Convolutional Neural Network - Deep Learning

What is a Convolutional Neural Network (CNN)?

(Reference : [link text (https://www.youtube.com/watch?v=QzY57FaENXg)](https://www.youtube.com/watch?v=QzY57FaENXg))

CNN is a specialised kind of neural network for processing data that has a known, grid-like topology. The most common example of grid-structured data is a 2D image. An important characteristic of CNN is its operation, which is referred to as convolution. Convolution is a special kind of linear operation.

In general CNN consists of one or more convolutional layers and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of CNN is designed in such a way that it takes advantage of the 2D structure of an input image. Another benefit of CNN is that they are easier to train and have many fewer parameters than a fully connected network with the same number of hidden layers.

Concept of Convolution Operator

(Reference : [link text (https://www.youtube.com/watch?v=Etksi-F5ug8)](https://www.youtube.com/watch?v=Etksi-F5ug8))

Convolution is a type of linear operator. It operates on two functions of real-valued arguments.

Let consider an example for better understanding, suppose we are using a laser sensor to track the location of a spaceship. where the laser sensor provides a single output $x(t)$ w.r.t. position at time t. Since both these variables are real-valued, we can get different output as the time changes. In order to get a less noisy estimate of the spaceship, we take the average of several measurements (or output). Assigning the weighted average to the recent measurements since it is more relevant. Using the weighting function $w(a)$, a is the time of a measurement.

Applying this function everytime will get a new function b resulting in the most accurate estimation of the position of the spaceship

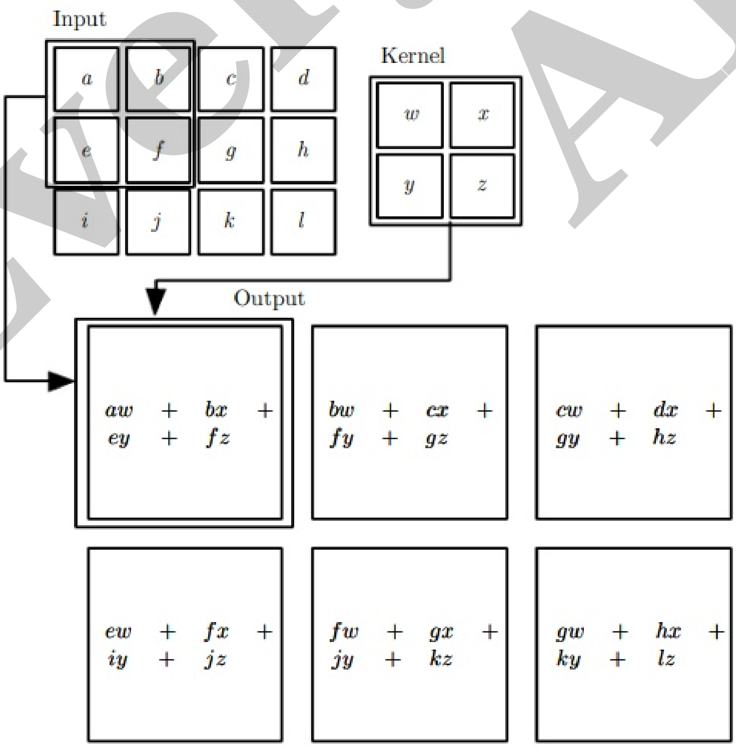
$$b(t) = \int x(a)w(t-a)da$$

This operator is called convolution. The operator is generally denoted with an asterisk:

$$s(w) = (x * w)(t)$$

In the above example , w needs to be 0 for all negative arguments. In CNN terminology, the first argument to the convolution is often referred to as the $input$ and the second argument as the $kernel$. And the output is the $featuremap$.

An example of convolution without kernel-flip is shown above. In the above fig. the output is restricted to only one position where the kernel lies entirely within the image, called $valid$ convolution. The boxes were drawn with arrows to indicate how the upper-left element of the output tensor is formed by applying the kernel to the upper-left region of the input tensor respectively.



1. Sparse Weight : In general every output unit interacts with every input unit. Convolutional networks typically have sparse weight i.e., making the kernel smaller than the input.
   For example, while processing an image, it consists of thousands to millions of pixels in the output image, but we can detect small features such as edges with kernels that occupy only tens or hundreds of pixels. This Results in fewer parameters, which reduces both the memory requirement and improves the statistical efficiency.
2. Equivariant representation : It means that if the input changes, the output changes in the same way. In the image, convolution creates a 2D map of where certain features appear in the input. If we move the object in the input, its representation will move the same amount in the output.
3. Parameter sharing : It refers to the same parameter for more than one function in a model.That is, each member of the kernel is used at every position of the input. The parameter sharing used by the convolution operation means that rather than learning a separate set of parameters for every location, it learns only one set. Although it does not affect the runtime of forward propagation, it further reduces the storage requirement of the model to k parameters.

Pooling

(Reference : [link text (https://www.youtube.com/watch?v=zg_AA3fZpE0)](https://www.youtube.com/watch?v=zg_AA3fZpE0))

A pooling function replaces the output of the net at a certain location with a summary statistic of the nearby outputs. In other words, pooling is a down-sampling operation which reduces the dimensionality of the feature map in order to introduce the translational invariance to small shifts and it reduces the number of learnable parameters.

Max Pooling : Max pooling extracts patches from the input feature maps, and provides the maximum output value in each patch, and discards all the other values. A max pooling with a filter of size 2×2 with a stride of 2 is commonly used in practice. This downsamples the in-plane dimension of feature maps by a factor of 2.Here the depth dimension of feature maps remains unchanged.

Consider an example of max pooling operation with a filter size of 2×2, no padding, and a stride of 2, which extracts 2×2 patches from the input tensors, outputs the maximum value in each patch, and discards all the other values, resulting in downsampling the in-plane dimension of an input tensor by a factor of 2.

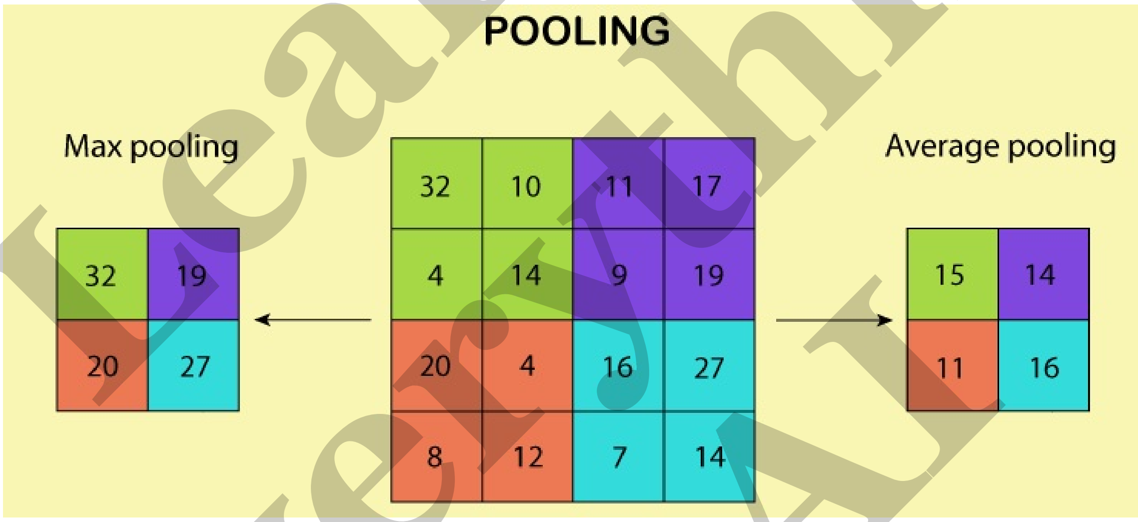


Image Classification using CNN

(Reference : [link text (https://www.youtube.com/watch?v=7HPwo4wnJeA)](https://www.youtube.com/watch?v=7HPwo4wnJeA))

To implement a CNN model we will be using the CIFAR-10 dataset. It has 60,000 colour images in 10 different classes. The image size is 32x32 and the dataset has 50,000 training images and 10,000 test images.

In [ ]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn.metrics as sm
```

In [ ]:

```
from tensorflow import keras
from keras.models import Sequential
from keras.optimizers import Adam
from keras.layers import Dense, Conv2D, Dropout, MaxPooling2D, Flatten
```

In [ ]:

```
from keras.datasets import cifar10
(x_train,y_train), (x_test,ytest) = cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 2s 0us/step
```

In [ ]:

```
x_train,y_train
```

Out[ ]:

```
(array([[[[ 59,  62,  63],
          [ 43,  46,  45],
          [ 50,  48,  43],
          ...,
          [158, 132, 108],
          [152, 125, 102],
          [148, 124, 103]],

         [[ 16,  20,  20],
          [  0,   0,   0],
          [ 18,   8,   0],
          ...,
          [123,  88,  55],
          [119,  83,  50],
          [122,  87,  57]],

         [[ 25,  24,  21],
          [ 16,   7,   0],
          [ 49,  27,   8],
          ...,
          [118,  84,  50],
          [120,  84,  50],
          [109,  73,  42]],

         ...,

         [[208, 170,  96],
          [201, 153,  34],
          [198, 161,  26],
          ...,
          [160, 133,  70],
          [ 56,  31,   7],
          [ 53,  34,  20]],

         [[180, 139,  96],
          [173, 123,  42],
          [186, 144,  30],
          ...,
          [184, 148,  94],
          [ 97,  62,  34],
          [ 83,  53,  34]],

         [[177, 144, 116],
          [168, 129,  94],
          [179, 142,  87],
          ...,
          [216, 184, 140],
          [151, 118,  84],
          [123,  92,  72]]],


        [[[154, 177, 187],
          [126, 137, 136],
          [105, 104,  95],
          ...,
          [ 91,  95,  71],
          [ 87,  90,  71],
          [ 79,  81,  70]],

         [[140, 160, 169],
          [145, 153, 154],
          [125, 125, 118],
          ...,
          [ 96,  99,  78],
          [ 77,  80,  62],
          [ 71,  73,  61]],

         [[140, 155, 164],
          [139, 146, 149],
          [115, 115, 112],
```

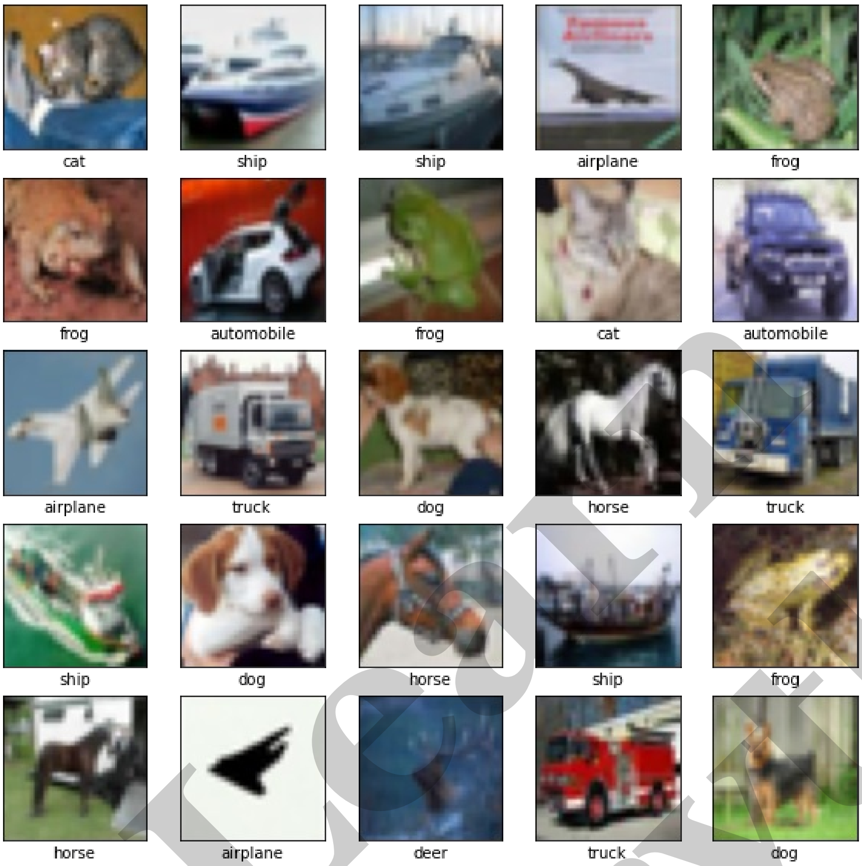
```
         ...,
         [ 79,  82,  64],
         [ 68,  70,  55],
         [ 67,  69,  55]],

        ...,

        [[175, 167, 166],
         [156, 154, 160],
         [154, 160, 170],
         ...,
         [ 42,  34,  36],
         [ 61,  53,  57],
         [ 93,  83,  91]],

        [[165, 154, 128],
         [156, 152, 130],
         [159, 161, 142],
         ...,
         [103,  93,  96],
         [123, 114, 120],
         [131, 121, 131]],

        [[163, 148, 120],
         [158, 148, 122],
         [163, 156, 133],
         ...,
         [143, 133, 139],
         [143, 134, 142],
         [143, 133, 144]]],


       [[[255, 255, 255],
         [253, 253, 253],
         [253, 253, 253],
         ...,
         [253, 253, 253],
         [253, 253, 253],
         [253, 253, 253]],

        [[255, 255, 255],
         [255, 255, 255],
         [255, 255, 255],
         ...,
         [255, 255, 255],
         [255, 255, 255],
         [255, 255, 255]],

        [[255, 255, 255],
         [254, 254, 254],
         [254, 254, 254],
         ...,
         [254, 254, 254],
         [254, 254, 254],
         [254, 254, 254]],

        ...,

        [[113, 120, 112],
         [111, 118, 111],
         [105, 112, 106],
         ...,
         [ 72,  81,  80],
         [ 72,  80,  79],
         [ 72,  80,  79]],

        [[111, 118, 110],
         [104, 111, 104],
         [ 99, 106,  98],
         ...,
         [ 68,  75,  73],
         [ 70,  76,  75],
         [ 78,  84,  82]],

        [[106, 113, 105],
         [ 99, 106,  98],
         [ 95, 102,  94],
         ...,
         [ 78,  85,  83],
         [ 79,  85,  83],
         [ 80,  86,  84]]],
```

```
        ...,


       [[[ 35, 178, 235],
         [ 40, 176, 239],
         [ 42, 176, 241],
         ...,
         [ 99, 177, 219],
         [ 79, 147, 197],
         [ 89, 148, 189]],

        [[ 57, 182, 234],
         [ 44, 184, 250],
         [ 50, 183, 240],
         ...,
         [156, 182, 200],
         [141, 177, 206],
         [116, 149, 175]],

        [[ 98, 197, 237],
         [ 64, 189, 252],
         [ 69, 192, 245],
         ...,
         [188, 195, 206],
         [119, 135, 147],
         [ 61,  79,  90]],

        ...,

        [[ 73,  79,  77],
         [ 53,  63,  68],
         [ 54,  68,  80],
         ...,
         [ 17,  40,  64],
         [ 21,  36,  51],
         [ 33,  48,  49]],

        [[ 61,  68,  75],
         [ 55,  70,  86],
         [ 57,  79, 103],
         ...,
         [ 24,  48,  72],
         [ 17,  35,  53],
         [  7,  23,  32]],

        [[ 44,  56,  73],
         [ 46,  66,  88],
         [ 49,  77, 105],
         ...,
         [ 27,  52,  77],
         [ 21,  43,  66],
         [ 12,  31,  50]]],


       [[[189, 211, 240],
         [186, 208, 236],
         [185, 207, 235],
         ...,
         [175, 195, 224],
         [172, 194, 222],
         [169, 194, 220]],

        [[194, 210, 239],
         [191, 207, 236],
         [190, 206, 235],
         ...,
         [173, 192, 220],
         [171, 191, 218],
         [167, 190, 216]],

        [[208, 219, 244],
         [205, 216, 240],
         [204, 215, 239],
         ...,
         [175, 191, 217],
         [172, 190, 216],
         [169, 191, 215]],

        ...,

        [[207, 199, 181],
         [203, 195, 175],
         [203, 196, 173],
```

```
  ...,
  [135, 132, 127],
  [162, 158, 150],
  [168, 163, 151]],

 [[198, 190, 170],
  [189, 181, 159],
  [180, 172, 147],
  ...,
  [178, 171, 160],
  [175, 169, 156],
  [175, 169, 154]],

 [[198, 189, 173],
  [189, 181, 162],
  [178, 170, 149],
  ...,
  [195, 184, 169],
  [196, 189, 171],
  [195, 190, 171]]],


[[[229, 229, 239],
  [236, 237, 247],
  [234, 236, 247],
  ...,
  [217, 219, 233],
  [221, 223, 234],
  [222, 223, 233]],

 [[222, 221, 229],
  [239, 239, 249],
  [233, 234, 246],
  ...,
  [223, 223, 236],
  [227, 228, 238],
  [210, 211, 220]],

 [[213, 206, 211],
  [234, 232, 239],
  [231, 233, 244],
  ...,
  [220, 220, 232],
  [220, 219, 232],
  [202, 203, 215]],

 ...,

 [[150, 143, 135],
  [140, 135, 127],
  [132, 127, 120],
  ...,
  [224, 222, 218],
  [230, 228, 225],
  [241, 241, 238]],

 [[137, 132, 126],
  [130, 127, 120],
  [125, 121, 115],
  ...,
  [181, 180, 178],
  [202, 201, 198],
  [212, 211, 207]],

 [[122, 119, 114],
  [118, 116, 110],
  [120, 116, 111],
  ...,
  [179, 177, 173],
  [164, 164, 162],
  [163, 163, 161]]]], dtype=uint8), array([[6],
[9],
[9],
...,
[9],
[1],
[1]], dtype=uint8))
```

In [ ]:

```
plt.figure(figsize=(10,10))
for i in range(25):
    plt.subplot(5,5,i+1)
    classNames = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
    plt.xticks([])
    plt.yticks([])
    plt.imshow(x_test[i])
    plt.xlabel(classNames[ytest[i][0]])
plt.show()
```



In [ ]:

```
# Converting input image data into float
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
```

In [ ]:

```
x_train = (x_train-x_train.mean())/x_train.max()
x_test = (x_test-x_test.mean())/x_test.max()
y_train = keras.utils.to_categorical(y_train,10)
y_test = keras.utils.to_categorical(ytest,10)
```

In [ ]:

```
# Difference between ytest and y_test
print(ytest)
print(y_test)
```

```
[[3]
 [8]
 [8]
 ...
 [5]
 [1]
 [7]]
[[0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 1. 0.]
 [0. 0. 0. ... 0. 1. 0.]
 ...
 [0. 0. 0. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 1. 0. 0.]]
```

In [ ]:

```python
List = [x_train.shape,x_test.shape,y_train.shape,y_test.shape]
print(List)
```

```
[(50000, 32, 32, 3), (10000, 32, 32, 3), (50000, 10), (10000, 10)]
```

In [ ]:

```python
model = Sequential()

model.add(Conv2D(32, (3, 3), padding='same',activation='relu', input_shape=(32,32,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3, 3), padding='same',activation='relu', input_shape=(32,32,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(10, activation='softmax'))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)          0
_________________________________________________________________
dropout (Dropout)            (None, 8, 8, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 4096)              0
_________________________________________________________________
dense (Dense)                (None, 256)               1048832
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2570
=================================================================
Total params: 1,070,794
Trainable params: 1,070,794
Non-trainable params: 0
_________________________________________________________________
```

In [ ]:

```python
model.compile(loss='categorical_crossentropy', optimizer=Adam(lr=1.0e-4), metrics = ['accuracy'])
```

In [ ]:

```python
model.fit(x_train, y_train, batch_size=256, epochs=200)
```

```
Epoch 1/200
196/196 [==============================] - 1s 7ms/step - loss: 1.9800 - accuracy: 0.2885
Epoch 2/200
196/196 [==============================] - 1s 7ms/step - loss: 1.6653 - accuracy: 0.4065
Epoch 3/200
196/196 [==============================] - 1s 7ms/step - loss: 1.5309 - accuracy: 0.4519
Epoch 4/200
196/196 [==============================] - 1s 7ms/step - loss: 1.4519 - accuracy: 0.4814
Epoch 5/200
196/196 [==============================] - 1s 7ms/step - loss: 1.3942 - accuracy: 0.5013
Epoch 6/200
196/196 [==============================] - 1s 7ms/step - loss: 1.3508 - accuracy: 0.5172
Epoch 7/200
196/196 [==============================] - 1s 7ms/step - loss: 1.3082 - accuracy: 0.5336
Epoch 8/200
196/196 [==============================] - 1s 7ms/step - loss: 1.2788 - accuracy: 0.5455
Epoch 9/200
196/196 [==============================] - 1s 7ms/step - loss: 1.2482 - accuracy: 0.5577
Epoch 10/200
196/196 [==============================] - 1s 7ms/step - loss: 1.2171 - accuracy: 0.5682
Epoch 11/200
```

```
196/196 [==============================] - 1s 7ms/step - loss: 1.1952 - accuracy: 0.5785
Epoch 12/200
196/196 [==============================] - 1s 7ms/step - loss: 1.1681 - accuracy: 0.5863
Epoch 13/200
196/196 [==============================] - 1s 7ms/step - loss: 1.1457 - accuracy: 0.5952
Epoch 14/200
196/196 [==============================] - 1s 7ms/step - loss: 1.1289 - accuracy: 0.6011
Epoch 15/200
196/196 [==============================] - 1s 7ms/step - loss: 1.1091 - accuracy: 0.6087
Epoch 16/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0918 - accuracy: 0.6159
Epoch 17/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0738 - accuracy: 0.6211
Epoch 18/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0574 - accuracy: 0.6290
Epoch 19/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0431 - accuracy: 0.6305
Epoch 20/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0281 - accuracy: 0.6386
Epoch 21/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0147 - accuracy: 0.6422
Epoch 22/200
196/196 [==============================] - 1s 7ms/step - loss: 1.0004 - accuracy: 0.6458
Epoch 23/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9913 - accuracy: 0.6528
Epoch 24/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9746 - accuracy: 0.6582
Epoch 25/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9627 - accuracy: 0.6613
Epoch 26/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9518 - accuracy: 0.6661
Epoch 27/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9392 - accuracy: 0.6709
Epoch 28/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9315 - accuracy: 0.6741
Epoch 29/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9193 - accuracy: 0.6783
Epoch 30/200
196/196 [==============================] - 1s 7ms/step - loss: 0.9095 - accuracy: 0.6805
Epoch 31/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8996 - accuracy: 0.6858
Epoch 32/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8883 - accuracy: 0.6892
Epoch 33/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8812 - accuracy: 0.6914
Epoch 34/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8693 - accuracy: 0.6972
Epoch 35/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8616 - accuracy: 0.6977
Epoch 36/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8540 - accuracy: 0.7026
Epoch 37/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8460 - accuracy: 0.7057
Epoch 38/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8390 - accuracy: 0.7058
Epoch 39/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8306 - accuracy: 0.7098
Epoch 40/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8209 - accuracy: 0.7150
Epoch 41/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8139 - accuracy: 0.7144
Epoch 42/200
196/196 [==============================] - 1s 7ms/step - loss: 0.8019 - accuracy: 0.7195
Epoch 43/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7984 - accuracy: 0.7210
Epoch 44/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7901 - accuracy: 0.7230
Epoch 45/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7859 - accuracy: 0.7248
Epoch 46/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7760 - accuracy: 0.7295
Epoch 47/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7713 - accuracy: 0.7304
Epoch 48/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7662 - accuracy: 0.7326
Epoch 49/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7591 - accuracy: 0.7356
Epoch 50/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7520 - accuracy: 0.7396
Epoch 51/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7465 - accuracy: 0.7396
Epoch 52/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7364 - accuracy: 0.7422
```

```
Epoch 53/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7330 - accuracy: 0.7445
Epoch 54/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7248 - accuracy: 0.7470
Epoch 55/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7200 - accuracy: 0.7494
Epoch 56/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7106 - accuracy: 0.7528
Epoch 57/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7047 - accuracy: 0.7524
Epoch 58/200
196/196 [==============================] - 1s 7ms/step - loss: 0.7046 - accuracy: 0.7546
Epoch 59/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6953 - accuracy: 0.7546
Epoch 60/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6894 - accuracy: 0.7597
Epoch 61/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6844 - accuracy: 0.7611
Epoch 62/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6796 - accuracy: 0.7642
Epoch 63/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6718 - accuracy: 0.7648
Epoch 64/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6667 - accuracy: 0.7697
Epoch 65/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6611 - accuracy: 0.7704
Epoch 66/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6549 - accuracy: 0.7725
Epoch 67/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6516 - accuracy: 0.7722
Epoch 68/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6500 - accuracy: 0.7725
Epoch 69/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6400 - accuracy: 0.7778
Epoch 70/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6338 - accuracy: 0.7791
Epoch 71/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6304 - accuracy: 0.7794
Epoch 72/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6286 - accuracy: 0.7806
Epoch 73/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6173 - accuracy: 0.7829
Epoch 74/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6147 - accuracy: 0.7851
Epoch 75/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6064 - accuracy: 0.7892
Epoch 76/200
196/196 [==============================] - 1s 7ms/step - loss: 0.6016 - accuracy: 0.7913
Epoch 77/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5999 - accuracy: 0.7912
Epoch 78/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5972 - accuracy: 0.7929
Epoch 79/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5862 - accuracy: 0.7940
Epoch 80/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5823 - accuracy: 0.7960
Epoch 81/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5798 - accuracy: 0.7975
Epoch 82/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5786 - accuracy: 0.7989
Epoch 83/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5734 - accuracy: 0.8001
Epoch 84/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5692 - accuracy: 0.8014
Epoch 85/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5593 - accuracy: 0.8056
Epoch 86/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5586 - accuracy: 0.8035
Epoch 87/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5541 - accuracy: 0.8064
Epoch 88/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5494 - accuracy: 0.8076
Epoch 89/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5423 - accuracy: 0.8087
Epoch 90/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5445 - accuracy: 0.8094
Epoch 91/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5382 - accuracy: 0.8134
Epoch 92/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5298 - accuracy: 0.8150
Epoch 93/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5309 - accuracy: 0.8137
Epoch 94/200
```

```
196/196 [==============================] - 1s 7ms/step - loss: 0.5201 - accuracy: 0.8178
Epoch 95/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5170 - accuracy: 0.8195
Epoch 96/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5160 - accuracy: 0.8194
Epoch 97/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5142 - accuracy: 0.8185
Epoch 98/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5088 - accuracy: 0.8223
Epoch 99/200
196/196 [==============================] - 1s 7ms/step - loss: 0.5044 - accuracy: 0.8239
Epoch 100/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4974 - accuracy: 0.8264
Epoch 101/200
196/196 [==============================] - 2s 8ms/step - loss: 0.4931 - accuracy: 0.8277
Epoch 102/200
196/196 [==============================] - 2s 8ms/step - loss: 0.4934 - accuracy: 0.8268
Epoch 103/200
196/196 [==============================] - 2s 8ms/step - loss: 0.4863 - accuracy: 0.8310
Epoch 104/200
196/196 [==============================] - 2s 8ms/step - loss: 0.4834 - accuracy: 0.8306
Epoch 105/200
196/196 [==============================] - 1s 8ms/step - loss: 0.4788 - accuracy: 0.8330
Epoch 106/200
196/196 [==============================] - 2s 8ms/step - loss: 0.4788 - accuracy: 0.8306
Epoch 107/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4742 - accuracy: 0.8349
Epoch 108/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4678 - accuracy: 0.8378
Epoch 109/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4677 - accuracy: 0.8353
Epoch 110/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4576 - accuracy: 0.8401
Epoch 111/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4579 - accuracy: 0.8377
Epoch 112/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4542 - accuracy: 0.8412
Epoch 113/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4524 - accuracy: 0.8414
Epoch 114/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4466 - accuracy: 0.8428
Epoch 115/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4445 - accuracy: 0.8444
Epoch 116/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4423 - accuracy: 0.8451
Epoch 117/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4379 - accuracy: 0.8462
Epoch 118/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4345 - accuracy: 0.8475
Epoch 119/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4325 - accuracy: 0.8484
Epoch 120/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4277 - accuracy: 0.8489
Epoch 121/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4232 - accuracy: 0.8498
Epoch 122/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4217 - accuracy: 0.8525
Epoch 123/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4173 - accuracy: 0.8533
Epoch 124/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4178 - accuracy: 0.8513
Epoch 125/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4128 - accuracy: 0.8545
Epoch 126/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4093 - accuracy: 0.8572
Epoch 127/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4045 - accuracy: 0.8597
Epoch 128/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4034 - accuracy: 0.8589
Epoch 129/200
196/196 [==============================] - 1s 7ms/step - loss: 0.4017 - accuracy: 0.8598
Epoch 130/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3963 - accuracy: 0.8608
Epoch 131/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3988 - accuracy: 0.8595
Epoch 132/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3917 - accuracy: 0.8610
Epoch 133/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3855 - accuracy: 0.8652
Epoch 134/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3865 - accuracy: 0.8620
Epoch 135/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3860 - accuracy: 0.8643
```

```
Epoch 136/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3809 - accuracy: 0.8660
Epoch 137/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3752 - accuracy: 0.8699
Epoch 138/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3767 - accuracy: 0.8671
Epoch 139/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3732 - accuracy: 0.8681
Epoch 140/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3682 - accuracy: 0.8713
Epoch 141/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3688 - accuracy: 0.8712
Epoch 142/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3643 - accuracy: 0.8737
Epoch 143/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3616 - accuracy: 0.8724
Epoch 144/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3626 - accuracy: 0.8716
Epoch 145/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3555 - accuracy: 0.8746
Epoch 146/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3543 - accuracy: 0.8759
Epoch 147/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3475 - accuracy: 0.8758
Epoch 148/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3496 - accuracy: 0.8762
Epoch 149/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3469 - accuracy: 0.8775
Epoch 150/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3433 - accuracy: 0.8794
Epoch 151/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3430 - accuracy: 0.8805
Epoch 152/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3449 - accuracy: 0.8789
Epoch 153/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3379 - accuracy: 0.8807
Epoch 154/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3357 - accuracy: 0.8836
Epoch 155/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3303 - accuracy: 0.8855
Epoch 156/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3319 - accuracy: 0.8819
Epoch 157/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3291 - accuracy: 0.8852
Epoch 158/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3273 - accuracy: 0.8839
Epoch 159/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3240 - accuracy: 0.8865
Epoch 160/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3218 - accuracy: 0.8863
Epoch 161/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3208 - accuracy: 0.8853
Epoch 162/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3165 - accuracy: 0.8883
Epoch 163/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3189 - accuracy: 0.8881
Epoch 164/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3127 - accuracy: 0.8914
Epoch 165/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3123 - accuracy: 0.8895
Epoch 166/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3091 - accuracy: 0.8903
Epoch 167/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3110 - accuracy: 0.8897
Epoch 168/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3079 - accuracy: 0.8920
Epoch 169/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3043 - accuracy: 0.8916
Epoch 170/200
196/196 [==============================] - 1s 7ms/step - loss: 0.3016 - accuracy: 0.8939
Epoch 171/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2978 - accuracy: 0.8955
Epoch 172/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2997 - accuracy: 0.8960
Epoch 173/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2973 - accuracy: 0.8952
Epoch 174/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2924 - accuracy: 0.8950
Epoch 175/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2881 - accuracy: 0.8991
Epoch 176/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2896 - accuracy: 0.8974
Epoch 177/200
```

```
196/196 [==============================] - 1s 7ms/step - loss: 0.2883 - accuracy: 0.8989
Epoch 178/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2864 - accuracy: 0.8996
Epoch 179/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2849 - accuracy: 0.8996
Epoch 180/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2841 - accuracy: 0.8988
Epoch 181/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2801 - accuracy: 0.9014
Epoch 182/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2785 - accuracy: 0.9006
Epoch 183/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2812 - accuracy: 0.9009
Epoch 184/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2713 - accuracy: 0.9054
Epoch 185/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2770 - accuracy: 0.9029
Epoch 186/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2761 - accuracy: 0.9030
Epoch 187/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2729 - accuracy: 0.9028
Epoch 188/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2686 - accuracy: 0.9063
Epoch 189/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2677 - accuracy: 0.9059
Epoch 190/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2645 - accuracy: 0.9086
Epoch 191/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2630 - accuracy: 0.9063
Epoch 192/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2612 - accuracy: 0.9088
Epoch 193/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2570 - accuracy: 0.9109
Epoch 194/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2598 - accuracy: 0.9090
Epoch 195/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2604 - accuracy: 0.9072
Epoch 196/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2558 - accuracy: 0.9092
Epoch 197/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2554 - accuracy: 0.9107
Epoch 198/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2563 - accuracy: 0.9100
Epoch 199/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2524 - accuracy: 0.9111
Epoch 200/200
196/196 [==============================] - 1s 7ms/step - loss: 0.2525 - accuracy: 0.9114
```

Out[ ]:

```
<tensorflow.python.keras.callbacks.History at 0x7f9f202231d0>
```

In [ ]:

```python
yhat = model.predict_classes(x_test)
print(sm.classification_report(ytest,yhat))
print(f'Accuracy of test data: {sm.accuracy_score(ytest,yhat)*100}%')
```

```
WARNING:tensorflow:From <ipython-input-13-a1a7593971d8>:1: Sequential.predict_classes (from tensorfl
ow.python.keras.engine.sequential) is deprecated and will be removed after 2021-01-01.
Instructions for updating:
Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classi
fication   (e.g. if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("i
nt32")`,   if your model does binary classification   (e.g. if it uses a `sigmoid` last-layer activa
tion).
              precision    recall  f1-score   support

           0       0.79      0.81      0.80      1000
           1       0.83      0.89      0.86      1000
           2       0.66      0.69      0.67      1000
           3       0.60      0.60      0.60      1000
           4       0.71      0.75      0.73      1000
           5       0.69      0.62      0.65      1000
           6       0.84      0.84      0.84      1000
           7       0.84      0.80      0.82      1000
           8       0.86      0.85      0.85      1000
           9       0.84      0.81      0.82      1000

    accuracy                           0.77     10000
   macro avg       0.77      0.77      0.76     10000
weighted avg       0.77      0.77      0.76     10000

Accuracy of test data: 76.53%
```

In [ ]:

```
cm = sm.confusion_matrix(ytest,yhat)
plt.clf()
plt.imshow(cm,cmap=plt.cm.autumn_r)
plt.title('Confusion Matrix - Test Data')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
tick_marks = np.arange(len(classNames))
plt.xticks(tick_marks,classNames,rotation=90)
plt.yticks(tick_marks,classNames)
for i in range(len(classNames)):
  for j in range(len(classNames)):
    plt.text(i,j,cm[i][j])
plt.show()
```