [==============================] - 2s 7ms/step - loss: 0.2133 -
accuracy: 0.9409
> 94.090
```



```
[ ]: # Use a slightly deeper model (2 layers)
     # 2 layers
     # define model
     opt = Adam(learning_rate=0.001)
     model = twoLayerModel(input_shape, neurons1=100, neurons2=100, opt=opt)
     # Print model summary
     model.summary()
     # fit model
     history = model.fit(trainX, trainY, epochs=50, batch_size=64,␣
      ↪validation_data=(testX, testY), verbose=1)
     # evaluate model
     _, acc = model.evaluate(testX, testY, verbose=1)
     print('> %.3f' % (acc * 100.0))

     # loss curves & accuracy
     summarize_diagnostics(history)
```

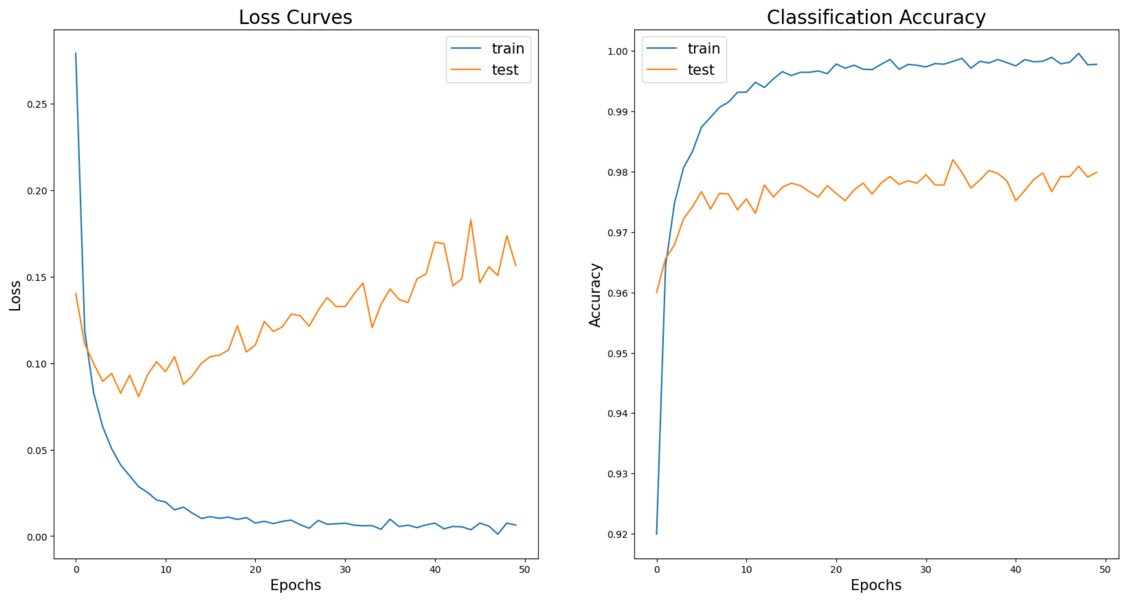
```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, 28, 28, 1)]       0
_________________________________________________________________
flatten_3 (Flatten)          (None, 784)               0
_________________________________________________________________
dense_7 (Dense)              (None, 100)               78500
_________________________________________________________________
dense_8 (Dense)              (None, 100)               10100
```

```
_________________________________________________________________
dense_9 (Dense)              (None, 10)                1010
=================================================================
Total params: 89,610
Trainable params: 89,610
Non-trainable params: 0
_________________________________________________________________
Epoch 1/50
  6/938 […] - ETA: 10s - loss: 2.1436 - accuracy:
0.2708

2022-11-15 14:57:41.090462: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - ETA: 0s - loss: 0.2791 - accuracy:
0.9199

2022-11-15 14:57:49.683589: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - 10s 10ms/step - loss: 0.2791 -
accuracy: 0.9199 - val_loss: 0.1402 - val_accuracy: 0.9600
Epoch 2/50
938/938 [==============================] - 9s 10ms/step - loss: 0.1187 -
accuracy: 0.9644 - val_loss: 0.1112 - val_accuracy: 0.9656
Epoch 3/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0827 -
accuracy: 0.9749 - val_loss: 0.0997 - val_accuracy: 0.9679
Epoch 4/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0633 -
accuracy: 0.9807 - val_loss: 0.0895 - val_accuracy: 0.9722
Epoch 5/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0505 -
accuracy: 0.9834 - val_loss: 0.0940 - val_accuracy: 0.9742
Epoch 6/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0411 -
accuracy: 0.9874 - val_loss: 0.0826 - val_accuracy: 0.9767
Epoch 7/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0349 -
accuracy: 0.9890 - val_loss: 0.0931 - val_accuracy: 0.9738
Epoch 8/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0286 -
accuracy: 0.9907 - val_loss: 0.0806 - val_accuracy: 0.9764
Epoch 9/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0252 -
accuracy: 0.9915 - val_loss: 0.0932 - val_accuracy: 0.9763
Epoch 10/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0209 -
```

```
accuracy: 0.9931 - val_loss: 0.1009 - val_accuracy: 0.9737
Epoch 11/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0197 -
accuracy: 0.9932 - val_loss: 0.0951 - val_accuracy: 0.9755
Epoch 12/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0151 -
accuracy: 0.9948 - val_loss: 0.1037 - val_accuracy: 0.9731
Epoch 13/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0168 -
accuracy: 0.9940 - val_loss: 0.0878 - val_accuracy: 0.9778
Epoch 14/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0133 -
accuracy: 0.9954 - val_loss: 0.0927 - val_accuracy: 0.9758
Epoch 15/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0102 -
accuracy: 0.9966 - val_loss: 0.1000 - val_accuracy: 0.9774
Epoch 16/50
938/938 [==============================] - 14s 15ms/step - loss: 0.0113 -
accuracy: 0.9959 - val_loss: 0.1038 - val_accuracy: 0.9781
Epoch 17/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0103 -
accuracy: 0.9965 - val_loss: 0.1046 - val_accuracy: 0.9777
Epoch 18/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0110 -
accuracy: 0.9965 - val_loss: 0.1075 - val_accuracy: 0.9767
Epoch 19/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0096 -
accuracy: 0.9967 - val_loss: 0.1215 - val_accuracy: 0.9758
Epoch 20/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0107 -
accuracy: 0.9963 - val_loss: 0.1064 - val_accuracy: 0.9777
Epoch 21/50
938/938 [==============================] - 12s 13ms/step - loss: 0.0075 -
accuracy: 0.9979 - val_loss: 0.1105 - val_accuracy: 0.9764
Epoch 22/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0086 -
accuracy: 0.9972 - val_loss: 0.1242 - val_accuracy: 0.9752
Epoch 23/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0072 -
accuracy: 0.9977 - val_loss: 0.1183 - val_accuracy: 0.9770
Epoch 24/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0085 -
accuracy: 0.9970 - val_loss: 0.1209 - val_accuracy: 0.9781
Epoch 25/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0092 -
accuracy: 0.9969 - val_loss: 0.1284 - val_accuracy: 0.9763
Epoch 26/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0066 -
```

```
accuracy: 0.9978 - val_loss: 0.1273 - val_accuracy: 0.9781
Epoch 27/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0046 -
accuracy: 0.9986 - val_loss: 0.1213 - val_accuracy: 0.9792
Epoch 28/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0091 -
accuracy: 0.9970 - val_loss: 0.1306 - val_accuracy: 0.9779
Epoch 29/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0068 -
accuracy: 0.9978 - val_loss: 0.1380 - val_accuracy: 0.9785
Epoch 30/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0071 -
accuracy: 0.9976 - val_loss: 0.1327 - val_accuracy: 0.9781
Epoch 31/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0074 -
accuracy: 0.9974 - val_loss: 0.1327 - val_accuracy: 0.9795
Epoch 32/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0063 -
accuracy: 0.9979 - val_loss: 0.1400 - val_accuracy: 0.9778
Epoch 33/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0060 -
accuracy: 0.9978 - val_loss: 0.1464 - val_accuracy: 0.9778
Epoch 34/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0060 -
accuracy: 0.9983 - val_loss: 0.1205 - val_accuracy: 0.9820
Epoch 35/50
938/938 [==============================] - 10s 11ms/step - loss: 0.0040 -
accuracy: 0.9988 - val_loss: 0.1340 - val_accuracy: 0.9799
Epoch 36/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0098 -
accuracy: 0.9972 - val_loss: 0.1429 - val_accuracy: 0.9773
Epoch 37/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0055 -
accuracy: 0.9983 - val_loss: 0.1368 - val_accuracy: 0.9786
Epoch 38/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0063 -
accuracy: 0.9980 - val_loss: 0.1350 - val_accuracy: 0.9802
Epoch 39/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0049 -
accuracy: 0.9986 - val_loss: 0.1488 - val_accuracy: 0.9797
Epoch 40/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0065 -
accuracy: 0.9981 - val_loss: 0.1514 - val_accuracy: 0.9785
Epoch 41/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0075 -
accuracy: 0.9975 - val_loss: 0.1699 - val_accuracy: 0.9752
Epoch 42/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0042 -
```

```
accuracy: 0.9986 - val_loss: 0.1691 - val_accuracy: 0.9769
Epoch 43/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0056 -
accuracy: 0.9982 - val_loss: 0.1447 - val_accuracy: 0.9787
Epoch 44/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0054 -
accuracy: 0.9983 - val_loss: 0.1488 - val_accuracy: 0.9798
Epoch 45/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0037 -
accuracy: 0.9990 - val_loss: 0.1829 - val_accuracy: 0.9767
Epoch 46/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0075 -
accuracy: 0.9979 - val_loss: 0.1465 - val_accuracy: 0.9792
Epoch 47/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0058 -
accuracy: 0.9981 - val_loss: 0.1557 - val_accuracy: 0.9792
Epoch 48/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0011 -
accuracy: 0.9996 - val_loss: 0.1507 - val_accuracy: 0.9809
Epoch 49/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0075 -
accuracy: 0.9977 - val_loss: 0.1737 - val_accuracy: 0.9791
Epoch 50/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0064 -
accuracy: 0.9978 - val_loss: 0.1564 - val_accuracy: 0.9799
313/313 [==============================] - 2s 8ms/step - loss: 0.1564 -
accuracy: 0.9799
> 97.990
```



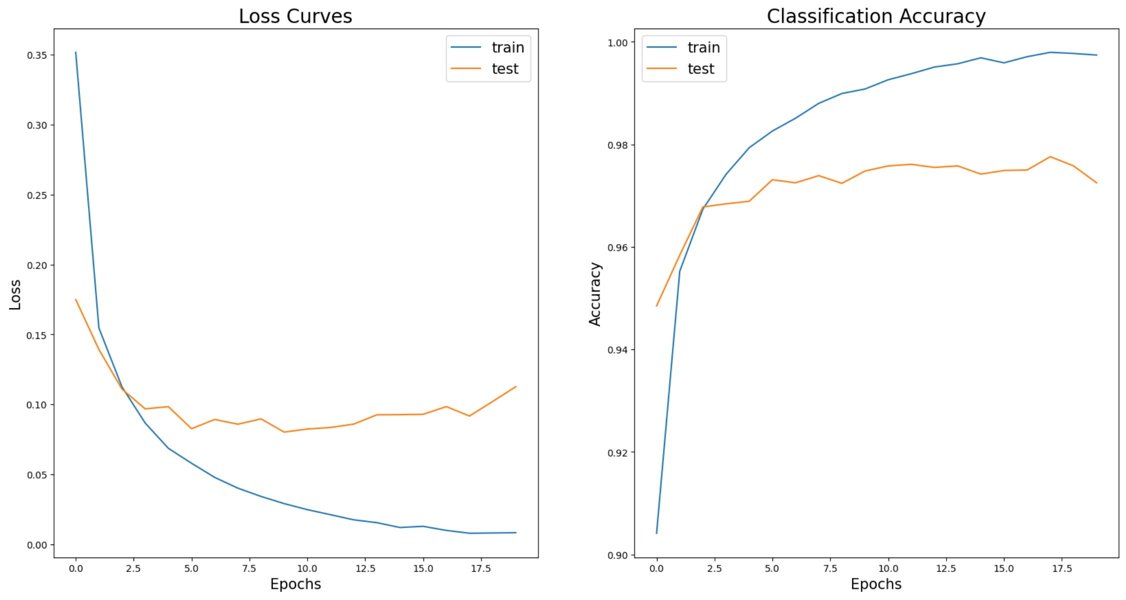
## 6 How do we find the point where the model starts to overfit and stop training there?

```
[ ]: # We will use the EarlyStopping API in keras
     # early stopping
     es = EarlyStopping(monitor='val_loss',
                        mode='min',
                        verbose=1,
                        patience=10,
                        restore_best_weights=True)

     # define model
     opt = Adam(learning_rate=0.0005)
     model = twoLayerModel(input_shape, neurons1=100, neurons2=100, opt=opt)
     # Print model summary
     model.summary()
     # fit model
     history = model.fit(trainX, trainY,
                         epochs=50,
                         batch_size=64,
                         validation_data=(testX, testY),
                         verbose=1,
                         callbacks=[es])
     # evaluate model
     _, acc = model.evaluate(testX, testY, verbose=1)
     print('> %.3f' % (acc * 100.0))

     # loss curves & accuracy
     summarize_diagnostics(history)
```

```
Model: "model_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         [(None, 28, 28, 1)]       0
_________________________________________________________________
flatten_4 (Flatten)          (None, 784)               0
_________________________________________________________________
dense_10 (Dense)             (None, 100)               78500
_________________________________________________________________
dense_11 (Dense)             (None, 100)               10100
_________________________________________________________________
dense_12 (Dense)             (None, 10)                1010
=================================================================
Total params: 89,610
Trainable params: 89,610
Non-trainable params: 0
```

```
----------------------------------------------------------------
Epoch 1/50
  6/938 […] - ETA: 9s - loss: 2.2248 - accuracy:
0.1641

2022-11-15 15:05:51.834238: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - ETA: 0s - loss: 0.3517 - accuracy:
0.9042

2022-11-15 15:06:02.035227: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - 12s 12ms/step - loss: 0.3517 -
accuracy: 0.9042 - val_loss: 0.1749 - val_accuracy: 0.9485
Epoch 2/50
938/938 [==============================] - 10s 11ms/step - loss: 0.1546 -
accuracy: 0.9553 - val_loss: 0.1393 - val_accuracy: 0.9584
Epoch 3/50
938/938 [==============================] - 10s 10ms/step - loss: 0.1124 -
accuracy: 0.9674 - val_loss: 0.1109 - val_accuracy: 0.9678
Epoch 4/50
938/938 [==============================] - 12s 13ms/step - loss: 0.0866 -
accuracy: 0.9742 - val_loss: 0.0968 - val_accuracy: 0.9684
Epoch 5/50
938/938 [==============================] - 12s 13ms/step - loss: 0.0686 -
accuracy: 0.9794 - val_loss: 0.0984 - val_accuracy: 0.9689
Epoch 6/50
938/938 [==============================] - 11s 11ms/step - loss: 0.0579 -
accuracy: 0.9826 - val_loss: 0.0826 - val_accuracy: 0.9731
Epoch 7/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0478 -
accuracy: 0.9851 - val_loss: 0.0892 - val_accuracy: 0.9725
Epoch 8/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0401 -
accuracy: 0.9880 - val_loss: 0.0858 - val_accuracy: 0.9739
Epoch 9/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0342 -
accuracy: 0.9899 - val_loss: 0.0896 - val_accuracy: 0.9724
Epoch 10/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0291 -
accuracy: 0.9908 - val_loss: 0.0802 - val_accuracy: 0.9748
Epoch 11/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0247 -
accuracy: 0.9926 - val_loss: 0.0824 - val_accuracy: 0.9758
Epoch 12/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0211 -
```

```
accuracy: 0.9938 - val_loss: 0.0835 - val_accuracy: 0.9761
Epoch 13/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0175 -
accuracy: 0.9951 - val_loss: 0.0859 - val_accuracy: 0.9755
Epoch 14/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0154 -
accuracy: 0.9957 - val_loss: 0.0925 - val_accuracy: 0.9758
Epoch 15/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0120 -
accuracy: 0.9969 - val_loss: 0.0926 - val_accuracy: 0.9742
Epoch 16/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0128 -
accuracy: 0.9959 - val_loss: 0.0929 - val_accuracy: 0.9749
Epoch 17/50
938/938 [==============================] - 10s 10ms/step - loss: 0.0099 -
accuracy: 0.9971 - val_loss: 0.0984 - val_accuracy: 0.9750
Epoch 18/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0079 -
accuracy: 0.9980 - val_loss: 0.0917 - val_accuracy: 0.9776
Epoch 19/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0081 -
accuracy: 0.9977 - val_loss: 0.1021 - val_accuracy: 0.9758
Epoch 20/50
938/938 [==============================] - 9s 10ms/step - loss: 0.0083 -
accuracy: 0.9974 - val_loss: 0.1126 - val_accuracy: 0.9725
Restoring model weights from the end of the best epoch.
Epoch 00020: early stopping
313/313 [==============================] - 2s 7ms/step - loss: 0.0802 -
accuracy: 0.9748
> 97.480
```



## 7 Convolutional Neural Networks (CNNs)

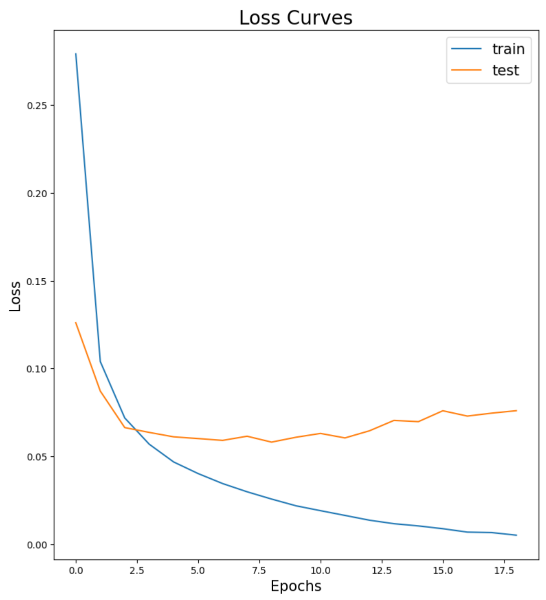
```
[ ]: # define a simple CNN architecture
def simpleCNN(input_size, filters=32,opt=Adam(learning_rate=0.01)):
  inputs = Input(shape=input_shape)
  conv1 = Conv2D(filters, kernel_size=(3,3), activation='relu', padding='same',␣
 ↪input_shape=input_size)(inputs)
  flatten1 = Flatten()(conv1)
  outputs = Dense(10, activation='softmax')(flatten1)
  model = Model(inputs, outputs)
  model.compile(optimizer=opt, loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
  return model
```

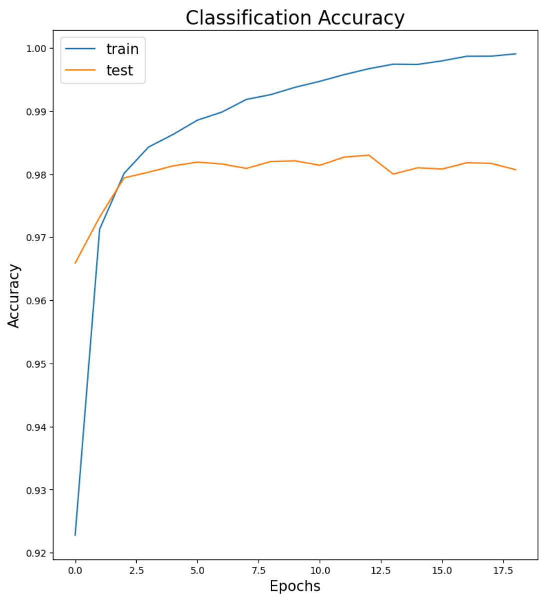
```
[ ]: # early stopping
es = EarlyStopping(monitor='val_loss',
                   mode='min',
                   verbose=1,
                   patience=10,
                   restore_best_weights=True)

# define model
opt = Adam(learning_rate=0.0005)
modelCNN = simpleCNN(input_shape, filters=32, opt=opt)
# Print model summary
modelCNN.summary()
# fit model
history = modelCNN.fit(trainX, trainY, epochs=50, batch_size=64,␣
 ↪validation_data=(testX, testY), verbose=1, callbacks=[es])
# evaluate model
_, acc = modelCNN.evaluate(testX, testY, verbose=1)
print('> %.3f' % (acc * 100.0))

# loss curves & accuracy
summarize_diagnostics(history)
```

```
Model: "model_5"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_6 (InputLayer)         [(None, 28, 28, 1)]       0

_________________________________________________________________
conv2d (Conv2D)              (None, 28, 28, 32)        320

_________________________________________________________________
```

```
flatten_5 (Flatten)          (None, 25088)             0

_________________________________________________________________
dense_13 (Dense)             (None, 10)                250890
=================================================================
Total params: 251,210
Trainable params: 251,210
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50

2022-11-15 15:09:12.682032: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - ETA: 0s - loss: 0.2792 - accuracy:
0.9228

2022-11-15 15:09:24.002652: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

938/938 [==============================] - 12s 12ms/step - loss: 0.2792 -
accuracy: 0.9228 - val_loss: 0.1260 - val_accuracy: 0.9659
Epoch 2/50
938/938 [==============================] - 11s 12ms/step - loss: 0.1040 -
accuracy: 0.9713 - val_loss: 0.0871 - val_accuracy: 0.9732
Epoch 3/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0718 -
accuracy: 0.9801 - val_loss: 0.0663 - val_accuracy: 0.9794
Epoch 4/50
938/938 [==============================] - 11s 11ms/step - loss: 0.0569 -
accuracy: 0.9843 - val_loss: 0.0636 - val_accuracy: 0.9803
Epoch 5/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0467 -
accuracy: 0.9863 - val_loss: 0.0611 - val_accuracy: 0.9813
Epoch 6/50
938/938 [==============================] - 11s 11ms/step - loss: 0.0401 -
accuracy: 0.9886 - val_loss: 0.0601 - val_accuracy: 0.9819
Epoch 7/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0345 -
accuracy: 0.9898 - val_loss: 0.0590 - val_accuracy: 0.9816
Epoch 8/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0298 -
accuracy: 0.9918 - val_loss: 0.0614 - val_accuracy: 0.9809
Epoch 9/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0256 -
accuracy: 0.9926 - val_loss: 0.0581 - val_accuracy: 0.9820
Epoch 10/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0218 -
accuracy: 0.9938 - val_loss: 0.0608 - val_accuracy: 0.9821
```

```
Epoch 11/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0190 -
accuracy: 0.9947 - val_loss: 0.0630 - val_accuracy: 0.9814
Epoch 12/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0163 -
accuracy: 0.9958 - val_loss: 0.0604 - val_accuracy: 0.9827
Epoch 13/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0136 -
accuracy: 0.9967 - val_loss: 0.0645 - val_accuracy: 0.9830
Epoch 14/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0116 -
accuracy: 0.9974 - val_loss: 0.0704 - val_accuracy: 0.9800
Epoch 15/50
938/938 [==============================] - 12s 13ms/step - loss: 0.0103 -
accuracy: 0.9974 - val_loss: 0.0697 - val_accuracy: 0.9810
Epoch 16/50
938/938 [==============================] - 12s 12ms/step - loss: 0.0087 -
accuracy: 0.9980 - val_loss: 0.0759 - val_accuracy: 0.9808
Epoch 17/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0068 -
accuracy: 0.9987 - val_loss: 0.0729 - val_accuracy: 0.9818
Epoch 18/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0065 -
accuracy: 0.9987 - val_loss: 0.0746 - val_accuracy: 0.9817
Epoch 19/50
938/938 [==============================] - 11s 12ms/step - loss: 0.0051 -
accuracy: 0.9991 - val_loss: 0.0760 - val_accuracy: 0.9807
Restoring model weights from the end of the best epoch.
Epoch 00019: early stopping
313/313 [==============================] - 2s 7ms/step - loss: 0.0581 -
accuracy: 0.9820
> 98.200
```





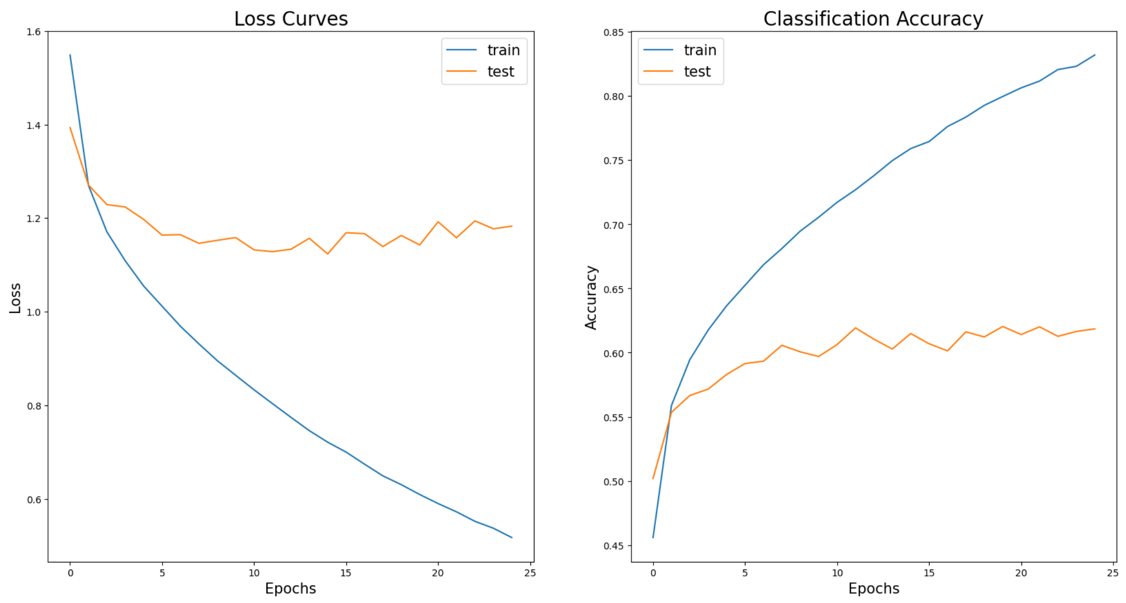
[ ]:
```
# Let us go back to CIFAR 10 and see if the CNN works better!

# load dataset
trainX, trainY, testX, testY = load_dataset()
# prepare pixel data
trainX, testX = prep_pixels(trainX, testX)
# Get shape of input
input_shape = trainX[0].shape

# early stopping
es = EarlyStopping(monitor='val_loss',
                   mode='min',
                   verbose=1,
                   patience=10,
                   restore_best_weights=True)

# define model
opt = Adam(learning_rate=0.0005)
modelCNN = simpleCNN(input_shape, filters=32, opt=opt)
# Print model summary
modelCNN.summary()
# fit model
history = modelCNN.fit(trainX, trainY, epochs=50, batch_size=64,␣
 ↪validation_data=(testX, testY), verbose=1, callbacks=[es])
# evaluate model
_, acc = modelCNN.evaluate(testX, testY, verbose=1)
print('> %.3f' % (acc * 100.0))
```

```
# loss curves & accuracy
summarize_diagnostics(history)
```

```
Model: "model_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_7 (InputLayer)         [(None, 32, 32, 3)]       0

_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        896

_________________________________________________________________
flatten_6 (Flatten)          (None, 32768)             0

_________________________________________________________________
dense_14 (Dense)             (None, 10)                327690
=================================================================
Total params: 328,586
Trainable params: 328,586
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50

2022-11-15 15:12:51.650173: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - ETA: 0s - loss: 1.5487 - accuracy:
0.4558

2022-11-15 15:13:04.646546: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 14s 17ms/step - loss: 1.5487 -
accuracy: 0.4558 - val_loss: 1.3934 - val_accuracy: 0.5019
Epoch 2/50
782/782 [==============================] - 11s 14ms/step - loss: 1.2700 -
accuracy: 0.5588 - val_loss: 1.2707 - val_accuracy: 0.5535
Epoch 3/50
782/782 [==============================] - 10s 13ms/step - loss: 1.1707 -
accuracy: 0.5946 - val_loss: 1.2290 - val_accuracy: 0.5666
Epoch 4/50
782/782 [==============================] - 10s 13ms/step - loss: 1.1087 -
accuracy: 0.6177 - val_loss: 1.2237 - val_accuracy: 0.5716
Epoch 5/50
782/782 [==============================] - 10s 13ms/step - loss: 1.0548 -
accuracy: 0.6366 - val_loss: 1.1973 - val_accuracy: 0.5830
Epoch 6/50
782/782 [==============================] - 10s 13ms/step - loss: 1.0118 -
```

```
accuracy: 0.6526 - val_loss: 1.1636 - val_accuracy: 0.5915
Epoch 7/50
782/782 [==============================] - 10s 13ms/step - loss: 0.9687 -
accuracy: 0.6685 - val_loss: 1.1646 - val_accuracy: 0.5933
Epoch 8/50
782/782 [==============================] - 10s 13ms/step - loss: 0.9313 -
accuracy: 0.6812 - val_loss: 1.1461 - val_accuracy: 0.6057
Epoch 9/50
782/782 [==============================] - 10s 13ms/step - loss: 0.8953 -
accuracy: 0.6948 - val_loss: 1.1526 - val_accuracy: 0.6006
Epoch 10/50
782/782 [==============================] - 11s 14ms/step - loss: 0.8640 -
accuracy: 0.7056 - val_loss: 1.1584 - val_accuracy: 0.5970
Epoch 11/50
782/782 [==============================] - 10s 13ms/step - loss: 0.8330 -
accuracy: 0.7172 - val_loss: 1.1322 - val_accuracy: 0.6063
Epoch 12/50
782/782 [==============================] - 10s 13ms/step - loss: 0.8034 -
accuracy: 0.7270 - val_loss: 1.1283 - val_accuracy: 0.6193
Epoch 13/50
782/782 [==============================] - 10s 13ms/step - loss: 0.7743 -
accuracy: 0.7380 - val_loss: 1.1335 - val_accuracy: 0.6105
Epoch 14/50
782/782 [==============================] - 10s 13ms/step - loss: 0.7457 -
accuracy: 0.7497 - val_loss: 1.1569 - val_accuracy: 0.6028
Epoch 15/50
782/782 [==============================] - 10s 13ms/step - loss: 0.7211 -
accuracy: 0.7591 - val_loss: 1.1234 - val_accuracy: 0.6149
Epoch 16/50
782/782 [==============================] - 10s 13ms/step - loss: 0.6999 -
accuracy: 0.7646 - val_loss: 1.1688 - val_accuracy: 0.6069
Epoch 17/50
782/782 [==============================] - 10s 13ms/step - loss: 0.6741 -
accuracy: 0.7763 - val_loss: 1.1668 - val_accuracy: 0.6014
Epoch 18/50
782/782 [==============================] - 10s 13ms/step - loss: 0.6491 -
accuracy: 0.7837 - val_loss: 1.1393 - val_accuracy: 0.6162
Epoch 19/50
782/782 [==============================] - 10s 13ms/step - loss: 0.6303 -
accuracy: 0.7928 - val_loss: 1.1630 - val_accuracy: 0.6122
Epoch 20/50
782/782 [==============================] - 10s 13ms/step - loss: 0.6092 -
accuracy: 0.7997 - val_loss: 1.1427 - val_accuracy: 0.6204
Epoch 21/50
782/782 [==============================] - 10s 13ms/step - loss: 0.5899 -
accuracy: 0.8064 - val_loss: 1.1921 - val_accuracy: 0.6141
Epoch 22/50
782/782 [==============================] - 10s 13ms/step - loss: 0.5722 -
```

```
accuracy: 0.8117 - val_loss: 1.1580 - val_accuracy: 0.6201
Epoch 23/50
782/782 [==============================] - 10s 13ms/step - loss: 0.5519 -
accuracy: 0.8207 - val_loss: 1.1942 - val_accuracy: 0.6127
Epoch 24/50
782/782 [==============================] - 10s 13ms/step - loss: 0.5372 -
accuracy: 0.8232 - val_loss: 1.1772 - val_accuracy: 0.6165
Epoch 25/50
782/782 [==============================] - 10s 13ms/step - loss: 0.5173 -
accuracy: 0.8320 - val_loss: 1.1829 - val_accuracy: 0.6185
Restoring model weights from the end of the best epoch.
Epoch 00025: early stopping
313/313 [==============================] - 2s 7ms/step - loss: 1.1234 -
accuracy: 0.6149
> 61.490
```



## 8 Is a deeper CNN better?

[ ]:
```
# Define a two-layer CNN model
def twoLayerCNN(input_size, filters1=32, filters2=64, opt=Adam(learning_rate=0.
↪01)):
  inputs = Input(shape=input_shape)
  conv1 = Conv2D(filters1, kernel_size=(3,3), activation='relu',␣
↪padding='same', input_shape=input_size)(inputs)
  conv2 = Conv2D(filters2, kernel_size=(3,3), activation='relu',␣
↪padding='same')(conv1)
```

```
  flatten1 = Flatten()(conv2)
  outputs = Dense(10, activation='softmax')(flatten1)
  model = Model(inputs, outputs)
  model.compile(optimizer=opt, loss='categorical_crossentropy',␣
↪metrics=['accuracy'])
  return model
```

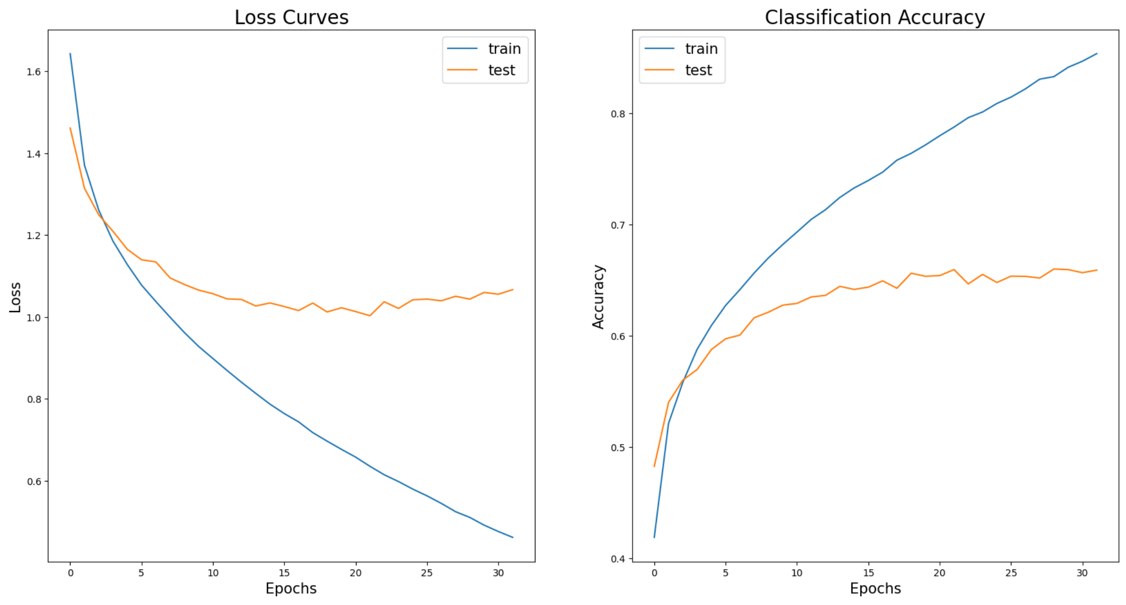
[ ]:
```
# define model
opt = Adam(learning_rate=0.0001)
modelCNN = twoLayerCNN(input_shape, filters1=32, filters2=64, opt=opt)
# Print model summary
modelCNN.summary()
# fit model
history = modelCNN.fit(trainX, trainY, epochs=50, batch_size=64,␣
↪validation_data=(testX, testY), verbose=1, callbacks=[es])
# evaluate model
_, acc = modelCNN.evaluate(testX, testY, verbose=1)
print('> %.3f' % (acc * 100.0))

# loss curves & accuracy
summarize_diagnostics(history)
```

```
Model: "model_7"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_8 (InputLayer)         [(None, 32, 32, 3)]       0

_________________________________________________________________
conv2d_2 (Conv2D)            (None, 32, 32, 32)        896

_________________________________________________________________
conv2d_3 (Conv2D)            (None, 32, 32, 64)        18496

_________________________________________________________________
flatten_7 (Flatten)          (None, 65536)             0

_________________________________________________________________
dense_15 (Dense)             (None, 10)                655370
=================================================================
Total params: 674,762
Trainable params: 674,762
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50

2022-11-15 15:17:19.023226: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - ETA: 0s - loss: 1.6426 - accuracy:
0.4185
```

```
2022-11-15 15:17:40.188520: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 23s 28ms/step - loss: 1.6426 -
accuracy: 0.4185 - val_loss: 1.4610 - val_accuracy: 0.4824
Epoch 2/50
782/782 [==============================] - 24s 31ms/step - loss: 1.3705 -
accuracy: 0.5210 - val_loss: 1.3142 - val_accuracy: 0.5399
Epoch 3/50
782/782 [==============================] - 22s 28ms/step - loss: 1.2607 -
accuracy: 0.5579 - val_loss: 1.2492 - val_accuracy: 0.5599
Epoch 4/50
782/782 [==============================] - 22s 28ms/step - loss: 1.1849 -
accuracy: 0.5876 - val_loss: 1.2089 - val_accuracy: 0.5694
Epoch 5/50
782/782 [==============================] - 23s 29ms/step - loss: 1.1280 -
accuracy: 0.6090 - val_loss: 1.1651 - val_accuracy: 0.5874
Epoch 6/50
782/782 [==============================] - 22s 28ms/step - loss: 1.0777 -
accuracy: 0.6271 - val_loss: 1.1396 - val_accuracy: 0.5971
Epoch 7/50
782/782 [==============================] - 23s 29ms/step - loss: 1.0377 -
accuracy: 0.6413 - val_loss: 1.1344 - val_accuracy: 0.6004
Epoch 8/50
782/782 [==============================] - 22s 29ms/step - loss: 0.9993 -
accuracy: 0.6563 - val_loss: 1.0952 - val_accuracy: 0.6160
Epoch 9/50
782/782 [==============================] - 23s 29ms/step - loss: 0.9620 -
accuracy: 0.6698 - val_loss: 1.0792 - val_accuracy: 0.6210
Epoch 10/50
782/782 [==============================] - 23s 29ms/step - loss: 0.9280 -
accuracy: 0.6818 - val_loss: 1.0655 - val_accuracy: 0.6273
Epoch 11/50
782/782 [==============================] - 22s 28ms/step - loss: 0.8985 -
accuracy: 0.6931 - val_loss: 1.0567 - val_accuracy: 0.6289
Epoch 12/50
782/782 [==============================] - 22s 28ms/step - loss: 0.8691 -
accuracy: 0.7045 - val_loss: 1.0440 - val_accuracy: 0.6347
Epoch 13/50
782/782 [==============================] - 22s 29ms/step - loss: 0.8407 -
accuracy: 0.7132 - val_loss: 1.0427 - val_accuracy: 0.6361
Epoch 14/50
782/782 [==============================] - 22s 28ms/step - loss: 0.8135 -
accuracy: 0.7241 - val_loss: 1.0268 - val_accuracy: 0.6442
Epoch 15/50
782/782 [==============================] - 23s 29ms/step - loss: 0.7871 -
accuracy: 0.7327 - val_loss: 1.0342 - val_accuracy: 0.6415
```

```
Epoch 16/50
782/782 [==============================] - 23s 29ms/step - loss: 0.7642 -
accuracy: 0.7394 - val_loss: 1.0254 - val_accuracy: 0.6435
Epoch 17/50
782/782 [==============================] - 23s 29ms/step - loss: 0.7439 -
accuracy: 0.7469 - val_loss: 1.0157 - val_accuracy: 0.6492
Epoch 18/50
782/782 [==============================] - 23s 29ms/step - loss: 0.7176 -
accuracy: 0.7577 - val_loss: 1.0339 - val_accuracy: 0.6425
Epoch 19/50
782/782 [==============================] - 31s 40ms/step - loss: 0.6969 -
accuracy: 0.7638 - val_loss: 1.0122 - val_accuracy: 0.6560
Epoch 20/50
782/782 [==============================] - 28s 36ms/step - loss: 0.6770 -
accuracy: 0.7714 - val_loss: 1.0223 - val_accuracy: 0.6533
Epoch 21/50
782/782 [==============================] - 22s 29ms/step - loss: 0.6578 -
accuracy: 0.7796 - val_loss: 1.0132 - val_accuracy: 0.6539
Epoch 22/50
782/782 [==============================] - 23s 29ms/step - loss: 0.6355 -
accuracy: 0.7874 - val_loss: 1.0029 - val_accuracy: 0.6593
Epoch 23/50
782/782 [==============================] - 23s 29ms/step - loss: 0.6148 -
accuracy: 0.7959 - val_loss: 1.0371 - val_accuracy: 0.6464
Epoch 24/50
782/782 [==============================] - 22s 29ms/step - loss: 0.5983 -
accuracy: 0.8010 - val_loss: 1.0209 - val_accuracy: 0.6550
Epoch 25/50
782/782 [==============================] - 22s 28ms/step - loss: 0.5799 -
accuracy: 0.8086 - val_loss: 1.0419 - val_accuracy: 0.6477
Epoch 26/50
782/782 [==============================] - 22s 29ms/step - loss: 0.5634 -
accuracy: 0.8144 - val_loss: 1.0436 - val_accuracy: 0.6534
Epoch 27/50
782/782 [==============================] - 23s 29ms/step - loss: 0.5450 -
accuracy: 0.8218 - val_loss: 1.0394 - val_accuracy: 0.6532
Epoch 28/50
782/782 [==============================] - 22s 28ms/step - loss: 0.5247 -
accuracy: 0.8305 - val_loss: 1.0505 - val_accuracy: 0.6517
Epoch 29/50
782/782 [==============================] - 21s 27ms/step - loss: 0.5107 -
accuracy: 0.8328 - val_loss: 1.0435 - val_accuracy: 0.6598
Epoch 30/50
782/782 [==============================] - 20s 26ms/step - loss: 0.4919 -
accuracy: 0.8412 - val_loss: 1.0598 - val_accuracy: 0.6593
Epoch 31/50
782/782 [==============================] - 19s 24ms/step - loss: 0.4763 -
accuracy: 0.8467 - val_loss: 1.0555 - val_accuracy: 0.6565
```

```
Epoch 32/50
782/782 [==============================] - 19s 25ms/step - loss: 0.4621 -
accuracy: 0.8535 - val_loss: 1.0666 - val_accuracy: 0.6588
Restoring model weights from the end of the best epoch.
Epoch 00032: early stopping
313/313 [==============================] - 3s 9ms/step - loss: 1.0029 -
accuracy: 0.6593
> 65.930
```



## 9 Reducing overfitting!

### 9.1 Let's use dropout (The network drops out the contribution of a certain percentage of neurons when training)

[ ]:
```
# Redefine to include the option for dropout
def twoLayerCNN(input_size, filters1=32, filters2=64, opt=Adam(learning_rate=0.
↪01), dropout=True):
  inputs = Input(shape=input_shape)
  conv1 = Conv2D(filters1, kernel_size=(3,3), activation='relu',␣
↪padding='same', input_shape=input_size)(inputs)
  if dropout:
    conv1 = Dropout(0.5)(conv1)
  conv2 = Conv2D(filters2, kernel_size=(3,3), activation='relu',␣
↪padding='same')(conv1)
  if dropout:
    conv2 = Dropout(0.5)(conv2)
  flatten1 = Flatten()(conv2)
```

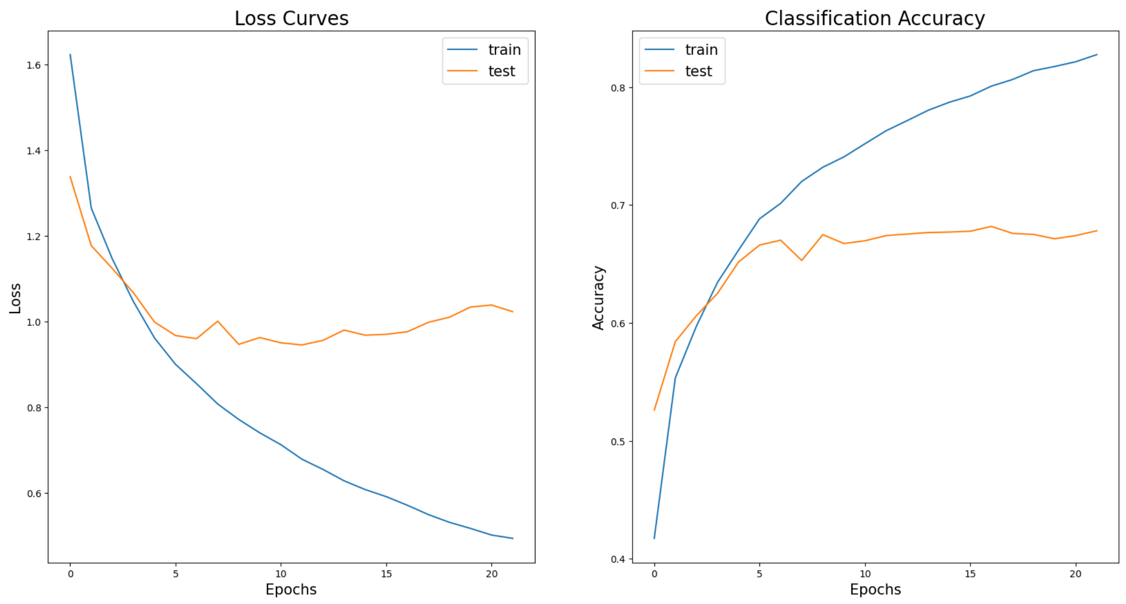
```
    outputs = Dense(10, activation='softmax')(flatten1)
    model = Model(inputs, outputs)
    model.compile(optimizer=opt, loss='categorical_crossentropy',␣
  ↪metrics=['accuracy'])
    return model
```

```
[ ]: # define model
     opt = Adam(learning_rate=0.0005)
     modelCNN = twoLayerCNN(input_shape, filters1=32, filters2=64, opt=opt,␣
      ↪dropout=True)
     # Print model summary
     modelCNN.summary()
     # fit model
     history = modelCNN.fit(trainX, trainY, epochs=50, batch_size=64,␣
      ↪validation_data=(testX, testY), verbose=1, callbacks=[es])
     # evaluate model
     _, acc = modelCNN.evaluate(testX, testY, verbose=1)
     print('> %.3f' % (acc * 100.0))

     # loss curves & accuracy
     summarize_diagnostics(history)
```

```
Model: "model_8"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_9 (InputLayer)         [(None, 32, 32, 3)]       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
dropout (Dropout)            (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 32, 32, 64)        18496
_________________________________________________________________
dropout_1 (Dropout)          (None, 32, 32, 64)        0
_________________________________________________________________
flatten_8 (Flatten)          (None, 65536)             0
_________________________________________________________________
dense_16 (Dense)             (None, 10)                655370
=================================================================
Total params: 674,762
Trainable params: 674,762
Non-trainable params: 0
_________________________________________________________________
Epoch 1/50

2022-11-15 15:29:31.346429: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
```

```
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - ETA: 0s - loss: 1.6230 - accuracy:
0.4173

2022-11-15 15:31:58.519166: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 149s 190ms/step - loss: 1.6230 -
accuracy: 0.4173 - val_loss: 1.3378 - val_accuracy: 0.5260
Epoch 2/50
782/782 [==============================] - 31s 40ms/step - loss: 1.2649 -
accuracy: 0.5534 - val_loss: 1.1773 - val_accuracy: 0.5841
Epoch 3/50
782/782 [==============================] - 31s 39ms/step - loss: 1.1462 -
accuracy: 0.5972 - val_loss: 1.1238 - val_accuracy: 0.6061
Epoch 4/50
782/782 [==============================] - 31s 40ms/step - loss: 1.0467 -
accuracy: 0.6343 - val_loss: 1.0674 - val_accuracy: 0.6250
Epoch 5/50
782/782 [==============================] - 30s 38ms/step - loss: 0.9619 -
accuracy: 0.6617 - val_loss: 0.9990 - val_accuracy: 0.6516
Epoch 6/50
782/782 [==============================] - 30s 39ms/step - loss: 0.9001 -
accuracy: 0.6883 - val_loss: 0.9674 - val_accuracy: 0.6660
Epoch 7/50
782/782 [==============================] - 32s 40ms/step - loss: 0.8549 -
accuracy: 0.7014 - val_loss: 0.9601 - val_accuracy: 0.6701
Epoch 8/50
782/782 [==============================] - 30s 38ms/step - loss: 0.8077 -
accuracy: 0.7200 - val_loss: 1.0009 - val_accuracy: 0.6529
Epoch 9/50
782/782 [==============================] - 30s 38ms/step - loss: 0.7715 -
accuracy: 0.7320 - val_loss: 0.9468 - val_accuracy: 0.6748
Epoch 10/50
782/782 [==============================] - 30s 39ms/step - loss: 0.7403 -
accuracy: 0.7407 - val_loss: 0.9627 - val_accuracy: 0.6673
Epoch 11/50
782/782 [==============================] - 33s 42ms/step - loss: 0.7126 -
accuracy: 0.7519 - val_loss: 0.9504 - val_accuracy: 0.6696
Epoch 12/50
782/782 [==============================] - 28s 35ms/step - loss: 0.6788 -
accuracy: 0.7629 - val_loss: 0.9454 - val_accuracy: 0.6740
Epoch 13/50
782/782 [==============================] - 29s 38ms/step - loss: 0.6549 -
accuracy: 0.7715 - val_loss: 0.9562 - val_accuracy: 0.6753
Epoch 14/50
782/782 [==============================] - 31s 40ms/step - loss: 0.6283 -
```

```
accuracy: 0.7803 - val_loss: 0.9801 - val_accuracy: 0.6766
Epoch 15/50
782/782 [==============================] - 31s 40ms/step - loss: 0.6079 -
accuracy: 0.7871 - val_loss: 0.9682 - val_accuracy: 0.6770
Epoch 16/50
782/782 [==============================] - 31s 40ms/step - loss: 0.5914 -
accuracy: 0.7924 - val_loss: 0.9702 - val_accuracy: 0.6777
Epoch 17/50
782/782 [==============================] - 31s 40ms/step - loss: 0.5713 -
accuracy: 0.8008 - val_loss: 0.9762 - val_accuracy: 0.6818
Epoch 18/50
782/782 [==============================] - 31s 40ms/step - loss: 0.5495 -
accuracy: 0.8063 - val_loss: 0.9982 - val_accuracy: 0.6759
Epoch 19/50
782/782 [==============================] - 31s 40ms/step - loss: 0.5314 -
accuracy: 0.8138 - val_loss: 1.0102 - val_accuracy: 0.6750
Epoch 20/50
782/782 [==============================] - 30s 39ms/step - loss: 0.5172 -
accuracy: 0.8174 - val_loss: 1.0339 - val_accuracy: 0.6713
Epoch 21/50
782/782 [==============================] - 33s 42ms/step - loss: 0.5017 -
accuracy: 0.8214 - val_loss: 1.0385 - val_accuracy: 0.6740
Epoch 22/50
782/782 [==============================] - 31s 40ms/step - loss: 0.4940 -
accuracy: 0.8275 - val_loss: 1.0231 - val_accuracy: 0.6781
Restoring model weights from the end of the best epoch.
Epoch 00022: early stopping
313/313 [==============================] - 3s 10ms/step - loss: 0.9454 -
accuracy: 0.6740
> 67.400
```



## 9.2 Another strategy is to use max pooling layers after convolution

The rationale is to retain only the filter outputs which are the strongest in a neighborhood. Additionally, it reduces the number of parameters a lot. Another desirable effect of max pooling is to introduce some translational invariance.

[ ]:
```
# Redefine to include the option for dropout
def twoLayerCNN(input_size, filters1=32, filters2=64, opt=Adam(learning_rate=0.
↪01), dropout=True, maxpooling=True):
  inputs = Input(shape=input_shape)

  conv1 = Conv2D(filters1, kernel_size=(3,3), activation='relu',␣
↪padding='same', input_shape=input_size)(inputs)
  if maxpooling:
    conv1 = MaxPooling2D((2, 2))(conv1)
  if dropout:
    conv1 = Dropout(0.5)(conv1)

  conv2 = Conv2D(filters2, kernel_size=(3,3), activation='relu',␣
↪padding='same')(conv1)
  if maxpooling:
    conv2 = MaxPooling2D((2, 2))(conv2)
  if dropout:
    conv2 = Dropout(0.5)(conv2)

  flatten1 = Flatten()(conv2)

  outputs = Dense(10, activation='softmax')(flatten1)

  model = Model(inputs, outputs)
  model.compile(optimizer=opt, loss='categorical_crossentropy',␣
↪metrics=['accuracy'])
  return model
```

[ ]:
```
# define model
opt = Adam(learning_rate=0.0005)
modelCNN = twoLayerCNN(input_shape,
                       filters1=32,
                       filters2=64,
                       opt=opt,
                       dropout=True,
                       maxpooling=True)
# Print model summary
modelCNN.summary()
# fit model
```

```
history = modelCNN.fit(trainX, trainY,
                       epochs=50,
                       batch_size=64,
                       validation_data=(testX, testY),
                       verbose=1,
                       callbacks=[es])
# evaluate model
_, acc = modelCNN.evaluate(testX, testY, verbose=1)
print('> %.3f' % (acc * 100.0))

# loss curves & accuracy
summarize_diagnostics(history)
```

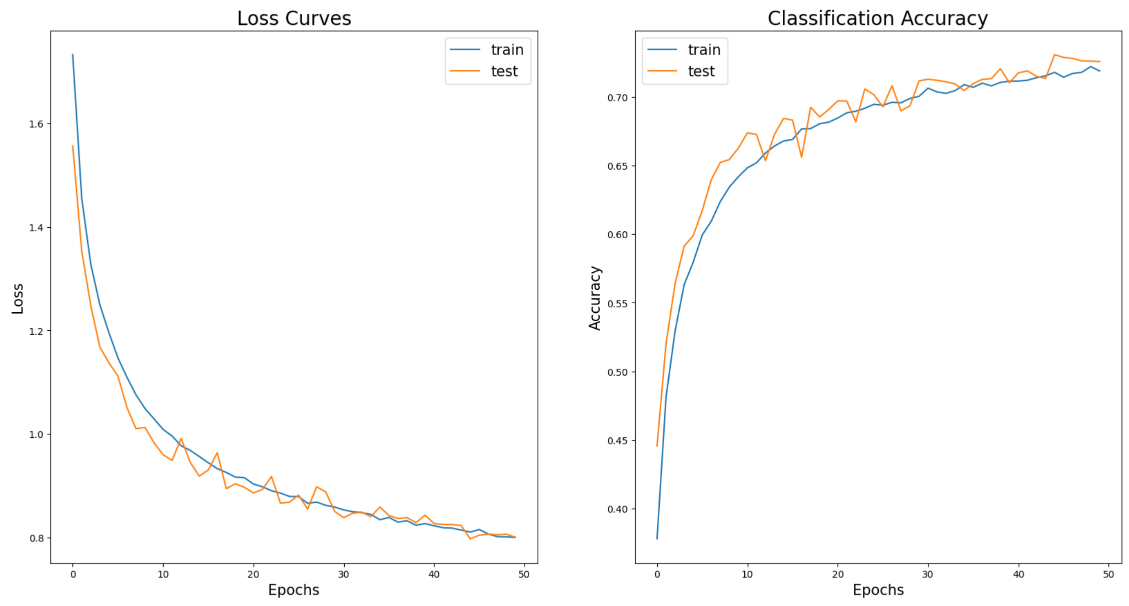
```
Model: "model_9"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
input_10 (InputLayer)       [(None, 32, 32, 3)]       0

_________________________________________________________________
conv2d_6 (Conv2D)           (None, 32, 32, 32)        896

_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)       0

_________________________________________________________________
dropout_2 (Dropout)         (None, 16, 16, 32)        0

_________________________________________________________________
conv2d_7 (Conv2D)           (None, 16, 16, 64)        18496

_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)         0

_________________________________________________________________
dropout_3 (Dropout)         (None, 8, 8, 64)          0

_________________________________________________________________
flatten_9 (Flatten)         (None, 4096)              0

_________________________________________________________________
dense_17 (Dense)            (None, 10)                40970
=================================================================
Total params: 60,362
Trainable params: 60,362
Non-trainable params: 0

_________________________________________________________________
Epoch 1/50

2022-11-15 15:42:53.511144: I
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - ETA: 0s - loss: 1.7323 - accuracy:
0.3778

2022-11-15 15:43:09.701862: I
```

```
tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:112]
Plugin optimizer for device_type GPU is enabled.

782/782 [==============================] - 18s 21ms/step - loss: 1.7323 -
accuracy: 0.3778 - val_loss: 1.5561 - val_accuracy: 0.4456
Epoch 2/50
782/782 [==============================] - 16s 21ms/step - loss: 1.4546 -
accuracy: 0.4816 - val_loss: 1.3527 - val_accuracy: 0.5205
Epoch 3/50
782/782 [==============================] - 16s 21ms/step - loss: 1.3276 -
accuracy: 0.5295 - val_loss: 1.2477 - val_accuracy: 0.5642
Epoch 4/50
782/782 [==============================] - 16s 21ms/step - loss: 1.2497 -
accuracy: 0.5632 - val_loss: 1.1679 - val_accuracy: 0.5913
Epoch 5/50
782/782 [==============================] - 16s 21ms/step - loss: 1.1958 -
accuracy: 0.5796 - val_loss: 1.1377 - val_accuracy: 0.5990
Epoch 6/50
782/782 [==============================] - 16s 21ms/step - loss: 1.1472 -
accuracy: 0.5994 - val_loss: 1.1120 - val_accuracy: 0.6173
Epoch 7/50
782/782 [==============================] - 18s 23ms/step - loss: 1.1097 -
accuracy: 0.6095 - val_loss: 1.0513 - val_accuracy: 0.6397
Epoch 8/50
782/782 [==============================] - 16s 21ms/step - loss: 1.0757 -
accuracy: 0.6239 - val_loss: 1.0104 - val_accuracy: 0.6524
Epoch 9/50
782/782 [==============================] - 16s 21ms/step - loss: 1.0492 -
accuracy: 0.6344 - val_loss: 1.0126 - val_accuracy: 0.6545
Epoch 10/50
782/782 [==============================] - 16s 21ms/step - loss: 1.0292 -
accuracy: 0.6419 - val_loss: 0.9831 - val_accuracy: 0.6628
Epoch 11/50
782/782 [==============================] - 16s 21ms/step - loss: 1.0089 -
accuracy: 0.6485 - val_loss: 0.9600 - val_accuracy: 0.6740
Epoch 12/50
782/782 [==============================] - 17s 21ms/step - loss: 0.9960 -
accuracy: 0.6521 - val_loss: 0.9490 - val_accuracy: 0.6728
Epoch 13/50
782/782 [==============================] - 16s 20ms/step - loss: 0.9773 -
accuracy: 0.6592 - val_loss: 0.9915 - val_accuracy: 0.6536
Epoch 14/50
782/782 [==============================] - 16s 20ms/step - loss: 0.9683 -
accuracy: 0.6644 - val_loss: 0.9454 - val_accuracy: 0.6728
Epoch 15/50
782/782 [==============================] - 16s 21ms/step - loss: 0.9566 -
accuracy: 0.6681 - val_loss: 0.9188 - val_accuracy: 0.6845
Epoch 16/50
```

```
782/782 [==============================] - 20s 25ms/step - loss: 0.9446 -
accuracy: 0.6692 - val_loss: 0.9305 - val_accuracy: 0.6833
Epoch 17/50
782/782 [==============================] - 18s 23ms/step - loss: 0.9331 -
accuracy: 0.6768 - val_loss: 0.9639 - val_accuracy: 0.6561
Epoch 18/50
782/782 [==============================] - 18s 23ms/step - loss: 0.9258 -
accuracy: 0.6771 - val_loss: 0.8945 - val_accuracy: 0.6926
Epoch 19/50
782/782 [==============================] - 18s 23ms/step - loss: 0.9168 -
accuracy: 0.6806 - val_loss: 0.9039 - val_accuracy: 0.6856
Epoch 20/50
782/782 [==============================] - 18s 23ms/step - loss: 0.9157 -
accuracy: 0.6818 - val_loss: 0.8970 - val_accuracy: 0.6910
Epoch 21/50
782/782 [==============================] - 18s 23ms/step - loss: 0.9034 -
accuracy: 0.6848 - val_loss: 0.8863 - val_accuracy: 0.6973
Epoch 22/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8981 -
accuracy: 0.6886 - val_loss: 0.8929 - val_accuracy: 0.6972
Epoch 23/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8905 -
accuracy: 0.6898 - val_loss: 0.9181 - val_accuracy: 0.6820
Epoch 24/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8858 -
accuracy: 0.6919 - val_loss: 0.8663 - val_accuracy: 0.7060
Epoch 25/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8794 -
accuracy: 0.6948 - val_loss: 0.8684 - val_accuracy: 0.7018
Epoch 26/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8792 -
accuracy: 0.6941 - val_loss: 0.8821 - val_accuracy: 0.6929
Epoch 27/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8661 -
accuracy: 0.6963 - val_loss: 0.8550 - val_accuracy: 0.7083
Epoch 28/50
782/782 [==============================] - 17s 22ms/step - loss: 0.8686 -
accuracy: 0.6959 - val_loss: 0.8981 - val_accuracy: 0.6900
Epoch 29/50
782/782 [==============================] - 16s 21ms/step - loss: 0.8623 -
accuracy: 0.6991 - val_loss: 0.8884 - val_accuracy: 0.6938
Epoch 30/50
782/782 [==============================] - 16s 20ms/step - loss: 0.8589 -
accuracy: 0.7007 - val_loss: 0.8505 - val_accuracy: 0.7119
Epoch 31/50
782/782 [==============================] - 16s 21ms/step - loss: 0.8535 -
accuracy: 0.7066 - val_loss: 0.8385 - val_accuracy: 0.7131
Epoch 32/50
```

```
782/782 [==============================] - 18s 23ms/step - loss: 0.8499 -
accuracy: 0.7038 - val_loss: 0.8471 - val_accuracy: 0.7123
Epoch 33/50
782/782 [==============================] - 22s 29ms/step - loss: 0.8481 -
accuracy: 0.7028 - val_loss: 0.8491 - val_accuracy: 0.7112
Epoch 34/50
782/782 [==============================] - 19s 25ms/step - loss: 0.8449 -
accuracy: 0.7048 - val_loss: 0.8404 - val_accuracy: 0.7096
Epoch 35/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8344 -
accuracy: 0.7091 - val_loss: 0.8589 - val_accuracy: 0.7048
Epoch 36/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8389 -
accuracy: 0.7072 - val_loss: 0.8426 - val_accuracy: 0.7099
Epoch 37/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8300 -
accuracy: 0.7102 - val_loss: 0.8369 - val_accuracy: 0.7130
Epoch 38/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8325 -
accuracy: 0.7082 - val_loss: 0.8385 - val_accuracy: 0.7135
Epoch 39/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8239 -
accuracy: 0.7108 - val_loss: 0.8290 - val_accuracy: 0.7208
Epoch 40/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8270 -
accuracy: 0.7116 - val_loss: 0.8434 - val_accuracy: 0.7106
Epoch 41/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8229 -
accuracy: 0.7117 - val_loss: 0.8271 - val_accuracy: 0.7177
Epoch 42/50
782/782 [==============================] - 18s 24ms/step - loss: 0.8190 -
accuracy: 0.7124 - val_loss: 0.8255 - val_accuracy: 0.7192
Epoch 43/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8183 -
accuracy: 0.7142 - val_loss: 0.8251 - val_accuracy: 0.7152
Epoch 44/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8145 -
accuracy: 0.7156 - val_loss: 0.8233 - val_accuracy: 0.7136
Epoch 45/50
782/782 [==============================] - 20s 26ms/step - loss: 0.8107 -
accuracy: 0.7181 - val_loss: 0.7977 - val_accuracy: 0.7310
Epoch 46/50
782/782 [==============================] - 18s 23ms/step - loss: 0.8156 -
accuracy: 0.7145 - val_loss: 0.8046 - val_accuracy: 0.7290
Epoch 47/50
782/782 [==============================] - 20s 25ms/step - loss: 0.8069 -
accuracy: 0.7173 - val_loss: 0.8065 - val_accuracy: 0.7283
Epoch 48/50
```

```
782/782 [==============================] - 18s 23ms/step - loss: 0.8018 -
accuracy: 0.7181 - val_loss: 0.8056 - val_accuracy: 0.7265
Epoch 49/50
782/782 [==============================] - 16s 20ms/step - loss: 0.8014 -
accuracy: 0.7223 - val_loss: 0.8066 - val_accuracy: 0.7263
Epoch 50/50
782/782 [==============================] - 16s 20ms/step - loss: 0.8004 -
accuracy: 0.7191 - val_loss: 0.8007 - val_accuracy: 0.7260
313/313 [==============================] - 2s 8ms/step - loss: 0.8007 -
accuracy: 0.7260
> 72.600
```



```
[ ]:
```