Task 13

Registration ID : SIRSS1038

Leela Satya Kartheek Raja

In [1]:

```
from keras.datasets import cifar10

import matplotlib.pyplot as plt
from keras import models,layers
from tensorflow.keras.utils import to_categorical
from keras import optimizers
```

In [2]:

```
(xtrain, ytrain), (xtest, ytest) = cifar10.load_data()
print(xtrain.shape)
print(xtest.shape)
print(ytrain.shape)
print(ytest.shape)
```

```
(50000, 32, 32, 3)
(10000, 32, 32, 3)
(50000, 1)
(10000, 1)
```

In [3]:

```
# convert the pixel values in float
xtrain = xtrain.astype('float32')
xtest = xtest.astype('float32')
# scale the images
xtrain /= 255  # ths is eqvalent to xtrain = xtrain/255
xtest /= 255
ytrain = to_categorical(ytrain)
ytest = to_categorical(ytest)
```

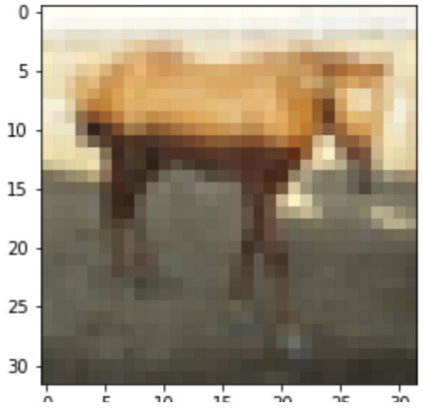
In [4]:

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

In [5]:

```
#explore data
print(ytrain[2000])
plt.imshow(xtrain[2000])
plt.show()
```

```
[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
```



0 5 10 15 20 25 30

In [6]:

```
from datetime import datetime
def timer(start_time=None):
    if not start_time:
        print(datetime.now())
        start_time = datetime.now()
        return start_time
    elif start_time:
        thour, temp_sec = divmod((datetime.now() - start_time).total_seconds(), 3600)
        tmin, tsec = divmod(temp_sec, 60)
        print('Time taken: %i hours %i minutes and %s seconds.' % (thour, tmin, round(ts
ec, 2)))
```

In [7]:

```
print(xtrain[10])
```

```
[[[0.20784314 0.25490198 0.20784314]
  [0.21176471 0.24705882 0.20392157]
  [0.21960784 0.23529412 0.19607843]
  ...
  [0.18431373 0.2        0.19607843]
  [0.16078432 0.1764706  0.17254902]
  [0.09411765 0.10980392 0.10588235]]

 [[0.18039216 0.23137255 0.16078432]
  [0.20784314 0.24313726 0.1764706 ]
  [0.21176471 0.23137255 0.17254902]
  ...
  [0.16470589 0.18039216 0.1764706 ]
  [0.15294118 0.16862746 0.16470589]
  [0.10980392 0.1254902  0.12156863]]

 [[0.1764706  0.23137255 0.14901961]
  [0.19607843 0.23529412 0.16078432]
  [0.18039216 0.20392157 0.13333334]
  ...
  [0.14901961 0.16470589 0.16078432]
  [0.14117648 0.15686275 0.15294118]
  [0.11372549 0.12941177 0.1254902 ]]

 ...

 [[0.2784314  0.3254902  0.25882354]
  [0.2901961  0.3254902  0.25882354]
  [0.3137255  0.33333334 0.2627451 ]
  ...
  [0.2        0.21176471 0.12941177]
  [0.18039216 0.19215687 0.13333334]
  [0.19215687 0.19607843 0.16078432]]

 [[0.29411766 0.32156864 0.2627451 ]
  [0.30980393 0.33333334 0.27058825]
  [0.31764707 0.33333334 0.2627451 ]
  ...
  [0.23921569 0.25490198 0.16078432]
  [0.2509804  0.2627451  0.1882353 ]
  [0.1882353  0.19607843 0.16078432]]

 [[0.33333334 0.3254902  0.2784314 ]
  [0.33333334 0.32941177 0.2784314 ]
  [0.3372549  0.3372549  0.27450982]
  ...
  [0.23921569 0.25490198 0.15294118]
  [0.2509804  0.2627451  0.18039216]
  [0.19215687 0.19607843 0.16078432]]]
```

In [8]:

```
ytrain[10]
```

Out[8]:

```
array([0., 0., 0., 0., 1., 0., 0., 0., 0., 0.], dtype=float32)
```

Creating the Model Layers

In [9]:

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), padding='same', activation='relu', input_shape=(32,3
2,3)))
model.add(layers.Conv2D(32,(3,3), padding='same',activation='relu'))
model.add(layers.MaxPooling2D(pool_size=2))
model.add(layers.Dropout(0.2))

model.add(layers.Conv2D(64,(3,3), padding='same',activation='relu'))
model.add(layers.Conv2D(128,(3,3), padding='same',activation='relu'))
model.add(layers.MaxPooling2D(pool_size=2))
model.add(layers.Dropout(0.2))

model.add(layers.Conv2D(64,(3,3), padding='same',activation='relu'))
model.add(layers.Conv2D(64,(3,3), padding='same',activation='relu'))
model.add(layers.MaxPooling2D(pool_size=2))
model.add(layers.Dropout(0.2))

#Classification layers
model.add(layers.Flatten())
model.add(layers.Dense(512,activation='relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(64,activation='relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(10,activation='softmax'))  # this is the actual output layer

# initiate Adam optimizer
opt = optimizers.Adam(learning_rate=1e-4, decay=1e-6)

# Let's train the model
model.compile(loss='categorical_crossentropy',
              optimizer=opt,
              metrics=['accuracy'])
```

Viewing the Model Summary

In [10]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 128)       73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 128)         0
_________________________________________________________________
dropout_1 (Dropout)          (None, 8, 8, 128)         0
```

```
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 8, 8, 64)          73792
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 8, 8, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 4, 4, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 512)               524800
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                32832
_________________________________________________________________
dropout_4 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 771,498
Trainable params: 771,498
Non-trainable params: 0
_________________________________________________________________
```

Fitting the Model

In [11]:

```
start_time=timer(None)
result = model.fit(
    xtrain,
    ytrain,
    validation_split=0.1,
    verbose=True,
    epochs=125,
    steps_per_epoch= 64,
    batch_size=512
    )
timer(start_time)
```

```
2021-07-31 14:09:42.877233
Epoch 1/125
64/64 [==============================] - 21s 66ms/step - loss: 2.2923 - accuracy: 0.1154
- val_loss: 2.0805 - val_accuracy: 0.2270
Epoch 2/125
64/64 [==============================] - 4s 62ms/step - loss: 2.0810 - accuracy: 0.2230 -
val_loss: 1.9673 - val_accuracy: 0.2928
Epoch 3/125
64/64 [==============================] - 3s 46ms/step - loss: 1.9688 - accuracy: 0.2804 -
val_loss: 1.8098 - val_accuracy: 0.3646
Epoch 4/125
64/64 [==============================] - 3s 46ms/step - loss: 1.8392 - accuracy: 0.3283 -
val_loss: 1.7014 - val_accuracy: 0.3844
Epoch 5/125
64/64 [==============================] - 3s 47ms/step - loss: 1.7562 - accuracy: 0.3562 -
val_loss: 1.6218 - val_accuracy: 0.4160
Epoch 6/125
64/64 [==============================] - 3s 46ms/step - loss: 1.6964 - accuracy: 0.3828 -
val_loss: 1.5585 - val_accuracy: 0.4442
Epoch 7/125
64/64 [==============================] - 3s 47ms/step - loss: 1.6321 - accuracy: 0.4048 -
val_loss: 1.5359 - val_accuracy: 0.4456
Epoch 8/125
64/64 [==============================] - 3s 47ms/step - loss: 1.5901 - accuracy: 0.4229 -
val_loss: 1.4851 - val_accuracy: 0.4610
Epoch 9/125
64/64 [==============================] - 3s 47ms/step - loss: 1.5536 - accuracy: 0.4357 -
val_loss: 1.4549 - val_accuracy: 0.4756
Epoch 10/125
```

```
Epoch 10/125
64/64 [==============================] - 3s 47ms/step - loss: 1.5333 - accuracy: 0.4464 -
val_loss: 1.4348 - val_accuracy: 0.4798
Epoch 11/125
64/64 [==============================] - 3s 47ms/step - loss: 1.4971 - accuracy: 0.4549 -
val_loss: 1.3760 - val_accuracy: 0.5054
Epoch 12/125
64/64 [==============================] - 3s 47ms/step - loss: 1.4510 - accuracy: 0.4747 -
val_loss: 1.3732 - val_accuracy: 0.5026
Epoch 13/125
64/64 [==============================] - 3s 48ms/step - loss: 1.4602 - accuracy: 0.4714 -
val_loss: 1.3458 - val_accuracy: 0.5108
Epoch 14/125
64/64 [==============================] - 3s 47ms/step - loss: 1.4152 - accuracy: 0.4878 -
val_loss: 1.3190 - val_accuracy: 0.5252
Epoch 15/125
64/64 [==============================] - 3s 47ms/step - loss: 1.3961 - accuracy: 0.4977 -
val_loss: 1.3011 - val_accuracy: 0.5302
Epoch 16/125
64/64 [==============================] - 3s 48ms/step - loss: 1.3641 - accuracy: 0.5114 -
val_loss: 1.2536 - val_accuracy: 0.5508
Epoch 17/125
64/64 [==============================] - 3s 48ms/step - loss: 1.3632 - accuracy: 0.5090 -
val_loss: 1.2481 - val_accuracy: 0.5512
Epoch 18/125
64/64 [==============================] - 3s 48ms/step - loss: 1.3380 - accuracy: 0.5225 -
val_loss: 1.2209 - val_accuracy: 0.5628
Epoch 19/125
64/64 [==============================] - 3s 48ms/step - loss: 1.3154 - accuracy: 0.5272 -
val_loss: 1.2183 - val_accuracy: 0.5660
Epoch 20/125
64/64 [==============================] - 3s 49ms/step - loss: 1.3010 - accuracy: 0.5348 -
val_loss: 1.1944 - val_accuracy: 0.5752
Epoch 21/125
64/64 [==============================] - 3s 49ms/step - loss: 1.2694 - accuracy: 0.5468 -
val_loss: 1.1770 - val_accuracy: 0.5832
Epoch 22/125
64/64 [==============================] - 3s 48ms/step - loss: 1.2734 - accuracy: 0.5495 -
val_loss: 1.1851 - val_accuracy: 0.5774
Epoch 23/125
64/64 [==============================] - 3s 49ms/step - loss: 1.2398 - accuracy: 0.5597 -
val_loss: 1.1700 - val_accuracy: 0.5796
Epoch 24/125
64/64 [==============================] - 3s 48ms/step - loss: 1.2341 - accuracy: 0.5591 -
val_loss: 1.1551 - val_accuracy: 0.5888
Epoch 25/125
64/64 [==============================] - 3s 48ms/step - loss: 1.2262 - accuracy: 0.5607 -
val_loss: 1.1133 - val_accuracy: 0.6066
Epoch 26/125
64/64 [==============================] - 3s 48ms/step - loss: 1.2095 - accuracy: 0.5703 -
val_loss: 1.1146 - val_accuracy: 0.6086
Epoch 27/125
64/64 [==============================] - 3s 49ms/step - loss: 1.1931 - accuracy: 0.5759 -
val_loss: 1.0671 - val_accuracy: 0.6230
Epoch 28/125
64/64 [==============================] - 3s 48ms/step - loss: 1.1821 - accuracy: 0.5789 -
val_loss: 1.1069 - val_accuracy: 0.6082
Epoch 29/125
64/64 [==============================] - 3s 48ms/step - loss: 1.1696 - accuracy: 0.5811 -
val_loss: 1.0591 - val_accuracy: 0.6250
Epoch 30/125
64/64 [==============================] - 3s 48ms/step - loss: 1.1552 - accuracy: 0.5895 -
val_loss: 1.0540 - val_accuracy: 0.6316
Epoch 31/125
64/64 [==============================] - 3s 49ms/step - loss: 1.1525 - accuracy: 0.5916 -
val_loss: 1.0346 - val_accuracy: 0.6322
Epoch 32/125
64/64 [==============================] - 3s 49ms/step - loss: 1.1286 - accuracy: 0.5951 -
val_loss: 1.0199 - val_accuracy: 0.6412
Epoch 33/125
64/64 [==============================] - 3s 48ms/step - loss: 1.1238 - accuracy: 0.6016 -
val_loss: 1.0166 - val_accuracy: 0.6396
Epoch 34/125
```

```
Epoch 34/125
64/64 [==============================] - 3s 49ms/step - loss: 1.1103 - accuracy: 0.6054 -
val_loss: 1.0054 - val_accuracy: 0.6468
Epoch 35/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0949 - accuracy: 0.6126 -
val_loss: 1.0102 - val_accuracy: 0.6452
Epoch 36/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0889 - accuracy: 0.6138 -
val_loss: 0.9725 - val_accuracy: 0.6594
Epoch 37/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0755 - accuracy: 0.6213 -
val_loss: 0.9720 - val_accuracy: 0.6616
Epoch 38/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0787 - accuracy: 0.6186 -
val_loss: 0.9875 - val_accuracy: 0.6520
Epoch 39/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0501 - accuracy: 0.6268 -
val_loss: 0.9496 - val_accuracy: 0.6638
Epoch 40/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0639 - accuracy: 0.6223 -
val_loss: 0.9517 - val_accuracy: 0.6630
Epoch 41/125
64/64 [==============================] - 3s 50ms/step - loss: 1.0268 - accuracy: 0.6350 -
val_loss: 0.9528 - val_accuracy: 0.6690
Epoch 42/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0430 - accuracy: 0.6329 -
val_loss: 0.9637 - val_accuracy: 0.6652
Epoch 43/125
64/64 [==============================] - 3s 49ms/step - loss: 1.0207 - accuracy: 0.6359 -
val_loss: 0.9181 - val_accuracy: 0.6768
Epoch 44/125
64/64 [==============================] - 3s 50ms/step - loss: 1.0210 - accuracy: 0.6403 -
val_loss: 0.9089 - val_accuracy: 0.6852
Epoch 45/125
64/64 [==============================] - 3s 50ms/step - loss: 1.0048 - accuracy: 0.6443 -
val_loss: 0.9039 - val_accuracy: 0.6832
Epoch 46/125
64/64 [==============================] - 3s 50ms/step - loss: 1.0099 - accuracy: 0.6469 -
val_loss: 0.9224 - val_accuracy: 0.6752
Epoch 47/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9947 - accuracy: 0.6464 -
val_loss: 0.8978 - val_accuracy: 0.6900
Epoch 48/125
64/64 [==============================] - 3s 49ms/step - loss: 0.9824 - accuracy: 0.6541 -
val_loss: 0.8926 - val_accuracy: 0.6862
Epoch 49/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9808 - accuracy: 0.6533 -
val_loss: 0.9074 - val_accuracy: 0.6886
Epoch 50/125
64/64 [==============================] - 3s 49ms/step - loss: 0.9625 - accuracy: 0.6619 -
val_loss: 0.8698 - val_accuracy: 0.6946
Epoch 51/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9506 - accuracy: 0.6622 -
val_loss: 0.8714 - val_accuracy: 0.6978
Epoch 52/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9488 - accuracy: 0.6649 -
val_loss: 0.8614 - val_accuracy: 0.6998
Epoch 53/125
64/64 [==============================] - 3s 49ms/step - loss: 0.9402 - accuracy: 0.6701 -
val_loss: 0.8508 - val_accuracy: 0.7064
Epoch 54/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9352 - accuracy: 0.6694 -
val_loss: 0.8602 - val_accuracy: 0.7010
Epoch 55/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9361 - accuracy: 0.6704 -
val_loss: 0.8457 - val_accuracy: 0.7070
Epoch 56/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9273 - accuracy: 0.6753 -
val_loss: 0.8688 - val_accuracy: 0.6996
Epoch 57/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9155 - accuracy: 0.6738 -
val_loss: 0.8376 - val_accuracy: 0.7114
Epoch 58/125
```

```
Epoch 58/125
64/64 [==============================] - 3s 50ms/step - loss: 0.9168 - accuracy: 0.6792 -
val_loss: 0.8337 - val_accuracy: 0.7074
Epoch 59/125
64/64 [==============================] - 3s 51ms/step - loss: 0.9046 - accuracy: 0.6822 -
val_loss: 0.8412 - val_accuracy: 0.7096
Epoch 60/125
64/64 [==============================] - 3s 51ms/step - loss: 0.8969 - accuracy: 0.6855 -
val_loss: 0.8336 - val_accuracy: 0.7088
Epoch 61/125
64/64 [==============================] - 3s 51ms/step - loss: 0.9021 - accuracy: 0.6805 -
val_loss: 0.8266 - val_accuracy: 0.7166
Epoch 62/125
64/64 [==============================] - 3s 51ms/step - loss: 0.8872 - accuracy: 0.6898 -
val_loss: 0.8320 - val_accuracy: 0.7122
Epoch 63/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8827 - accuracy: 0.6901 -
val_loss: 0.8143 - val_accuracy: 0.7218
Epoch 64/125
64/64 [==============================] - 3s 51ms/step - loss: 0.8802 - accuracy: 0.6921 -
val_loss: 0.8119 - val_accuracy: 0.7184
Epoch 65/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8700 - accuracy: 0.6945 -
val_loss: 0.8193 - val_accuracy: 0.7170
Epoch 66/125
64/64 [==============================] - 3s 51ms/step - loss: 0.8650 - accuracy: 0.6981 -
val_loss: 0.7987 - val_accuracy: 0.7250
Epoch 67/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8569 - accuracy: 0.7001 -
val_loss: 0.8000 - val_accuracy: 0.7238
Epoch 68/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8538 - accuracy: 0.6985 -
val_loss: 0.7911 - val_accuracy: 0.7318
Epoch 69/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8512 - accuracy: 0.7030 -
val_loss: 0.7975 - val_accuracy: 0.7268
Epoch 70/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8440 - accuracy: 0.7047 -
val_loss: 0.7850 - val_accuracy: 0.7360
Epoch 71/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8376 - accuracy: 0.7050 -
val_loss: 0.7831 - val_accuracy: 0.7324
Epoch 72/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8504 - accuracy: 0.7058 -
val_loss: 0.7780 - val_accuracy: 0.7356
Epoch 73/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8358 - accuracy: 0.7081 -
val_loss: 0.7858 - val_accuracy: 0.7286
Epoch 74/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8338 - accuracy: 0.7088 -
val_loss: 0.7759 - val_accuracy: 0.7392
Epoch 75/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8138 - accuracy: 0.7111 -
val_loss: 0.7792 - val_accuracy: 0.7312
Epoch 76/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8172 - accuracy: 0.7076 -
val_loss: 0.7649 - val_accuracy: 0.7424
Epoch 77/125
64/64 [==============================] - 3s 50ms/step - loss: 0.8005 - accuracy: 0.7183 -
val_loss: 0.7653 - val_accuracy: 0.7376
Epoch 78/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7996 - accuracy: 0.7209 -
val_loss: 0.7561 - val_accuracy: 0.7436
Epoch 79/125
64/64 [==============================] - 3s 51ms/step - loss: 0.8078 - accuracy: 0.7110 -
val_loss: 0.7705 - val_accuracy: 0.7318
Epoch 80/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7933 - accuracy: 0.7226 -
val_loss: 0.7595 - val_accuracy: 0.7366
Epoch 81/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7913 - accuracy: 0.7238 -
val_loss: 0.7639 - val_accuracy: 0.7368
Epoch 82/125
```

Epoch 82/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7787 - accuracy: 0.7277 - val_loss: 0.7664 - val_accuracy: 0.7340
Epoch 83/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7788 - accuracy: 0.7274 - val_loss: 0.7558 - val_accuracy: 0.7418
Epoch 84/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7695 - accuracy: 0.7331 - val_loss: 0.7547 - val_accuracy: 0.7400
Epoch 85/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7613 - accuracy: 0.7319 - val_loss: 0.7411 - val_accuracy: 0.7456
Epoch 86/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7642 - accuracy: 0.7322 - val_loss: 0.7436 - val_accuracy: 0.7460
Epoch 87/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7608 - accuracy: 0.7314 - val_loss: 0.7552 - val_accuracy: 0.7454
Epoch 88/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7663 - accuracy: 0.7339 - val_loss: 0.7421 - val_accuracy: 0.7476
Epoch 89/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7388 - accuracy: 0.7422 - val_loss: 0.7322 - val_accuracy: 0.7508
Epoch 90/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7492 - accuracy: 0.7372 - val_loss: 0.7570 - val_accuracy: 0.7386
Epoch 91/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7494 - accuracy: 0.7375 - val_loss: 0.7261 - val_accuracy: 0.7542
Epoch 92/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7320 - accuracy: 0.7435 - val_loss: 0.7303 - val_accuracy: 0.7496
Epoch 93/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7337 - accuracy: 0.7393 - val_loss: 0.7299 - val_accuracy: 0.7522
Epoch 94/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7261 - accuracy: 0.7436 - val_loss: 0.7382 - val_accuracy: 0.7486
Epoch 95/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7176 - accuracy: 0.7451 - val_loss: 0.7172 - val_accuracy: 0.7554
Epoch 96/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7161 - accuracy: 0.7469 - val_loss: 0.7309 - val_accuracy: 0.7448
Epoch 97/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7179 - accuracy: 0.7484 - val_loss: 0.7133 - val_accuracy: 0.7566
Epoch 98/125
64/64 [==============================] - 3s 50ms/step - loss: 0.7023 - accuracy: 0.7553 - val_loss: 0.7148 - val_accuracy: 0.7540
Epoch 99/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6971 - accuracy: 0.7564 - val_loss: 0.7078 - val_accuracy: 0.7586
Epoch 100/125
64/64 [==============================] - 3s 51ms/step - loss: 0.7025 - accuracy: 0.7514 - val_loss: 0.7118 - val_accuracy: 0.7582
Epoch 101/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6948 - accuracy: 0.7541 - val_loss: 0.7163 - val_accuracy: 0.7524
Epoch 102/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6987 - accuracy: 0.7513 - val_loss: 0.7075 - val_accuracy: 0.7576
Epoch 103/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6835 - accuracy: 0.7604 - val_loss: 0.7113 - val_accuracy: 0.7582
Epoch 104/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6819 - accuracy: 0.7614 - val_loss: 0.7101 - val_accuracy: 0.7566
Epoch 105/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6665 - accuracy: 0.7661 - val_loss: 0.7010 - val_accuracy: 0.7612
Epoch 106/125

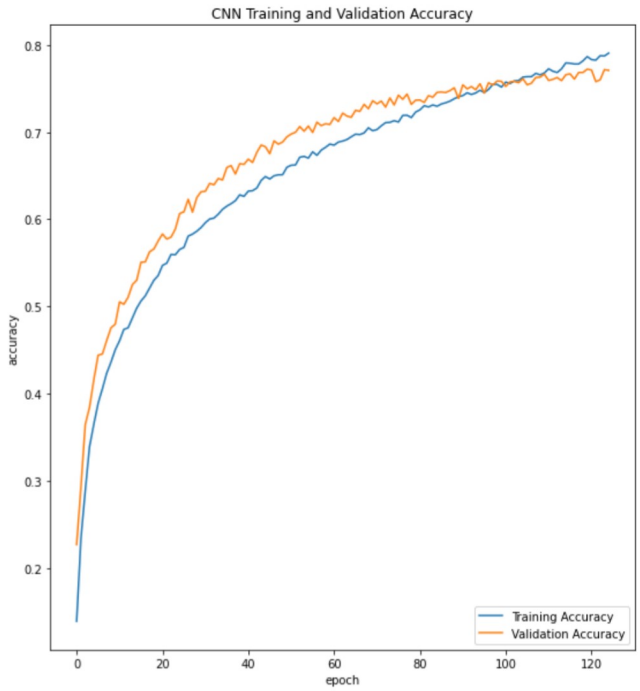
```
Epoch 106/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6742 - accuracy: 0.7611 -
val_loss: 0.7115 - val_accuracy: 0.7542
Epoch 107/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6756 - accuracy: 0.7631 -
val_loss: 0.7005 - val_accuracy: 0.7562
Epoch 108/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6757 - accuracy: 0.7616 -
val_loss: 0.7019 - val_accuracy: 0.7626
Epoch 109/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6673 - accuracy: 0.7653 -
val_loss: 0.7057 - val_accuracy: 0.7630
Epoch 110/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6684 - accuracy: 0.7671 -
val_loss: 0.6952 - val_accuracy: 0.7664
Epoch 111/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6462 - accuracy: 0.7728 -
val_loss: 0.7045 - val_accuracy: 0.7592
Epoch 112/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6550 - accuracy: 0.7711 -
val_loss: 0.7025 - val_accuracy: 0.7606
Epoch 113/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6439 - accuracy: 0.7676 -
val_loss: 0.6899 - val_accuracy: 0.7626
Epoch 114/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6488 - accuracy: 0.7704 -
val_loss: 0.6968 - val_accuracy: 0.7590
Epoch 115/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6315 - accuracy: 0.7796 -
val_loss: 0.6862 - val_accuracy: 0.7658
Epoch 116/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6249 - accuracy: 0.7800 -
val_loss: 0.6900 - val_accuracy: 0.7668
Epoch 117/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6307 - accuracy: 0.7794 -
val_loss: 0.6996 - val_accuracy: 0.7608
Epoch 118/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6417 - accuracy: 0.7764 -
val_loss: 0.6912 - val_accuracy: 0.7684
Epoch 119/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6200 - accuracy: 0.7823 -
val_loss: 0.6843 - val_accuracy: 0.7684
Epoch 120/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6102 - accuracy: 0.7844 -
val_loss: 0.6778 - val_accuracy: 0.7722
Epoch 121/125
64/64 [==============================] - 3s 51ms/step - loss: 0.6142 - accuracy: 0.7857 -
val_loss: 0.6721 - val_accuracy: 0.7712
Epoch 122/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6133 - accuracy: 0.7828 -
val_loss: 0.7006 - val_accuracy: 0.7578
Epoch 123/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6077 - accuracy: 0.7864 -
val_loss: 0.6921 - val_accuracy: 0.7600
Epoch 124/125
64/64 [==============================] - 3s 50ms/step - loss: 0.6067 - accuracy: 0.7845 -
val_loss: 0.6793 - val_accuracy: 0.7716
Epoch 125/125
64/64 [==============================] - 3s 51ms/step - loss: 0.5957 - accuracy: 0.7900 -
val_loss: 0.6718 - val_accuracy: 0.7708
Time taken: 0 hours 7 minutes and 36.47 seconds.
```

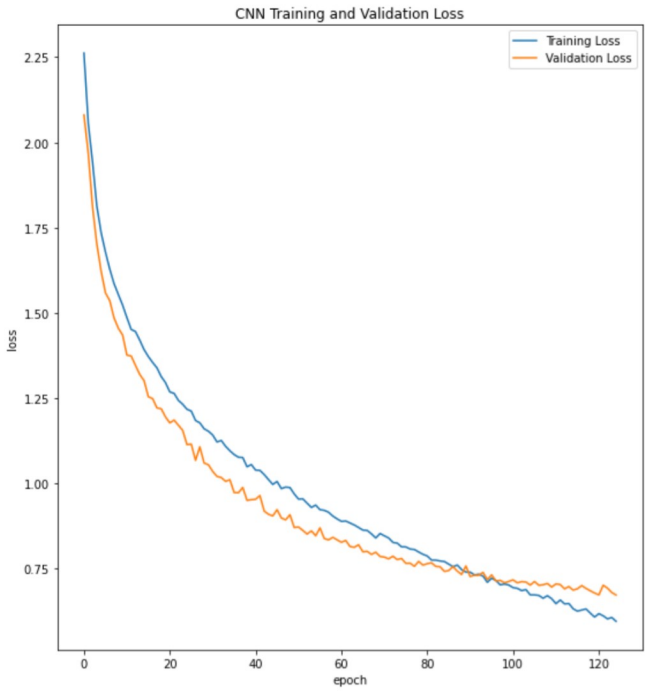
Visualizing the performance

In [12]:

```
plt.figure(figsize=(20, 10))
plt.subplot(1, 2, 1)
plt.title("CNN Training and Validation Accuracy")
plt.plot(result.history["accuracy"], label='Training Accuracy')
```

```
plt.plot(result.history['val_accuracy'], label='Validation Accuracy')
plt.legend(loc='lower right')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.subplot(1, 2, 2)
plt.plot( result.history["loss"], label='Training Loss')
plt.plot( result.history["val_loss"], label='Validation Loss')
plt.legend(loc='upper right')
plt.title('CNN Training and Validation Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.show()
```





Check performance on test

In [13]:

```
# check performance on test
scores = model.evaluate(xtest, ytest, verbose=1)
print('Test loss:', scores[0])
print('Test accuracy:', scores[1])
```

```
313/313 [==============================] - 1s 3ms/step - loss: 0.7058 - accuracy: 0.7556
Test loss: 0.7057846188545227
Test accuracy: 0.7555999755859375
```

Result:

So the model's final Training Accuracy is 79%

Validation Accuracy is 77%

Testing Accuracy is 76%